Реализация и исследование 8-точечного алгоритма оценивания существенной матрицы

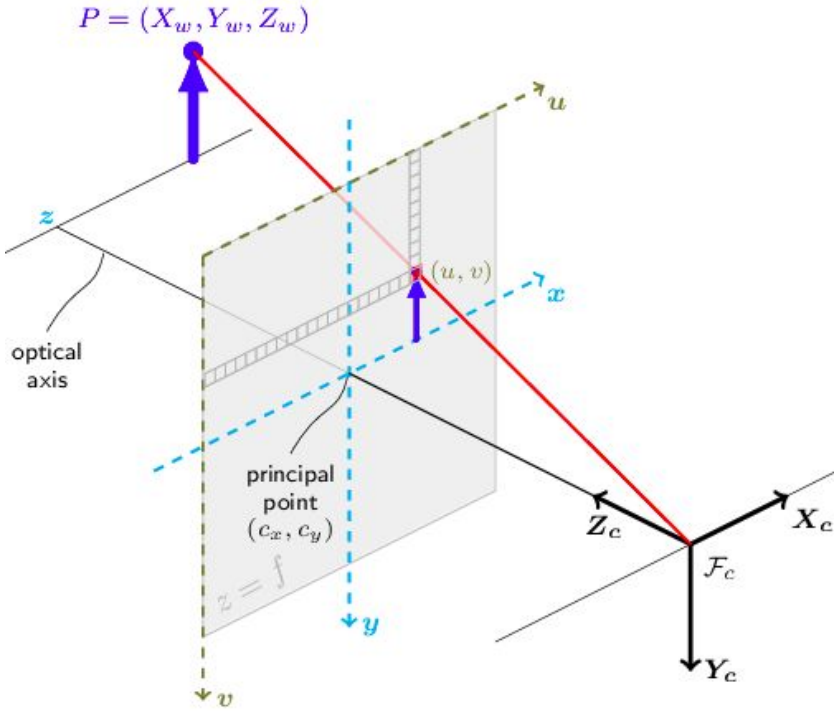
Команда “Восприятие”
Дмитрий Клёстов
Иван Карелин
Сергей Фамуляк
Данила Бочарников

Постановка задачи

- Требуется оценить взаимное положение двух камер в трёхмерном мире
- Это преобразование моделируется существенной матрицей
- 8 точечный алгоритм позволяет оценить существенную матрицу с хорошей точностью
- Приложения: компьютерное зрение, робототехника, виртуальная/дополненная реальность
- Метрика качества - норма вектора ошибки

Описание метода

Используем соотношения из проективной геометрии



Описание метода

Используем соотношения из проективной геометрии

Координаты точки в кадре

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}.$$

Координаты точки в трёхмерном мире

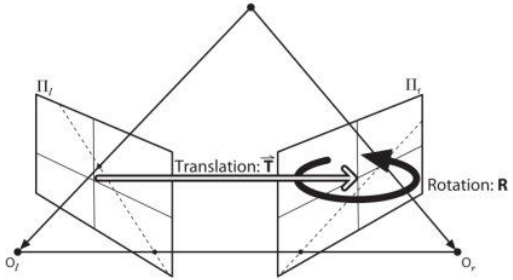
матрица параметров камеры
K

Описание метода

$\lambda_1 x_1 = X \; ; \; \lambda_2 x_2 = RX + T$ $\quad x_1, \; x_2 \; -$ обратные проекции

$$\lambda_2 x_2 = R(\lambda_1 x_1) + T$$

$$\lambda_2 \widehat{T} x_2 = \lambda_1 \widehat{T} R x_1$$



$$\widehat{T} = \begin{pmatrix} 0 & -T_3 & T_2 \\ T_3 & 0 & -T_1 \\ -T_2 & T_1 & 0 \end{pmatrix}$$

