

Звіт

# 1 Компактна форма кубічного рівняння для алгоритму оцінки фундаментальної матриці за сімома точками

Результати даного дослідження апробовано на XV Всеукраїнській науково-практичній конференції студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики» у 2020 році [15].

Фундаментальна матриця грає дуже важливу роль у комп'ютерному стереозорі, для побудови просторової моделі за набором зображень [4, 14]. Даний розділ присвячено одному з алгоритмів оцінки цієї матриці. Цей алгоритм має гарну сходимість та стійкість до шуму, проте його ефективна реалізація потребує ефективного розв'язку кубічного рівняння. У зарубіжній літературі алгоритм відомий як 7-points algorithm (семиточковий алгоритм). Для спрощення кубічного рівняння, що в ньому виникає, можна використовувати інструменти автоматичного виводу формул на кшталт Macaulay2 [3], SymPy [11], Mathematica [6] та інших.

Про актуальність такого дослідження свідчать роботи, в яких міститься спрощення рівнянь, які виникають при оцінці істотної матриці, коли внутрішні параметри камери з точковою діафрагмою відомі [12] або відомі частково [7]. Даний розділ присвячено випадку, коли внутрішні параметри камери невідомі.

Подібні спрощення рівнянь потрібні не тільки для математичної краси: їх прагматичне значення полягає в тому, що більш прості рівняння простіше запрограмувати, перевірити коректність їх розв'язку, швидше знаходити помилки.

## 1.1 Алгоритм

На вході маємо два зображення що відображають одну й ту ж саму сцену, які здебільшого відображають одні й ті ж самі об'єкти. За допомогою деякого алгоритму (наприклад, SIFT [8], SURF [1], ORB [13] та інші) отримуємо  $n$  пар точок  $X \subset \mathbb{R}^{n \times 2 \times 3}$ , що потенційно можуть відповідати одна одній на даних двох зображеннях (третя координата дорівнює одиниці і вводиться у даному випадку для лінійності). Ми хочемо знайти матрицю  $F \in \mathbb{R}^{3 \times 3}$  рангу 2, для якої

$$\mathbf{x}^T F \mathbf{x} = 0, \quad \forall \langle \mathbf{x}', \mathbf{x} \rangle \in X.$$

Наявність шуму та невірних відповідностей призводить до задачі

$$|\mathbf{x}'^T F \mathbf{x}| < \varepsilon, \quad \forall \langle \mathbf{x}', \mathbf{x} \rangle \in X' \subset X, \quad (1)$$

де  $\varepsilon \sim 10^{-3}$ , а  $X'$  це множина тих відповідностей, для яких нерівність виконується. Кількість елементів у множині  $X'$  є показником якості знайденої  $F$ .

Алгоритм складається з чергування кроків

1. обрати сім пар відповідних точок з множини  $X$ ;
2. знайти фундаментальну матрицю, що відповідає обраній сімці відповідностей за допомогою розв'язку рівнянь, що наведено далі;
3. підрахувати кількість пар, що задовольняють (1).

В результаті (після тисяч ітерацій) потрібно обрати найкращу матрицю — ту, за якої  $|X'|$  сягає найбільшого значення.

Вираз  $\mathbf{x}'^T F \mathbf{x}$  є лінійним відносно компонент матриці  $F$

$$\begin{aligned} \mathbf{x}'^T F \mathbf{x} &= \begin{bmatrix} x'_1 & x'_2 & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = \\ &= \text{vec} (\mathbf{x}' \mathbf{x}^T)^T \text{vec} (F), \end{aligned}$$

де  $\text{vec}$  є оператором конкатенації рядків у матриці в один вектор-рядок. Запишемо систему лінійних рівнянь

$$\mathbf{x}_i^T F \mathbf{x}_i = 0, \quad i = \overline{1, 7}$$

у більш звичному вигляді

$$\begin{bmatrix} \text{vec} (\mathbf{x}_1' \mathbf{x}_1^T)^T \\ \vdots \\ \text{vec} (\mathbf{x}_7' \mathbf{x}_7^T)^T \end{bmatrix} \text{vec} (F) = 0. \quad (2)$$

Система містить сім рівнянь та дев'ять невідомих. Це означає, що матриця коефіцієнтів має двовимірне ядро, базисні вектори якого позначимо  $\mathbf{f}_1$  та  $\mathbf{f}_2$ . Для зручності переведемо ці вектори у матриці за допомогою оберненого до оператору векторизації матриці:  $F_1 = \text{vec}^{-1} \mathbf{f}_1$  та  $F_2 = \text{vec}^{-1} \mathbf{f}_2$ . Якщо не вводити додаткових обмежень, нетривіальним розв'язком системи (2) є будь-яка матриця виду

$$F = \alpha_1 F_1 + \alpha_2 F_2, \quad \alpha_1^2 + \alpha_2^2 > 0. \quad (3)$$

Проте обмеження  $\epsilon$ . Воно потребує, щоб матриця  $F$  мала ранг 2. Це обмеження прийнято спрощувати: вимагається, щоб матриця  $F$  мала нульовий визначник. Отже, треба знайти такі  $\alpha_1$  і  $\alpha_2$ , що

$$\det(F) = \det(\alpha_1 F_1 + \alpha_2 F_2) = 0, \quad \alpha_1^2 + \alpha_2^2 > 0.$$

