# Розділ 1: Задача розмітки на ланцюгах
# Домашнє завдання

## Перш ніж почати

### Вимоги

- якомога більше дій повинно бути описано словами: це дозволяє перевірити розуміння матеріалу та перевіряти роботи на унікальність;
- для кожної змінної, константи і функції повинно бути вказано множину (або область значень і визначення), якій вона належить: це дозволяє запобігати багатьох помилок та покращує зрозумілість роботи;
- письмові роботи виконуються індивідуально.

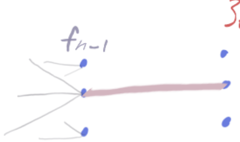
### Рекомендації

- ретельно перевіряйте свою роботу перед тим, як відправляти її викладачеві;
- якщо завдання виконується на аркуші паперу і фотографується, намагайтеся задіяти достатньо освітлення і фотографувати якомога рівніше, щоб усі частини сторінки з розв'язком було добре видно.

### Ілюстрації

Рекомендації до виконання письмових завдань на прикладах і картинках

Звідки такий висновок?




$\Rightarrow d_{n-1} \in \underset{d}{\arg\min}\, f_{n-1}(d) = F_{n-1}(d)$ ← колізія імені $d$

$d$ ← в якій множині шукати $d$?

Малюнок не є доведенням

---

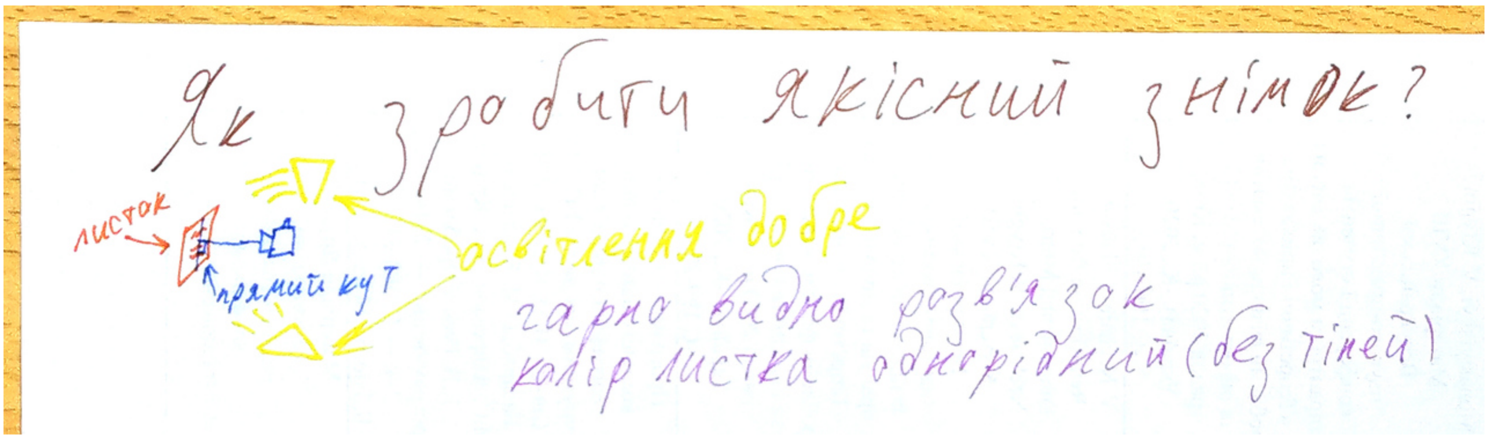
$$\min_{d \in D} \left[ f_{n-1}(d) + \lambda(d_n) \right] =$$

