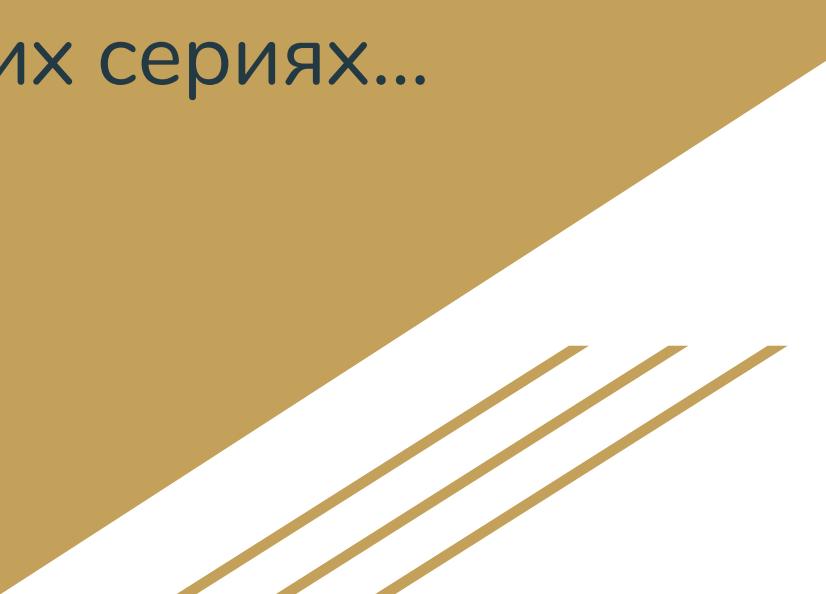
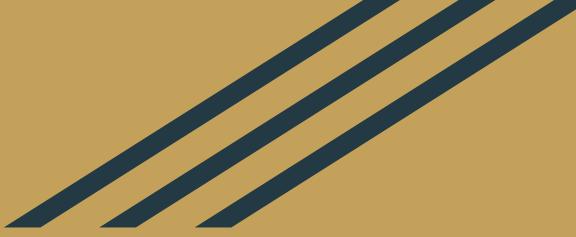


# Компьютерное зрение

## Лекция 4 Optical Flow, 3D зрение.

04.06.2020  
Руслан Алиев



В предыдущих сериях...

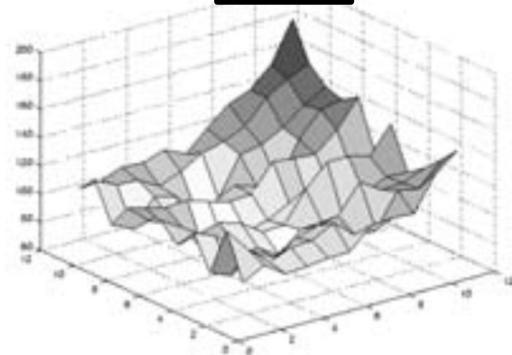
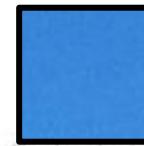
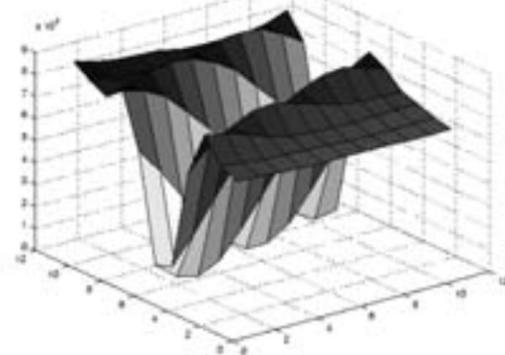
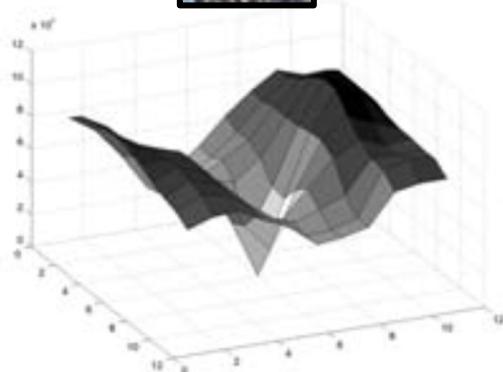
# Мера непохожести

