Обробка зображень і мультимедіа

Олег Гутік

Лекція 7: Стиснення зображень

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Сучасне поле стиснення інформації дуже широке, у ньому зійшло безліч різних способів компресії різних типів даних: текстів, зображень, відео й звуку. Серед цього різноманіття методів особливе місце займає стиснення зображень, оскільки, по-перше, це перша область, де користувачі мають справу з великою кількістю файлів, які необхідно ефективно стискати, а по-друге, тут ми вперше зустрічаємо стиснення даних із частковою втратою інформації.

У кодуванні ASCII кожен символ займає один байт. Типова книга у чверть мільйона слів налічує приблизно мільйон символів та займає близько 1MB. Однак, якщо до цієї книги додати хоч одну картинку для ілюстрації, то обсяг пам'яті може подвоїтися, оскільки файл, що містить зображення, може займати 1MB і навіть більше.

Файл зображення такий великий, тому що він представляє собою двовимірний образ. Крім того, сучасні дисплеї та інші засоби представлення зображень здатні передавати величезну кількість кольорів та відтінків, які вимагають багато бітів (зазвичай 24) для зображення кольору кожного окремого пікселя.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень

Стиснення тексту завжди робиться без втрати інформації. Можна уявити документи, у яких частина інформації не дуже важлива, і тому її можна опустити без спотворення сенсу всього тексту, однак, не існує алгоритмів, які могли б дати комп'ютеру можливість самостійно, без участі людини, визначати, що в тексті важливо, а що — ні. На відміну від стиснення текстів, для компресії зображень такі алгоритми є у великій кількості. Такий алгоритм може видалити більшу частину інформації із зображення, що призводить до чудової компресії. Перевіркою якості стиснення зображення є візуальне порівняння оригінального зображення та зображення, отриманого з його стисненого образу (із втратою частини інформації). Якщо випробувач неспроможний відрізнити одне одного, то стиск із втратою допускається. У деяких випадках людське око здатне вловити різницю, але все одно стиск вважається задовільним.

Обговорення стиснення зображень у цій лекції носить переважно розмаїття теоретичний характер. Тут викладатимуться основні підходи до вирішення проблеми стиснення зображень. З деяких розглянутих конкретних методів компресії, виділяється найбільш важливий з них — це метод JPEG.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

Сучасні комп'ютери дуже активно застосовують графіку. Операційні системи з інтерфейсом віконного типу використовують зображення, наприклад, для відображення директорій або папок. Деякі вчинені системою дії, наприклад завантаження та пересилання файлів, також відображаються графічно. Багато програм та додатків пропонують користувачеві графічний інтерфейс (GUI), який значно спрощує роботу користувача і дозволяє легко інтерпретувати отримані результати. Комп'ютерна графіка використовується у багатьох галузях повсякденної діяльності під час перекладу складних масивів даних у графічне представлення. Отже, графічні зображення є вкрай важливими, але вони вимагають так багато місця! Оскільки сучасні дисплеї передають безліч кольорів, то кожен піксел прийнято інтерпретувати у вигляді 24-бітового числа, в якому компоненти червоного, зеленого та блакитного кольорів займають по 8 біт кожен. Такий 24-бітовий піксель може відображати $2^{24} \approx 16.78$ мільйонів кольорів. Тоді зображення з роздільною здатністю 512×512 пікселів займатиме 786432 байти. А зображення розміром 1024×1024 пікселя знадобиться 3145728 байт для його зберігання. Мультфільми і анімація, які також дуже популярні в комп'ютерних додатках, вимагатимуть більшого обсягу. Усе це пояснює важливість стиснення зображень. На щастя, зображення допускають часткову втрату інформації під час стиснення. Зрештою зображення покликані для того, щоб люди на них дивилися, тому та частина зображення, яка не сприймається оком, може бути опущена. Ця основна ідея вже три десятиліття рухає розробниками методів стиснення, допускають деяку втрату даних.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

У випадку, інформацію можна стиснути, якщо у ній присутня надмірність. У цих лекціях книзі часто повторювалося твердження, що компресія інформації робиться шляхом видалення чи скорочення надмірності даних. При стисканні з втратами ми маємо справу з новою концепцією, з новою парадигмою, яка передбачає стиснення шляхом видалення неістотної, нерелевантної інформації. Зображення можна стиснути з втратою, видаливши несуттєву інформацію, навіть якщо в ньому немає надмірності.

Зазначимо, що зображення без надмірності не обов'язково є випадковим. Ми обговорювали два типи надмірності, властивої графічним образам. Найважливішою є кореляція (залежність) пікселів. У поодиноких випадках кореляція пікселів може бути слабкою або зовсім відсутня, але зображення все ще не буде випадковим і навіть осмисленим.

Ідея про допустимість втрати графічної інформації стане більш приємною після того, як ми розглянемо процес утворення цифрових зображень. Ось три приклади: (1) реальне зображення може бути зіскановано з фотографії або малюнка і оцифровано (переведено в матрицю пікселів); (2) образ може бути записаний відеокамерою, яка сама породжує пікселі та зберігає їх у пам'яті; (3) намальований на екрані комп'ютера за допомогою спеціальної комп'ютерної програми. У всіх трьох випадках деяка інформація втрачається під час перекладу в цифрову форму. Глядач готовий дозволити деяку втрату інформації за умови, що це буде терпима та малопомітна втрата.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

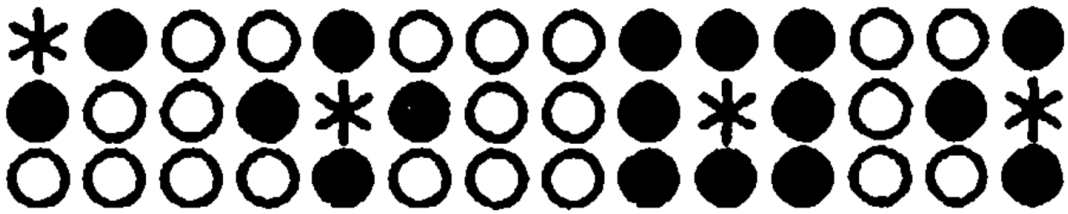
Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

Стиснення зображень. Вступ

Оцифрування зображень складається з двох етапів: вибірки зразків та квантування. Вибірка зразків, дискретизація, полягає у розбитті двовимірного зображення на дрібні області, пікселі, а під квантуванням мається на увазі процес присвоєння кожному пікселю деякого числа, що означає його колір. Зауважимо, що оцифровування звуку складається з тих самих етапів, але тільки звуковий потік є одновимірним.

Наведемо простий тест на якісне визначення втрати інформації під час стиснення зображення. Нехай даний образ (1) стиснутий B, (2) розтиснутий в C, і (3) їхня різниця позначена $D = C - A$. Якщо A був стиснутий без втрати інформації і розтиснути належним чином, то C повинен бути ідентичним до A, і образ D має бути рівномірно білим. Що більше інформації втрачено, то далі буде D від рівномірно білого образу.

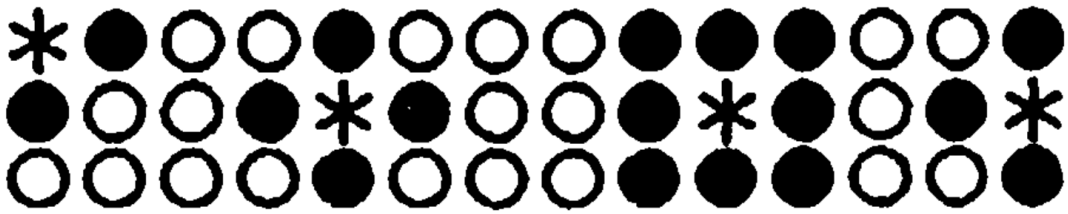
Стиснення зображень. Вступ



Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

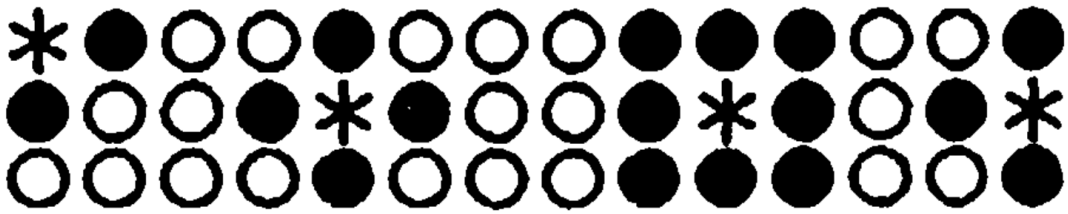
1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.



Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

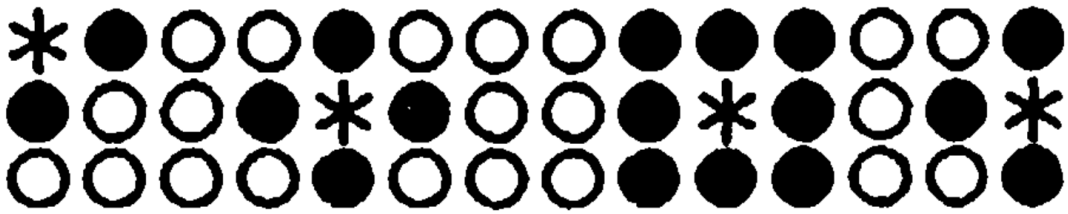


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

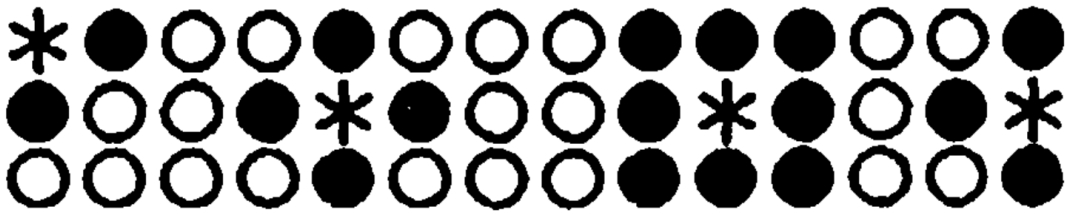


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

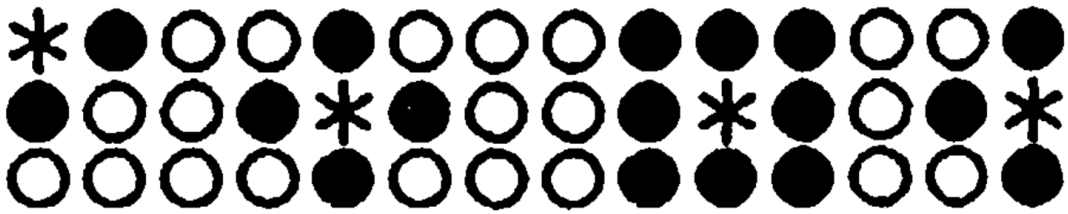


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

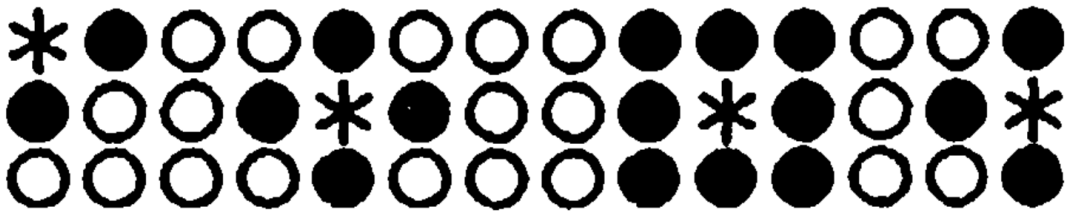


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

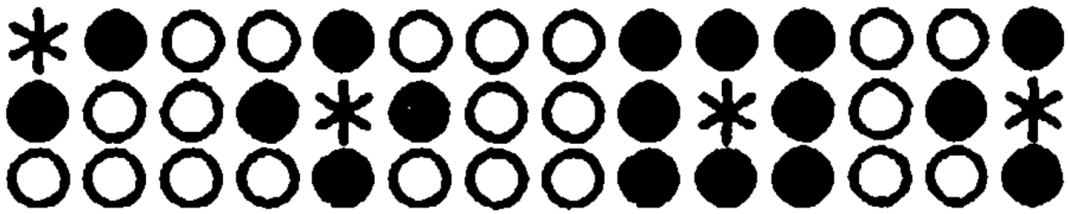


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері "і" передують літери "н", "р", "л", "с", "п", а за нею слідують літери "е", "в", "н", "м", "р". В інших абзацах та сама буква "і" може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя "$*$" показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

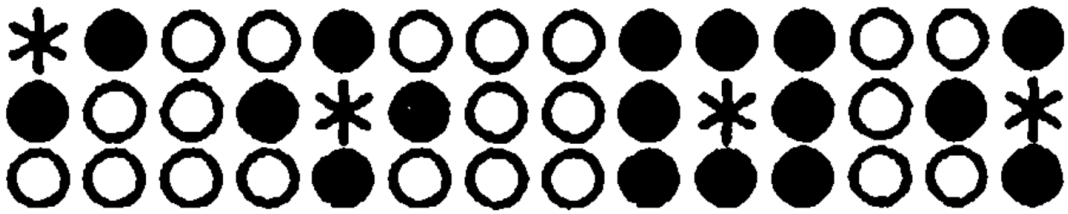


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

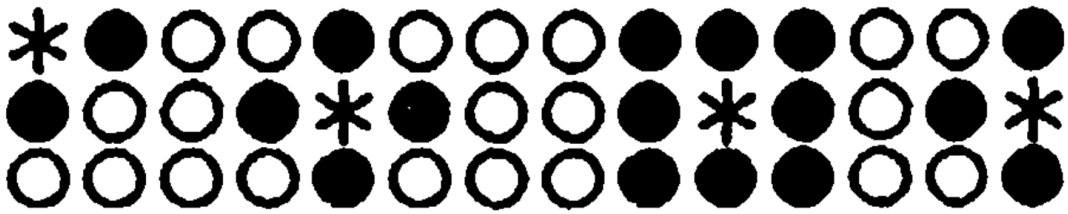


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

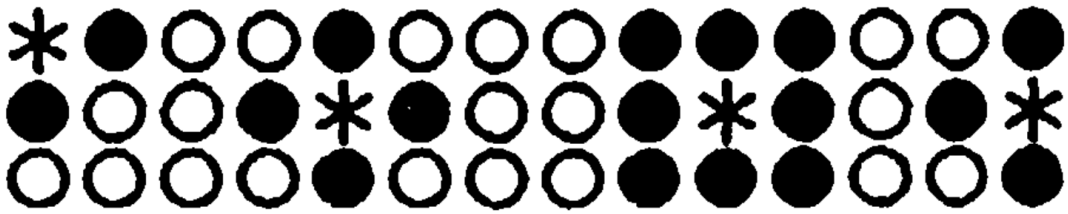


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

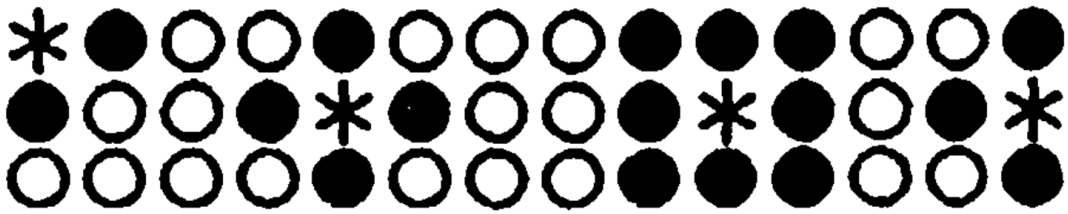


Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

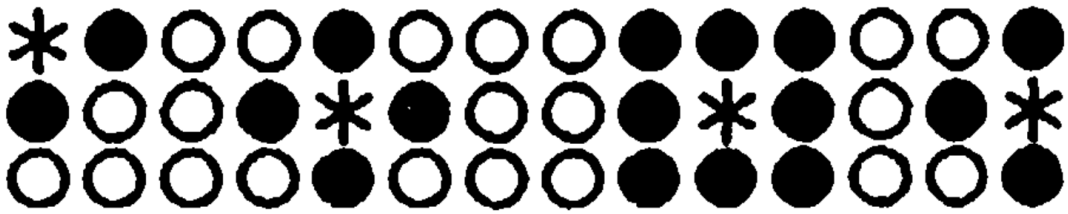
Стиснення зображень. Вступ



Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.



Чотири найближчих та вісім сусідніх пікселів

Нові методи стиснення повинні враховувати три основні відмінності між текстами та графічними зображеннями:

1. Текст одновимірний, а зображення має розмірність 2. Весь текст можна розглядати як один довгий рядок символів. Кожна літера тексту має двох сусідів, ліворуч та праворуч. Усі сусіди дуже слабко кореловані між собою. Наприклад, у цьому абзаці літері “і” передують літери “н”, “р”, “л”, “с”, “п”, а за нею слідують літери “е”, “в”, “н”, “м”, “р”. В інших абзацах та сама буква “і” може мати інших сусідів. У зображенні піксель має чотири безпосередніх сусідів і вісім найближчих (за винятком пікселів, що лежать на кордоні, див. рис., де вісім найближчих сусідів пікселя ”$*$” показані чорним кольором), і між ними існує сильна кореляція.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

2. Текст складається з відносно небагатьох символів алфавіту. Зазвичай це 128 кодів ASCII або 256 байтів довжини по 8 біт кожен. Навпаки, кожен піксель зображення представимо 24 бітами, а тому може бути до $2^{24} \approx 16.78$ мільйонів різних пікселів. Отже, кількість елементарних "символів" у зображенні може бути величезним числом.

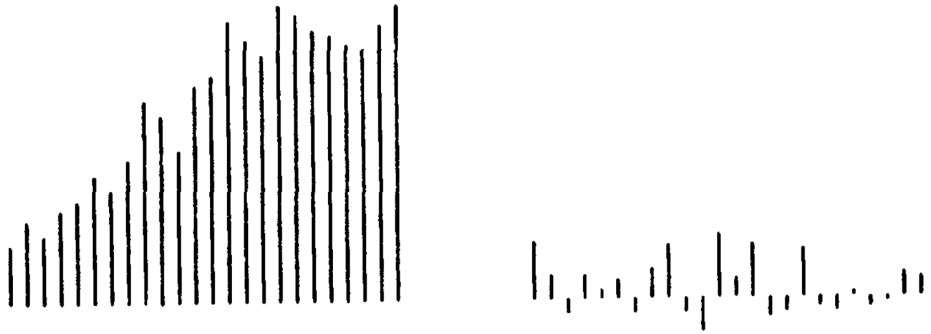
3. Не відомий алгоритм, який визначав би, яка частина тексту є неважливою або малозначущою, і її можна видалити без шкоди для всього тексту, але існують методи, що автоматично видаляють неважливу інформацію з графічного образу. Цим досягається значний ступінь компресії.

Отже, методи стиснення текстової інформації стають малоефективними та незадовільними під час роботи із зображеннями. Тому в цьому розділі обговорюватимуться зовсім інші підходи до вирішення цього завдання. Вони різні, але вони видаляють надмірність з допомогою наступного принципу.

Принцип стиснення зображень. *Якщо випадково вибрати піксель зображення, то з великою ймовірністю ближчі до нього пікселі матимуть той же або близький колір.*

Отже, стиснення зображень ґрунтується на сильній кореляції сусідніх пікселів. Ця кореляція також називається *просторовою надмірністю*.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

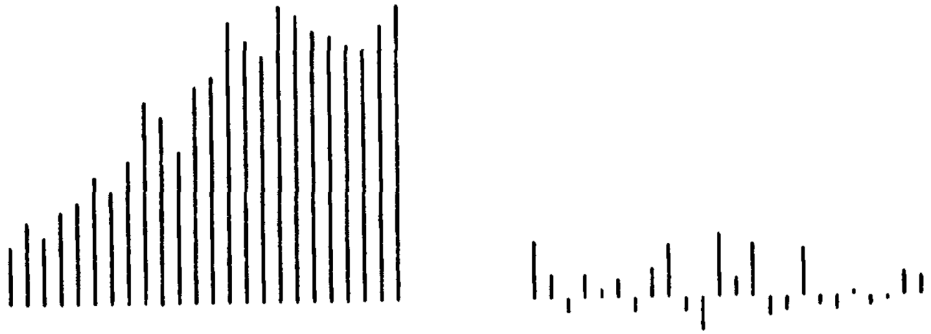
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

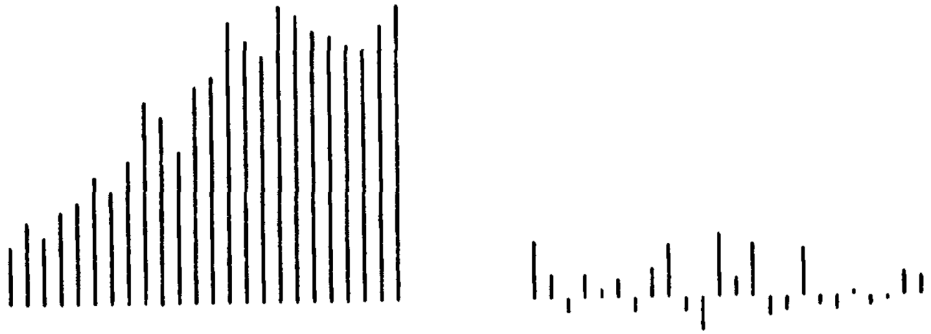
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

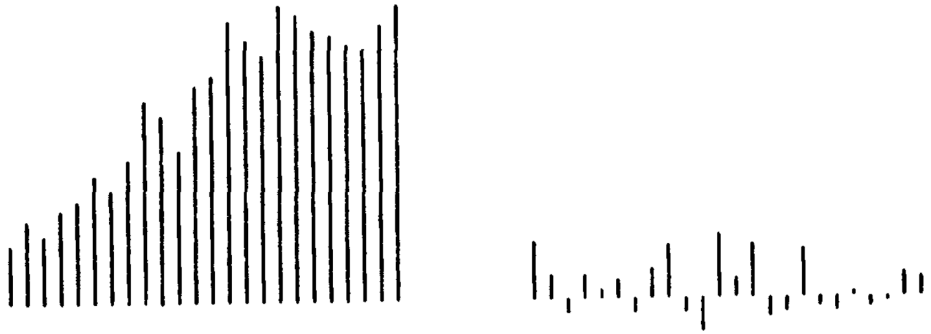
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

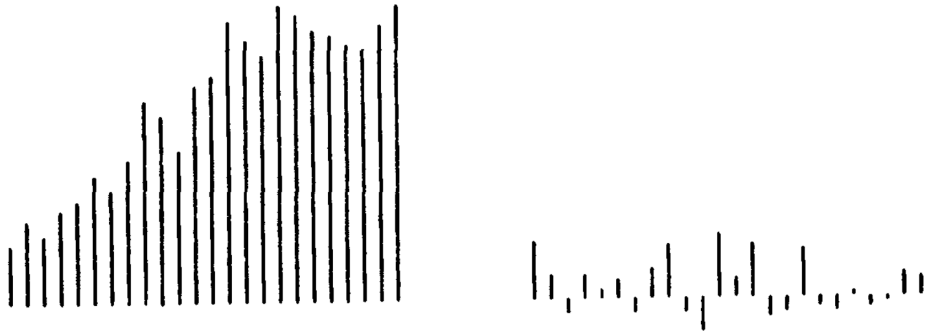
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

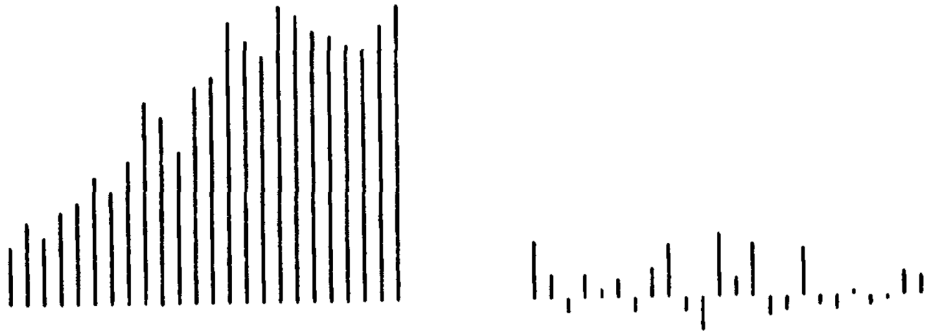
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3.