Якщо одна з цих матриць вже має нульовий визначник, саме її можна обрати в якості розв'язку на даній ітерації. В іншому випадку ми точно впевнені у тому, що ані  $\alpha_1$ , ані  $\alpha_2$  не дорівнюють нулеві, адже, якщо хоча б одне з них дорівнює нулеві, то й інше теж повинно бути нульовим, щоб визначник дорівнював нулеві, що дає тривіальний розв'язок, який нас не цікавить. Це означає, що матриці  $F$ ,  $F_1$  і  $F_2$  можна розділити на  $\alpha_1$  (чи  $\alpha_2$ )

$$\det\left(\frac{F}{\alpha_1}\right) = \det\left(F_1 + \frac{\alpha_2}{\alpha_1} \cdot F_2\right) = 0, \quad \alpha_1 \neq 0, \quad \alpha_2 \neq 0.$$

Розв'язок системи (2) не залежить від ненульового константного множника матриці  $F$ , тому обмеження на визначник можна спростити до

$$\det F = \det(F_1 + \alpha F_2) = 0, \quad \alpha \neq 0. \quad (4)$$

Оскільки матриці  $F_1$  і  $F_2$  мають розміри  $3 \times 3$ , перед нами кубічне рівняння відносно  $\alpha$ , спрощенню якого присвячено даний розділ звіту.

## 1.2 Виведення компактної форми кубічного рівняння

**Теорема 1.** *Нехай  $A$  і  $B$  — невироджені матриці  $3 \times 3$ . Тоді*

$$\det(A + B) = \det(A) + \det(A) \cdot \text{tr}(B \cdot A^{-1}) + \det(B) \cdot \text{tr}(A \cdot B^{-1}) + \det(B)$$

*Доказательство.* Нехай  $\epsilon_{ijk}$  — символ Леві-Чивіти. Тоді

$$\det(A) = \sum_{i,j,k} \epsilon_{ijk} a_{i1} a_{j2} a_{k3}$$

і

$$\begin{aligned}
\det(A + B) &= \\
&= \sum_{i,j,k} \epsilon_{ijk} \cdot (a_{i1} + b_{i1}) \cdot (a_{j2} + b_{j2}) \cdot (a_{k3} + b_{k3}) = \\
&= \sum_{i,j,k} \epsilon_{ijk} \cdot a_{i1} \cdot a_{j2} \cdot a_{k3} + \sum_{i,j,k} \epsilon_{ijk} \cdot a_{i1} \cdot a_{j2} \cdot b_{k3} + \\
&+ \sum_{i,j,k} \epsilon_{ijk} \cdot a_{i1} \cdot b_{j2} \cdot a_{k3} + \sum_{i,j,k} \epsilon_{ijk} \cdot a_{i1} \cdot b_{j2} \cdot b_{k3} + \\
&+ \sum_{i,j,k} \epsilon_{ijk} \cdot b_{i1} \cdot a_{j2} \cdot a_{k3} + \sum_{i,j,k} \epsilon_{ijk} \cdot b_{i1} \cdot a_{j2} \cdot b_{k3} + \\
&+ \sum_{i,j,k} \epsilon_{ijk} \cdot b_{i1} \cdot b_{j2} \cdot a_{k3} + \sum_{i,j,k} \epsilon_{ijk} \cdot b_{i1} \cdot b_{j2} \cdot b_{k3}.
\end{aligned}$$

Нехай  $C_{ij}^A$  і  $C_{ij}^B$  — алгебраїчні доповнення відповідних мінорів матриць  $A$  і  $B$  відповідно. Тоді

$$\begin{aligned}
\det(A + B) &= \\
&= \det(A) + \sum_k C_{k3}^A \cdot b_{k3} + \sum_j C_{j2}^A \cdot b_{j2} + \sum_i C_{i1}^B \cdot a_{i1} + \\
&+ \sum_i C_{i1}^A \cdot b_{i1} + \sum_j C_{j2}^B \cdot a_{j2} + \sum_k C_{k3}^B \cdot a_{k3} + \det(B) = \\
&= \det(A) + \sum_{ij} C_{ij}^A \cdot b_{ij} + \sum_{ij} C_{ij}^B \cdot a_{ij} + \det(B).
\end{aligned}$$

Перепишемо отриманий результат окремо, адже він також є важливим через ефективність його обчислювання

$$\det(A + B) = \det(A) + \sum_{ij} C_{ij}^A \cdot b_{ij} + \sum_{ij} C_{ij}^B \cdot a_{ij} + \det(B). \quad (5)$$

Відомо, що  $A^{-T} = \frac{1}{\det(A)} C^A$ . Позначимо елементи оберненої матриці  $A^{-1}$  як  $A_{ij}^{-1}$ . Тоді

$$\begin{aligned}
\det(A + B) &= \\
&= \det(A) + \det(A) \cdot \sum_{ij} A_{ji}^{-1} \cdot b_{ij} + \det(B) \cdot \sum_{ij} B_{ji}^{-1} \cdot a_{ij} + \det(B) = \\
&= \det(A) + \det(A) \cdot \text{tr}(B \cdot A^{-1}) + \det(B) \cdot \text{tr}(A \cdot B^{-1}) + \det(B).
\end{aligned}$$

□

Оскільки матриці  $F_1$  і  $F_2$  невироджені, ми можемо переписати рівняння (4) у вигляді

$$\begin{aligned}\det(F_1 + \alpha F_2) &= \\ &= \det(F_1) + \det(F_1) \operatorname{tr}(\alpha F_2 F_1^{-1}) + \det(\alpha F_2) \operatorname{tr}(F_1 (\alpha F_2)^{-1}) + \det(\alpha F_2).\end{aligned}$$

Оскільки матриця  $F_2$  має розміри  $3 \times 3$ ,

$$\det(\alpha F_2) = \alpha^3 \det(F_2).$$

Маємо *компактну форму* кубічного рівняння відносно  $\alpha$

$$\begin{aligned}\det(F_1 + \alpha F_2) &= \\ &= \det(F_1) + \alpha \cdot \det(F_1) \cdot \operatorname{tr}(F_2 F_1^{-1}) + \\ &\quad + \alpha^2 \cdot \det(F_2) \cdot \operatorname{tr}(F_1 F_2^{-1}) + \alpha^3 \cdot \det(F_2).\end{aligned}\tag{6}$$