---

Небо: везде низкая

Край: низкая вдоль границы

Угол: везде высокая



## Мера непохожести

---

- $\sum_d \sum_{x,y} (I(x,y) - I(x+d_x, y+d_y))^2$
- Все еще много вычислений
- Нужна аппроксимация
  - подробные математические выкладки - Szeliski, стр. 212

# Structure matrix

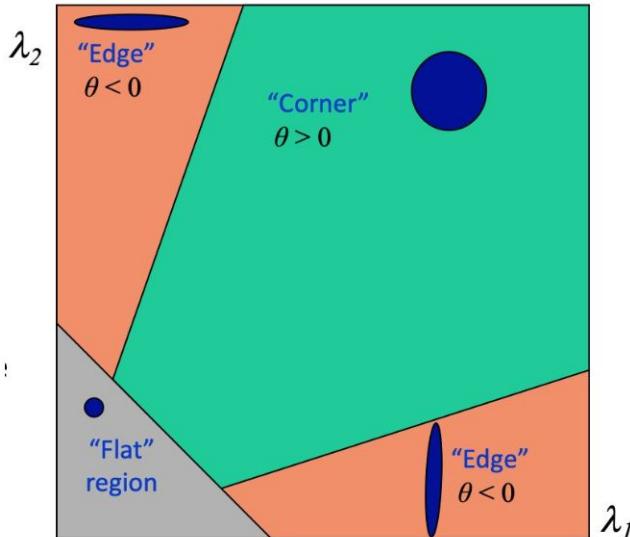
---

- Взвешенная сумма градиентов соседей
  - $\begin{vmatrix} \sum_i w_i |x(i)|_x(i) & \sum_i w_i |x(i)|_y(i) \\ \sum_i w_i |x(i)|_y(i) & \sum_i w_i |y(i)|_y(i) \end{vmatrix}$
  - $\begin{vmatrix} \sum_i w_i |y(i)|_x(i) & \sum_i w_i |y(i)|_y(i) \\ \sum_i w_i |y(i)|_y(i) & \sum_i w_i |x(i)|_y(i) \end{vmatrix}$
- Можем использовать Гауссиану (опять)
- Собственные числа этой матрицы описывают поведение градиентов в округе
- $\lambda_1$  и  $\lambda_2$  - собственные числа
  - $\lambda_1$  и  $\lambda_2$  оба маленькие: flat region
  - $\lambda_1 \gg \lambda_2$ : граница
  - $\lambda_1$  и  $\lambda_2$  похожи: угол

# ФУНКЦИЯ ОТКЛИКА УГЛА

---

- Несколько методов:
  - $\det(S) = \lambda_1 * \lambda_2$
  - $\text{trace}(S) = \lambda_1 + \lambda_2$
- $\Theta = \det(S) - \alpha \text{trace}(S)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$



# Детектор углов Харриса

---

- Вычислить производные  $I_x$  и  $I_y$
- Вычислить 3 вспомогательные матрицы  $\mathbf{I}_{xx}$ ,  $\mathbf{I}_{yy}$ ,  $\mathbf{I}_{xy}$
- Посчитать взвешенную сумму
  - Взвешенная?
  - Фильтр Гаусса!
- Посчитать функцию отклика угла
- Non-max suppression (как в детекторе Кэнни)

