Точечные особенности изображения и их отслеживание

Сергей Кривохатский

СПбГУ, Современное программирование

16 сентября 2022 г.

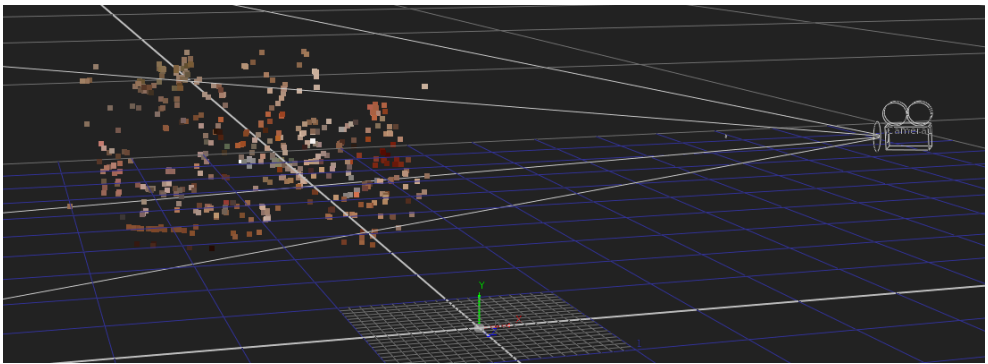
Трекинг камеры / Structure from Motion I

Входные данные — набор изображений (кадров) одной и той же сцены, снятой с разных ракурсов

Трекинг камеры / Structure from Motion II

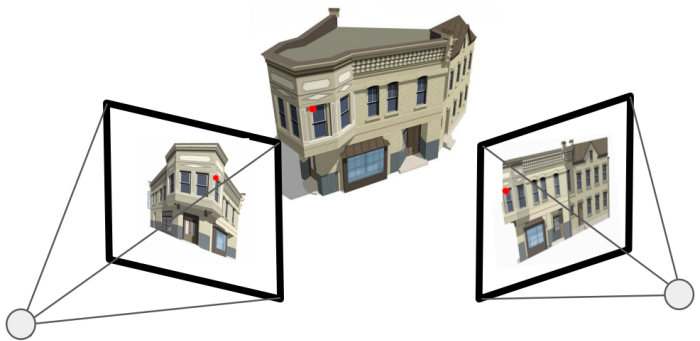
Выходные данные — 3D-структура сцены

- позиции камер, с которых сняты изображения
- облако точек — набор 3D-координат некоторых точек сцены



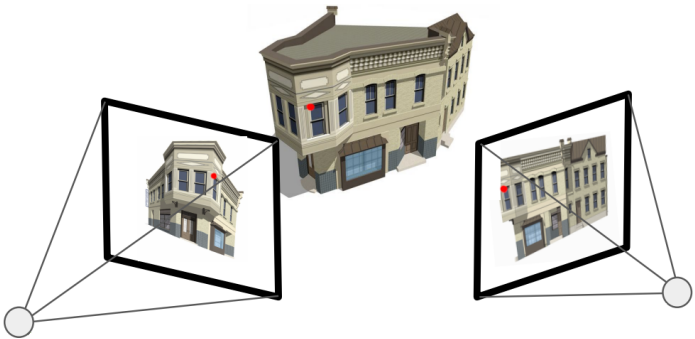
Что происходит при съемке?

- 3D-точки, попадающие в объектив, проецируются в 2D-точки на кадре
- одна и та же 3D-точка попадает в разные 2D-позиции на разных кадрах



Как начать восстанавливать 3D-информацию?