Отримане рівняння компактне, проте його безпосередня реалізація може бути обчислювально неефективною. Згадаємо, що протягом доведення теореми 1, було знайдено важливий проміжний результат (5), який має більш компактний вигляд, проте потребує менше обчислень

$$\det(F_1 + \alpha F_2) = \det(F_1) + \alpha \sum_{ij} C_{ij}^{F_1} f_{2ij} + \alpha^2 \sum_{ij} C_{ij}^{F_2} f_{1ij} + \alpha^3 \det(F_2).\tag{7}$$

### 1.3 Перевірка результату за допомогою бібліотеки символьних обчислень

Наступний код, що використовує бібліотеку SymPy, створює матриці  $A$  і  $B$  розміром  $3 \times 3$ , обчислює їх визначник за допомогою методу `det`, обернену матрицю за допомогою методу `inv`, слід за допомогою методу `trace`, а за допомогою функції `cancel` спрощується вираз (розкриваються дужки, скорочуються подібні вирази тощо)

```
>>> from sympy import *
>>> A = Matrix(3, 3, symbols('a:3:3'))
>>> B = Matrix(3, 3, symbols('b:3:3'))
>>> d = (A + B).det()
>>> s = (
...     A.det() + A.det() * (A.inv() * B).trace()
...     + B.det() * (B.inv() * A).trace() + B.det()
... )
>>> cancel(d - s)
0
```

В результаті отримаємо 0. Отже вирази, що було розраховано за двома формулами (візначенник звичайної суми та компактна формула), дають однакові результати.

Розглянемо, який вигляд мають коефіцієнти поліному, корені якого треба знайти, якщо не використовувати компактну формулу

$$\begin{aligned}
 \det(A + x \cdot B) &= c_0 + c_1 \cdot x + c_2 \cdot x^2 + c_3 \cdot x^3, \\
 c_0 &= a_{11} \cdot a_{22} \cdot a_{33} - a_{11} \cdot a_{23} \cdot a_{32} - a_{12} \cdot a_{21} \cdot a_{33} + \\
 &\quad + a_{12} \cdot a_{23} \cdot a_{31} + a_{13} \cdot a_{21} \cdot a_{32} - a_{13} \cdot a_{22} \cdot a_{31}, \\
 c_1 &= a_{11} \cdot a_{22} \cdot b_{33} - a_{11} \cdot a_{23} \cdot b_{32} - a_{11} \cdot a_{32} \cdot b_{23} + \\
 &\quad + a_{11} \cdot a_{33} \cdot b_{22} - a_{12} \cdot a_{21} \cdot b_{33} + a_{12} \cdot a_{23} \cdot b_{31} + \\
 &\quad + a_{12} \cdot a_{31} \cdot b_{23} - a_{12} \cdot a_{33} \cdot b_{21} + a_{13} \cdot a_{21} \cdot b_{32} - \\
 &\quad - a_{13} \cdot a_{22} \cdot b_{31} - a_{13} \cdot a_{31} \cdot b_{22} + a_{13} \cdot a_{32} \cdot b_{21} + \\
 &\quad + a_{21} \cdot a_{32} \cdot b_{13} - a_{21} \cdot a_{33} \cdot b_{12} - a_{22} \cdot a_{31} \cdot b_{13} + \\
 &\quad + a_{22} \cdot a_{33} \cdot b_{11} + a_{23} \cdot a_{31} \cdot b_{12} - a_{23} \cdot a_{32} \cdot b_{11}, \\
 c_2 &= a_{11} \cdot b_{22} \cdot b_{33} - a_{11} \cdot b_{23} \cdot b_{32} - a_{12} \cdot b_{21} \cdot b_{33} + \\
 &\quad + a_{12} \cdot b_{23} \cdot b_{31} + a_{13} \cdot b_{21} \cdot b_{32} - a_{13} \cdot b_{22} \cdot b_{31} - \\
 &\quad - a_{21} \cdot b_{12} \cdot b_{33} + a_{21} \cdot b_{13} \cdot b_{32} + a_{22} \cdot b_{11} \cdot b_{33} - \\
 &\quad - a_{22} \cdot b_{13} \cdot b_{31} - a_{23} \cdot b_{11} \cdot b_{32} + a_{23} \cdot b_{12} \cdot b_{31} + \\
 &\quad + a_{31} \cdot b_{12} \cdot b_{23} - a_{31} \cdot b_{13} \cdot b_{22} - a_{32} \cdot b_{11} \cdot b_{23} + \\
 &\quad + a_{32} \cdot b_{13} \cdot b_{21} + a_{33} \cdot b_{11} \cdot b_{22} - a_{33} \cdot b_{12} \cdot b_{21}, \\
 c_3 &= b_{11} \cdot b_{22} \cdot b_{33} - b_{11} \cdot b_{23} \cdot b_{32} - b_{12} \cdot b_{21} \cdot b_{33} + \\
 &\quad + b_{12} \cdot b_{23} \cdot b_{31} + b_{13} \cdot b_{21} \cdot b_{32} - b_{13} \cdot b_{22} \cdot b_{31}.
 \end{aligned}$$

Ми бачимо, що наведену в даному підрозділі компактну форму цього рівняння доцільніше використовувати, аніж ту, що отримується безспосереднім обрахунком візначенника суми матриць.