Умножим слева и справа на x_2^T

$$\lambda_1 x_2^T \widehat{T} R x_1 = 0$$

$$x_2^T \widehat{T} R x_1 = 0$$

epipolar constraint

$E = \widehat{T}R \in \mathbb{R}^{3x3}$ – существенная матрица

Описание метода

$Epipolar\ constraint\ x_2^T E x_1 = 0$ означает, что векторы

$$x_1,\ x_2 \text{ и } T$$

лежат в одной плоскости. Следовательно $rank(E) = 2$

$$SVD:\ E = U\Sigma V^T;\quad \Sigma = diag\{\sigma,\ \sigma,\ 0\}$$

Из существенной матрицы восстанавливаются $R,\ T$:

$$R = UR_z\left(\pm\frac{\pi}{2}\right)V^T\ ;\ T = UR_z\left(\pm\frac{\pi}{2}\right)\Sigma U^T$$

Описание метода

Как найти существенную матрицу E

Поставим задачу линейно:

- Перепишем E в виде вектора

$$E^{est} = (e_{11}, e_{12}, e_{13}, e_{21}, e_{22}, e_{23}, e_{31}, e_{32}, e_{33})^T \in \mathbb{R}^9$$

- Введём комбинацию векторов x1 и x2

$$a = (x_1x_2, x_1y_2, x_1z_2, y_1x_2, y_1y_2, y_1z_2, z_1x_2, z_1y_2, z_1z_2)^T \in \mathbb{R}^9$$

- Тогда *epipolar constraint*

$$x_2^T E x_1 = a^T E^{est} = 0 \qquad \chi E^{est} = 0, \text{ где } \chi = \left(a^1, a^2, \ldots, a^n\right)^T$$

Описание метода

Решение полученной линейной однородной системы E^{est} не обязано удовлетворять свойствам существенной матрицы и скорее всего не будет. Поэтому будем искать ближайшую к нему (например, по норме Фробениуса) существенную матрицу

$$E^{est} = U \cdot diag\{\sigma_1, \sigma_2, \sigma_3\} V^T$$
$$\sigma_1 \geq \sigma_2 \geq \sigma_3$$

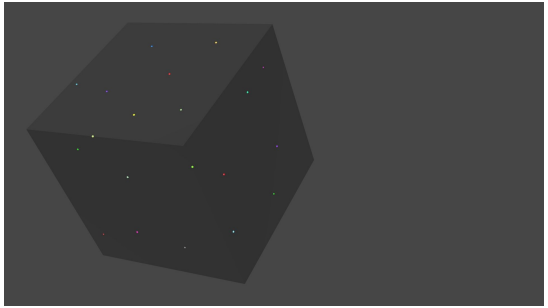
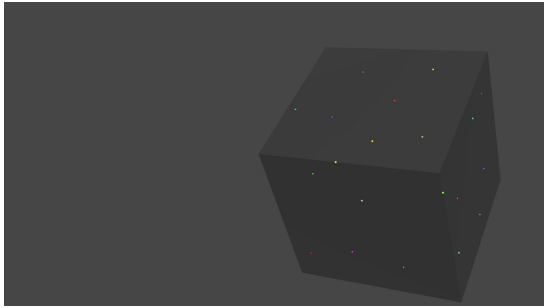
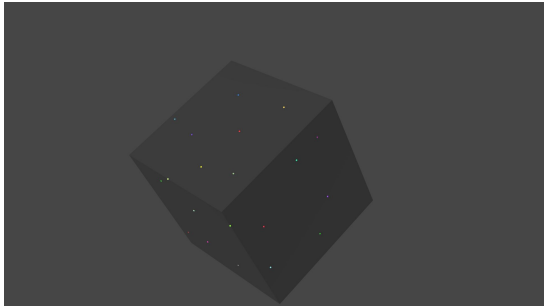
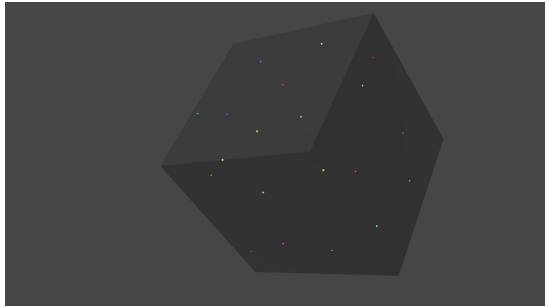
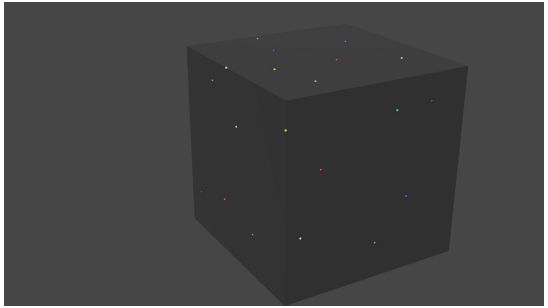
$$E = U \cdot diag\{\sigma, \sigma, 0\} V^T$$
$$\sigma = \frac{\sigma_1 + \sigma_2}{2}$$