Віднімання кожного попереднього символу дасть послідовність

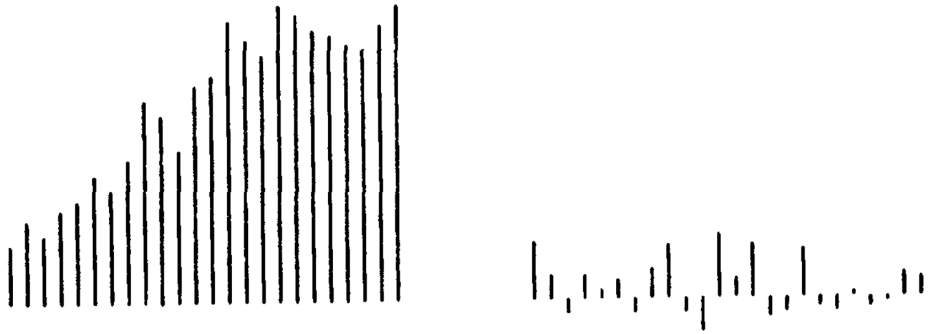
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

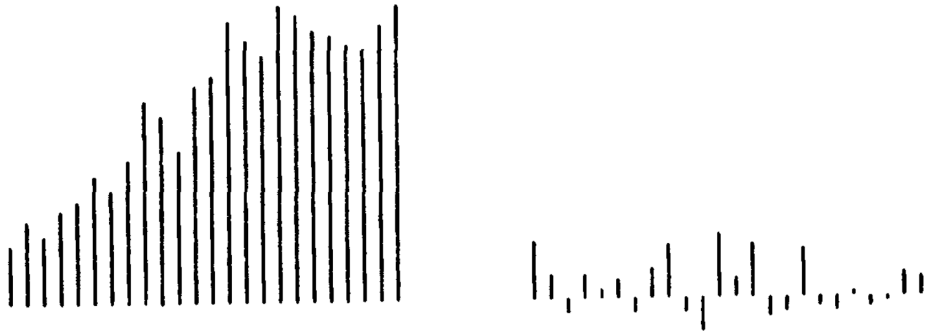
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

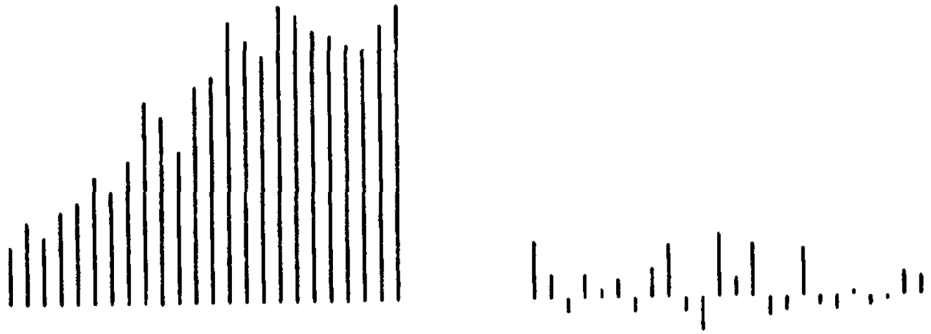
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

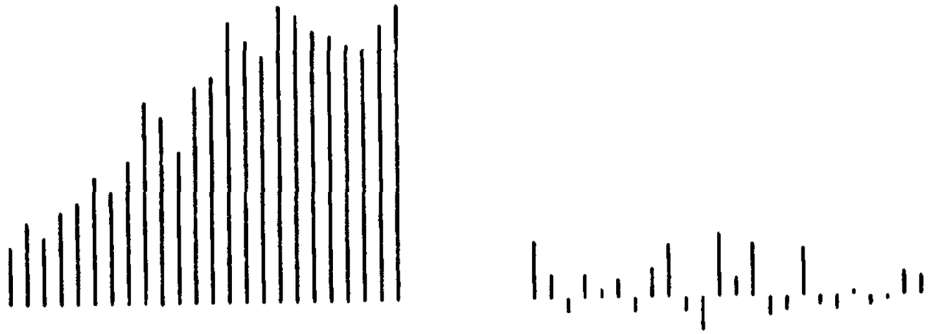
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

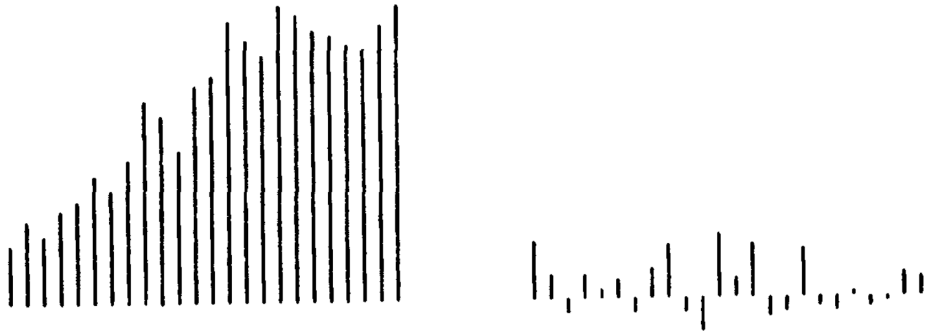
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

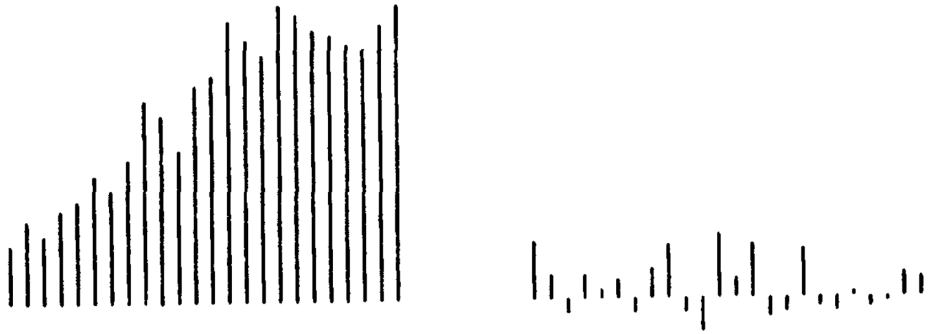
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

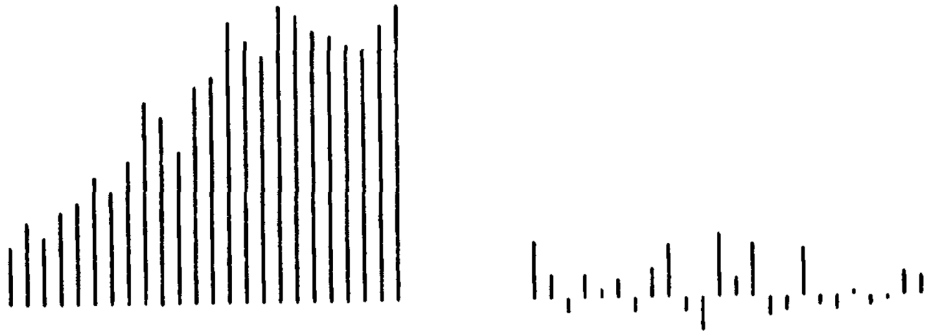
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

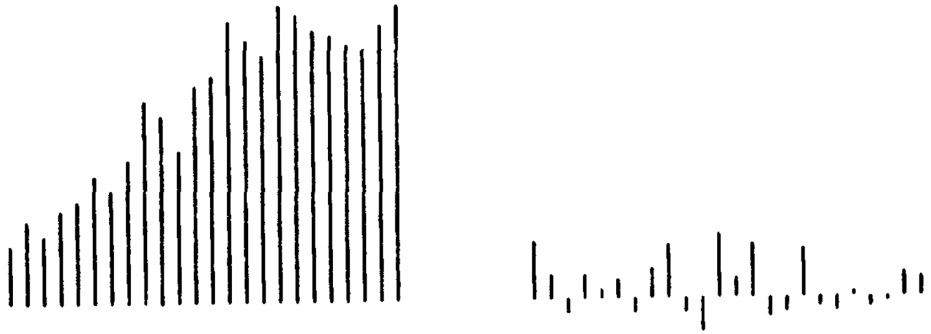
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

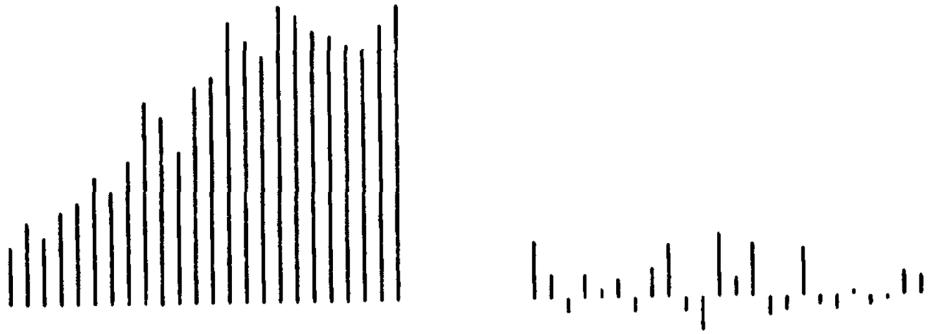
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

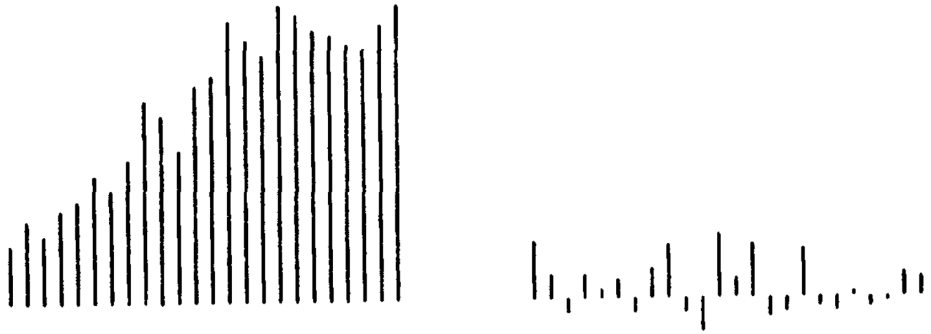
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

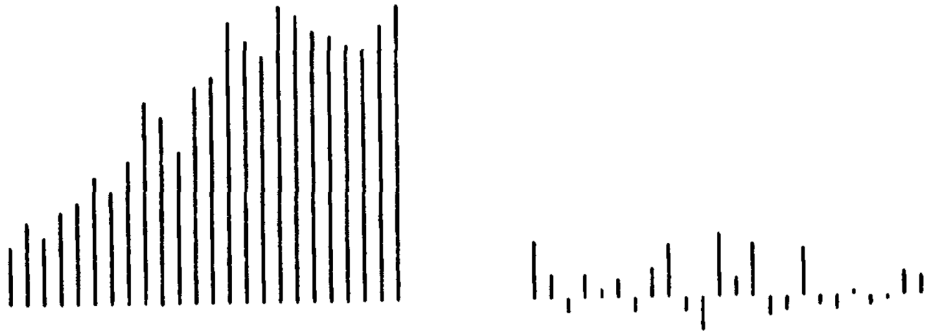
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

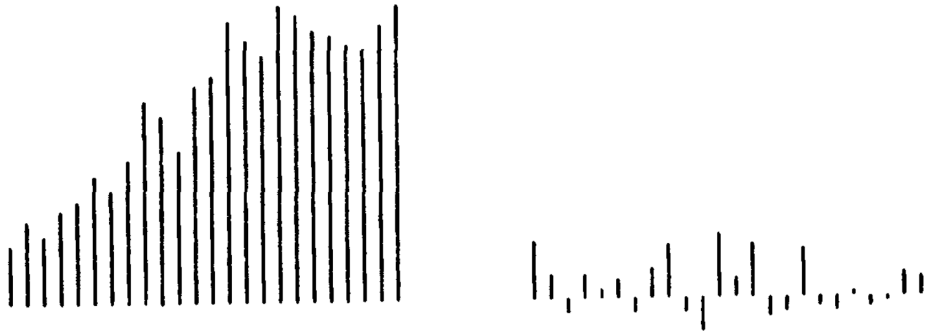
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

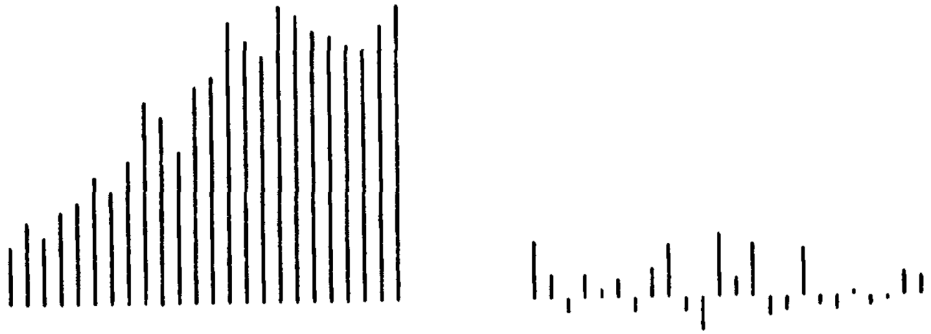
$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ



Ось простий приклад того, що можна зробити із корельованими пікселами. Наступна послідовність чисел відображає інтенсивність 24 суміжних пікселів в одному рядку деякого неперервно тонового зображення:

$$12, 17, 14, 19, 21, 26, 23, 29, 41, 38, 31, 44, 46, 57, 53, 50, 60, 58, 55, 54, 52, 51, 56, 60.$$

Тут лише два пікселі збігаються. Їхнє середнє значення дорівнює 40.3. Віднімання кожного попереднього символу дасть послідовність

$$12, 5, -3, 5, 2, 5, -3, 6, 12, -3, -7, 13, 2, 11, -4, -3, 10, -2, -3, -1, -2, -1, 5, 4.$$

Ці послідовності пікселів ілюстровані графічно на рис. який показує потенціал стиснення: (1) різниця пікселів істотно менша від їх абсолютних величин. Їхнє середнє дорівнює 2.58. (2) Вони повторюються. Є лише 15 різних значень. У принципі, кожен із них можна закодувати 4 бітами. Вони декорельовані, тобто різниці сусідніх значень, у середньому, не зменшуються. Це видно з послідовності других різниць:

$$12, -7, -8, 8, -3, 3, -8, 9, 6, -15, -4, 20, -11, 9, -15, -1, 13, -12, -1, 2, -1, -1, 6, 1.$$

Ці різниці вже більші за вихідні величини.

Стиснення зображень. Вступ

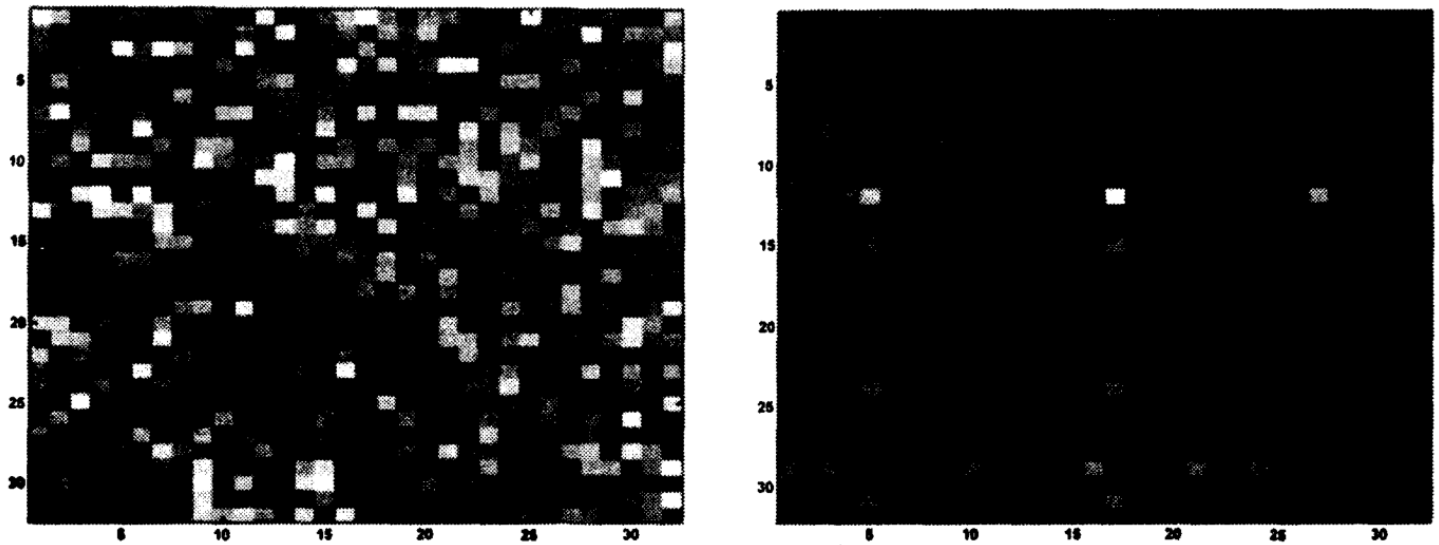


Рис. пропонує іншу ілюстрацію “корельованих величин”. Матриця розміром 32×32 заповнена випадковими числами, та її значення малюнку (a) зображені квадратиками різних сірих відтінків. Випадкова природа цих елементів є очевидною. Потім цю матрицю обернули і результат — матрицю — зобразили на малюнку (b). Цього разу цей масив квадратиків 32×32 виглядає більш структурованим.

Стиснення зображень. Вступ

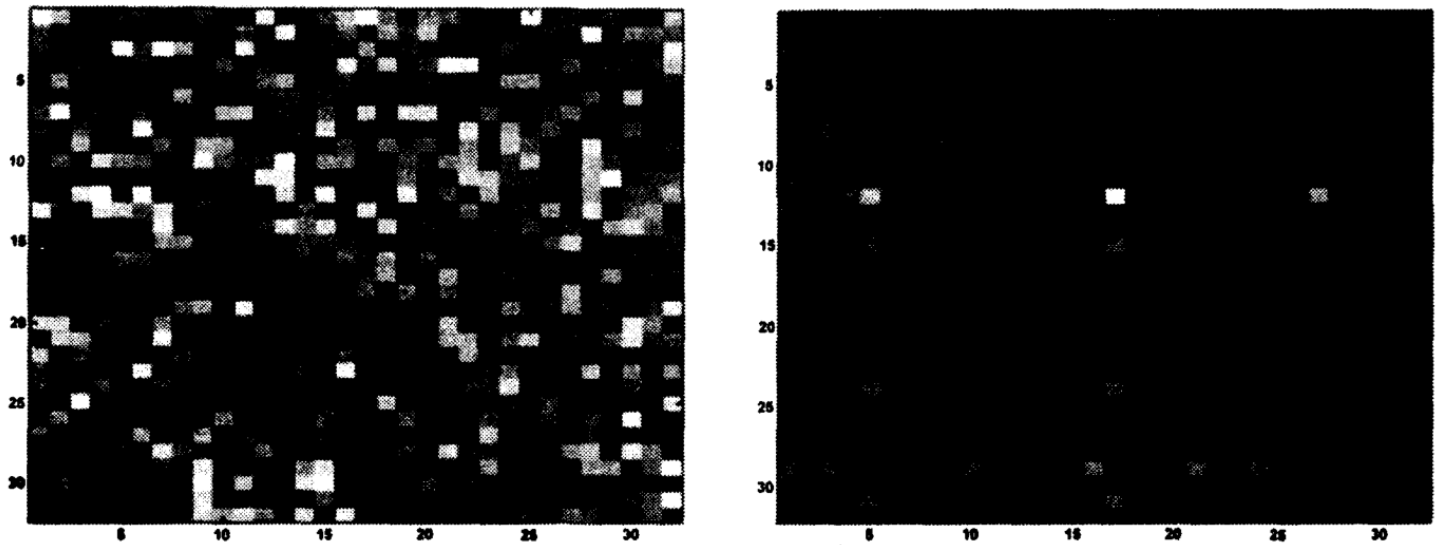


Рис. пропонує іншу ілюстрацію “корельованих величин”. Матриця розміром 32×32 заповнена випадковими числами, та її значення малюнку (a) зображені квадратиками різних сірих відтінків. Випадкова природа цих елементів є очевидною. Потім цю матрицю обернули і результат — матрицю — зобразили на малюнку (b). Цього разу цей масив квадратиків 32×32 виглядає більш структурованим.