## 1.4 Висновки

У даному розділі розглянуто компактний (6) та напівкомпактний (7) вид кубічного рівняння, що виникає в алгоритмі оцінки фундаментальної матриці за сімома точками. Ця компактна форма має наступні переваги над автоматичним виводом:

- її можливо запам'ятати;
- її одержання корисне для згадування лінійної алгебри, що знадобиться у викладанні предметів, пов'язаних з обчислювальною геометрією у комп'ютерному зорі;

- вона проста в реалізації та відлажоджуванні у коді прикладного програмного забезпечення.

## 2 Вирівнювання стереопари за допомогою векторного добутку та фундаментальної матриці

Результати даного дослідження апробовано на XV Всеукраїнській науково-практичній конференції студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики» у 2020 році [16].

Фундаментальна матриця дозволяє визначити епіпольярні лінії та епіпольярні точки на стереопарі. Ці лінії допомагають зменшити розмір простору, де шукається розв'язок задачі бінокулярного стереозору. Іноді безпосередньому розв'язку задачі стереозору передує вирівнювання стереопарі, в результаті якого відповідні епіпольярні лінії обох зображень мають однакове розташування та напрям, а також стають паралельними до горизонтальної вісі. Саме такий процедурі вирівнювання (ректифікації) присвячено цей розділ звіту. Для отримання більш якісних результатів потрібно використовувати істотну матрицю, проте вона потребує знати внутрішні параметри камери, такі як фокусна відстань, форма пікселів (вони можуть бути прямокутні або ж мати форму косих паралелограмів) та центр зображення. Ці дані не завжди відомі з достатньою точністю, а бувають і зовсім невідомі. Саме тому методи вирівнювання стереопар за фундаментальною матрицею представляють не тільки теоретичний, а й практичний інтерес.

Спочатку в даному розділі описується алгоритм вирівнювання правого зображення за статтею [5]. Потім представлено сам алгоритм вирівнювання лівого зображення, що базується на вирівненому правому зображення і фундаментальній матриці, в основі якого лежить векторний добуток. Наприкінці проводиться аналіз запропонованого алгоритму, де особлива увага приділяється випадкам, коли він не працює коректно; проте в результаті аналізу з'ясовується, що такі випадки можна виявити до початку застосування алгоритму — за фундаментальною матрицею.

### 2.1 Вирівнювання правого зображення

Ми шукаємо такі лінійні перетворення, що лінії, на яких можуть знаходитися відповідні пари пікселів, були на одній горизонтальній лінії, щоб подальша задача стереозору полягала в тому, щоб для кожного пікселя одного зображення потрібно було знайти відповідний піксель на іншому зображені лише на тій самій горизонтальній прямій (а не

на всьому зображенні).

Фундаментальна матриця містить правило для двох точок з двох зображень, яке необхідне (проте не є достатнім) для того, щоб вони могли відповісти одна одній

$$\mathbf{x}'^T \cdot F \cdot \mathbf{x} = 0.$$

Один з методів оцінки цієї матриці наведено у попередньому підрозділі (\*\*\*) .

Фундаментальна матриця  $F$  має ранг 2, тому її ядра є одновимірними просторами. Вектор  $\mathbf{e} = [e_x \ e_y \ 1]^T$  з правого ядра називається епіполярною точкою лівого зображення. Геометрично епіполярна точка зображення є проекцією оптичного центру іншої камери. Аналогічно вектор  $\mathbf{e}' = [e'_x \ e'_y \ 1]^T$  з лівого ядра називається епіполярною точкою правого зображення.

Надалі вектори  $\mathbf{e}$  і  $\mathbf{e}'$  позначатимуть одиничні вектори з правого та лівого ядра фундаментальної матриці  $F$  відповідно. Це не вплине на загальність викладених результатів, проте спростить формули. Також ми розглядаємо камеру з точковою діафрагмою, центром зображення в точці  $[0 \ 0 \ 1]^T$  та центром камери в точці  $[0 \ 0 \ 0]^T$ . Якщо це не так, ми можемо змістити, повернути та масштабувати систему координат таким чином, щоб координати вказаних точок стали саме такими, що, звісно, не вплине на взаємне розташування камер та об'єктів реального світу.

Щоб зробити епіполярні лінії горизонтальними, нам потрібно змістити епіполярну точку в  $[1 \ 0 \ 0]^T$  (вправо на нескінченість). В однорідних координатах це можна зробити за допомогою лінійного перетворення, що називається ректифікацією. Назовемо це відображення  $R_r$

$$R_r = G \cdot R,$$

де  $R$  — матриця повороту, що обертає зображення так, щоб його епіполярна точка лежала на вісі абсцис, а матриця  $G$  — матриця зкошування, яка робить епіполярні лінії паралельними одну до одної, тобто зміщує епіполярну точку вправо на нескінченість.

Існують різні варіанти матриці  $R_r$  [5, 10]. В даній роботі використовуються наступні формули. Матриця  $R$  складається з ортонормованих

векторів

$$R = \begin{bmatrix} \frac{(\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T}{\|(\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T\|} \\ \frac{(\mathbf{k} \times \mathbf{e}')^T}{\|(\mathbf{k} \times \mathbf{e}')^T\|} \\ \mathbf{k}^T \end{bmatrix},$$

де  $\mathbf{k} = [0 \ 0 \ 1]^T$  — один з базисних векторів простору, а ортогональність векторів виконується за властивістю векторного добутку. Матриця  $G$  має вигляд

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{-e'_z}{x} & 0 & 1 \end{bmatrix},$$

де

$$x = (R \cdot \mathbf{e}')_x = (\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T \cdot \mathbf{e}'.$$
 (8)

Отже,

$$R_r = G \cdot R = \begin{bmatrix} (\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T \\ (\mathbf{k} \times \mathbf{e}')^T \\ (\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T \cdot \frac{-e'_z}{x} + \mathbf{k}^T \end{bmatrix}.$$

Більш докладно про ці висновки можна дізнатися з робіт [5, 4].