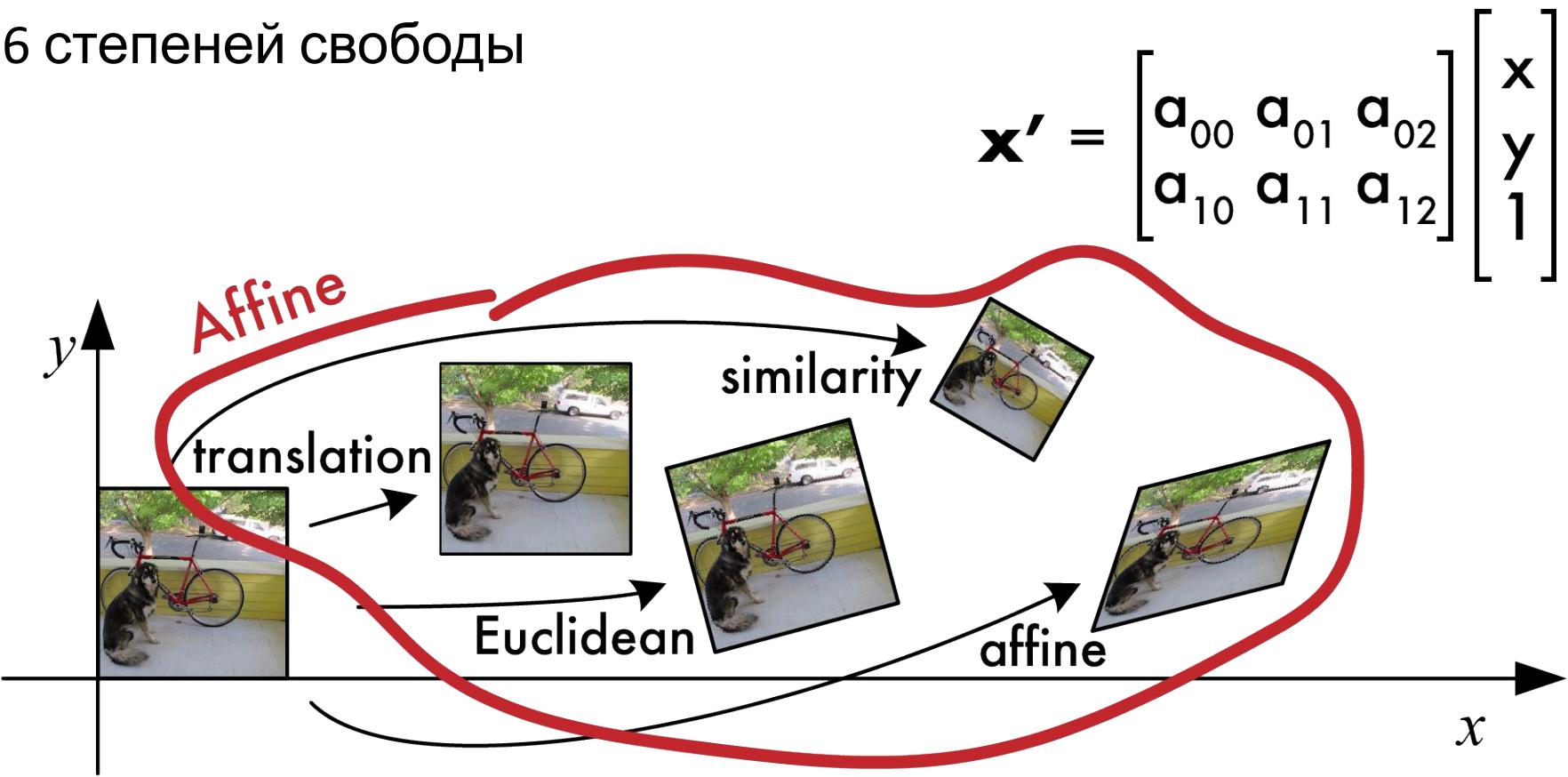
# Дескрипторы

---

- Мы хотим как-то описывать регионы изображения
- Можно просто брать значения пикселей в этом регионе
- Матчить просто, метрика расстояния:
  - $\sum_{x,y} (I(x,y) - J(x,y))^2$
  - Какие тут возникают проблемы?
- Дескрипторы могут быть сложнее
  - Использовать градиенты
  - Нормализовывать
  - HoG и SIFT