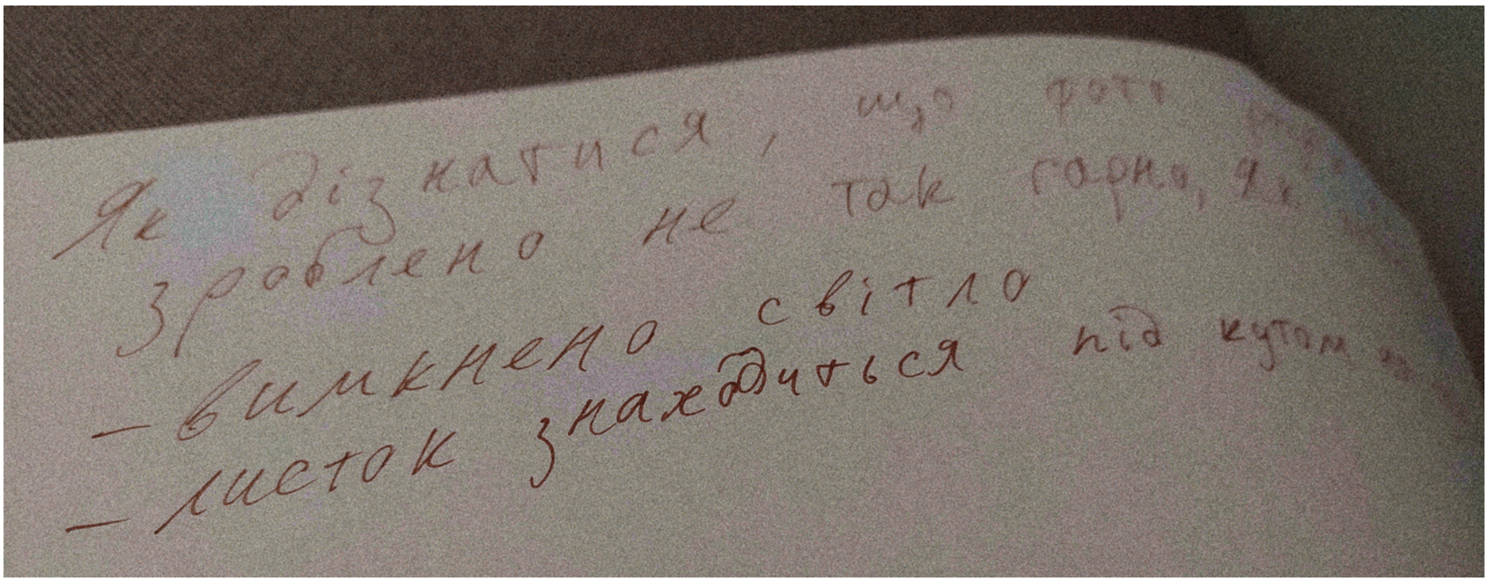
Оскільки $\lambda(d_n)$ не залежить від $d$, його можна винести як константу за межі мінімізації

детальні коментарі

$$= \left[ \min_{d \in D} f_{n-1}(d) \right] + \lambda(d_n), \ \forall d_n \in D, \forall n = N..1.$$

вказано множини    визначено всі змінні




# 1 Основне (2 бали)

## 1.1 Задача розмітки на довільному напівкільці

Для довільного напівкільця $\langle R, \oplus, \otimes \rangle$, відомих скінченних $K$ і $n \in \mathbb{N}_+$ і функції $g : \{1, \ldots, n\} \times K^2 \to R$ вивести ефективний алгоритм пошуку значення

$$G = \bigoplus_{k \in K^{n+1}} \bigotimes_{i=1}^{n} g_i \left( k_{i-1}, k_i \right)$$

та розрахувати його складність. Навести приклади можливих задач із визначенням напівкільця та множини $K$.

## 1.2 Мінімізація часткового ризику

Показати, що для таких стратегій $q^*$, що

$$q^* \in \operatorname*{argmin}_{q : X \to D} \sum_{\substack{k \in K \\ x \in X}} w \left( q \left( x \right), k \right) \cdot p \left( x, k \right),$$

виконується

$$\forall x \in X : q^* \left( x \right) \in \operatorname*{argmin}_{d \in D} \sum_{k \in K} w \left( d, k \right) \cdot p \left( x, k \right).$$

Навіщо цей факт потрібен на практиці?

# 2 Додаткове (3 бали)

## 2.1 Задача розмітки на довільному напівкільці та ациклічній структурі

Для довільного напівкільця $\langle R, \oplus, \otimes \rangle$, відомого орієнтованого дерева $\langle T, \tau \rangle$, відомої скінченної множини $K$ і функції $g : \tau \times K^2 \to R$ вивести ефективний алгоритм пошуку значення

$$G = \bigoplus_{k \in K^T} \bigotimes_{tt' \in \tau} g_{tt'} \left( k_t, k_{t'} \right)$$

та розрахувати його складність.

# 3 Комп'ютерне (4 бали)

## Перш ніж почати

- спочатку розв'язується теорія, а вже потім на її основі пишеться код: це дозволяє не починати роботу з реалізації алгоритму, що є невірнім;
- бажано перевірити алгоритм на папірці: отримані дані також можна буде використовувати в автотестах;
- мова програмування довільна;
- дозволяється написання коду в бригадах розміром до двох людей; за commit messages буде визначатися внесок кожного учаснику, і на основі цього робитися висновок щодо того, чи було роботу виконано бригадою чи однією людиною;
- бажано використовувати технології, що спрощують збірку й запуск коду: CMake для C, C++ і Fortran, setuptools для Python, npm для NodeJS, Maven/Ant для Java/Scala і так далі.