## 2.2 Запропонований алгоритм ректифікації

У даному підрозділі описано сам алгоритм, обґрунтування та пояснення якого знаходиться в наступному підрозділі.

Щоб лінеаризувати векторний добуток (для зручності переходів у формулах) введемо матрицю

$$[\mathbf{e}']_{\times} = \begin{bmatrix} 0 & -e'_z & e'_y \\ e'_z & 0 & -e'_x \\ -e'_y & e'_x & 0 \end{bmatrix},$$

для якої виконується

$$[\mathbf{e}']_{\times} \cdot \mathbf{x} = \mathbf{e}' \times \mathbf{x}, \quad \forall \mathbf{x} \in \mathbb{R}^3.$$

Побудуємо проміжну матрицю (зауважимо, що вона має ранг 2)

$$M = R_r \cdot [\mathbf{e}']_{\times} \cdot F$$

та позначимо її останні дві строки символами  $\mathbf{m}_y^T$  і  $\mathbf{m}_z^T$  відповідно, й побудуємо нову матрицю

$$M' = \begin{bmatrix} \frac{(\mathbf{m}_y \times \mathbf{m}_z)^T}{\sqrt{\|\mathbf{m}_y \times \mathbf{m}_z\|}} \\ \mathbf{m}_y^T \\ \mathbf{m}_z^T \end{bmatrix}.$$

Оскільки після вирівнювання епіполярні лінії стають паралельними вісі абсцис, а нам важливо сберігати лише відносний порядок точок зображення на епіполярних лініях, рекомендується після вирівнювання лівого зображення додатково стиснути (або розтягнути) його по горизонталі, щоб пікселі на ньому мали координати, що якомога близчі до координат відповідних їм пікселів на правому зображенні [5], адже зазвичай вирівнювання дуже сильно змінює масштаб на цій вісі або переміщує зображення по ній занадто далеко. Тому для вирівнювання лівого зображення рекомендується застосовувати матрицю  $R_l$

$$R_l = \begin{bmatrix} a & b & c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot M',$$

в якій коефіцієнти  $a$ ,  $b$  і  $c$  є розв'язками оптимізаційної задачі

$$\sum_{\langle \mathbf{x}, \mathbf{x}' \rangle \in X'} \left( \begin{bmatrix} a \\ b \\ c \end{bmatrix}^T \cdot \frac{M' \cdot \mathbf{x}}{(M' \cdot \mathbf{x})_z} - \frac{(R_r \cdot \mathbf{x}')_x}{(R_r \cdot \mathbf{x}')_z} \right)^2 \rightarrow \min_{a, b, c \in \mathbb{R}},$$

де  $X'$ , як і у минулому розділі, є множиною пар точок, що відповідають одна одній, та використовувалися для оцінки фундаментальної матриці. Ділення на координату  $z$  необідне для нормалізації координат точок.

На рис. 1 зображено результати одного з експериментів, де порівнюються результати, отримані за допомогою функції `stereoRectifyUncalibrated` з популярної бібліотеки OpenCV [2] (рис. 1b та рис. 1a), і результати, отримані за допомогою запропонованого алгоритму (рис. 1f та рис. 1e). Можна побачити, що обидва методи дали візуально схожі результати. На рис. 2 зображено ще один результат роботи наведеного алгоритму, але з позначенням епіполярних ліній.

### 2.3 Аналіз запропонованого алгоритму

Ми хочемо знайти таку матрицю  $M$ , що перетворить ліве зображення таким чином, щоб його епіполярні лінії співпадали з відповідними епіпо-