Стиснення зображень. Вступ

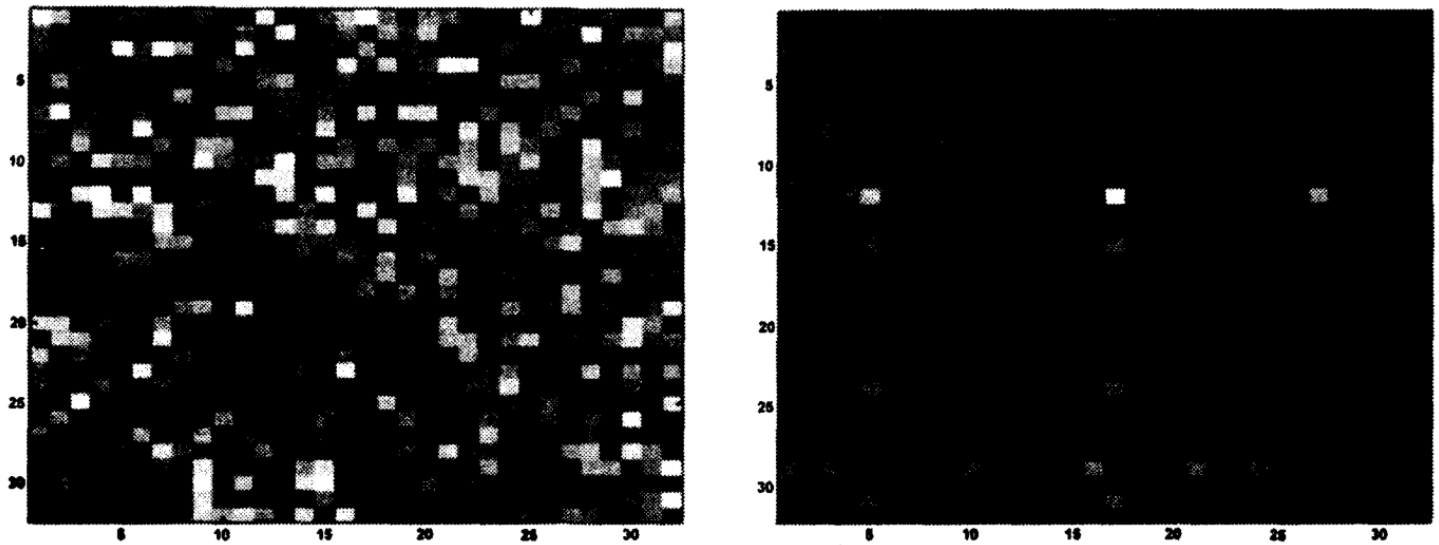


Рис. пропонує іншу ілюстрацію “корельованих величин”. Матриця розміром 32×32 заповнена випадковими числами, та її значення малюнку (a) зображені квадратиками різних сірих відтінків. Випадкова природа цих елементів є очевидною. Потім цю матрицю обернули і результат — матрицю — зобразили на малюнку (b). Цього разу цей масив квадратиків 32×32 виглядає більш структурованим.

Стиснення зображень. Вступ

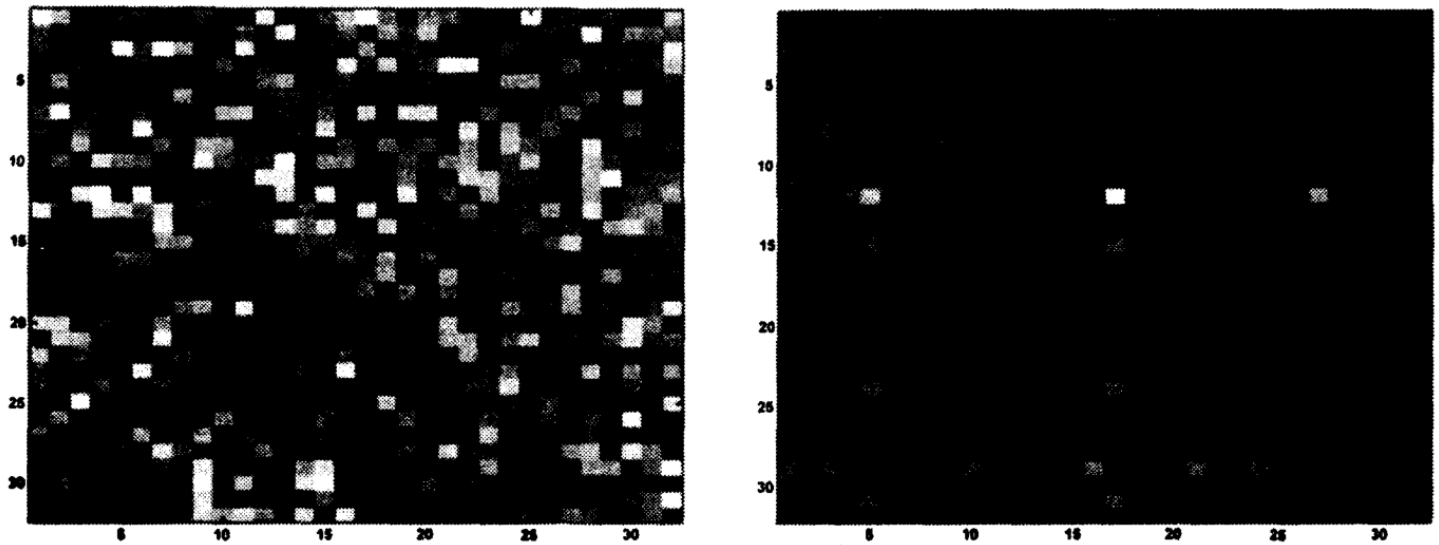
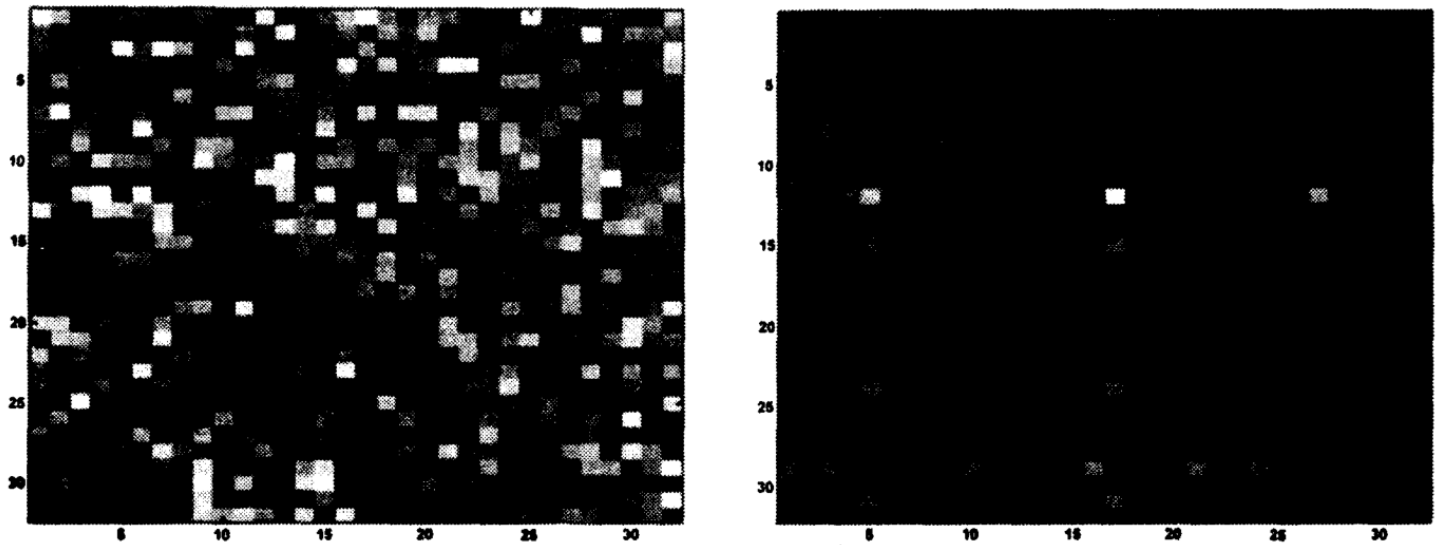


Рис. пропонує іншу ілюстрацію “корельованих величин”. Матриця розміром 32×32 заповнена випадковими числами, та її значення малюнку (a) зображені квадратиками різних сірих відтінків. Випадкова природа цих елементів є очевидною. Потім цю матрицю обернули і результат — матрицю — зобразили на малюнку (b). Цього разу цей масив квадратиків 32×32 виглядає більш структурованим.

Стиснення зображень. Вступ



(a) (b)

Рис. пропонує іншу ілюстрацію “корельованих величин”. Матриця розміром 32×32 заповнена випадковими числами, та її значення малюнку (a) зображені квадратиками різних сірих відтінків. Випадкова природа цих елементів є очевидною. Потім цю матрицю обернули і результат — матрицю — зобразили на малюнку (b). Цього разу цей масив квадратиків 32×32 виглядає більш структурованим.

Стиснення зображень. Вступ

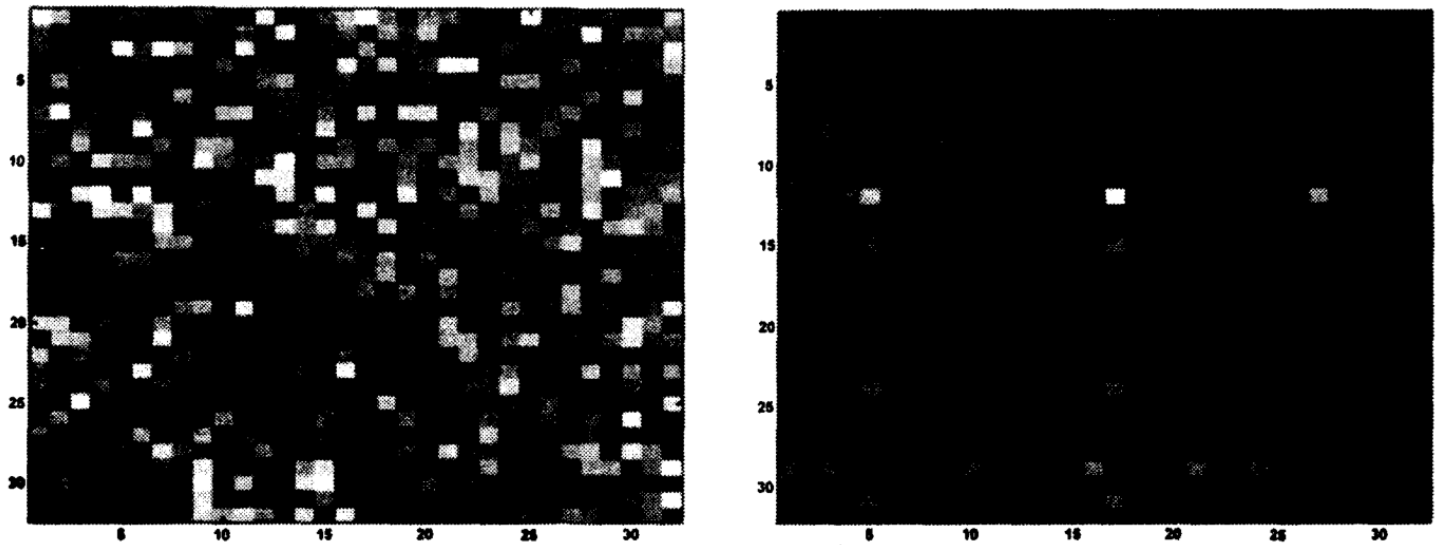


Рис. пропонує іншу ілюстрацію “корельованих величин”. Матриця розміром 32×32 заповнена випадковими числами, та її значення малюнку (a) зображені квадратиками різних сірих відтінків. Випадкова природа цих елементів є очевидною. Потім цю матрицю обернули і результат — матрицю — зобразили на малюнку (b). Цього разу цей масив квадратиків 32×32 виглядає більш структурованим.

Стиснення зображень. Вступ

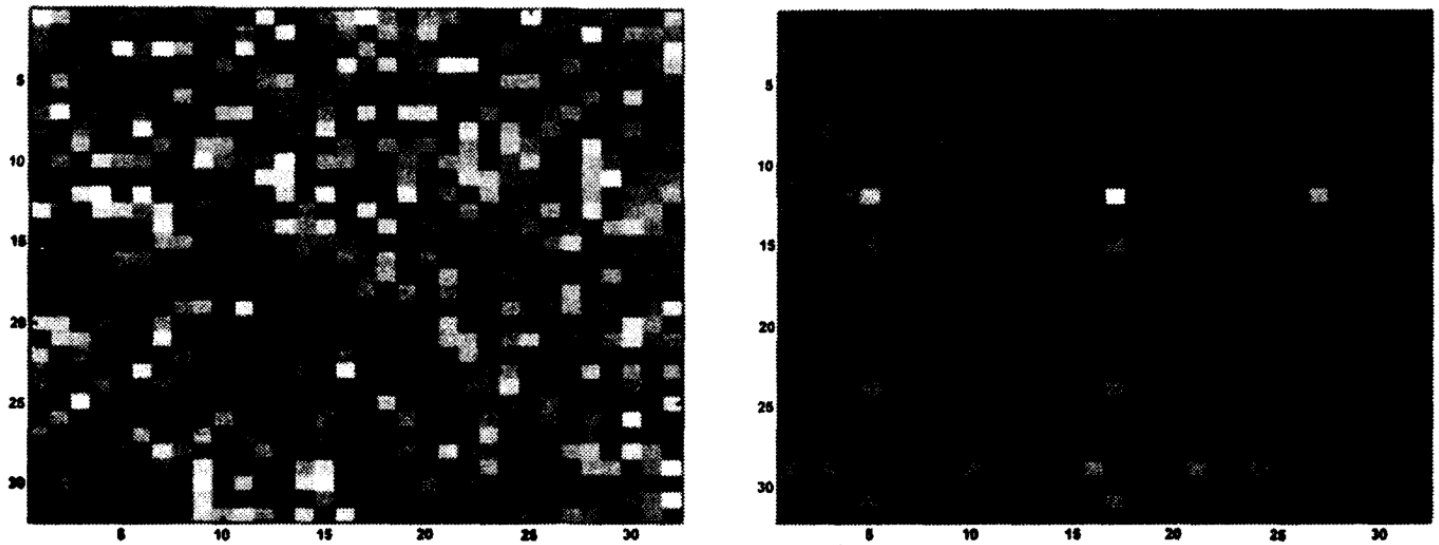
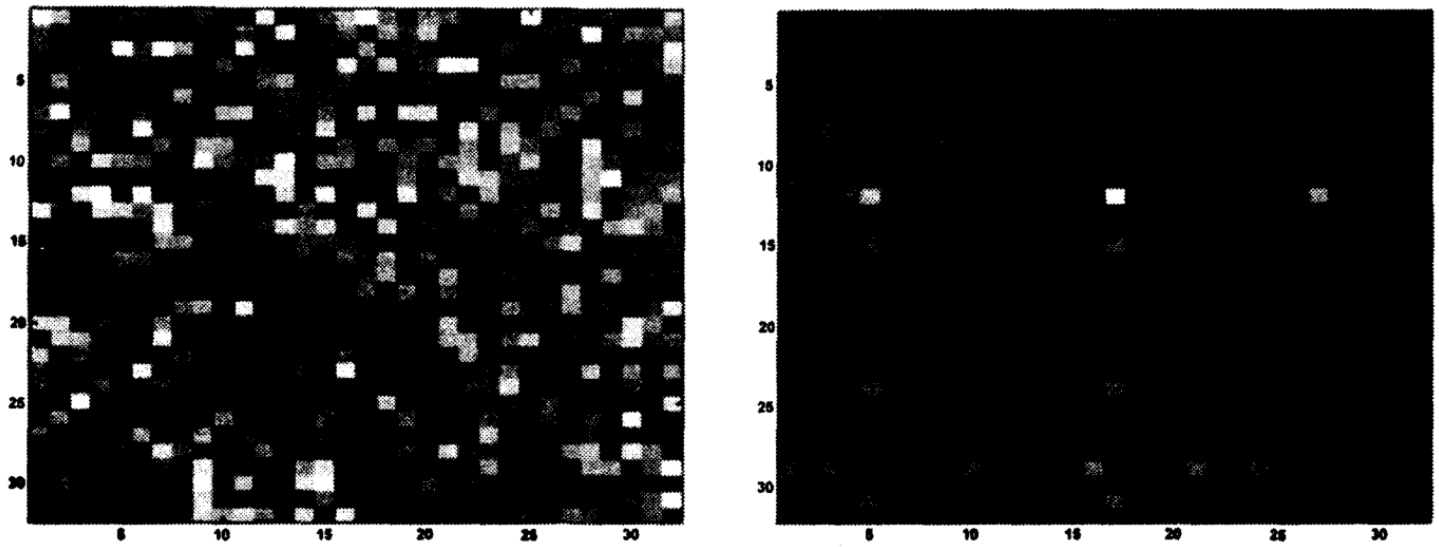


Рис. пропонує іншу ілюстрацію “корельованих величин”. Матриця розміром 32×32 заповнена випадковими числами, та її значення малюнку (a) зображені квадратиками різних сірих відтінків. Випадкова природа цих елементів є очевидною. Потім цю матрицю обернули і результат — матрицю — зобразили на малюнку (b). Цього разу цей масив квадратиків 32×32 виглядає більш структурованим.

Стиснення зображень. Вступ



(a) (b)

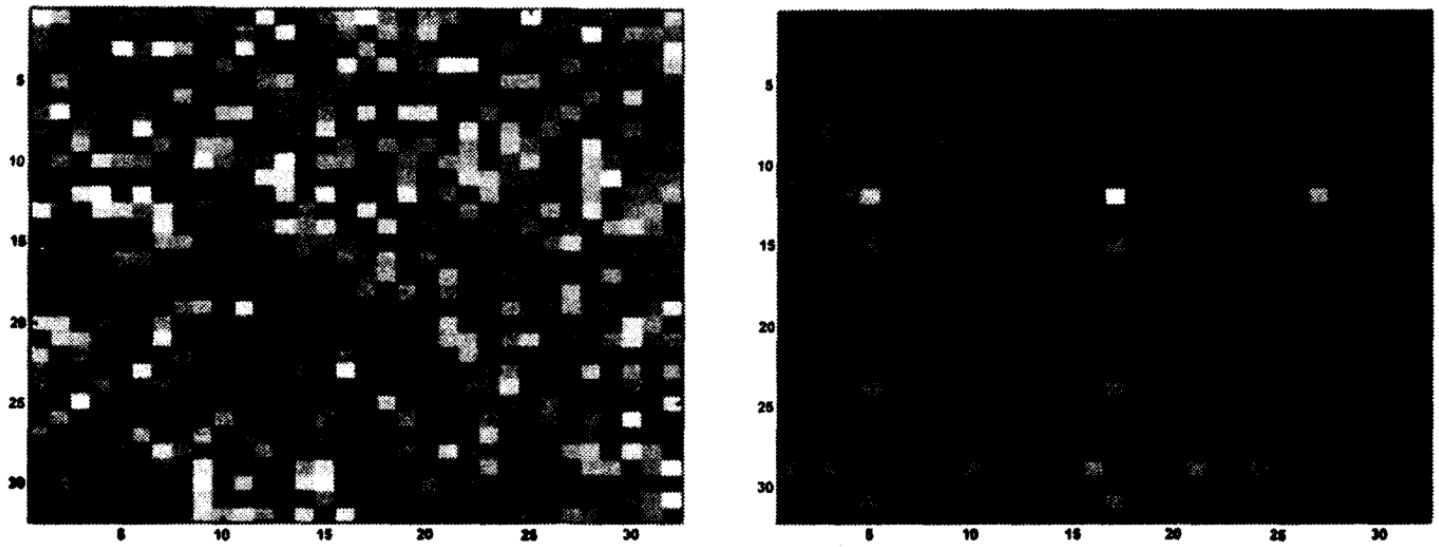
Пряме обчислення кореляції за допомогою коефіцієнта Пірсона за формулою (1) стверджує, що перехресна кореляція верхніх двох рядків матриці A є відносно малим числом 0.0412, у той час, як відповідна величини, обчислена для верхніх рядків матриці дорівнює відносно великому числу 10. Справа в тому, що кожен елемент матриці залежить від всіх елементів матриці

$$R = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{\left(n\sum x_i^2 - (\sum x_i)^2\right)\left(n\sum y_i^2 - (\sum y_i)^2\right)}}. \qquad (1)$$

```
n=32; a=rand(n); imagesc(a); colormap(gray)
b=inv(a); imagesc(b)
```

Программа на Matlab для рис.