## Критерії оцінювання

- 2 бали (обов'язкова умова): коректно працююча програма з кодом у доступному викладачеві репозитарії (наприклад, GitHub, GitLab, BitBucket) та наявність теоретичного розв'язку поставленої задачі (теорія здається кожним учасником бригади окремо; вимоги ті ж самі, що й до основного завдання);
- 1 бал: наявність автотестів (наприклад, doctest у Python, Boost.Test у C++, JUnit у Java);
- 1 бал: наявність змістовних commit messages та CI (наприклад, GitHub Actions, GitLab CI, BitBucket Pipelines, TravisCI, Jenkins).

Якщо не знаєте, що обрати для тестування або CI, або як писати тести чи прив'язати до свого репозитарію CI, зверніться до викладача.

## 3.1 Бінарна штрафна функція

### Задача

Беремо англійську абетку (з пробілом) $T$ та бінарні еталонні зображення букв $G : T \to \{0, 1\}^{n \times m}$. Зображення тексту генерується шляхом конкатенації відповідних еталонів по горизонталі та незалежною інверсією кожного пікселя з ймовірністю $p$, яку вказує користувач. Ймовірності біграм дані. Ймовірність букви знаходитися на першому місці вважати рівною ймовірності того, що букві передує пробіл.

### Мета

Закріпити навички розпізнавання станів прихованих моделей Маркова.

### Завдання

На вхід програмі подається

- шлях до папки, що містить еталони букв (еталони мають назви a.png, b.png, ..., z.png та space.png для пробілу);
- рівень шуму від 0 до 1;
- зашумлене зображення з однією строкою тексту.

Програма виводить розпізнаний текст.

Ймовірності надає викладач. Також програма повинна містити режим розпізнавання за рівномірних ймовірностей переходів (незалежне розпізнавання кожного символу).

# Розділ 2: Алгоритм дифузії
# Домашнє завдання

## 1 Основне (2 бали)

### 1.1 Супермодулярність і репараметризація

Показати, що для двох супермодулярних функцій $g_1 : K^2 \to \mathbb{R}$ і $g_2 : K^2 \to \mathbb{R}$ функція

$$g\left(k, k'\right) = \max_{\varkappa \in K} \left\{g_1\left(k, \varkappa\right) + g_2\left(\varkappa, k'\right)\right\}$$

також супермодулярна.

### 1.2 Супермодулярність і увігнутість

У деяких випадках зручно використовувати увігнуті функції в якості вагів. Доведіть, що для будь-якої увігнутої $f : \mathbb{R} \to \mathbb{R}$ та скінченної множини міток $K \subset \mathbb{R}$ (де впорядкованість та різниця визначаються природньо) функція

$$g\left(k, k'\right) = f\left(k - k'\right)$$

супермодулярна.

## 2 Додаткове (3 бали)

### 2.1 Навчання на задачах розмітки

Нехай $T = \{1, \ldots, n\}^2$ — поле зору, $\tau \subset T^2$ — структура другого порядку, визначена на полі зору, $X \ni x : T \to \{0, 1\}$ — зображення, задане на полі зору $T$, $q : T \times X \times \{0, 1\} \to \mathbb{R}$ — відома унарна штрафна функція, що залежить від зображення, $g : \tau \times \{0, 1\}^2 \to \mathbb{R}$ — **не**відома бінарна штрафна функція, що **не** залежить від зображення. Маємо експоненційний розподіл виду

$$p\left(k; x\right) = \frac{1}{Z\left(x\right)} \cdot \exp\left\{\sum_{t \in T} q_t\left(k_t; x\right) + \sum_{tt' \in \tau} g_{tt'}\left(k_t, k_{t'}\right)\right\}, \quad k : T \to \{0, 1\}, \quad x : T \to \{0, 1\}.$$

Побудувати ефективний алгоритм, що на вхід приймає таку множину пар $\left\{\left\langle x^i, k^i\right\rangle : i = \overline{1, m}\right\}$, що

$$p\left(k^i; x^i\right) > p\left(k; x^i\right), \quad \forall k \neq k^i, \quad \forall i = \overline{1, m},$$

а на виході дає бінарну штрафну функцію $g : \tau \times \{0, 1\}^2 \to \mathbb{R}$, для якої ці нерівності виконуться для всієї навчальної множини.

Вважати, що ми знаємо ефективний алгоритм пошуку найкращої розмітки, а також інший алгоритм, який для скінченної множини $X \subset \mathbb{R}^m$ довільної скінченної розмірності $m$, такої що існує невідоме $\alpha \in \mathbb{R}^m$, для якого

$$\alpha \cdot x > 0, \quad x \in X,$$

знайде одне з можливих $\alpha$ за скінченну кількість кроків (наприклад, алгоритм навчання персептрону або алгоритм Козинця).