# Аффинное: масштабирование, поворот, сдвиг, искажение

6 степеней свободы



# Композиция - тоже аффинное преобразование

---

Допустим мы хотим сдвинуть, затем повернуть, затем снова сдвинуть, затем масштабировать

$$\mathbf{x}' = \mathbf{S} \mathbf{t} \mathbf{R} \mathbf{t} \bar{\mathbf{x}} = \mathbf{M} \bar{\mathbf{x}},$$

Если  $\mathbf{M} = (\mathbf{S} \mathbf{t} \mathbf{R} \mathbf{t})$

$\mathbf{M}$  - это все еще аффинное преобразование

Но это матрицы  $2 \times 3$ , как мы можем их перемножать?

## Добавим строку к матрицам

---

$$\bar{\mathbf{x}}' = \begin{bmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

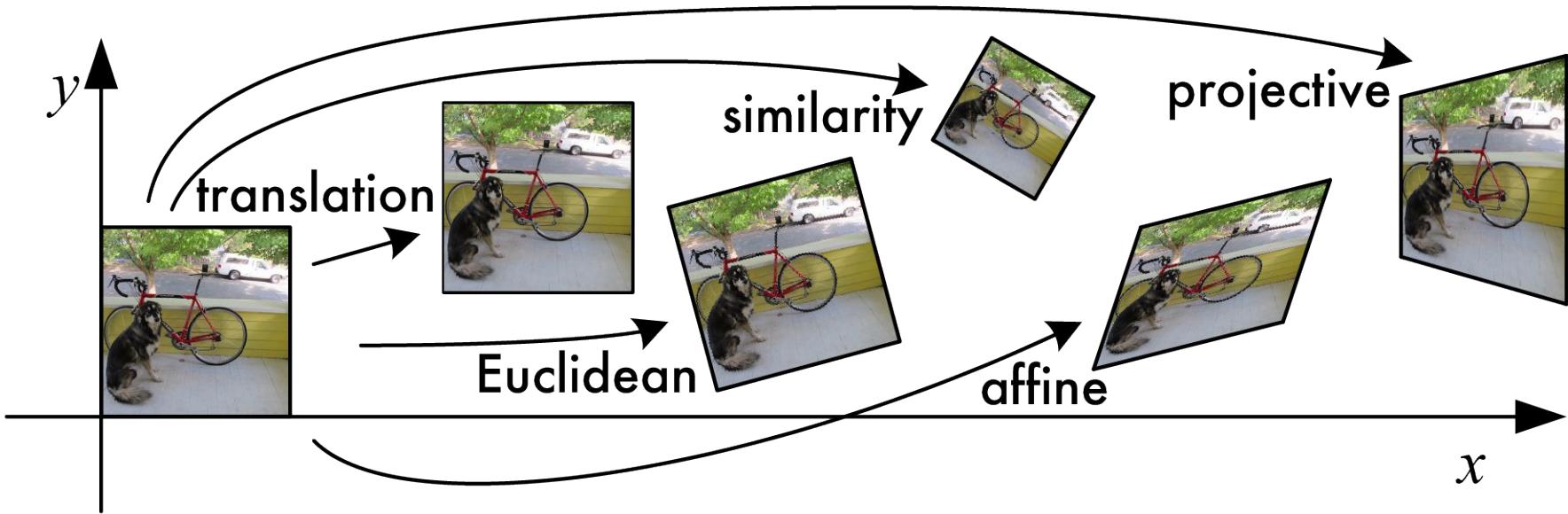
$$\bar{\mathbf{x}}' = \begin{bmatrix} \cos\theta & -\sin\theta & dx \\ \sin\theta & \cos\theta & dy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\bar{\mathbf{x}}' = \begin{bmatrix} a & -b & dx \\ b & a & dy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\bar{\mathbf{x}}' = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# Проективное преобразование

- АКА гомология
- Аффинное -  $2 \times 3$  матрицы, а проективное - ...?