Стиснення зображень. Вступ



(a) (b)

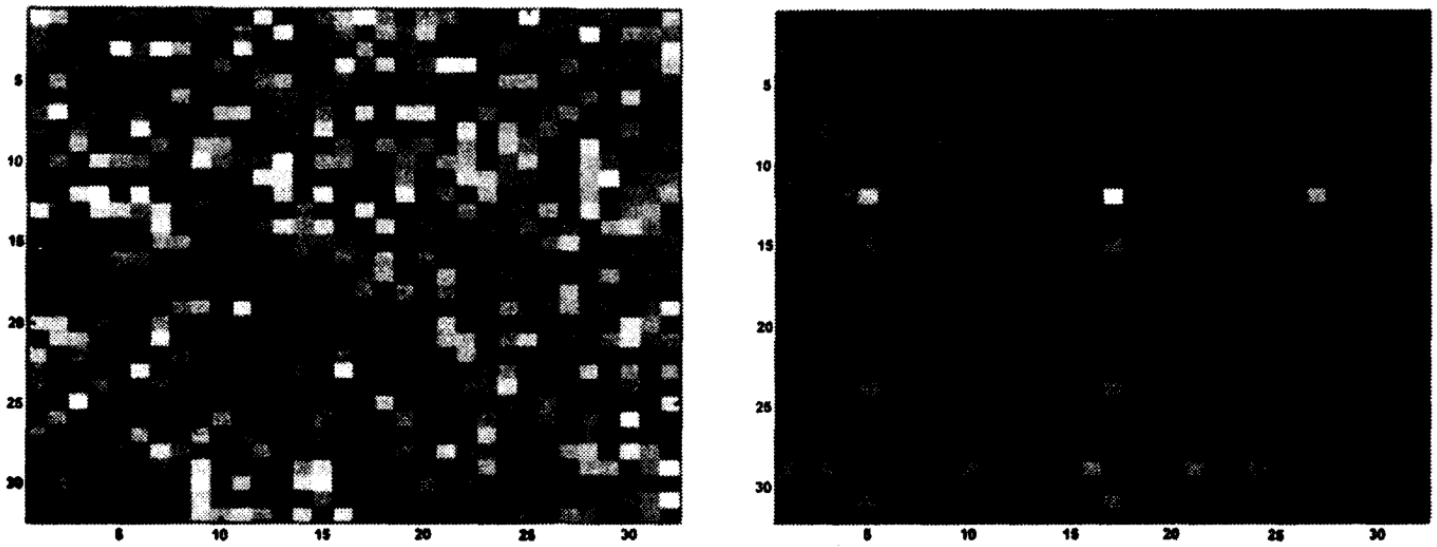
Пряме обчислення кореляції за допомогою коефіцієнта Пірсона за формулою (1) стверджує, що перехресна кореляція верхніх двох рядків матриці A є відносно малим числом 0.0412, у той час, як відповідна величини, обчислена для верхніх рядків матриці дорівнює відносно великому числу 10. Справа в тому, що кожен елемент матриці залежить від всіх елементів матриці

$$R = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{\left(n\sum x_i^2 - (\sum x_i)^2\right)\left(n\sum y_i^2 - (\sum y_i)^2\right)}}. \quad (1)$$

```
n=32; a=rand(n); imagesc(a); colormap(gray)
b=inv(a); imagesc(b)
```

Программа на Matlab для рис.

Стиснення зображень. Вступ



(a) (b)

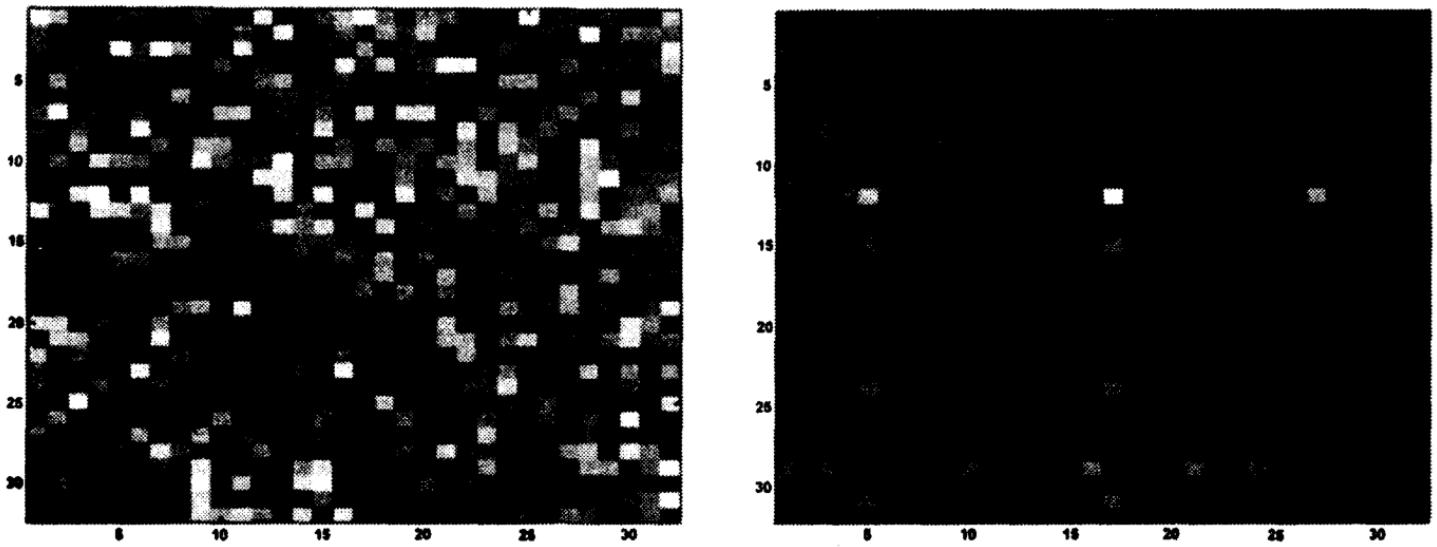
Пряме обчислення кореляції за допомогою коефіцієнта Пірсона за формулою (1) стверджує, що перехресна кореляція верхніх двох рядків матриці A є відносно малим числом 0.0412, у той час, як відповідна величини, обчислена для верхніх рядків матриці дорівнює відносно великому числу 10. Справа в тому, що кожен елемент матриці залежить від всіх елементів матриці

$$R = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{\left(n\sum x_i^2 - \left(\sum x_i\right)^2\right)\left(n\sum y_i^2 - \left(\sum y_i\right)^2\right)}}. \quad (1)$$

```
n=32; a=rand(n); imagesc(a); colormap(gray)
b=inv(a); imagesc(b)
```

Программа на Matlab для рис.

Стиснення зображень. Вступ



(a) (b)

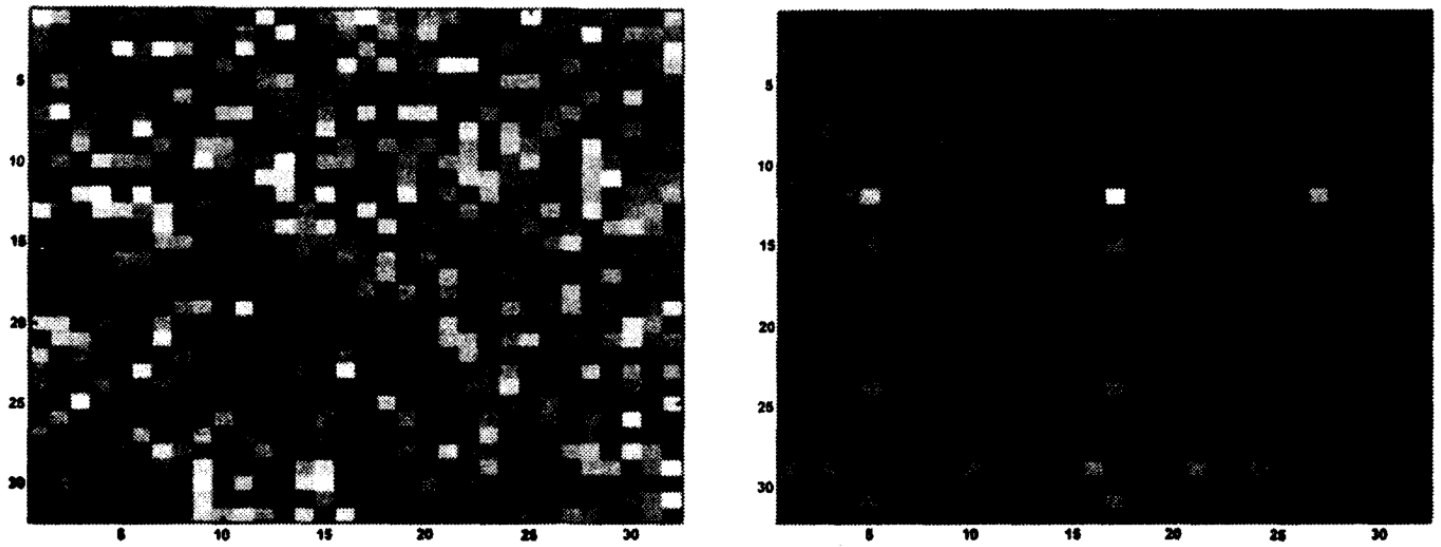
Пряме обчислення кореляції за допомогою коефіцієнта Пірсона за формулою (1) стверджує, що перехресна кореляція верхніх двох рядків матриці A є відносно малим числом 0.0412, у той час, як відповідна величини, обчислена для верхніх рядків матриці дорівнює відносно великому числу 10. Справа в тому, що кожен елемент матриці залежить від всіх елементів матриці

$$R = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{\left(n\sum x_i^2 - \left(\sum x_i\right)^2\right)\left(n\sum y_i^2 - \left(\sum y_i\right)^2\right)}}. \qquad (1)$$

```
n=32; a=rand(n); imagesc(a); colormap(gray)
b=inv(a); imagesc(b)
```

Программа на Matlab для рис.

Стиснення зображень. Вступ



(a) (b)

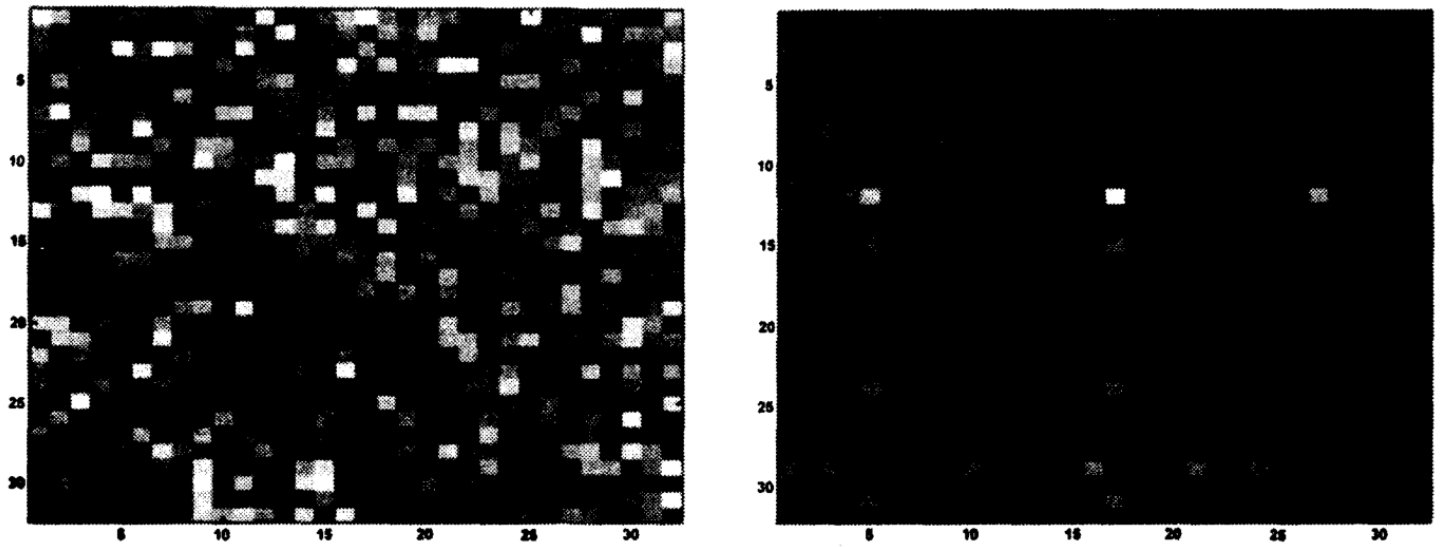
Пряме обчислення кореляції за допомогою коефіцієнта Пірсона за формулою (1) стверджує, що перехресна кореляція верхніх двох рядків матриці A є відносно малим числом 0.0412, у той час, як відповідна величини, обчислена для верхніх рядків матриці дорівнює відносно великому числу 10. Справа в тому, що кожен елемент матриці залежить від всіх елементів матриці

$$R = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{\left(n\sum x_i^2 - \left(\sum x_i\right)^2\right)\left(n\sum y_i^2 - \left(\sum y_i\right)^2\right)}}. \qquad (1)$$

```
n=32; a=rand(n); imagesc(a); colormap(gray)
b=inv(a); imagesc(b)
```

Программа на Matlab для рис.

Стиснення зображень. Вступ



(a) (b)

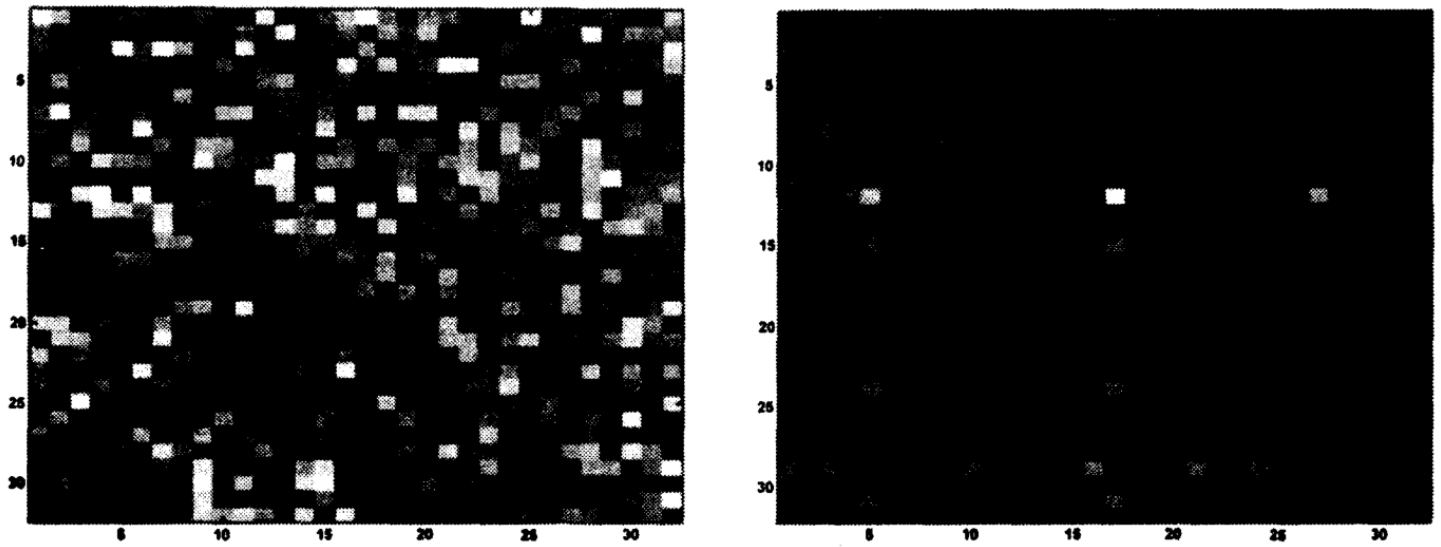
Пряме обчислення кореляції за допомогою коефіцієнта Пірсона за формулою (1) стверджує, що перехресна кореляція верхніх двох рядків матриці A є відносно малим числом 0.0412, у той час, як відповідна величини, обчислена для верхніх рядків матриці дорівнює відносно великому числу 10. Справа в тому, що кожен елемент матриці залежить від всіх елементів матриці

$$R = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{\left(n\sum x_i^2 - (\sum x_i)^2\right)\left(n\sum y_i^2 - (\sum y_i)^2\right)}}. \qquad (1)$$

```
n=32; a=rand(n); imagesc(a); colormap(gray)
b=inv(a); imagesc(b)
```

Программа на Matlab для рис.

Стиснення зображень. Вступ



(a) (b)

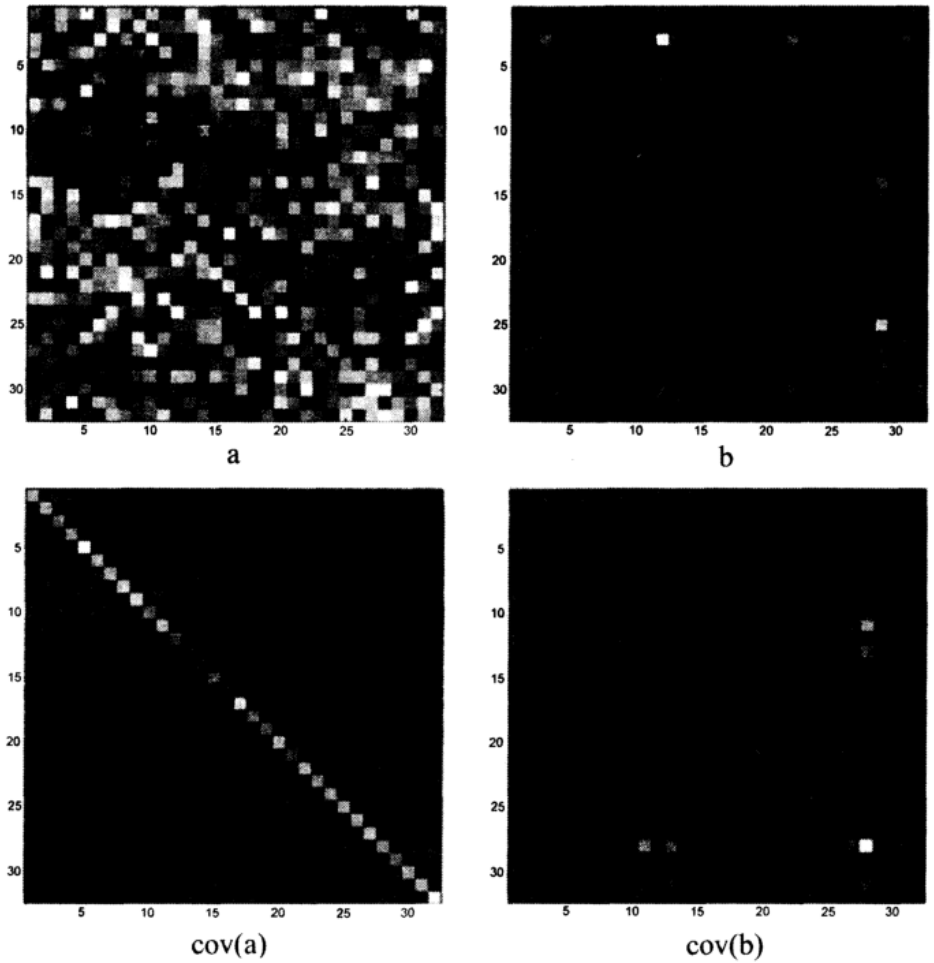
Пряме обчислення кореляції за допомогою коефіцієнта Пірсона за формулою (1) стверджує, що перехресна кореляція верхніх двох рядків матриці A є відносно малим числом 0.0412, у той час, як відповідна величини, обчислена для верхніх рядків матриці дорівнює відносно великому числу 10. Справа в тому, що кожен елемент матриці залежить від всіх елементів матриці

$$R = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{\left(n\sum x_i^2 - (\sum x_i)^2\right)\left(n\sum y_i^2 - (\sum y_i)^2\right)}}. \qquad (1)$$

```
n=32; a=rand(n); imagesc(a); colormap(gray)
b=inv(a); imagesc(b)
```

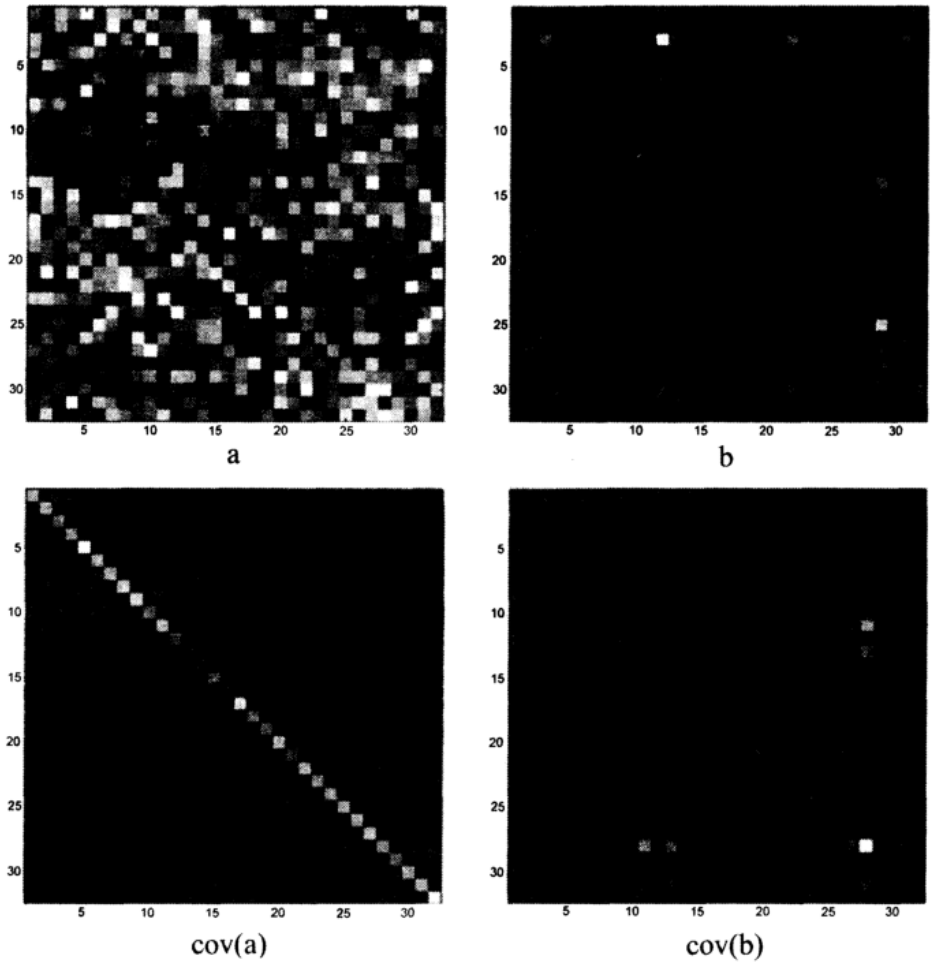
Программа на Matlab для рис.