т.к. E определена с точностью до умножения на константу,
можем использовать $\sigma = 1$

После нахождения существенной матрицы, её можно разложить на матрицу поворота и вектор переноса

Тестирование

Моделируем разные положения камер в среде blender

Тестирование

Небольшой набор данных (20 точек)

Оценка алгоритма - среднеквадратичное отклонение от ground truth $\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m}(a_{i,j}-b_{ij})^2/(n*m)}$

Оценка алгоритма:

mean с типом float (предполагается субпиксельная точность детекции точек):

cv- 5.413711608621522e-07, hand_essential - 3.8145220820578494e-07

При переводе к int (пиксельная точность детекции точек):

cv- 0.008322426910311598, hand_essential - 0.0032902498499065257

Сравнение устойчивости алгоритма к выбросам

Ошибка - среднеквадратичное отклонение от ground truth, при возмущении 15 процентов особых точек.

Результаты с помощью OpenCV

mean = 0.008661784819192447

std = 0.0048803357327973276

Результаты с помощью hand_essential

mean = 0.1341490759325452

std = 0.019371460987091374

Оценка алгоритма

Реализация	Среднеквадратичное отклонение от ground truth (float)	Среднеквадратичное отклонение от ground truth (int)
наша	3.81e-07	0.003
OpenCV	5.41e-07	0.008

Устойчивость к выбросам

Реализация	Mean среднее квадратическое отклонение от ground truth	Std среднеквадратичное отклонение от ground truth
наша	0.134	0.019
OpenCV	0.008	0.005

Скорость работы

Большие наборы данных (1e3 - 1e5 точек)

# точек	Время работы
1e3 точек	0.02 с
1e4 точек	4.3 с
1e5 точек	не вариант

Зависимость от реализации SVD

Реализация SVD	Среднеквадратичное отклонение от ground truth
QR based	3.81e-07
Якоби	3.83e-07

Выводы

- Основные результаты проекта
  Реализовали алгоритм, получили приемлемую в практических приложениях точность;
- Что планировалось
  оценить точность и сложность 8-точечного алгоритма
- Что получилось
  Оценили точность на малых данных, устойчивость к выбросам на малых наборах, время исполнения на больших наборах данных
- Как хотелось бы улучшить проект
  добавить устойчивость к шумам и выбросам, протестировать на устойчивость к выбросам на больших наборах данных

Вклад участников

- **Дмитрий Клёстов**: реализация алгоритма, создание тестовых данных
- **Иван Карелин**: написание алгоритма, сравнение точности с другими реализациями, тестирование
- **Сергей Фамуляк**: поиск различных реализаций алгоритма, работа с математическими источниками, оценка быстродействия
- **Данила Бочарников**: алгоритм, сравнение SVD реализаций, тестирование

Все вместе: brain storming, выбор и обсуждение задачи, поиск источников

Ссылка на проект https://github.com/biquaternion/nla_2023_essential_matrix