# Проективное преобразование

---

- АКА гомология
- Аффинное - любая  $2 \times 3$  матрица
- Гомология - любая  $3 \times 3$  матрица
- Умножение на скаляр ничего не меняет
  - $3^* H \sim H$

$$\tilde{\mathbf{x}}' = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix} \quad \tilde{\mathbf{x}}' = \tilde{H} \tilde{\mathbf{x}}$$

# Проективное преобразование

---

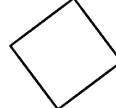
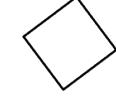
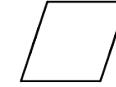
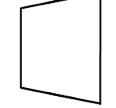
- Умножить  $\tilde{\mathbf{x}}^{\sim}$  на  $\mathbf{H}^{\sim}$ , чтобы получить  $\tilde{\mathbf{x}}'^{\sim}$
- Преобразуем в  $\tilde{\mathbf{x}}'$  просто деля  $w'^{\sim}$

$$\tilde{\mathbf{x}}' = \tilde{\mathbf{H}} \tilde{\mathbf{x}}$$

$$\begin{bmatrix} \tilde{x}' \\ \tilde{y}' \\ \tilde{w}' \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix} \quad \bar{\mathbf{x}} = \tilde{\mathbf{x}} / \tilde{w}$$

# Какое выбрать?

---

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \mathbf{I} & \mathbf{t} \end{bmatrix}_{2 \times 3}$	2	orientation	
rigid (Euclidean)	$\begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}_{2 \times 3}$	3	lengths	
similarity	$\begin{bmatrix} s\mathbf{R} & \mathbf{t} \end{bmatrix}_{2 \times 3}$	4	angles	
affine	$\begin{bmatrix} \mathbf{A} \end{bmatrix}_{2 \times 3}$	6	parallelism	
projective	$\begin{bmatrix} \tilde{\mathbf{H}} \end{bmatrix}_{3 \times 3}$	8	straight lines	

# Аффинное преобразование для матчинга

---

- Есть уже сматченные точки
- Хотим оценить матрицу  $A$ , преобразующую  $x$  в  $x'$
- $xA = x'$
- Сколько степеней свободы?
  - 6
- Сколько уравнений задает одна пара точек?  $mX = n$ 
  - 2
  - $n_x = a_{00} * m_x + a_{01} * m_y + a_{02} * 1$
  - $n_y = a_{10} * m_x + a_{11} * m_y + a_{12} * 1$

$$x' = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# Аффинное преобразование для матчинга

- Сколько уравнений задает одна пара точек?  $m \mathbf{A} = \mathbf{n}$

- $n_x = a_{00} * m_x + a_{01} * m_y + a_{02} * 1$
- $n_y = a_{10} * m_x + a_{11} * m_y + a_{12} * 1$
- Решим уравнение  $\mathbf{M} \mathbf{a} = \mathbf{b}$

- $\mathbf{M}^T \mathbf{M} \mathbf{a} = \mathbf{M}^T \mathbf{b}$
- $\mathbf{a} = (\mathbf{M}^T \mathbf{M})^{-1} \mathbf{M}^T \mathbf{b}$

- Работает даже если overdetermined
  - Почему???

$$\begin{bmatrix} \mathbf{M} & \mathbf{a} & \mathbf{b} \end{bmatrix} = \begin{bmatrix} m_{x1} & m_{y1} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & m_{x1} & m_{y1} & 1 \\ m_{x2} & m_{y2} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & m_{x2} & m_{y2} & 1 \\ m_{x3} & m_{y3} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & m_{x3} & m_{y3} & 1 \\ \dots & & & & & \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{01} \\ a_{02} \\ a_{10} \\ a_{11} \\ a_{12} \end{bmatrix} = \begin{bmatrix} n_{x1} \\ n_{y1} \\ n_{x2} \\ n_{y2} \\ n_{x3} \\ n_{y3} \end{bmatrix}$$