Стиснення зображень. Вступ



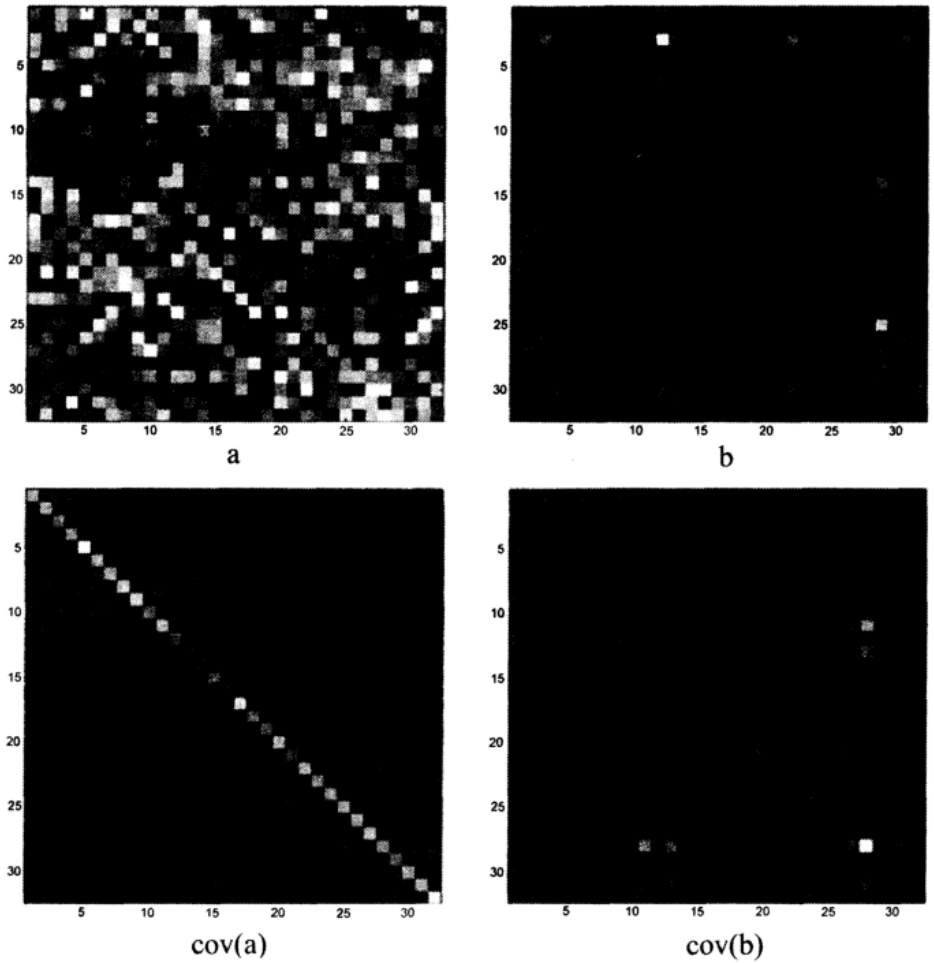
Приклад. Скористаємося спеціальною програмою для ілюстрації коварійної матриці для (1) матриці з корельованими елементами та (2) матриці з декорельованими елементами. На рис. зображено дві 32×32 матриці. Перша з них є випадковою (тобто, декорельованою), а друга матриця b є оборотною для матриці a (отже, вона корельована).

Стиснення зображень. Вступ



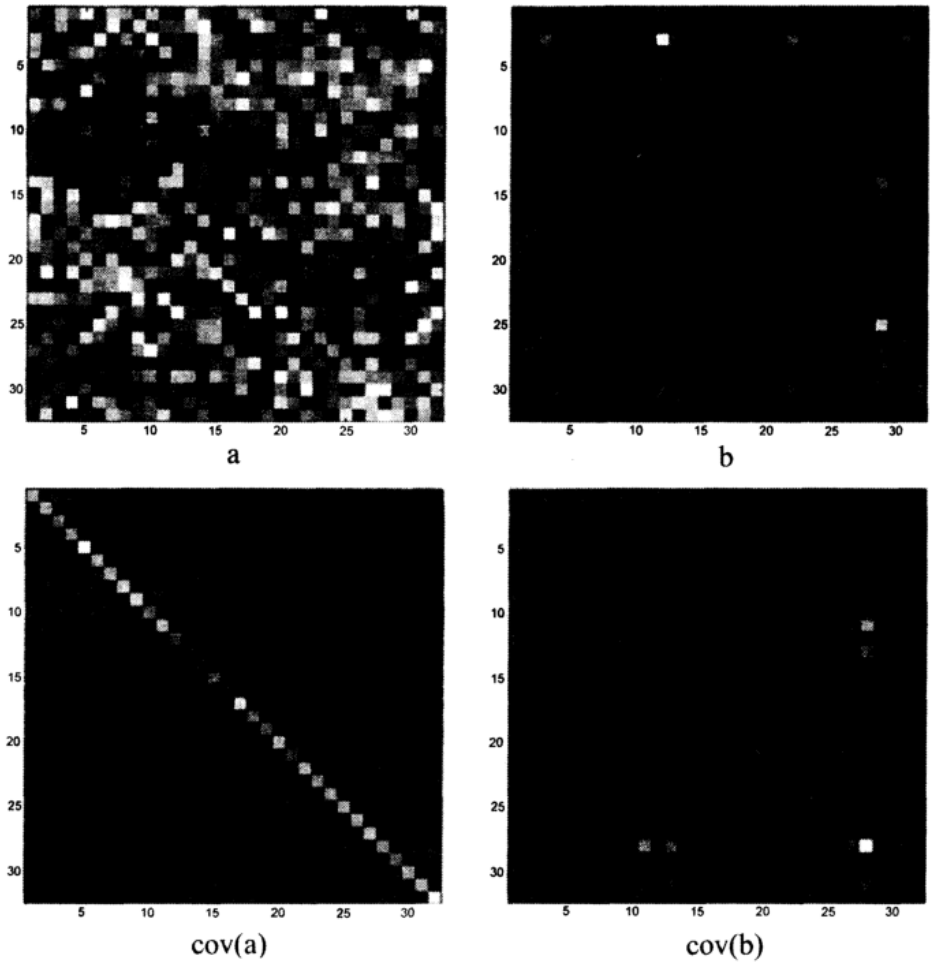
Приклад. Скористаємося спеціальною програмою для ілюстрації коварійної матриці для (1) матриці з корельованими елементами та (2) матриці з декорельованими елементами. На рис. зображено дві 32×32 матриці. Перша з них є випадковою (тобто, декорельованою), а друга матриця b є оборотною для матриці a (отже, вона корельована).

Стиснення зображень. Вступ



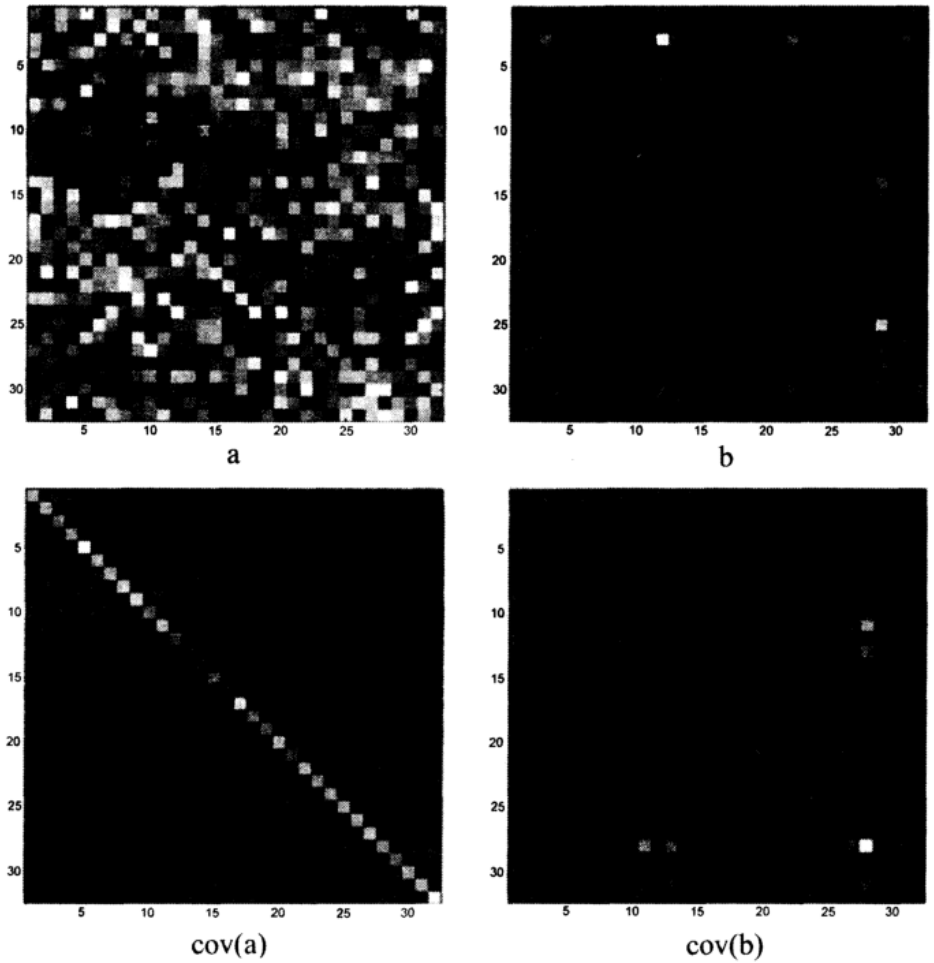
Приклад. Скористаємося спеціальною програмою для ілюстрації коварійної матриці для (1) матриці з корельованими елементами та (2) матриці з декорельованими елементами. На рис. зображено дві 32×32 матриці. Перша з них є випадковою (тобто, декорельованою), а друга матриця b є оборотною для матриці a (отже, вона корельована).

Стиснення зображень. Вступ



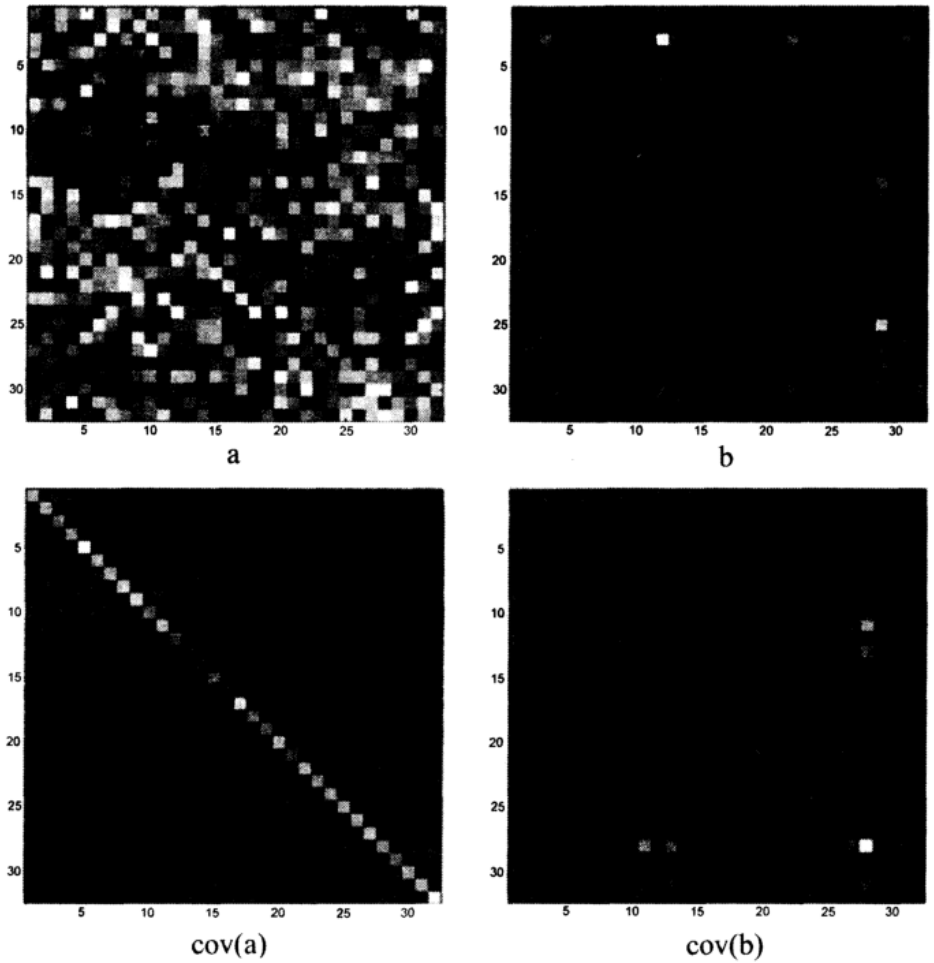
Приклад. Скористаємося спеціальною програмою для ілюстрації коварійної матриці для (1) матриці з корельованими елементами та (2) матриці з декорельованими елементами. На рис. зображено дві 32×32 матриці. Перша з них є випадковою (тобто, декорельованою), а друга матриця b є оборотною для матриці a (отже, вона корельована).

Стиснення зображень. Вступ



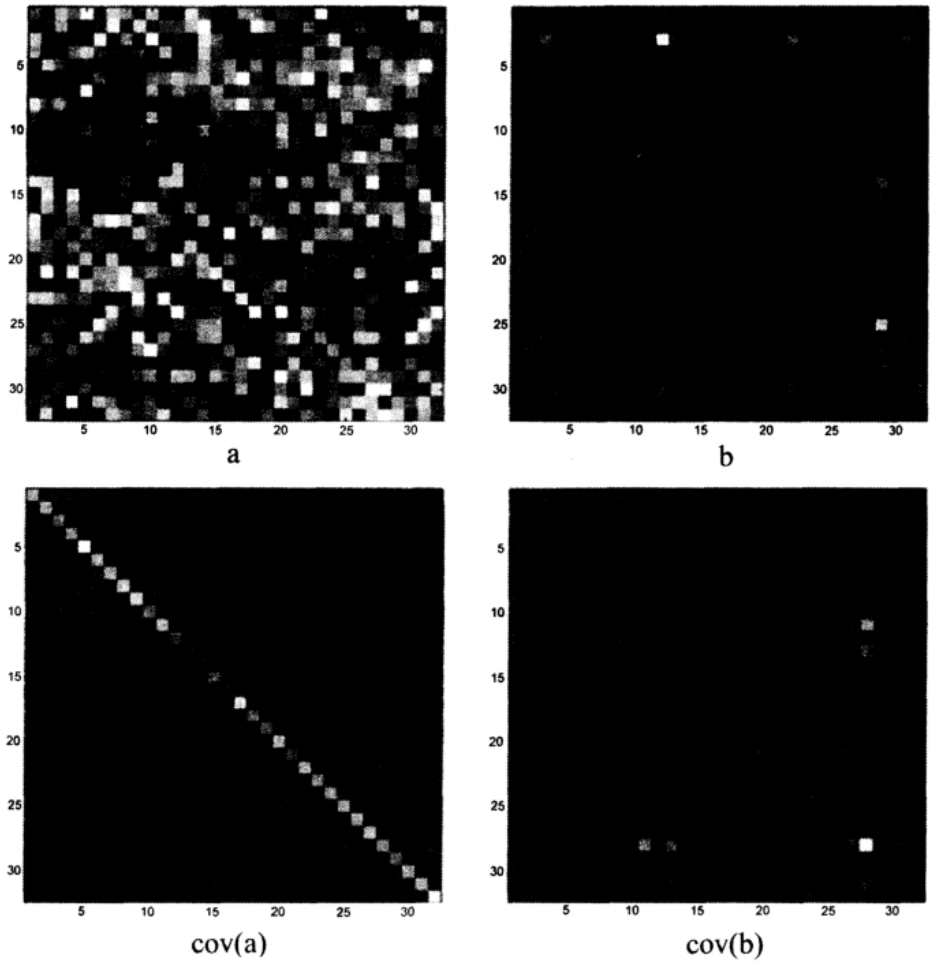
Приклад. Скористаємося спеціальною програмою для ілюстрації коварійної матриці для (1) матриці з корельованими елементами та (2) матриці з декорельованими елементами. На рис. зображено дві 32×32 матриці. Перша з них є випадковою (тобто, декорельованою), а друга матриця b є оборотною для матриці a (отже, вона корельована).

Стиснення зображень. Вступ



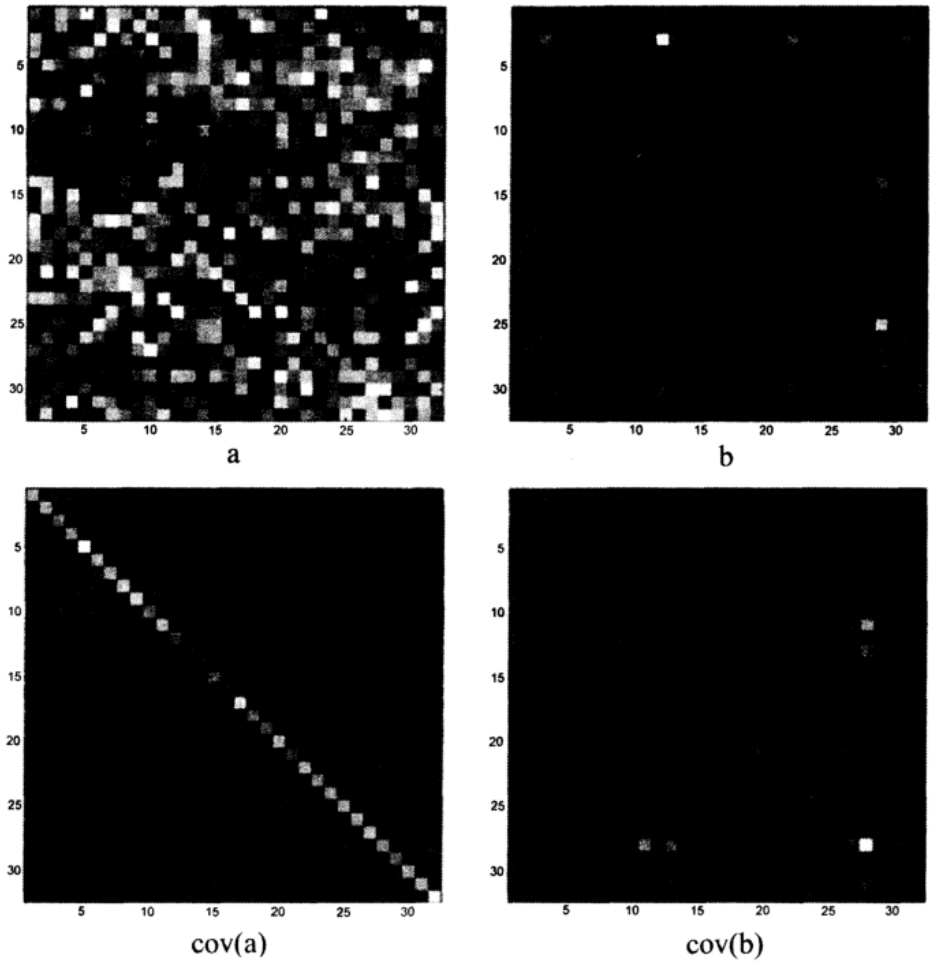
Приклад. Скористаємося спеціальною програмою для ілюстрації коварійної матриці для (1) матриці з корельованими елементами та (2) матриці з декорельованими елементами. На рис. зображено дві 32×32 матриці. Перша з них є випадковою (тобто, декорельованою), а друга матриця b є оборотною для матриці a (отже, вона корельована).