- нужно найти соответствия между 2D-точками, являющимися проекциями одной и той же 3D-точки
- для всех точек это сделать сложно, поэтому ограничимся хорошо отличимыми *особенными* точками



Локальные точечные особенности

Локальная точечная особенность или *ключевая точка* — точка изображения (вместе с небольшой окрестностью), заметно отличающаяся от других

Какие ключевые точки нужны нам? I

Возможны два варианта

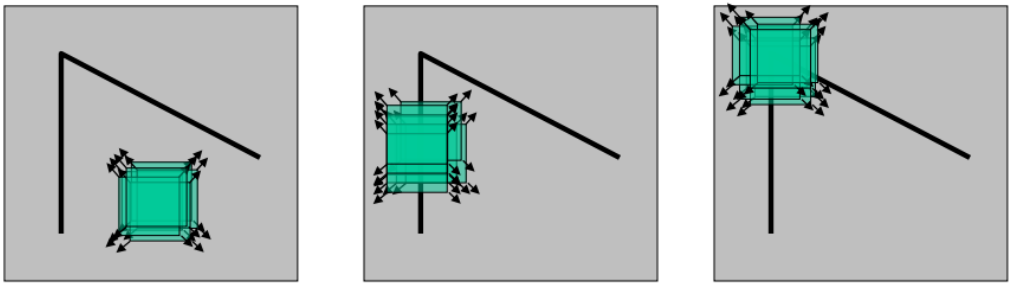
- кадры сняты независимо друг от друга
  - никаких специальных предположений сделать нельзя
- видео — кадры сняты последовательно
  - можно предполагать, что происходят только небольшие постепенные изменения

В этот раз речь пойдет о видео. А значит

- подойдут точки, перемещение которых легко заметить
- особенностям не обязательно выглядеть уникально

Какие ключевые точки нужны нам? II

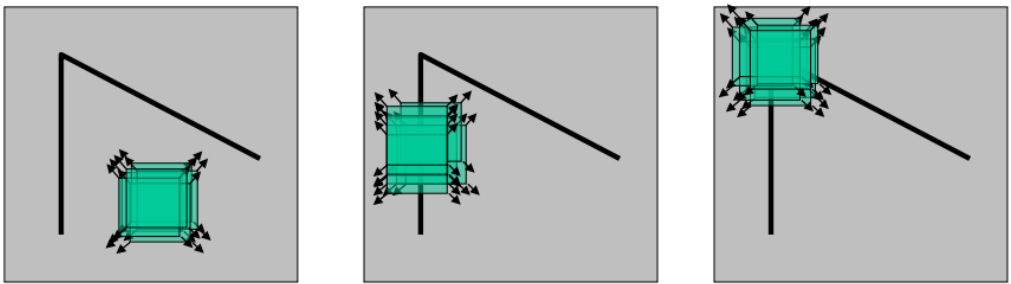
Зафиксируем прямоугольное «окно» на плоскости изображения и мысленно подвигаем изображение под ним



(или подвигаем окно над изображением в разные стороны)

Какие ключевые точки нужны нам? III

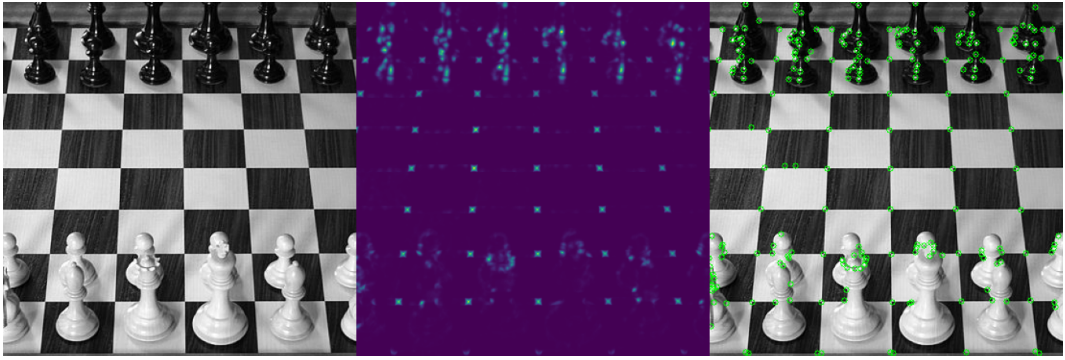
Если при сдвиге в любую сторону содержимое окна заметно меняется, движение такой особенности будет легко отследить