## 3 Комп'ютерне (4 бали)

### 3.1 Основи текстурної сегментації

#### Задача

Для зашумленого кольорового зображення $x : T \to \{0, \ldots, 255\}^3$ побудувати ітеративне наближення до оптимальної бінарної сегментації за допомогою алгоритму дифузії

$$k^* \in \operatorname*{argmax}_{k : T \to \{0,1\}} \frac{1}{Z\left(x\right)} \cdot \exp\left\{-\sum_{t \in T} \left[\!\left[\left\|x_t - c\left(k_t\right)\right\| > \left\|x_t - c\left(\tilde{k}_t\right)\right\|\right]\!\right] - \sum_{tt' \in \tau} \alpha \cdot \left[\!\left[k_t \neq k_{t'}\right]\!\right]\right\},$$

де $\tilde{k}$ означає заперечення до $k$, відображення $c : K \to \{0, \ldots, 255\}^3$ дає колір сегменту (студент самостійно обирає два кольори, на які хоче розбити зображення), константу $\alpha > 0$ студент обирає під конкретні зображення, $[\![\cdot]\!]$ — дужки Айверсона, $0 < Z$ — нормуючий множник розподілу, а $\tau$ — така структура сусідства, що сусідами пікселя $t$ з координатами $\langle t_x, t_y \rangle$ є

$$N(t) = \{\langle t_x - 1, t_y \rangle, \langle t_x + 1, t_y \rangle, \langle t_x, t_y - 1 \rangle, \langle t_x, t_y + 1 \rangle\}.$$

## Мета

Закріпити навички пошуку найбільш ймовірного стану випадкового поля.

## Завдання

На вхід програмі подається

- шлях до зашумленого зображення;
- значення $\alpha$;
- кольори двох сегментів (або інтенсивності у випадку чорно-білих зображень).

Програма малює результати кожної ітерації пошуку сегментації.

# Розділ 3: TRW-S на зображенні
# Домашнє завдання

## 1 Основне (2 бали)

### 1.1 TRW-S на ланцюзі Маркова

Навести алгоритм пошуку найкращої розмітки за допомогою TRW-S на ланцюзі

$$k^* = \operatorname*{argmax}_{k \in K^{n+1}} \left\{ \frac{1}{z} \cdot \exp\left[ -\sum_{i=0}^{n} q_i(k_i) - \sum_{i=1}^{n} g_i(k_{i-1}, k_i) \right] \right\}.$$

## 2 Додаткове (3 бали)

### 2.1 Метод напівглобальної оптимізації

Нехай $T = \{1, \ldots, n\}^2$ — поле зору. Дано скінченну множину міток $K$, унарну функцію $q : T \times K \to \mathbb{R}$ та симетричну бінарну функцію $g : \tau \times K^2 \to \mathbb{R}$.

Кожен піксель є коренем дерева, гілки якого напрямлено в чотири сторони: горизонтально вліво (до лівого краю), горизонтально вправо (до правого краю), вертикально вгору (до верхнього краю) і вертикально вниз (до нижнього краю). Для кожного пікселя треба знайти таку мітку, що є частиною кращої розмітки дерева, коренем якого він є

$$k_t^* \in \operatorname*{argmin}_{k_t \in K} \min_{k : T \setminus \{t\} \to K} \left\{ \sum_{t \in T} q_t(k_t) + \sum_{tt' \in \tau} g_{tt'}(k_t, k_{t'}) \right\}.$$

де

$$\sum_{tt' \in \tau} g_{tt'}(k_t, k_{t'}) = \sum_{x=t_x}^{n-1} g_{t_y,x}\left(k_{t_y,x}, k_{t_y,x+1}\right) + \sum_{x=t_x}^{2} g_{t_y,x}\left(k_{t_y,x}, k_{t_y,x-1}\right) + \sum_{y=t_y}^{n-1} g_{y,t_x}(k_{y,t_x}, k_{y+1,t_x}) + \sum_{y=t_y}^{2} g_{y,t_x}(k_{y,t_x}, k_{y-1,t_x}).$$

Описати алгоритм пошуку таких міток, що потребує не більше, ніж $O\left(n^2 \cdot |K|^2\right)$ порівнянь та додавань. Вважаємо, що $q$ та $g$ розраховуються за константний час, та пам'ятаємо, що $n$ в даному випадку — висота та ширина зображення, тобто $n^2$ — його площа.

Підказки:

- Виведення TRW-S потребувало розв'язок подібної задачі, тому ідею розв'язку можна здобути з лекцій.
- Схожі методи описано у статтях
  https://core.ac.uk/download/pdf/11134866.pdf
  та http://dev.ipol.im/~facciolo/mgm/mgm.pdf.