Стиснення зображень. Вступ



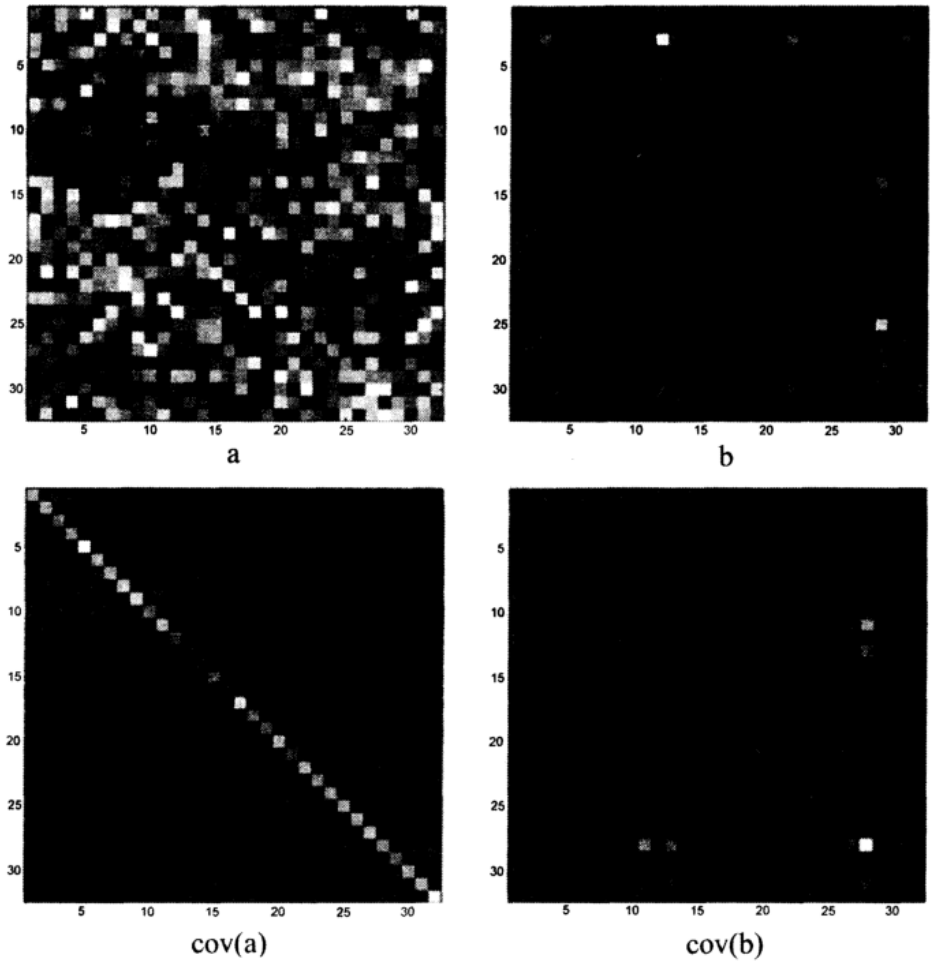
Їхні коваріаційні матриці також зображені. Очевидно, що матриця $\mathrm{cov}(\mathrm{a})$ близька до діагональної (недіагональні елементи дорівнюють нулю або близькі до нуля), а матриця $\mathrm{cov}(\mathrm{b})$ далека від діагональної. Далі наведено програму для системи Matlab, яка малює ці рисунки.

Стиснення зображень. Вступ



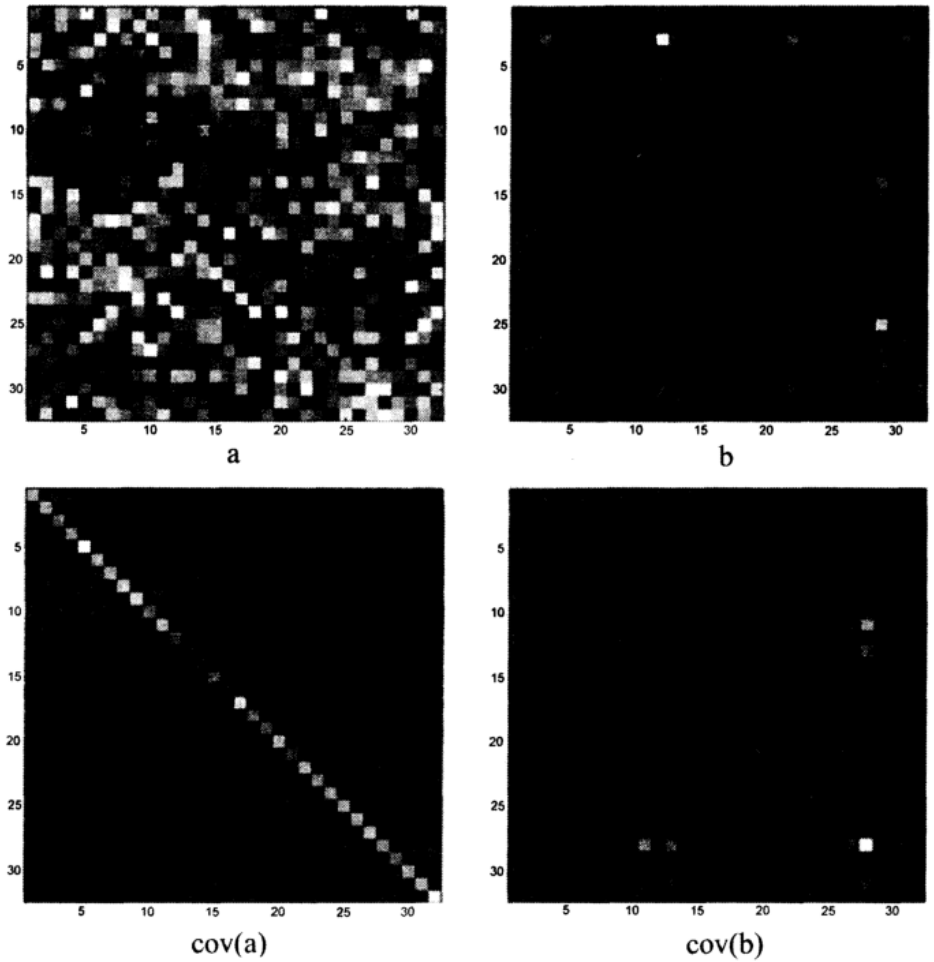
Їхні коваріаційні матриці також зображені. Очевидно, що матриця $\mathrm{cov}(\mathrm{a})$ близька до діагональної (недіагональні елементи дорівнюють нулю або близькі до нуля), а матриця $\mathrm{cov}(\mathrm{b})$ далека від діагональної. Далі наведено програму для системи Matlab, яка малює ці рисунки.

Стиснення зображень. Вступ



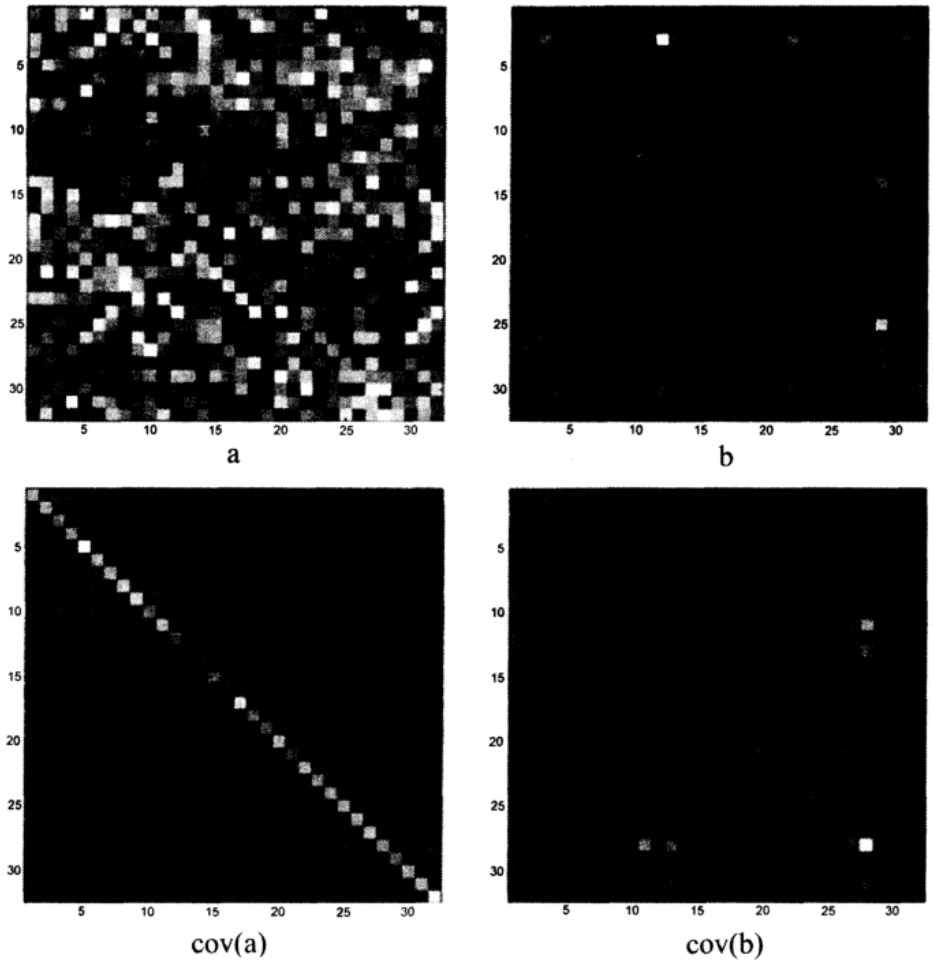
Їхні коваріаційні матриці також зображені. Очевидно, що матриця $\mathrm{cov(a)}$ близька до діагональної (недіагональні елементи дорівнюють нулю або близькі до нуля), а матриця $\mathrm{cov(b)}$ далека від діагональної. Далі наведено програму для системи Matlab, яка малює ці рисунки.

Стиснення зображень. Вступ



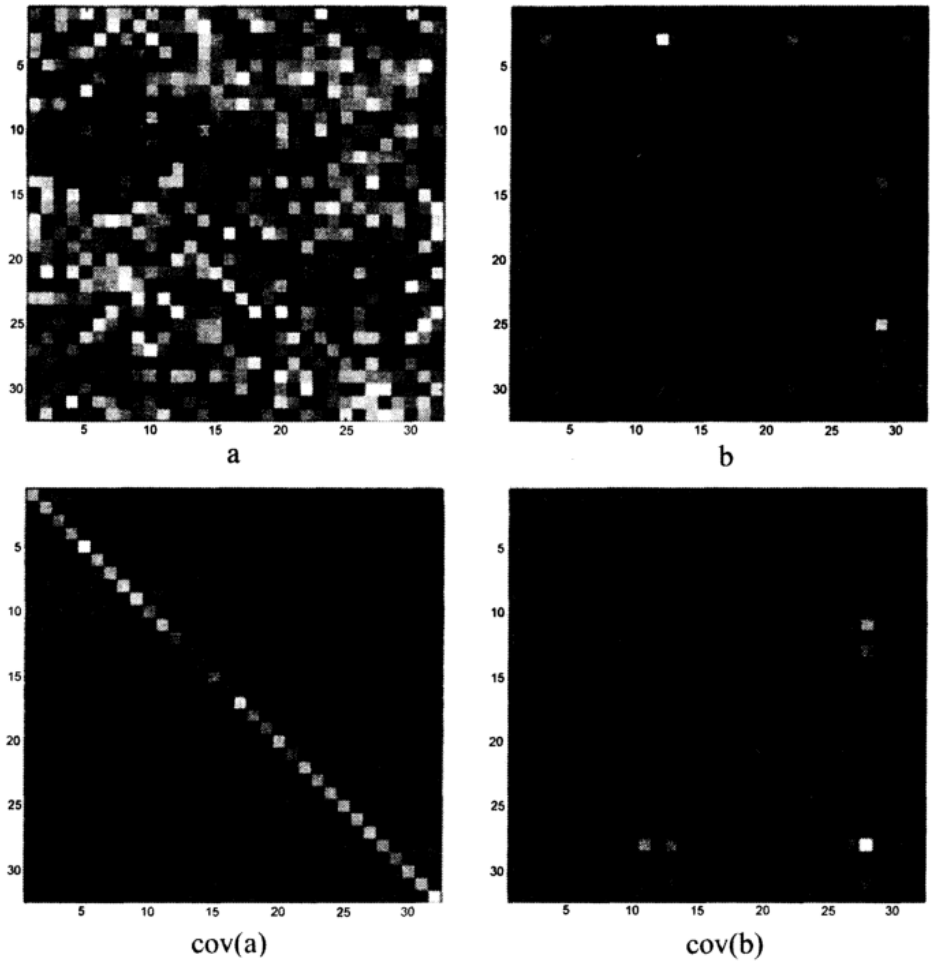
Їхні коваріаційні матриці також зображені. Очевидно, що матриця $\mathrm{cov}(\mathrm{a})$ близька до діагональної (недіагональні елементи дорівнюють нулю або близькі до нуля), а матриця $\mathrm{cov}(\mathrm{b})$ далека від діагональної. Далі наведено програму для системи Matlab, яка малює ці рисунки.

Стиснення зображень. Вступ



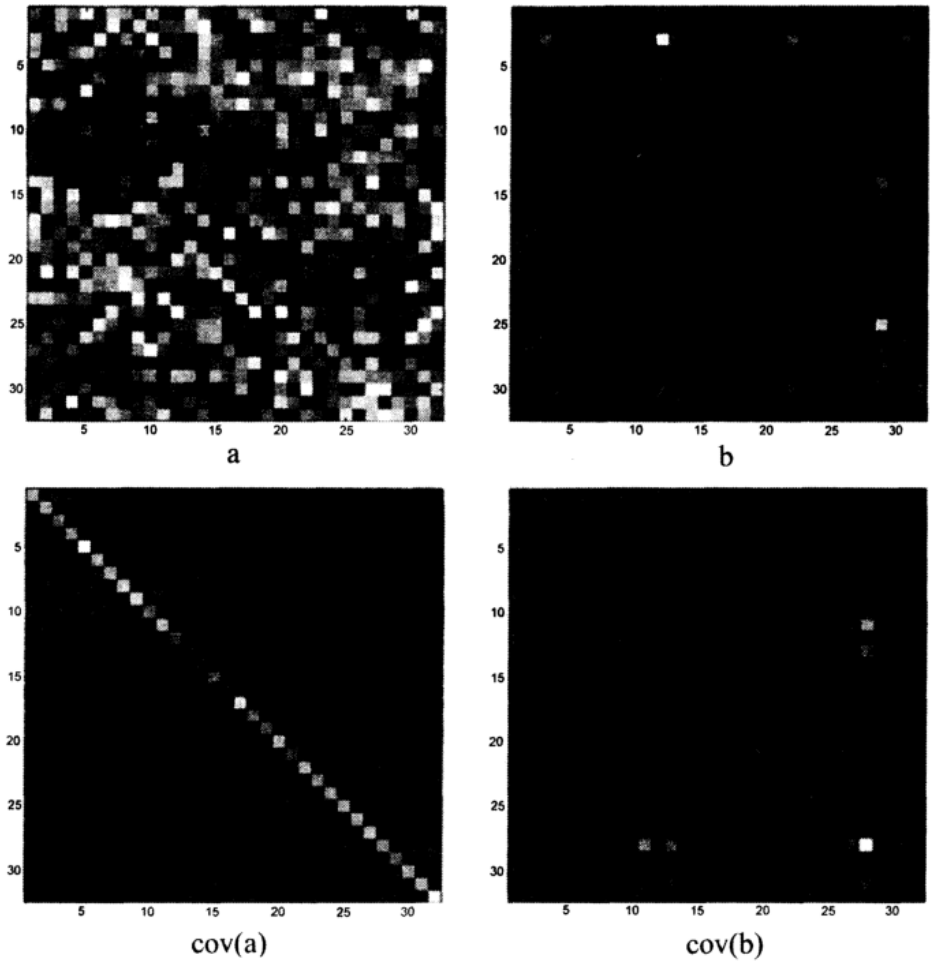
Їхні коваріаційні матриці також зображені. Очевидно, що матриця $\mathrm{cov}(\mathrm{a})$ близька до діагональної (недіагональні елементи дорівнюють нулю або близькі до нуля), а матриця $\mathrm{cov}(\mathrm{b})$ далека від діагональної. Далі наведено програму для системи Matlab, яка малює ці рисунки.

Стиснення зображень. Вступ



Їхні коваріаційні матриці також зображені. Очевидно, що матриця $\mathrm{cov(a)}$ близька до діагональної (недіагональні елементи дорівнюють нулю або близькі до нуля), а матриця $\mathrm{cov(b)}$ далека від діагональної. Далі наведено програму для системи Matlab, яка малює ці рисунки.

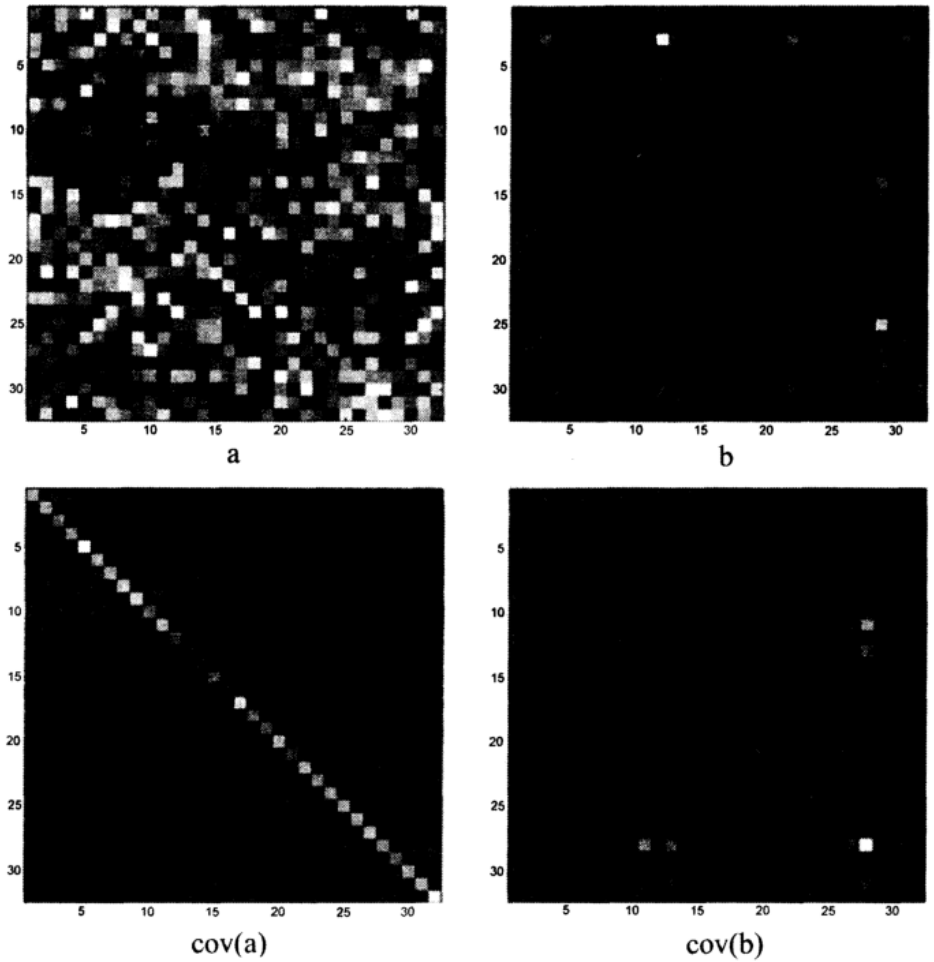
Стиснення зображень. Вступ




```
a=rand(32); b=inv(a);
figure(1); imagesc(a); colormap(gray); axis square
figure(2); imagesc(b); colormap(gray); axis square
figure(3); imagesc(cov(a)); colormap(gray); axis square
figure(4); imagesc(cov(b)); colormap(gray); axis square
```

Програма на Matlab для рис.

Стиснення зображень. Вступ




```
a=rand(32); b=inv(a);
figure(1); imagesc(a); colormap(gray); axis square
figure(2); imagesc(b); colormap(gray); axis square
figure(3); imagesc(cov(a)); colormap(gray); axis square
figure(4); imagesc(cov(b)); colormap(gray); axis square
```

Програма на Matlab для рис.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Вступ

Якщо поняття корелюваних величин стало більш зрозумілим, то можна легко відповісти на запитання: "Як перевірити чи стали пікселі зображення після деякого перетворення декорельованими чи ні?" Відповідь буде такою: "Якщо матриця містить декорельовані значення, то коваріація будь-якого її рядка з будь-яким стовпцем дорівнює нулю". В результаті ковариаційна матриця M буде діагональною. Статистичне поняття дисперсії, коваріації та кореляції обговорюються у будь-якому підручнику зі статистики.

Принцип стиснення зображень має й інший бік медалі. Нам добре відомо з досвіду, що *яскравість* близьких пікселів також корелює. Два суміжні пікселі можуть мати різні кольори. Один може бути близьким до червоного, а другий — до зеленого, однак, якщо перший був яскравим, то його сусід, як правило, теж буде яскравим. Цю властивість можна використовувати для перекладу уявлення пікселів RGB (red, green, blue) у вигляді трьох інших компонентів: яскравості та двох хроматичних компонентів, що визначають колір. Одним таким форматом (або простором кольорів) служить YCbCr, де Y (компонента "світимості") відповідає за яскравість пікселя, а Cb і Cr визначають його колір. Цей формат обговорюватиметься в наступних лекціях, але його перевагу легко зрозуміти. Наше око чутливе до маленьким змінам яскравості, але не кольору. Тому втрата інформації у компонентах Cb і Cr стискає образ, вносячи спотворення, які людське око не помічає. А спотворення інформації про компонент Y, навпаки, є більш помітним.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

Цифрове зображення представляє собою прямокутну таблицю точок, або елементів зображення, розташованих у рядках і стовпцях. Вираз $m \times n$ називається *роздільною здатністю зображення* (хоча іноді цей термін використовується для позначення кількості пікселів, що припадають на одиницю довжини зображення). Точки зображення називаються *пікселями* (за винятком випадків, коли зображення передається факсом або відео; у цих випадках точка називається *пелом*). Для цілей стиснення графічних образів зручно виділити такі типи зображень.

1. *Дворівневе* (або *монохроматичне*) *зображення*. У цьому випадку всі пікселі можуть мати лише два значення, які зазвичай називають чорним (двійкова одиниця, або основний колір) та білим (двійковий нуль або колір фону). Кожен піксель такого зображення зображається одним бітом, а тому це найпростіший тип зображення.