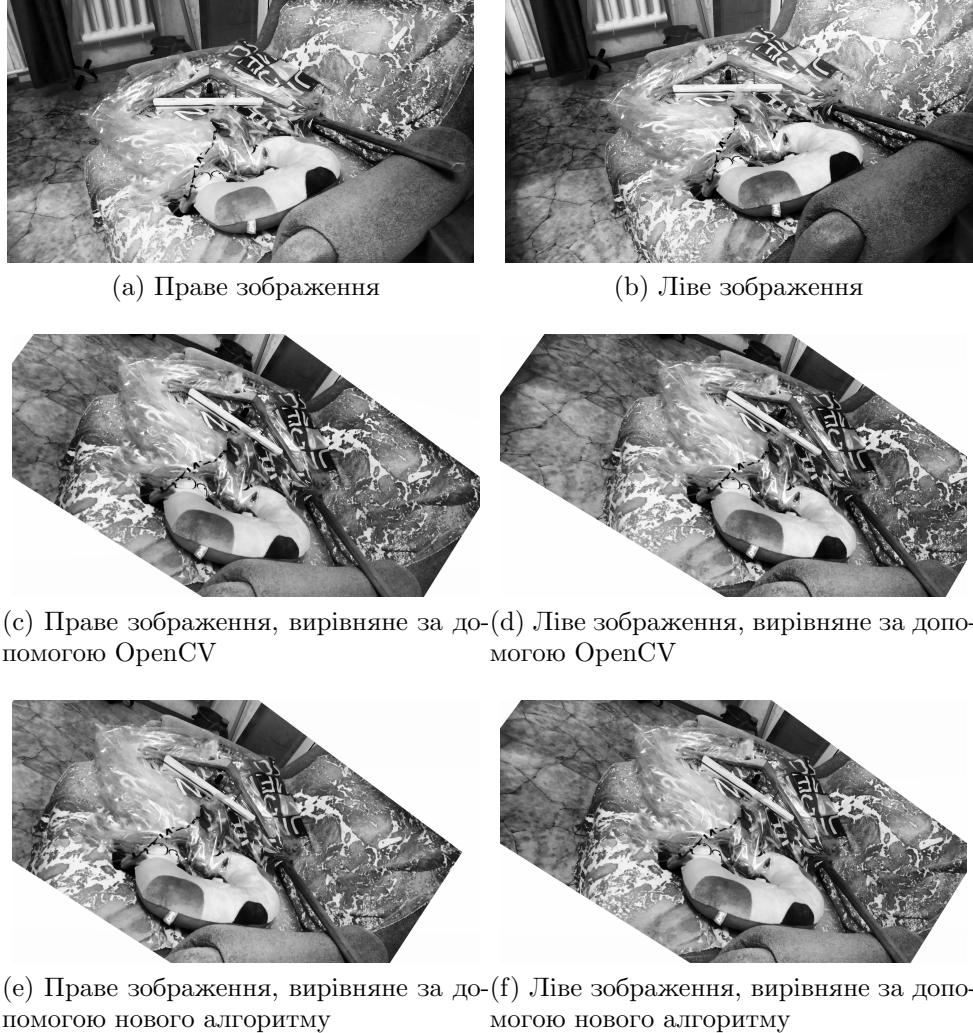


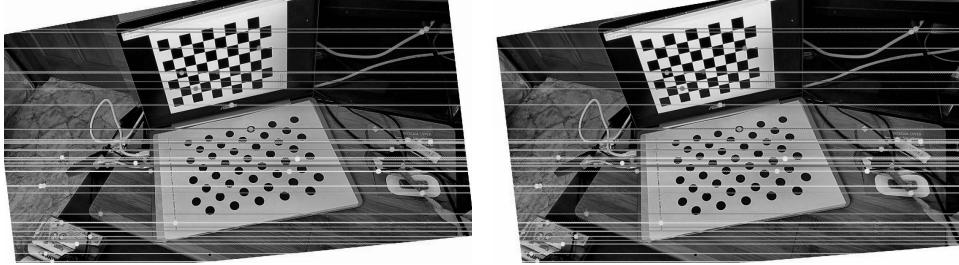
Рис. 1: Результати експериментів

лярними лініями правого зображення. Відомо [5], що фундаментальну матрицю  $F$  можна представити у вигляді

$$F = [\mathbf{e}']_x \cdot S,$$

де матриця  $S$  переносить деяку площину з правого зображення в ліве. Один з методів вилучення такої матриці, що задовольняє наведену рівність, є формула [9]

$$S = [\mathbf{e}']_x \cdot F.$$



(a) Праве зображення, вирівняне за до-(b) Ліве зображення, вирівняне за допомогою нового алгоритму  
допомогою нового алгоритму

Рис. 2: Результати експериментів

Формулу для обчислення  $M$

$$M = R_r \cdot S = G \cdot \begin{bmatrix} ((\mathbf{k} \times (\mathbf{k} \times \mathbf{e}')) \times \mathbf{e}')^T \\ ((\mathbf{k} \times \mathbf{e}') \times \mathbf{e}')^T \\ (\mathbf{k} \times \mathbf{e}')^T \end{bmatrix} \cdot F = G \cdot \begin{bmatrix} (\mathbf{k} \times \mathbf{e}')^T \\ ((\mathbf{k} \times \mathbf{e}') \times \mathbf{e}')^T \\ (\mathbf{k} \times \mathbf{e}')^T \end{bmatrix}^T \cdot F.$$

можна інтерпретувати наступним чином: ми переводимо ліве зображення у простір правого за допомогою  $S$ , і там вирівнюємо його за допомогою матриці  $R_r$ , яка застосовується для вирівнювання правого зображення. Ранг отриманої матриці  $M$  не перевищує 2, адже матриця  $S$  є добутком двох матриць, що мають ранг 2. Нам потрібно отримати з  $M$  матрицю з рангом 3. Для цього ми можемо змінити перший рядок цієї матриці, адже цей рядок впливає лише на координати  $x$  точок, що не впливає на горизонтальність епіпольлярних ліній (яку ми хочемо досягти). Для того, щоб така зміна мала сенс, другий та третій рядки матриці  $M$  повинні бути лінійно незалежними.

Матриця  $M$  в алгоритмі розраховується за формулою

$$M = R_r \cdot S = G \cdot R \cdot [\mathbf{e}']_{\times} \cdot F.$$

Розглянемо її частину  $G \cdot R \cdot [\mathbf{e}']_{\times}$ , а потім повернемося до повної формули. Оскільки

$$(\mathbf{k} \times (\mathbf{k} \times \mathbf{e}')) \times \mathbf{e}' = \mathbf{k} \times \mathbf{e}',$$

маємо

$$G \cdot R \cdot [\mathbf{e}']_{\times} = \begin{bmatrix} (\mathbf{k} \times \mathbf{e}')^T \\ (\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T \\ (\mathbf{k} \times \mathbf{e}')^T \cdot \left( \frac{-e'_z}{x} + 1 \right) \end{bmatrix}.$$

Перший та третій рядок колінеарні за будь-яких умов, отже розглянемо другий та третій. Оскільки за властивістю векторного добутку справедливо  $\mathbf{k} \times (\mathbf{k} \times \mathbf{e}') \perp (\mathbf{k} \times \mathbf{e}')$  для ненульових і не колінеарних  $\mathbf{k}$  і  $\mathbf{e}'$ , нас цікавить лише випадок, коли один з рядків є нульовим вектором. Другий рядок може стати нульовим тільки якщо  $\mathbf{k}$  і  $\mathbf{e}'$  колінеарні. Це означало б, що  $\mathbf{e}'$  знаходиться прямо в центрі зображення. У такому разі зміщення точки  $\mathbf{e}'$  на нескінченність означало б зміщення центру зображення на нескінченність, що дуже сильно деформує зображення, тому випадки, за яких епіпольярна точка знаходиться в межах картинки, не розглядається.