## 3 Комп'ютерне (4 бали)

### 3.1 Основи реставрації зображень

#### Задача

Нехай $T = \{1, \ldots, h\} \times \{1, \ldots, w\}$ — поле зору. Маємо набір відтінків сірого $C = \{0, \ldots, 255\}$ і спеціальне значення $\varepsilon$, що відповідає за відсутність інформації про колір (яке може співпадати з існуючим кольором — наприклад, чорним). На вході пошкоджене чорно-біле зображення $x : T \to C \cup \{\varepsilon\}$. Студент обирає набір міток з множини кольорів $K \subseteq C$ (наприклад, 5 відтінків сірого $K = \{0, 64, 128, 192, 255\}$) і коефіцієнт згладжування $\alpha > 0$. Побудувати ітеративне наближення кращої розмітки за допомогою алгоритму TRW-S для розподілу

$$k^* \in \operatorname*{argmax}_{k : T \to K} \frac{1}{Z(x)} \cdot \exp\left\{ -\sum_{t \in T} \|x_t - k_t\| \cdot [\![x_t \neq \varepsilon]\!] - \alpha \cdot \sum_{tt' \in \tau} \|k_t - k_{t'}\| \right\},$$

де $\tau$ — така структура сусідства, що сусідами пікселя $t$ з координатами $\langle t_x, t_y \rangle$ є

$$N(t) = \{\langle t_x - 1, t_y \rangle, \langle t_x + 1, t_y \rangle, \langle t_x, t_y - 1 \rangle, \langle t_x, t_y + 1 \rangle\}.$$

Допитливі можуть дізнатися більше про цю задачу (Image Denoising and Inpainting) в роботі `https://www.cs.cornell.edu/~rdz/Papers/SZSVKATR-PAMI08.pdf`.

## Мета

Закріпити навички оцінки найбільш ймовірного стану випадкового поля за допомогою TRW-S.

## Завдання

На вхід програмі подається

- шлях до пошкодженого зображення;
- значення $\alpha$;
- кількість відтінків у $K$ (або перелік самих кольорів).

Для 16 відтінків на зображенні розміром $500 \times 250$ один прохід алгоритму (прямий і зворотній в сумі) повинен тривати не довше за одну хвилину.

Для більшої наочності можна розбити кольорове зображення на кілька каналів та обробити окремо, а потім об'єднати їх знов.

## Приклад

Далі наведено пошкоджене зображення 1 та поновлене 2. Зауважте, що окрім малюнків алгоритм частково впорався з кракелюром та відблисками.