2. *Напівтонове зображення*. Кожен піксель такого зображення може мати 2^n значень від 0 до $2^n - 1$, що позначають одну з 2^n градацій сірого (або іншого) кольору. Число n зазвичай порівняно з розміром байта, тобто воно дорівнює 4, 8, 12, 16, 24 або кратне 4 або 8. Множина найбільш значущих бітів всіх пікселів утворює найбільш значиму бітову площину або шар зображення. Отже, напівтонове зображення зі шкалою з 2^n рівнів складено з n бітових шарів.

Стиснення зображень. Типи зображень

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

Стиснення зображень. Типи зображень

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

3. *Кольорове зображення*. Існує декілька методів визначення кольору, але у кожному з них беруть участь три параметри. Отже, кольоровий піксель складається із трьох частин. Зазвичай кольоровий піксель складається з трьох байтів. Типовими моделями кольорів є RGB, HLS і CMYK. Детальний опис моделей кольорів виходить за межі цього курсу, але деякий базовий розгляд питання буде наведено в подальших лекціях.

4. *Зображення з неперервним тоном*. Цей тип зображень може мати багато схожих кольорів (або напівтонів). Якщо сусідні пікселі відрізняються лише на одиницю, то оку практично неможливо розрізнити їхні кольори. В результаті такі зображення можуть містити області, в яких колір здається оку, що він неперервно змінюється. У цьому випадку піксель представляється або більшим кількістю (у напівтоновому випадку) або трьома компонентами (як кольорового образу). Зображення з неперервним тоном є природними (на відміну рукотворних, штучних); зазвичай вони виходять під час зйомки на цифрову фотокамеру або під час сканування фотографій або малюнків.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

5. *Дискретно-тонове зображення* (воно також називається *синтетичним*). Зазвичай це зображення отримується штучним шляхом. У ньому може бути лише декілька кольорів або багато кольорів, але в ньому немає шумів та плям природного зображення. Прикладами таких зображень можуть бути фотографії штучних об'єктів, машин або механізмів, сторінки тексту, карти, малюнки або зображення на дисплеї комп'ютера. (Не кожне штучне зображення буде обов'язково дискретно-тоновим. Згенероване комп'ютером зображення, яке має виглядати природним, матиме неперервні тони, незважаючи на своє штучне походження.) Штучні об'єкти, тексти, намальовані лінії мають форму, межі, що добре визначаються. Вони сильно контрастують на тлі решти зображення (фону). Прилеглі пікселі дискретно-тонового образу часто бувають одинарними або змінюють свої значення. Такі зображення погано стискаються методами з втратою даних, оскільки спотворення всього декількох пікселів літери робить її нерозбірливою, перетворює звичнайне зображення в абсолютно нерозрізнене. Методи стиснення зображень з неперервними тонами погано поводяться з чіткими краями дискретно-тонових образів, для яких слід розробляти особливі методи компресії. Зазначимо, що дискретно-тонові зображення, як правило, несуть у собі велику надмірність. Багато їхніх фрагментів повторюються багато разів у різних місцях зображення.

Стиснення зображень. Типи зображень

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

Стиснення зображень. Типи зображень

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

Стиснення зображень. Типи зображень

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

Стиснення зображень. Типи зображень

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

Стиснення зображень. Типи зображень

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

Стиснення зображень. Типи зображень

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

6. *Зображення, подібні до мультфільмів*. Це кольорові зображення, в яких є великі області одного кольору. При цьому області, що стикаються, можуть дуже відрізнятися за своїм кольором. Цю властивість можна використовувати для досягнення кращої компресії.

Інтуїтивно стає зрозуміло, що кожному типу зображень притаманна певна надмірність, але вони надмірні по-різному. Тому важко створити один метод, який однаково добре стискає будь-які типи зображень. Існують окремі методи для стиснення дворівневих образів, неперервно-тонових та дискретно-тонових зображень. Існують також методи, які намагаються розділити зображення на неперервно-тонову та дискретно-тонову частини та стискати їх окремо.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень

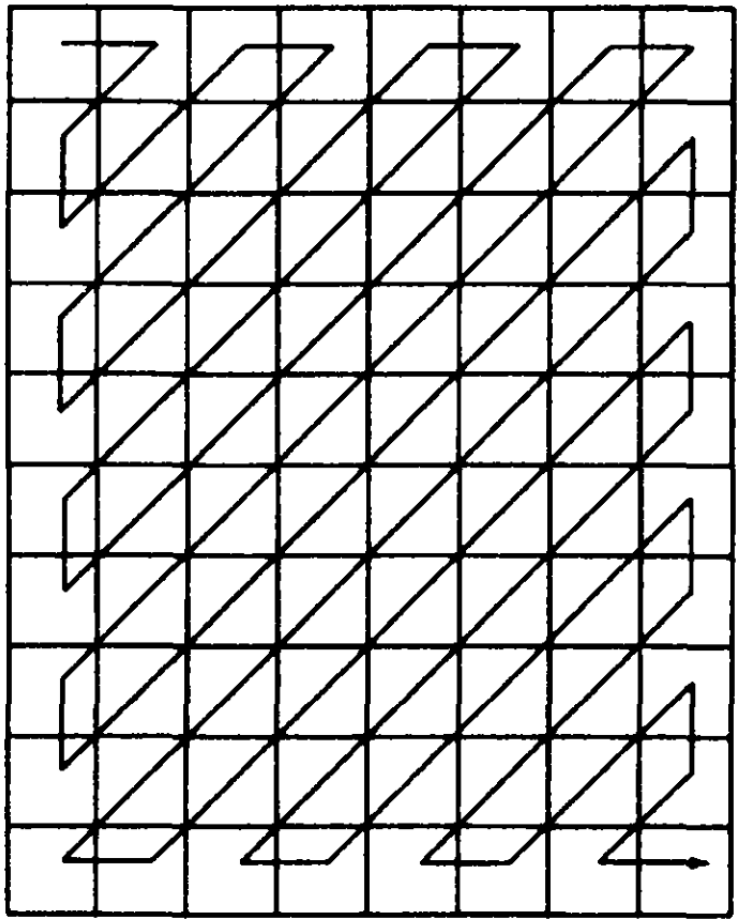
Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Методи стиснення зображень зазвичай розробляються конкретного типу зображень. Тут наведено різні підходи до компресії графічних образів. При цьому обговорюватимуться лише загальні принципи. Специфічні методи описані далі у наступних лекціях.

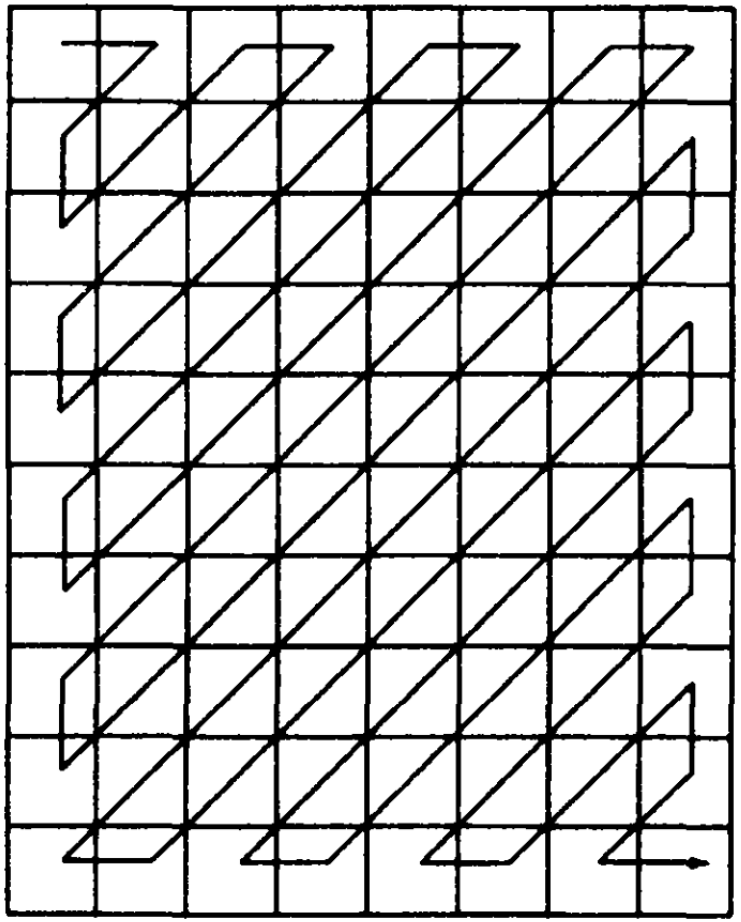
Підхід 1. Для стискання дворівневих зображень. Кожен піксель такого образу є одним бітом. Застосування принципу стиснення образів до компресії дворівневих зображень означає, що безпосередні сусіди пікселя P прагнуть збігатися з P. Тому є зміст використовувати методи кодування довжин серій (RLE) для стиснення таких зображень. Метод стиснення сканує образ рядок за рядком та обчислює довжини послідовних чорних та білих пікселів. Довжини кодуються кодами змінної довжини та записуються у стислий файл. Прикладом такого стиснення є факсимільна компресія.

Стиснення зображень. Підходи до стиснення зображень



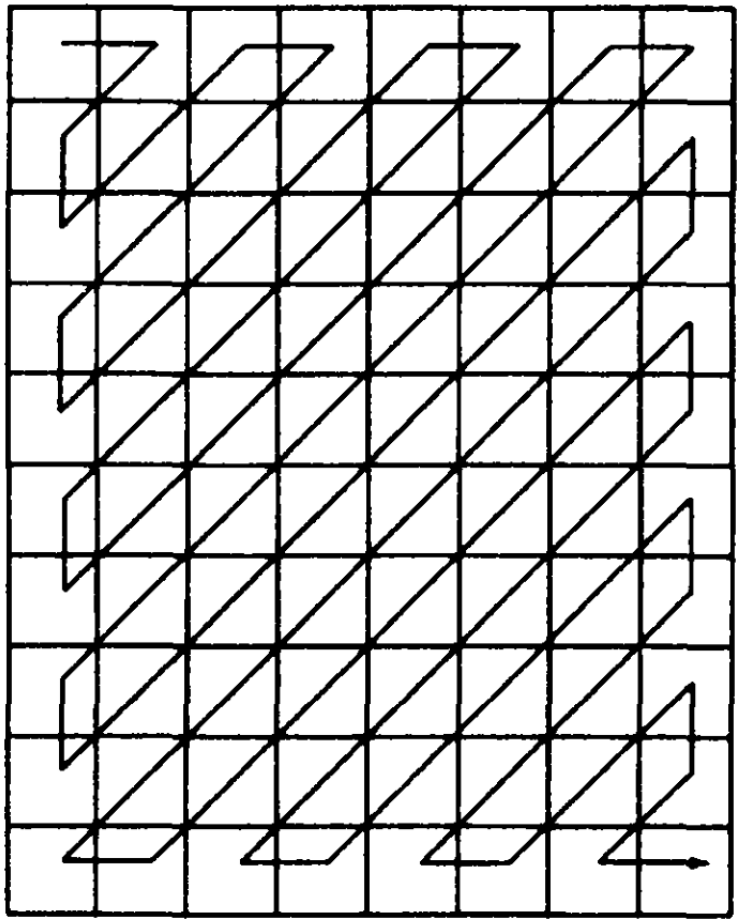
Ще раз підкреслимо, що це лише загальний принцип стиснення, а конкретні методи можуть відрізнятися один від іншого. Наприклад, метод може сканувати образ по рядках, а може зигзагоподібно (див. рис.), або сканувати зображення область за областю за допомогою кривих, що її заповнюють.

Стиснення зображень. Підходи до стиснення зображень



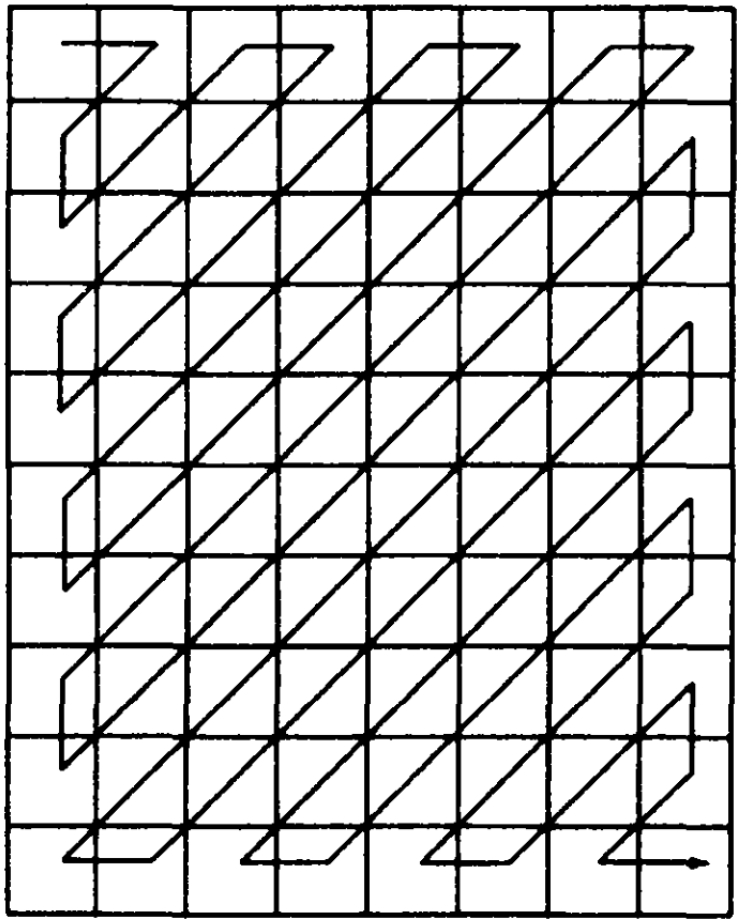
Ще раз підкреслимо, що це лише загальний принцип стиснення, а конкретні методи можуть відрізнятися один від іншого. Наприклад, метод може сканувати образ по рядках, а може зигзагоподібно (див. рис.), або сканувати зображення область за областю за допомогою кривих, що її заповнюють.

Стиснення зображень. Підходи до стиснення зображень



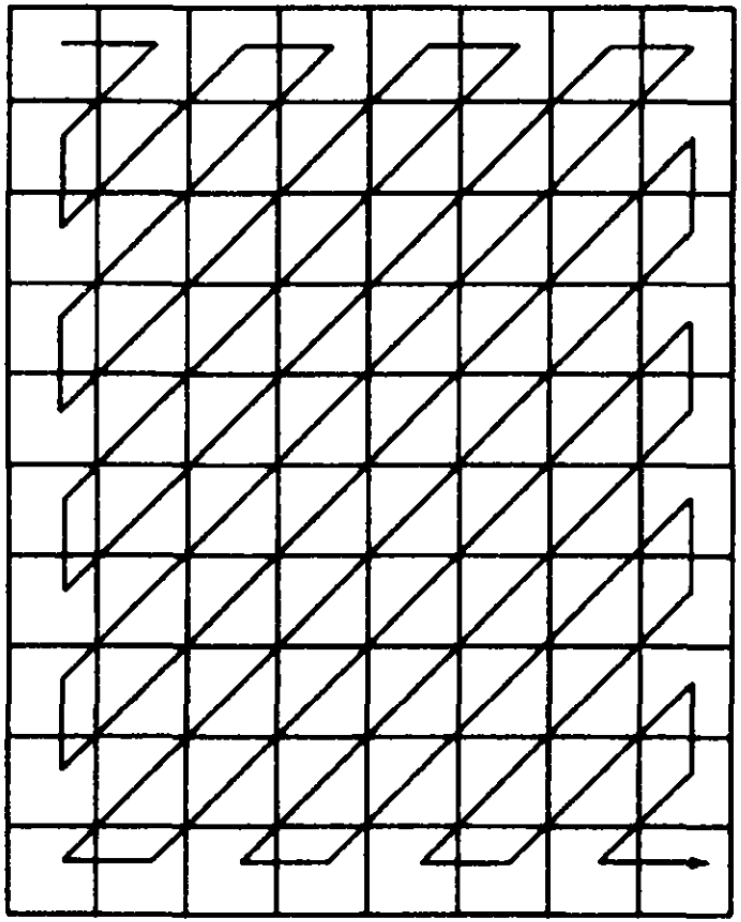
Ще раз підкреслимо, що це лише загальний принцип стиснення, а конкретні методи можуть відрізнятися один від іншого. Наприклад, метод може сканувати образ по рядках, а може зигзагоподібно (див. рис.), або сканувати зображення область за областю за допомогою кривих, що її заповнюють.

Стиснення зображень. Підходи до стиснення зображень



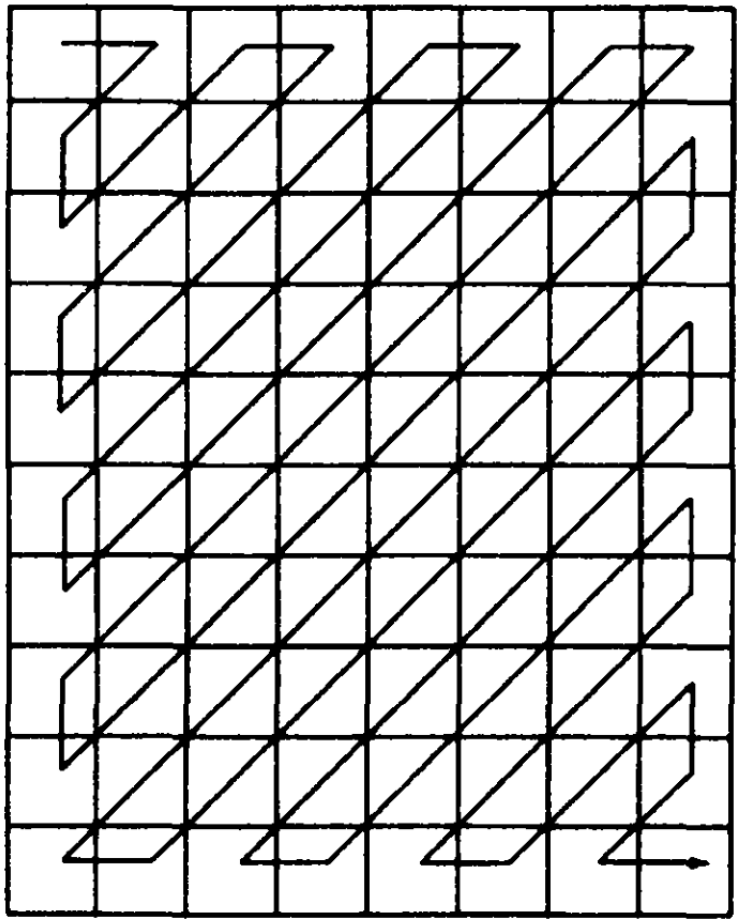
Ще раз підкреслимо, що це лише загальний принцип стиснення, а конкретні методи можуть відрізнятися один від іншого. Наприклад, метод може сканувати образ по рядках, а може зигзагоподібно (див. рис.), або сканувати зображення область за областю за допомогою кривих, що її заповнюють.

Стиснення зображень. Підходи до стиснення зображень



Ще раз підкреслимо, що це лише загальний принцип стиснення, а конкретні методи можуть відрізнятися один від іншого. Наприклад, метод може сканувати образ по рядках, а може зигзагоподібно (див. рис.), або сканувати зображення область за областю за допомогою кривих, що її заповнюють.

Стиснення зображень. Підходи до стиснення зображень



Ще раз підкреслимо, що це лише загальний принцип стиснення, а конкретні методи можуть відрізнятися один від іншого. Наприклад, метод може сканувати образ по рядках, а може зигзагоподібно (див. рис.), або сканувати зображення область за областю за допомогою кривих, що її заповнюють.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 2. Також є метод для дворівневих зображень. Знову використовується принцип збігу найближчих пікселів у розширеній формі: якщо поточний піксель має колір c (де c — білий або чорний), то пікселі того ж кольору, що спостерігалися в минулому (а також ті, що з'являться в майбутньому) матимуть таких найближчих сусідів.

Цей підхід розглядає n послідовних сусідів поточного пікселя і представляє їх у вигляді n-бітного числа. Це число називається *контекстом* пікселя. У принципі, може бути 2^n різних контекстів, але через надмірність образу ми очікуємо, що контексти розподілені нерівномірно. Деякі контексти зустрічаються частіше, а деякі — рідше.

Кодер обчислює скільки разів зустрічався кожен контекст для пікселів кольору c і надає контекстам відповідні ймовірності. Якщо поточний піксель має колір c і його контекст має ймовірність p, то кодер може використовувати адаптивне арифметичне кодування для кодування пікселя з цією ймовірністю. Такий підхід використано у методі JBIG.

Перейдемо тепер до напівтонових зображень. Піксель такого зображення представлений n бітами і може мати одне з 2^n значень. Застосовуючи принцип стиснення зображень, робимо висновок, що сусіди пікселя P прагнуть бути схожими на P, але не обов'язково збігаються з ним. Тому метод RLE тут годиться. Натомість розглянемо дві інші ідеї.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 3. Розшарувати напівтонову картинку на n дворівневих зображень і кожну стиснути за допомогою RLE та префіксних кодів. Тут здається, що можна застосувати основний принцип стиснення зображень, але стосовно кожного окремого шару напівтонового зображення. Однак це не так, і наступний приклад прояснює ситуацію. Уявімо напівтонове зображення з $n = 4$ (тобто маємо справу з 4-бітовими пікселями, або з 16 градаціями сірого кольору). Його можна розшарувати на 4 дворівневі зображення. Якщо два суміжні пікселі вихідного образу мали величини

$$0000 \quad \text{і} \quad 0001,$$

то вони схожі за кольором. Однак два пікселі зі значеннями

$$0111 \quad \text{і} \quad 1000$$

також близькі на напівтоновому зображенні (це відповідно числа 7 і 8), але вони різняться у всіх 4 шарах.