Подобные особенности называют *уголками*

Идея алгоритма детектирования уголков

- посчитаем, насколько похож на уголок каждый пиксель
- выберем лучшие точки



Подсчет качества уголка

Возьмем окно вокруг точки изображения и еще одно с небольшим сдвигом, посмотрим на разницу

$$d(W, \delta) := \sum_{p \in W} (I(p) - I(p + \delta))^2$$

где

- $I : \mathbb{R}^2 \to \mathbb{R}$ — изображение
- W — множество координат пикселей окна
- $\delta = \begin{pmatrix} \delta_x & \delta_y \end{pmatrix}^\mathsf{T}$ — сдвиг

Наивный подсчет качества

Промоделируем перемещение окна во все стороны

1. Определим множество сдвигов, например $\Delta \coloneqq \{\begin{pmatrix}\delta_x & \delta_y\end{pmatrix}^\mathsf{T} \mid \delta_x, \delta_y \in \{-2, -1, \ldots, 2\}\}$
2. Для каждого $\delta \in \Delta$ посчитаем $d(W, \delta)$
3. Как-то проанализируем полученные разницы и решим, уголок это или нет

Проблемы

- как правильно выполнять шаг 3?
- время работы — $\Omega(|W| \cdot |\Delta|)$

Эффективный подсчет качества I

$$I(p+\delta) = I(p) + \delta^{\mathsf{T}} \nabla I(p) + \cdots \implies$$

$$\begin{aligned} d(W,\delta) &= \sum_{p \in W} (I(p) - I(p+\delta))^2 \approx \\ &\approx \sum_{p \in W} (I(p) - (I(p) + \delta^{\mathsf{T}} \nabla I(p)))^2 = \\ &= \sum_{p \in W} (-\delta^{\mathsf{T}} \nabla I(p))^2 \end{aligned}$$

Эффективный подсчет качества II

$$d(W, \delta) \approx \sum_{p \in W} (\delta^\intercal \nabla I(p))^2 = \delta^\intercal A^\intercal A \delta = \delta^\intercal M \delta$$

где

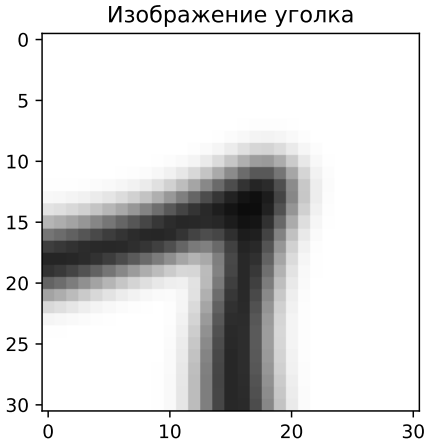
$$A = \begin{pmatrix} I'_x(p_1) & I'_y(p_1) \\ I'_x(p_2) & I'_y(p_2) \\ \dots & \\ I'_x(p_n) & I'_y(p_n) \end{pmatrix} \qquad M = \sum_{p \in W} \begin{pmatrix} I'_x(p)^2 & I'_x(p) I'_y(p) \\ I'_x(p) I'_y(p) & I'_y(p)^2 \end{pmatrix}$$

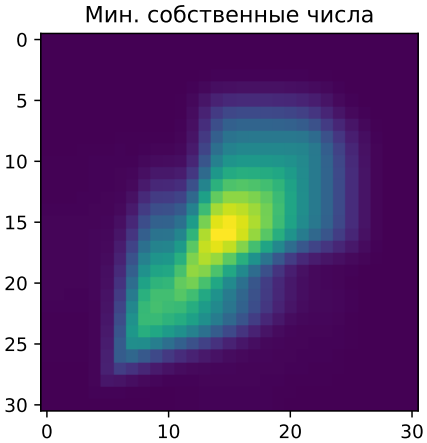
Эффективный подсчет качества III

- получаем квадратичную форму $\delta^{\intercal} M \delta$
- рассмотрим собственные числа λ_1 и λ_2 матрицы M
- каждое число характеризует силу изменения участка под окном W в направлении соответствующего собственного вектора
- в качестве качества уголка возьмем $\min\{\lambda_1, \lambda_2\}$