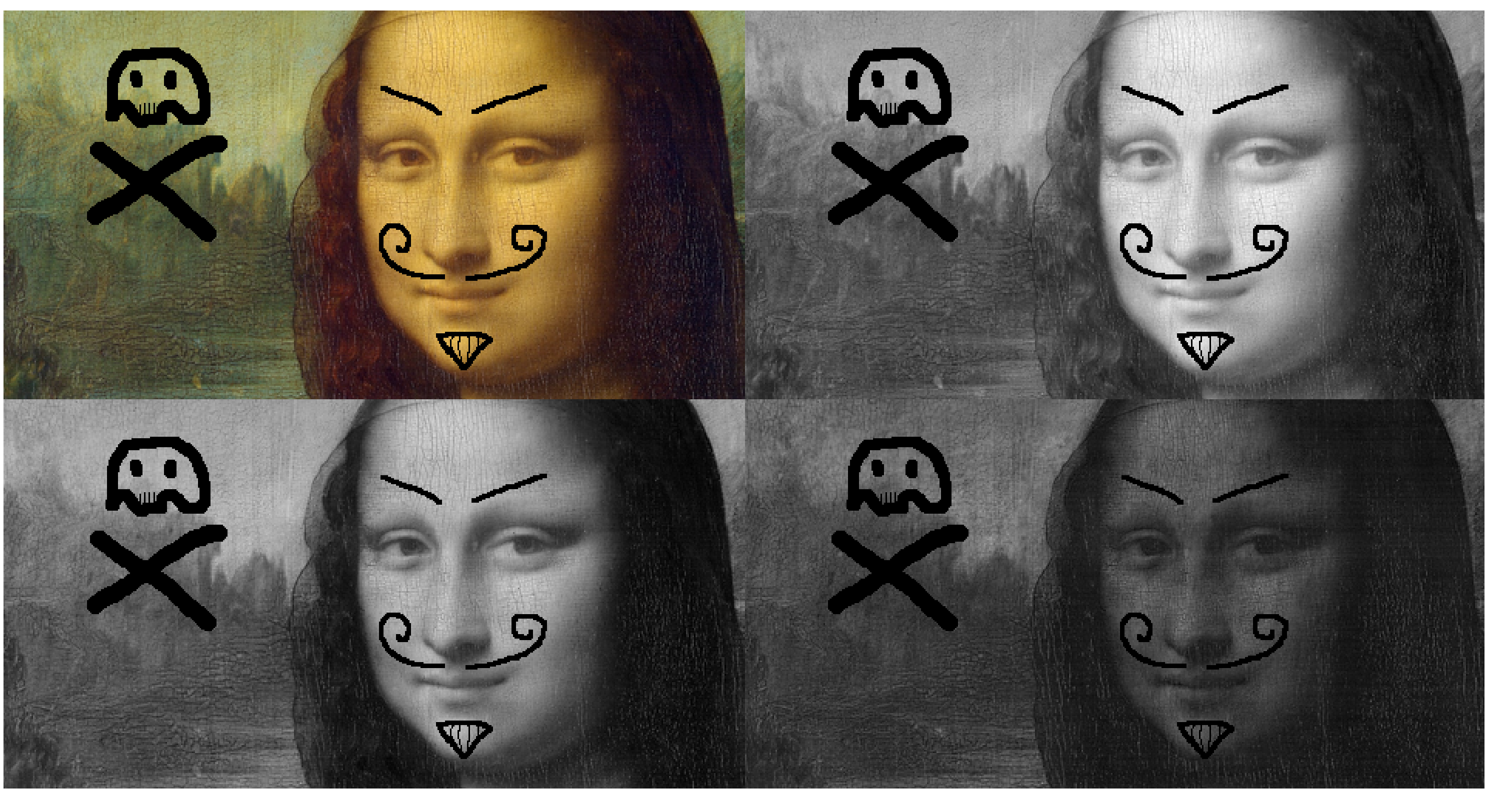


Рис. 1: Пошкоджене зображення з чорним кольором в якості $\varepsilon$, розкладене на канали