Єдиний граничний випадок, що нас цікавить, це

$$\frac{-e'_z}{x} = -1.$$

Розпишемо  $x$  за рівністю (8)

$$e'_z = (\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T \cdot \mathbf{e}'.$$

Оскільки одиничний вектор  $\mathbf{k}$  лежить на вісі аплікат,  $e'_z = \mathbf{k} \cdot \mathbf{e}'$ . Маємо

$$\mathbf{k}^T \cdot \mathbf{e}' = (\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T \cdot \mathbf{e}'. \quad (9)$$

Позначимо кут між векторами  $\mathbf{e}'$  і  $\mathbf{k}$  літерою  $\alpha$ . Оскільки  $\mathbf{k}$ ,  $\mathbf{e}'$  і  $\mathbf{k} \times (\mathbf{k} \times \mathbf{e}')$  колінеарні й вектор  $\mathbf{k} \times (\mathbf{k} \times \mathbf{e}')$  перпендикулярний до  $\mathbf{k}$ , ми можемо описати  $\mathbf{e}'$  як лінійну комбінацію векторів  $\mathbf{k} \times (\mathbf{k} \times \mathbf{e}')$  і  $\mathbf{k}$

$$\mathbf{e}' = \cos \alpha \cdot \mathbf{k} + \sin \alpha \cdot \mathbf{k} \times (\mathbf{k} \times \mathbf{e}').$$

Оскільки вектор  $\mathbf{e}'$  не колінеарний до  $\mathbf{k}$ , виконується  $\sin \alpha \neq 0$ , що дозволяє поділити на цей синус обидві частини рівності

$$\mathbf{k} \times (\mathbf{k} \times \mathbf{e}') = \frac{\mathbf{e}' - \cos \alpha \cdot \mathbf{k}}{\sin \alpha}.$$

Тоді (9) перетворюється на

$$\mathbf{k}^T \cdot \mathbf{e}' = \frac{\mathbf{e}'^T \cdot \mathbf{e}' - \cos \alpha \cdot \mathbf{k}^T \cdot \mathbf{e}'}{\sin \alpha},$$

з чого слідує

$$(\cos \alpha + \sin \alpha) \cdot \mathbf{k}^T \cdot \mathbf{e}' = \mathbf{e}'^T \cdot \mathbf{e}'.$$

За визначенням скалярного добутку

$$(\cos \alpha + \sin \alpha) \cdot \|\mathbf{k}\| \cdot \|\mathbf{e}'\| \cdot \cos a = \|\mathbf{e}'\|^2$$

Оскільки всі наведені вектори мають довжину 1,

$$(\cos \alpha + \sin \alpha) \cdot \cos \alpha = 1,$$

отже

$$\sin \alpha \cdot (\sin \alpha - \cos \alpha) = 0,$$

з чого слідує, що є наступні розв'язки для  $\alpha$

$$\begin{aligned} \alpha &= \pi \cdot n, \\ \alpha &= \pi \cdot n - \frac{pi}{4}, \quad \forall n \in \mathbb{Z}. \end{aligned}$$

Перший випадок означає, що епіполярна точка  $\mathbf{e}'$  колінеарна вісі апликат, але ми вже вказали, що такі випадки ми не розглядаємо. Отже, проблемними є ті випадки, коли вектор  $\mathbf{e}'$  утворюється кут  $45^\circ$  з одним із векторів  $\mathbf{k}$  або  $\mathbf{k} \times (\mathbf{k} \times \mathbf{e}')$ , а з іншим з них утворює кут  $135^\circ$ . У зв'язку з наявністю шуму метод буде працювати нестабільно й за умов, коли вказані кути близькі до вказаних значень.

Повернемося до повної формули матриці  $M$

$$M = G \cdot R \cdot [\mathbf{e}']_{\times} \cdot F$$

та розкриємо всі позначення

$$M = \begin{bmatrix} (\mathbf{k} \times \mathbf{e}')^T \cdot F \\ (\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'))^T \cdot F \\ (\mathbf{k} \times \mathbf{e}') \cdot F \cdot \left( \frac{-e_z}{x} + 1 \right) \end{bmatrix},$$

де  $F$  — фундаментальна матриця, що має ранг 2. Її ліве ядро має єдиний одиничний вектор  $\mathbf{e}'$  (інші мають іншу норму). Для будь-якого вектору  $\mathbf{x}$  знайдеться такий вектор  $\mathbf{y} \perp \mathbf{e}'$  і число  $c$ , що

$$\mathbf{x} = \mathbf{y} + c \cdot \mathbf{e}'.$$

Оскільки  $\mathbf{e}'$  належить лівому ядру фундаментальної матриці  $F$ , що означає  $c \cdot \mathbf{e}'^T \cdot F = 0$ , справедливим є ланцюг

$$\mathbf{x}^T \cdot F = (\mathbf{y} + c \cdot \mathbf{e}')^T \cdot F = \mathbf{y}^T \cdot F + c \cdot \mathbf{e}'^T \cdot F = \mathbf{y}^T \cdot F,$$

Отже, для будь-яких тривимірних векторів  $\mathbf{x}$  і  $\mathbf{y}$

$$\mathbf{x}^T \cdot F = \mathbf{y}^T \cdot F \iff \exists c \in \mathbb{R} : \mathbf{x} = \mathbf{y} + c \cdot \mathbf{e}', \mathbf{e}' \perp \mathbf{y}.$$

У попередньому аналізі ми не розглядали вплив матриці  $F$  на результат. Щоб матриця  $F$  могла впливати на стабільність алгоритму, потрібно, щоб она мала можливість робити другий та третій рядок колінеарними, що в свою чергу означало б

$$\mathbf{k} \times \mathbf{e}' = \mathbf{k} \times (\mathbf{k} \times \mathbf{e}') + c \cdot \mathbf{e}'.$$

що неможливо, бо  $\mathbf{k} \times \mathbf{e}' \perp \mathbf{k} \times (\mathbf{k} \times \mathbf{e}')$  і  $\mathbf{k} \times \mathbf{e}' \perp \mathbf{e}'$ , якщо  $\mathbf{e}' \neq \mathbf{k}$ .