Ця проблема виникає, оскільки у двійковому зображенні сусідні числа можуть відрізнятися у багатьох розрядах. Двійкові коди для 0 і 1 відрізняються в одному розряді, коди для 1 і 2 розрізняються в двох розрядах, а коди для 7 і 8 різняться вже у всіх чотирьох бітах. Розв'язком цієї проблеми може бути розробка спеціальних послідовностей двійкових кодів, в яких послідовні коди з номерами n і $n + 1$ відрізнялися б рівно на один біт. Прикладом таких кодів є рефлексні коди Грея.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

$$\Delta = P - A$$

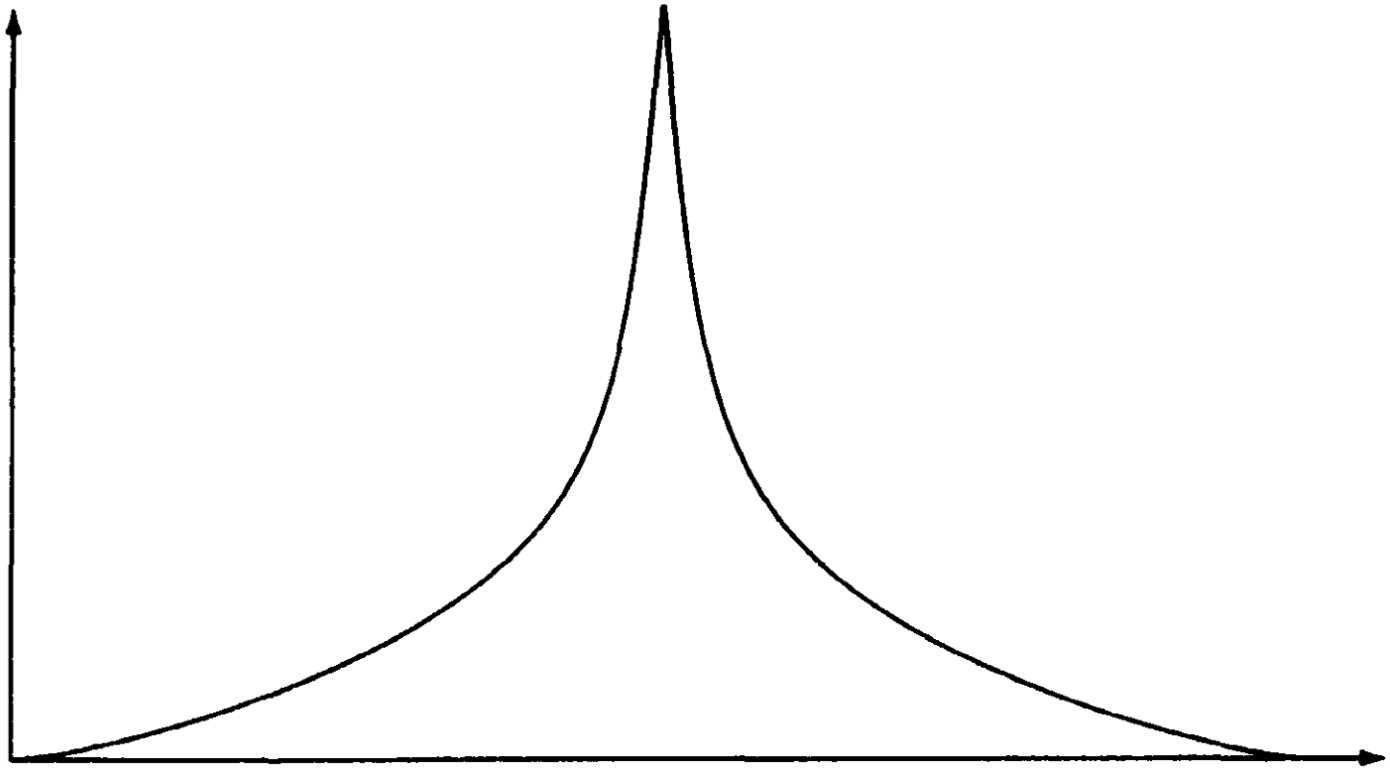
і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

Підхід 4. Використовувати контекст пікселу P, який складається із значень кількох найближчих сусідів. Вибираємо кілька сусідніх пікселів, обчислюємо середнє значення їхніх величин і робимо передбачення, що піксель P дорівнюватиме A. Основний принцип стиснення зображень дозволяє вважати, що в більшості випадків ми маємо рацію, у багатьох випадках будемо майже праві та лише в деяких випадках будемо абсолютно неправі. Можна сказати, що в передбаченій величині пікселя P міститься надмірна інформація про P. Обчислимо різницю

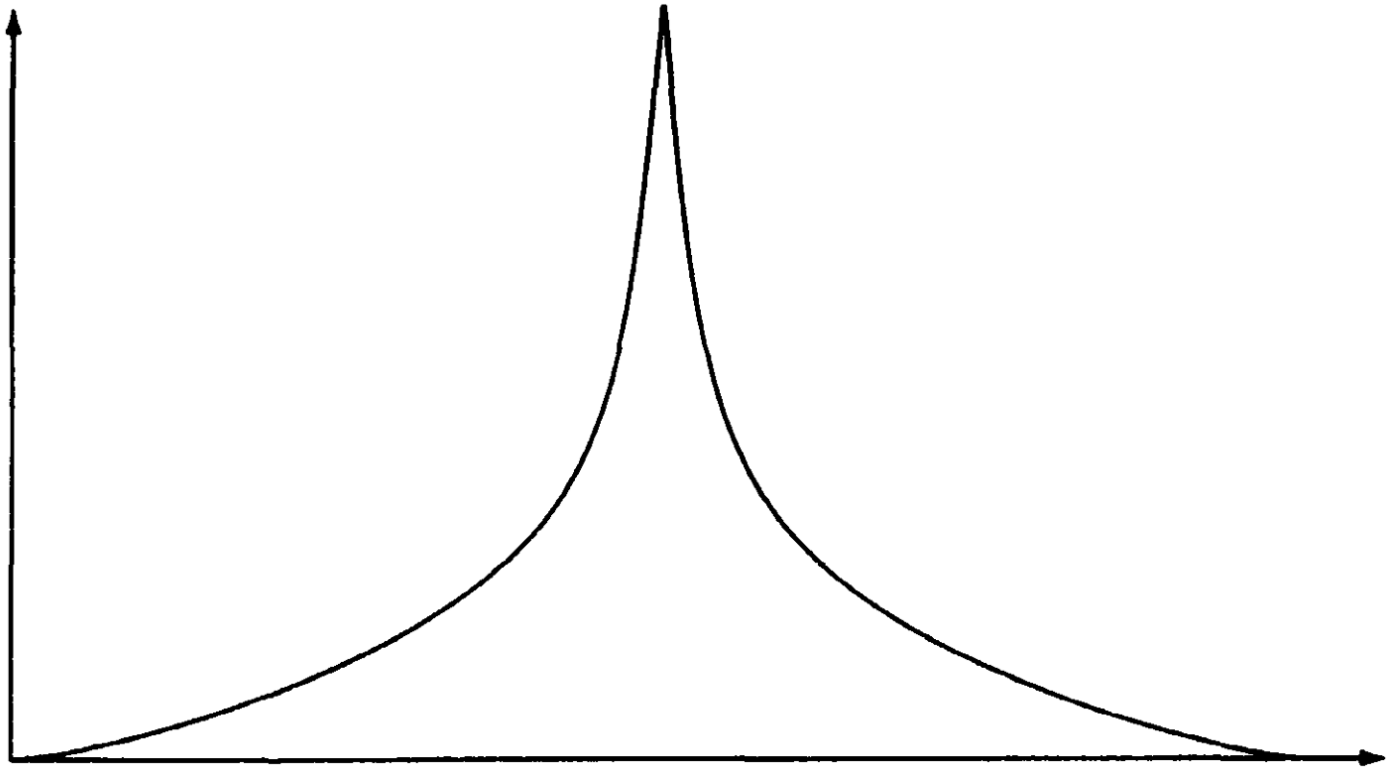
$$\Delta = P - A$$

і присвоємо деякий (префіксний) код змінної довжини величині A так, щоб малим величинам (які очікуються часто) відповідали короткі коди, а великим величинам (рідко очікувані) призначалися довгі коди. Якщо значення P лежить в інтервалі від 0 до $m-1$, то значення A потрапляють в інтервал $[-(m-1), +(m-1)]$, для чого потрібно $2(m-1)+1$ або $2m-1$ кодів.

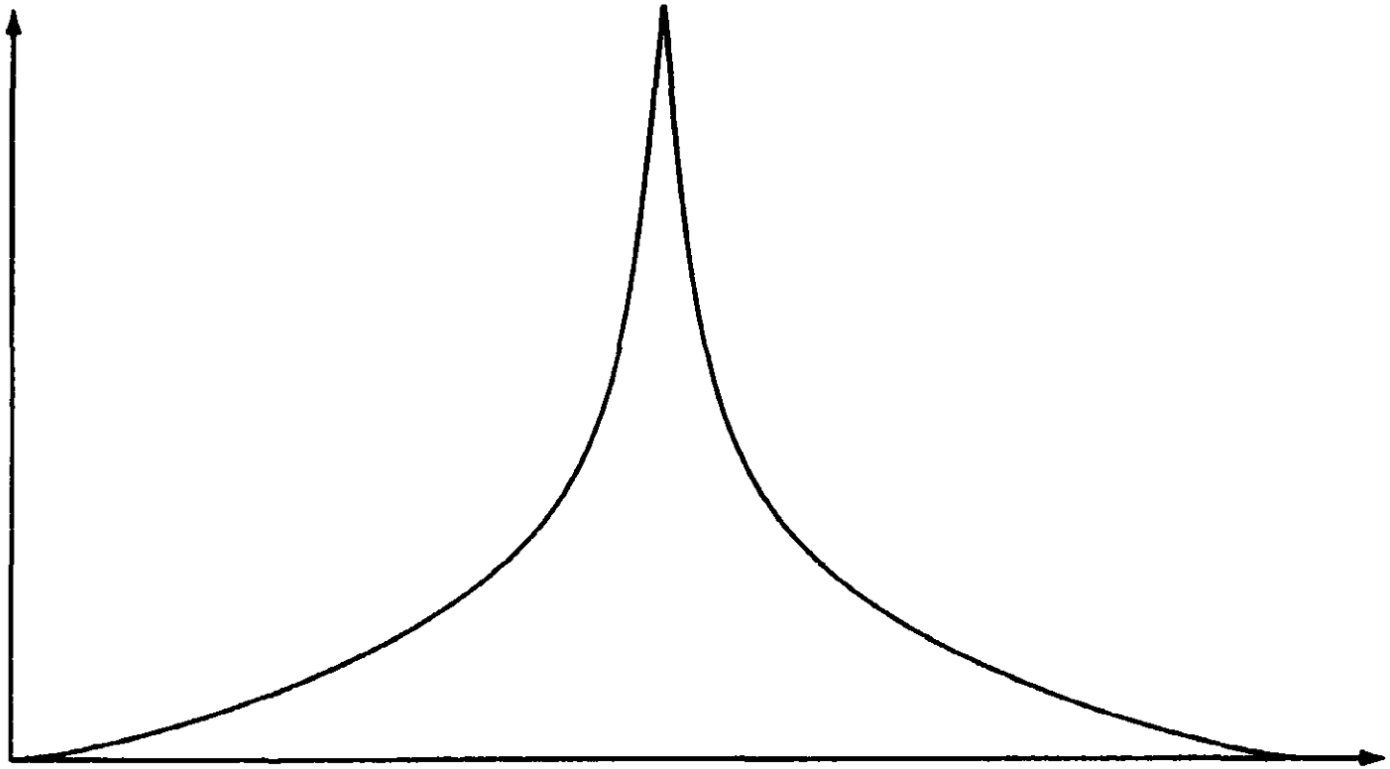
Стиснення зображень. Підходи до стиснення зображень



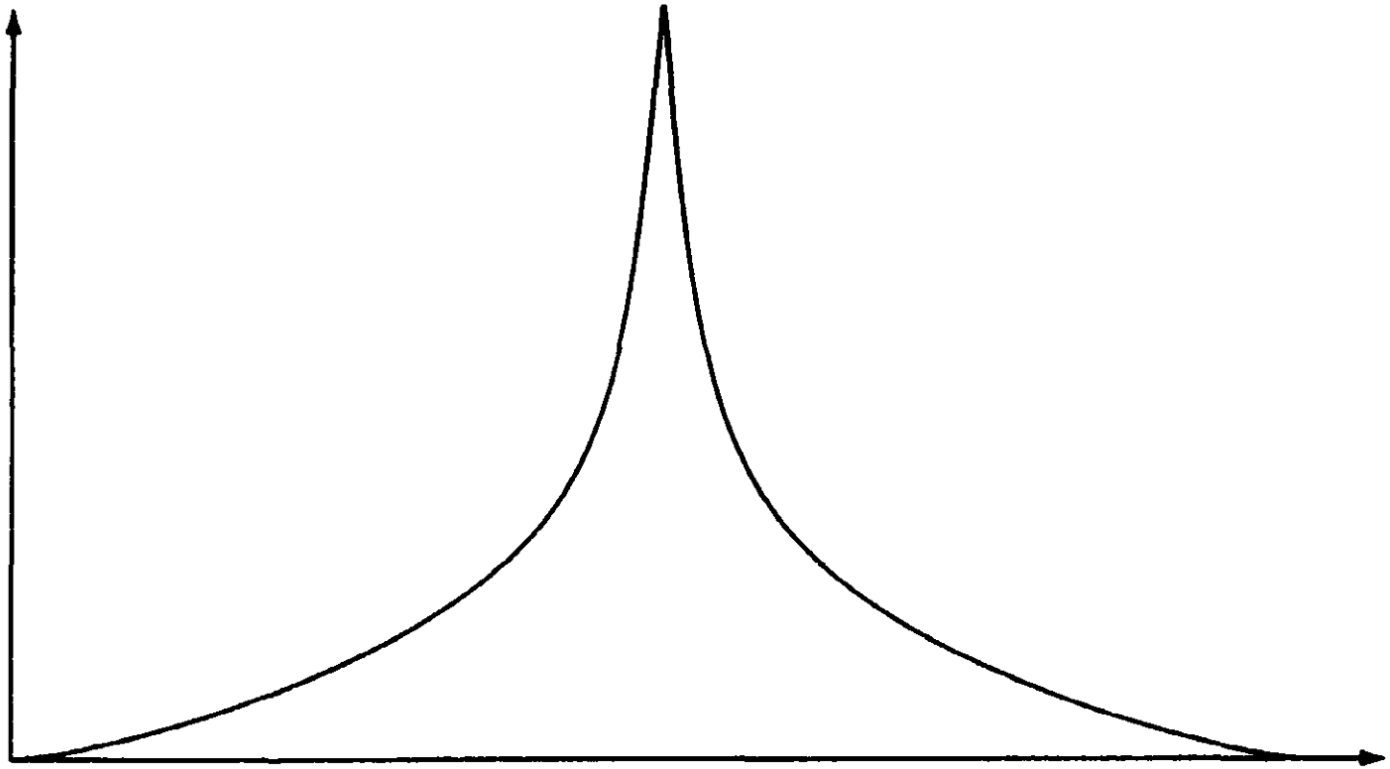
Експериментування з великими числами показує, що значення A прагнуть мати розподіл, близький до розподілу Лапласа (див. рис.), яке часто виникає у статистиці. Тоді метод стиснення може використовувати цей розподіл для присвоєння ймовірностей величинам A і дуже ефективно застосовувати арифметичне кодування для значення A. Цей підхід є основою методу стиснення MLP.



Експериментування з великими числами показує, що значення A прагнуть мати розподіл, близький до розподілу Лапласа (див. рис.), яке часто виникає у статистиці. Тоді метод стиснення може використовувати цей розподіл для присвоєння ймовірностей величинам A і дуже ефективно застосовувати арифметичне кодування для значення A. Цей підхід є основою методу стиснення MLP.



Експериментування з великими числами показує, що значення A прагнуть мати розподіл, близький до розподілу Лапласа (див. рис.), яке часто виникає у статистиці. Тоді метод стиснення може використовувати цей розподіл для присвоєння ймовірностей величинам A і дуже ефективно застосовувати арифметичне кодування для значення A. Цей підхід є основою методу стиснення MLP.



Експериментування з великими числами показує, що значення A прагнуть мати розподіл, близький до розподілу Лапласа (див. рис.), яке часто виникає у статистиці. Тоді метод стиснення може використовувати цей розподіл для присвоєння ймовірностей величинам A і дуже ефективно застосовувати арифметичне кодування для значення A. Цей підхід є основою методу стиснення MLP.

Стиснення зображень. Підходи до стиснення зображень

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага.

Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Контекст пікселя може складатися з одного або двох сусідів. Однак найкращі результати виходять, якщо використовувати декілька пікселів, причому кожному пікселю присвоюється деяка вага для обчислення зваженого середнього: більш далеким пікселям присвоюється менша вага. Інший важливий розгляд стосується декодування. Для декодування зображення необхідно вміти обчислювати контекст до пікселя. Це означає, що контекст має складатися з декодованих пікселів. Якщо зображення сканується рядок за рядком (зверху вниз і зліва направо), то контекст може складатися тільки з пікселів, які розташовані вище та лівіше за цей піксель.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення.

Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням.

Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 5. Зробити перетворення пікселів та кодувати перетворені значення. Поняття перетворення образу, а також найважливіші перетворення, що використовуються при компресії зображень, будуть розібрані у подальших лекціях. Також буде лекція, яка присвячена вейвлетним перетворенням. Нагадаємо, що компресія досягається шляхом видалення чи скорочення надмірності. Надмірність зображення утворюється за рахунок кореляції між пікселами, тому перетворення, які роблять декореляцію, одночасно видаляють надмірність. Квантуючи перетворені значення, можна отримати ефективне стиск із частковою втратою інформації. Ми хочемо перетворювати величини на незалежні, тому що для незалежних величин легше побудувати просту модель для їхнього кодування.

Звернемося тепер до кодування кольорових зображень. Пікселі таких зображень складаються із трьох компонентів, наприклад, червоного, зеленого та синього. Більшість кольорових образів є неперервно-тоновими чи дискретно-тоновими.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Підхід 6. Ідея цього методу полягає в поділі зображення на три напівтонових і в їхньому незалежному стисканні на підставі одного з підходів 2, 3 або 4.

Принцип стиснення безперервно-тонових зображень свідчить, що колір сусідніх пікселів мало змінюється, хоча не обов'язково є постійний. Проте близькість кольорів ще означає близькість величин пікселів. Розглянемо, наприклад, кольоровий 12-бітовий піксель, у якому кожна компонента має 4 біти. Отже, 12 біт

$$1000|0100|0000$$

позначають піксель, колір якого виходить змішуванням 8 одиниць червоного (близько 50%, а тому що всього їх 16), чотирьох одиниць зеленого (близько 25%) і зовсім без синього. Уявімо тепер два пікселі зі значеннями

$$0011|0101|0011 \qquad \text{і} \qquad 0010|0101|0011.$$

Їхні кольори дуже близькі, оскільки вони відрізняються лише в одній червоній компоненті, і там їхня відмінність полягає тільки в одному біті. Однак, якщо розглянути їх як 12-бітові числа, то два числа

$$0011|0101|0011 = 851 \qquad \text{і} \qquad 0010|0101|0011 = 595$$

відрізняються дуже значно.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Важлива особливість цього підходу полягає у використанні уявлення кольору у вигляді яскравості та кольоровості замість звичайного RGB. Ці поняття будуть обговорюватися у наступній лекції. Перевага цього уявлення полягає в тому, що око чутливе до маленьких змін яскравості, але не кольоровості. Це дозволяє допускати значну втрату в компоненті кольоровості, але при цьому декодувати без значного візуального погіршення якості зображення.

Підхід 7. Інші методи потрібні для компресії дискретно-тонових зображень. Нагадаємо, такі зображення складаються з великих одноколірних областей, причому кожна область може з'являтися в декількох місцях образу. Хорошим прикладом є вміст екрана комп'ютера. Такий образ складається з тексту та піктограм (іконок). Кожна літера, кожна піктограма займають деяку область на екрані, і кожна з цих областей може бути у різних місцях екрана. Можливий спосіб стиснення такого зображення полягає в його скануванні та виявленні таких областей, що повторюються. Якщо область B збігається з раніше виділеної областю A, то можна стиснути область B, записавши у файл покажчик на область A. Прикладом такого методу служить алгоритм блокової декомпозиції.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Підхід 8. Поділ зображення на частини (перекриваються чи ні) та стиск їх одна за одною. Припустимо, що наступна частина, що вимагає стиснення, має номер 15. Постараємося порівняти її з уже обробленими частинами 1–14. Припустимо, що її можна виразити, наприклад, за допомогою комбінації частин 5 (розтяг) і 11 (поворот), тоді достатньо зберегти лише кілька чисел, які описують необхідну комбінацію, а частину 15 можна відкинути. Якщо частина 15 не вдається висловити у вигляді комбінації колишніх частин, то її доведеться зберегти в "сирому вигляді".

Інший підхід є основою різних *фрактальних* методів стиснення зображень. Тут застосовується основний принцип стиснення зображень, але замість пікселів виступають частини зображення. Цей принцип говорить про те, що "цікаві" зображення (тобто ті, які стискатимуться на практиці) мають деякий ступінь *самоподібності*. Частина зображення збігається або близька до всього зображення.

Методи стиснення зображень зовсім не обмежуються перерахованими базовими підходами. Багатьма авторами обговорюються методи, які використовують контекстні дерева, марківські ланцюги, вейвлети та багато іншого. Додатково слід згадати метод *прогресивного стиску образів*, який додає ще один вимір стиску зображень.

Дякую за увагу!