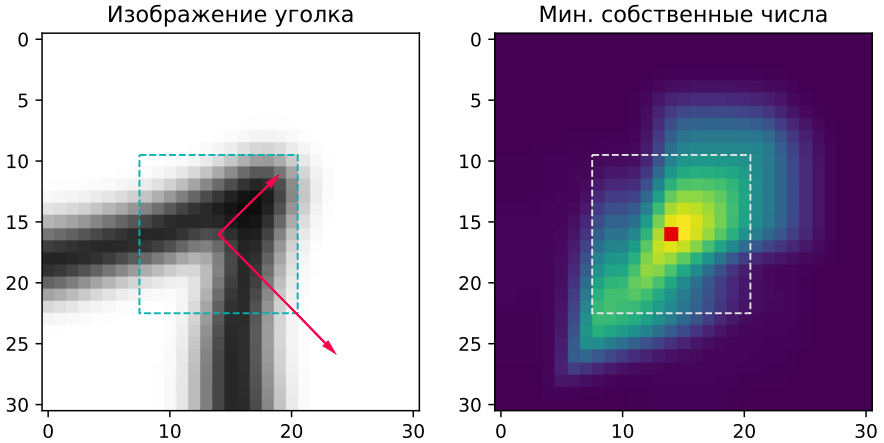
$$\text{Сложность вычислений} — \Theta(|W|)$$

Эффективный подсчет качества. Пример I



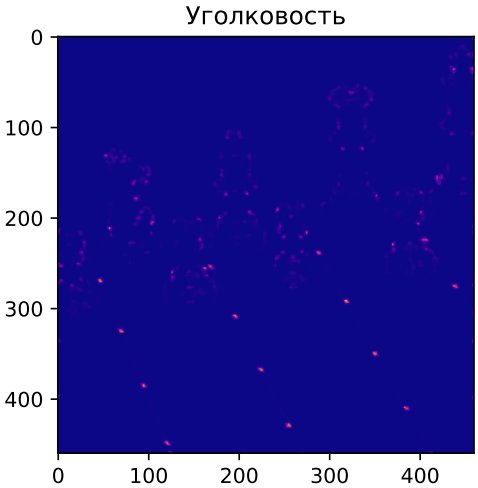


Эффективный подсчет качества. Пример II



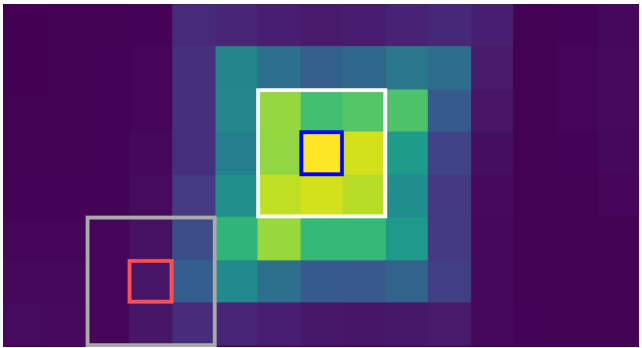
Длина стрелки пропорциональна величине собственного числа

Эффективный подсчет качества. Пример III



Non-maximum suppression

- чаще всего высокие значения качества уголков сгруппированы в небольшие участки
- такой участок скорее всего соответствует одному уголку
- будем брать только локальные максимумы



Детектор уголков Ши — Томаси

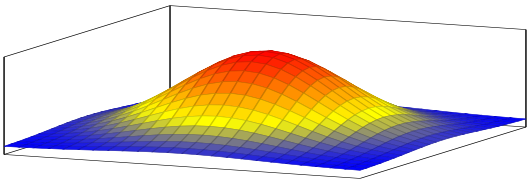
- зафиксируем размер окна
- для каждого пикселя посчитаем его качество
- в полученной матрице качества выделим локальные максимумы
- возьмем те максимумы, значение которых выше порогового