Робимо висновок, що запропонований алгоритм не може працювати у випадках, що близькі до одного з наступних трьох

$$\begin{aligned} \angle(\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'), \mathbf{e}') &= 135^\circ \text{ \& } \angle(\mathbf{k}, \mathbf{e}') = 45^\circ; \\ \angle(\mathbf{k} \times (\mathbf{k} \times \mathbf{e}'), \mathbf{e}') &= 45^\circ \text{ \& } \angle(\mathbf{k}, \mathbf{e}') = 135^\circ; \\ \angle(\mathbf{k}, \mathbf{e}') &= 0^\circ, \end{aligned} \tag{10}$$

за умови, що центр зображення знаходиться у  $\mathbf{k}$ .

## 2.4 Висновки

Наведений алгоритм вирівнювання стереопарі за допомогою векторного добутку та фундаментальної матриці є простим у реалізації й ефективним у використання. Проте він не є універсальним: він працюватиме некоректно, коли базисний вектор  $\mathbf{k}$  утворює кут  $0^\circ$ ,  $45^\circ$  або  $135^\circ$  з епіполярною точкою (10). Як ми бачимо, це досить специфічні випадки, проте вони можуть порушити роботу алгоритму.

## Список литературы

- [1] Herbert Bay и др. “Speeded-Up Robust Features (SURF)”. B: *Comput. Vis. Image Underst.* 110.3 (июнь 2008), 346–359. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2007.09.014. URL: <https://doi.org/10.1016/j.cviu.2007.09.014>.
- [2] G. Bradski. “The OpenCV Library”. B: *Dr. Dobb’s Journal of Software Tools* (2000).
- [3] Daniel R. Grayson и Michael E. Stillman. *Macaulay2, a software system for research in algebraic geometry*. Available at <http://www.math.uiuc.edu/Macaulay2/>.
- [4] R. I. Hartley и A. Zisserman. *Multiple View Geometry in Computer Vision*. Second. Cambridge University Press, ISBN: 0521540518, 2004.
- [5] Richard I. Hartley. “Theory and Practice of Projective Rectification”. B: *Int. J. Comput. Vis.* 35.2 (1999), c. 115—127. DOI: 10.1023/A:1008115206617. URL: <https://doi.org/10.1023/A:1008115206617>.
- [6] Wolfram Research, Inc. *Mathematica, Version 12.1*. Champaign, IL, 2020. URL: <https://www.wolfram.com/mathematica>.
- [7] Z. Kukelova, M. Bujnak и T. Pajdla. “Polynomial Eigenvalue Solutions to the 5-pt and 6-pt Relative Pose Problems”. B: *Proceedings of the British Machine Vision Conference*. doi:10.5244/C.22.56. BMVA Press, 2008, c. 56.1—56.10. ISBN: 1-901725-36-7.
- [8] David G. Lowe. “Object Recognition from Local Scale-Invariant Features”. B: *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*. ICCV ’99. USA: IEEE Computer Society, 1999, c. 1150. ISBN: 0769501648.
- [9] Q.-T. Luong и T. Viéville. “Canonical Representations for the Geometries of Multiple Projective Views”. B: *Computer Vision and Image Understanding* 64.2 (1996), c. 193 —229. ISSN: 1077-3142. DOI: <https://doi.org/10.1006/cviu.1996.0055>. URL: <http://www.sciencedirect.com/science/article/pii/S1077314296900557>.
- [10] John Mallon и Paul F. Whelan. “Projective rectification from the fundamental matrix”. B: *Image and Vision Computing* 23.7 (2005), c. 643 —650. ISSN: 0262-8856. DOI: <https://doi.org/10.1016/j.imavis.2005.03.002>. URL: <http://www.sciencedirect.com/science/article/pii/S0262885605000399>.

- [11] Aaron Meurer и др. “SymPy: symbolic computing in Python”. B: *PeerJ Computer Science* 3 (янв. 2017), e103. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.103. URL: <https://doi.org/10.7717/peerj-cs.103>.
- [12] David Nister. “An efficient solution to the five-point relative pose problem”. B: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.6 (2004), с. 756—770.
- [13] Ethan Rublee и др. “ORB: An Efficient Alternative to SIFT or SURF”. B: *International Conference on Computer Vision*. Barcelona, 2011.
- [14] Richard Szeliski. *Computer vision algorithms and applications*. 2011. URL: <http://dx.doi.org/10.1007/978-1-84882-935-0>.
- [15] O. Monastyrskyi V. Krygin. “A compact form of a cubic equation from the 7-point algorithm”. B: «Теоретичні і прикладні проблеми фізики, математики та інформатики», XV Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених (2020), с. 190—191.
- [16] V. Nechaieva V. Krygin. “Plausible Rectification from the Fundamental Matrix”. B: «Теоретичні і прикладні проблеми фізики, математики та інформатики», XV Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених (2020), с. 199—202.