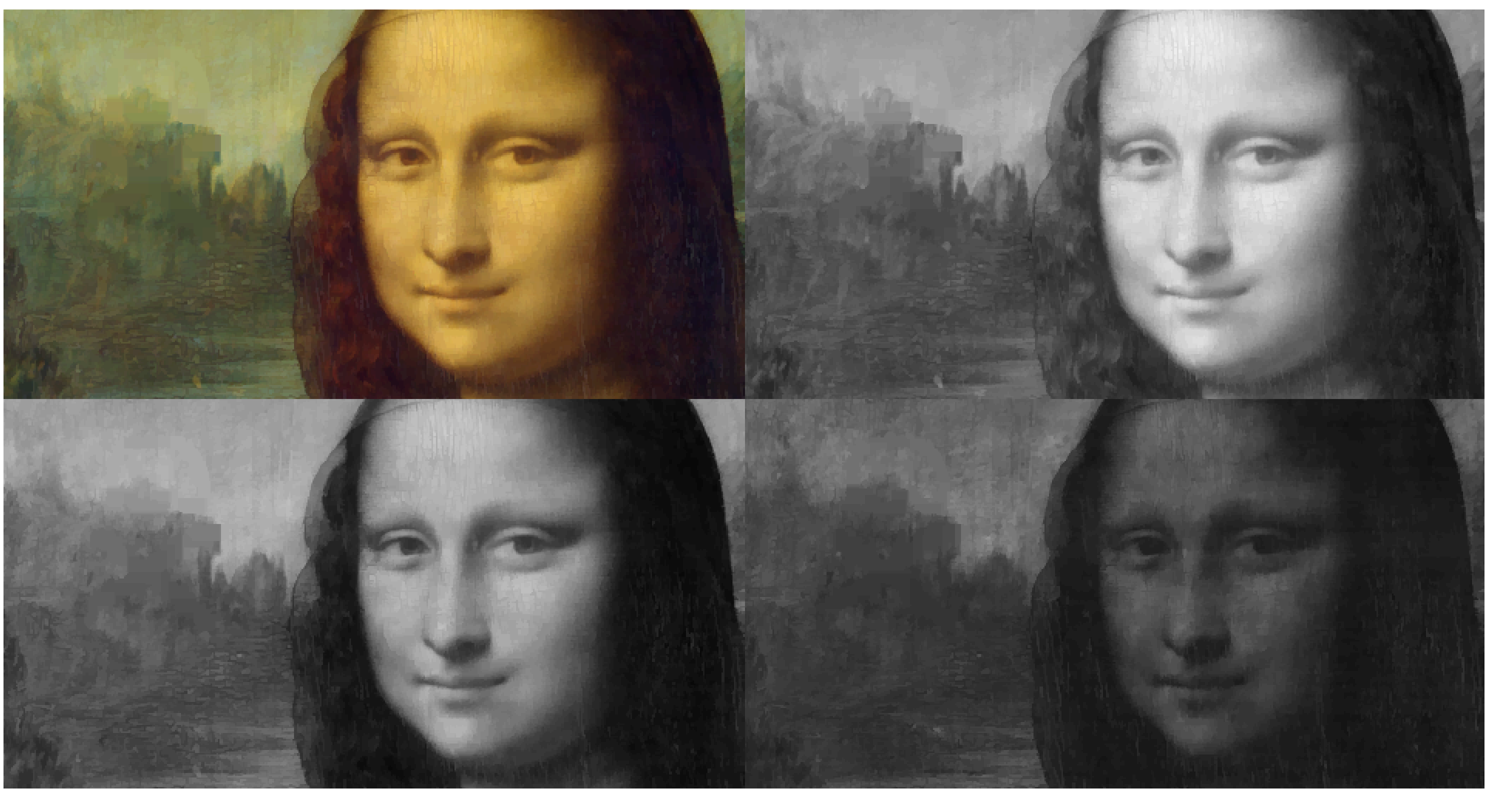


Рис. 2: Результати поновлення за 20 ітерацій з використанням 64 відтінків і $\alpha = 2.5$ для кожного каналу

# Розділ 4: ЕМ-алгоритм на умовних випадкових полях
# Домашнє завдання

## 1 Основне (2 бали)

### 1.1 Монотонність ЕМ-алгоритму

Сформулювати ЕМ-алгоритм самонавчання та показати, що на жодній ітерації він не зменшує функції правдоподібності.

## 2 Додаткове (3 бали)

### 2.1 ЕМ для GMM

Вивести ЕМ-алгоритм для кластеризації елементів з вибірки тривимірних нормально розподілених векторів, кожен з яких належить одному з двох розподілів з невідомими параметрами $\mu_1 \in \mathbb{R}^3$, $\mu_2 \in \mathbb{R}^3$, $\Sigma_1 \in \mathbb{R}^{3\times 3}$ та $\Sigma_2 \in \mathbb{R}^{3\times 3}$.

## 3 Комп'ютерне (4 бали)

### 3.1 Основи текстурної сегментації

#### Задача

Задано поле зору $T = \{1, \ldots, h\} \times \{1, \ldots, w\}$. Маємо кольорове триканальне (RGB) зображення $x : T \to C$ та його часткову сегментація $s : T \to \{b, f, \varepsilon\}$. Реалізувати алгоритм, що складається з чергування наступних кроків:

1. Навчання суміші тривимірних випадкових векторів, розподілених за нормальним законом (gaussian mixture model, GMM), використовуючи сегментацію $s$, що дає на виході набори $\left\{\left\langle p^f(i), \boldsymbol{\mu}_i^f, \Sigma_i^f\right\rangle : i = \overline{1, N^f}\right\}$ і $\left\{\left\langle p^b(i), \boldsymbol{\mu}_i^b, \Sigma_i^b\right\rangle : i = \overline{1, N^b}\right\}$, де $N^f \in \{3, \ldots, 8\}$, $N^b \in \{3, \ldots, 8\}$.

2. Розв'язок задачі розмітки на умовному випадковому полі

$$s \in \underset{s:T\to\{b,f\}}{\operatorname{argmax}}\, p(s; x) = \underset{s:T\to\{b,f\}}{\operatorname{argmax}} \frac{1}{Z(x)} \cdot \prod_{t\in T} q_t(s_t; x) \cdot \prod_{tt'\in\tau} g_{tt'}(s_t, s_{t'}; x),$$

де

$$q_t(s_t; x) = \sum_{i=1}^{N^{s_t}} p^{s_t}(i) \cdot (2\cdot\pi)^{-\frac{5}{2}} \cdot \det\left(\Sigma_i^{s_t}\right)^{-\frac{1}{2}} \cdot \exp\left\{-\frac{1}{2} \cdot \left(\boldsymbol{x}_t - \boldsymbol{\mu}_i^{s_t}\right)^T \cdot \left(\Sigma_i^{s_t}\right)^{-1} \left(\boldsymbol{x}_t - \boldsymbol{\mu}_i^{s_t}\right)\right\},$$

$$g_{tt'}(s_t, s_{t'}; x) = \exp\left\{-\left[\!\left[s_t \neq s_{t'}\right]\!\right] \cdot \gamma \cdot e^{-\frac{\left\|\boldsymbol{x}_t - \boldsymbol{x}_{t'}\right\|^2}{2\cdot\beta}}\right\},$$

$$\gamma \in [10; 100],$$

$$\beta = \frac{\sum\limits_{tt'\in\tau} \left\|\boldsymbol{x}_t - \boldsymbol{x}_{t'}\right\|^2}{|\tau|}.$$