Детектор уголков — детали I

- На практике разницы между пикселями окна могут учитываться с различными весами

$$d(W, \delta) := \sum_{p \in W} w(p)(I(p) - I(p + \delta))^2$$

где $w(p)$ — функция веса. Например, Гауссова



Детектор уголков — детали II

Подсчет матрицы M может быть эффективно реализован с помощью сверток

$$\begin{pmatrix} G_\sigma * I_x^2 & G_\sigma * (I_x I_y) \\ G_\sigma * (I_x I_y) & G_\sigma * I_y^2 \end{pmatrix}$$

где

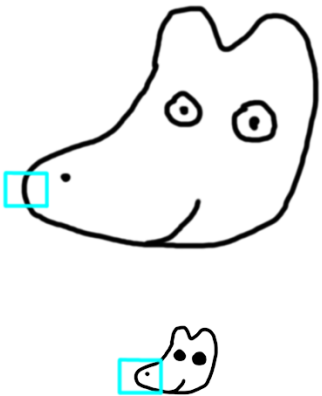
- $I_x = S_x * I$ — свертка с ядром фильтра Собеля по x
- $I_y = S_y * I$ — свертка с ядром фильтра Собеля по y
- G_σ — Гауссово ядро

Детектор уголков — масштаб

Масштаб имеет значение

- Маленькое окно
  - \+ Больше точек
  - \+ Быстрее работает
  - – Упускаем крупные детали
- Большое окно
  - \+ Ловим крупные детали
  - – Меньше точек
  - – Медленнее работает
  - – Лишние детали в окне

Ищем уголки на нескольких масштабах



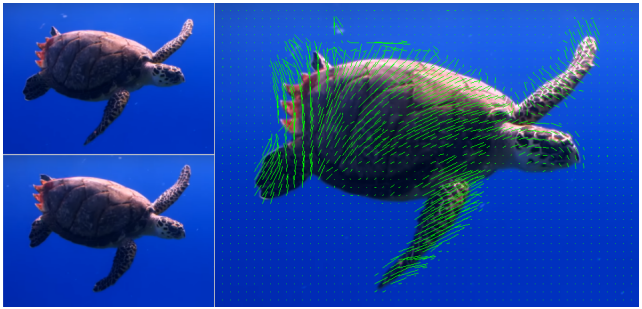
Пирамиды изображений

1. $I_0 = I$ — исходное изображение
2. I_1 — уменьшенный I_0
   - размыть по Гауссу
   - выбросить половину столбцов и половину строк
3. ...

$$1 + \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \cdots = \frac{4}{3}$$

Отслеживание уголков — оптический поток I

- Для каждого уголка текущего кадра необходимо определить, куда он переместится на следующем
- *Оптический поток* — характеристика видимого движения объектов сцены



Отслеживание уголков — оптический поток II

- Для каждого уголка текущего кадра необходимо определить, куда он переместится на следующем
- Предполагаем, что от кадра к кадру картинка меняется не очень сильно
- Делаем следующие допущения
  - точки в окрестности уголка (в окне) смещаются одинаково
  - яркость точек уголка не меняется со временем (практически)
  - смещение уголка невелико

Отслеживание уголка

Исходя из допущений будем искать минимум $f(v)$

$$f(v) := \sum_{p \in W} \left(J(p+v) - I(p)\right)^2$$

где

- I — текущий кадр, а J — следующий
- $\{p_1, \ldots, p_n\} \equiv W$ — окно вокруг уголка
- $v = \begin{pmatrix} v_x & v_y \end{pmatrix}^\mathsf{T}$ — искомое смещение уголка

Отслеживание уголка — упрощаем выражение

$$J(p+v) = J(p) + v^{\mathsf{T}}\nabla J(p) + \cdots \implies$$

$$\begin{aligned} f(v) &= \sum_{p\in W} \big(J(p+v) - I(p)\big)^2 \approx \\ &\approx \sum_{p\in W} \big(J(p) + v^{\mathsf{T}}\nabla J(p) - I(p)\big)^2 = \\ &= \sum_{p\in W} \big(v^{\mathsf{T}}\nabla J(p) - (I(p) - J(p))\big)^2 \end{aligned}$$

Отслеживание уголка — СЛАУ I

$$f(v) \approx \sum_{p \in W} \left(v^\intercal \nabla J(p) - (I(p) - J(p))\right)^2 =$$
$$= (Av - b)^\intercal (Av - b) =: \tilde{f}(v)$$

где

$$A = \begin{pmatrix} J'_x(p_1) & J'_y(p_1) \\ \dots & \\ J'_x(p_n) & J'_y(p_n) \end{pmatrix} \qquad v = \begin{pmatrix} v_x \\ v_y \end{pmatrix} \qquad b = \begin{pmatrix} I(p_1) - J(p_1) \\ \dots \\ I(p_n) - J(p_n) \end{pmatrix}$$

Отслеживание уголка — минимизация I

- Для отслеживания уголка минимизируем сумму квадратов

$$(Av - b)^\mathsf{T}(Av - b) \to \min$$

- Можно добавить веса, если хочется, чтобы пиксели окна влияли по-разному

$$(Av - b)^\mathsf{T} \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_n \end{pmatrix} (Av - b) \to \min$$

Отслеживание уголка — минимизация II

- Распишем минимизируемое выражение

$$\tilde{f}(v) = (Av - b)^\mathsf{T}(Av - b) = v^\mathsf{T}(A^\mathsf{T}A)v - 2b^\mathsf{T}Av + b^\mathsf{T}b$$

- Продифференцируем: $\nabla\tilde{f}(v) = 2A^\mathsf{T}Av - 2A^\mathsf{T}b$
- Приравняем $\nabla\tilde{f}(v)$ к нулю: $A^\mathsf{T}Av = A^\mathsf{T}b$
- Решение этой системы — минимум, т. к. $x^\mathsf{T}A^\mathsf{T}Ax = (Ax)^2$
- Матрица того же вида, что в детекторе уголков!

$$A^\mathsf{T}A = \sum_{p \in W} \begin{pmatrix} J'_x(p)^2 & J'_x(p)J'_y(p) \\ J'_x(p)J'_y(p) & J'_y(p)^2 \end{pmatrix}$$

Метод Лукаса — Канаде

- Из-за приближения получим не идеальное смещение
- Для уточнения метод применяется итеративно
  1. вычислить смещение v
  2. если оно близко к нулю, закончить алгоритм
  3. иначе подвинуть следующий кадр J на $-v$ и перейти к пункту 1

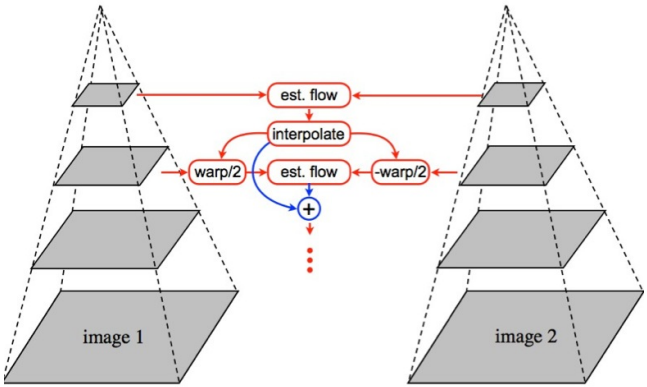
$$J_{i+1}(p) \coloneqq J_i(p + v)$$

где i — номер итерации, $J_0 \coloneqq J$

- При реализации важно проводить вычисления с субпиксельной точностью — интерполировать изображение

Метод Лукаса — Канаде — пирамиды

- Метод не работает для очень больших смещений
- Решить проблему можно с помощью пирамид



Спасибо за внимание!