Зі схожими методами сегментації, що використовують схожі вхідні дані та ті ж розподіли, можна ознайомитися за посиланнями

- https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.9971&rep=rep1&type=pdf
- https://cvg.ethz.ch/teaching/cvl/2012/grabcut-siggraph04.pdf
- https://sandipanweb.wordpress.com/2018/02/11/interactive-image-segmentation-with-graph-cut
- https://www.robots.ox.ac.uk/~varun/prs.pdf

Зверніть увагу на те, що в даних роботах використовується інший алгоритм пошуку найбільш ймовірного стану випадкового поля, а саме розв'язок задачі про мінімальний зріз графу або її двоїстої задачі — задачі пошуку максимального потоку.

### Мета

Ефективна реалізація оцінки найбільш ймовірного стану випадкового поля та EM-алгоритму кластеризації нормальних випадкових векторів.

### Завдання

На вхід програмі подається

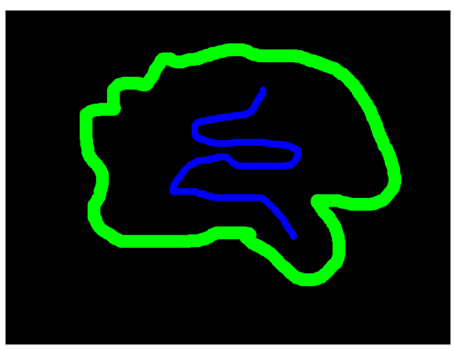
- шлях до зображення;
- шлях до зображення з частковою сегментацією, де синій та червоний кольори означають фон та об'єкт, а чорний позначає невідому частину, яку програма повинна сегментувати;
- параметри $\gamma$, $N^f$, $N^b$;
- максимальну кількість ітерацій EM-алгоритму;
- максимальну кількість ітерацій алгоритму пошуку найбільш ймовірного стану випадкового поля (якщо використовується ітеративний алгоритм);

EM-алгоритм та алгоритм пошуку найбільш ймовірного стану випадкового поля потрібно реалізувати власноруч.

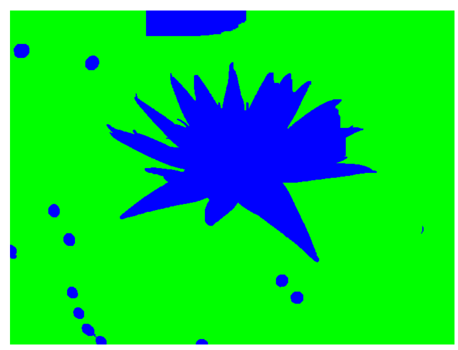
У випадку використання алгоритму дифузії чи TRW-S 10 проходів по зображенню у прямому та зворотньому напрямку не повинно виконуватись більше ніж 100 секунд на зображенні розміром $700 \times 500$.

### Приклади

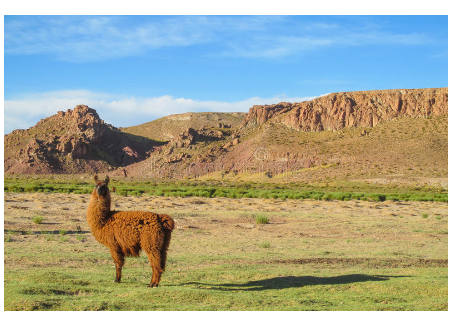
(a) Вхідне зображення $x$



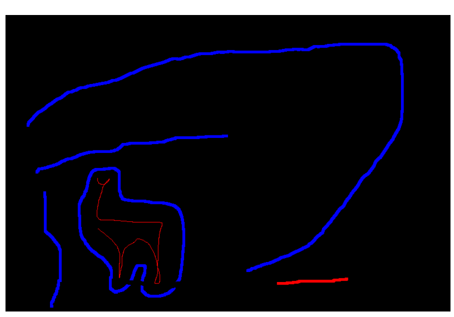
(b) Початкова сегментація $s$



(c) Результат роботи алгоритму з параметрами $N^f = N^b = 5$, $\gamma = 50$, 10 кроків алгоритму по 100 проходів TRW-S в кожному



(a) Вхідне зображення $x$



(b) Початкова сегментація $s$

(c) Результат роботи алгоритму з параметрами $N^f = N^b = 5$, $\gamma = 50$, 5 кроків алгоритму по 100 проходів TRW-S в кожному