## Linear least squares

---

Хотим минимизировать квадрат ошибок

$$\| b - Ma \|^2 = b^T b - 2a^T M^T b + a^T M^T M a$$

Это выпуклая функция, необходимое условие минимума - градиент = 0. Берем производную по  $a$  (это вектор!) и приравниваем к нулю.

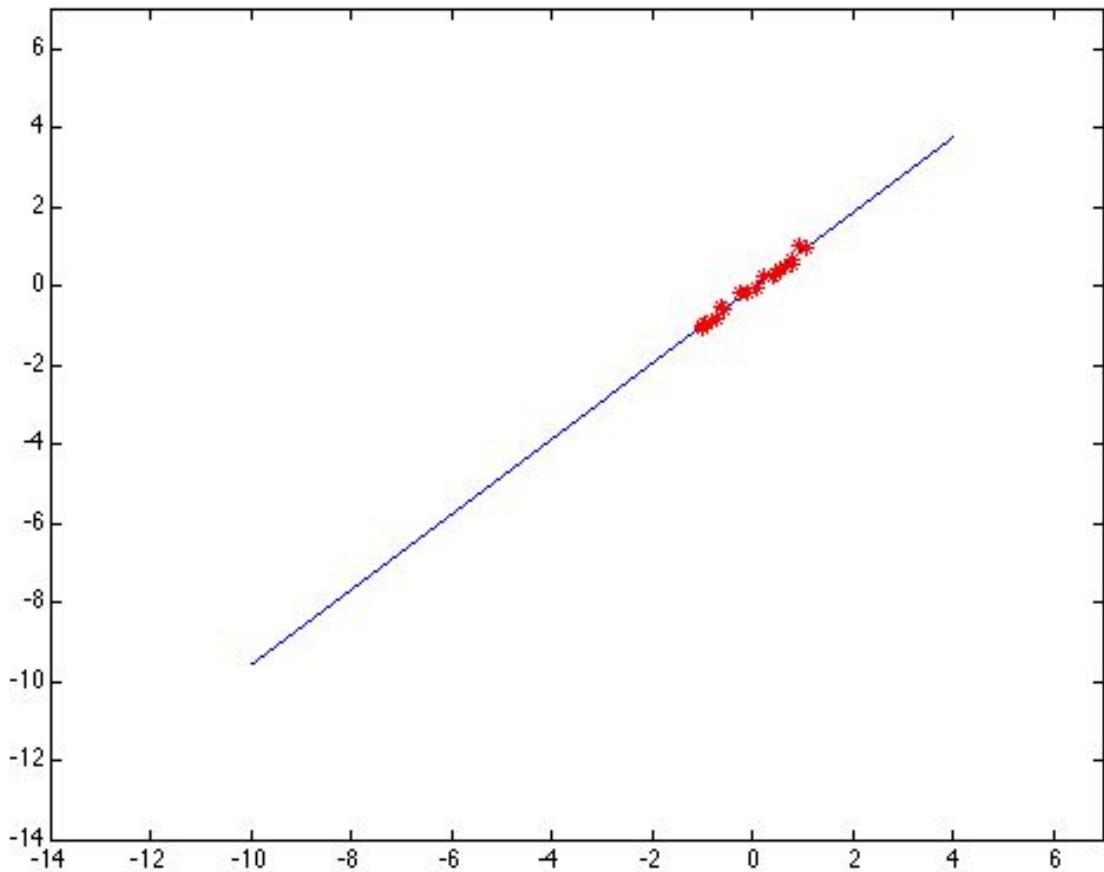
$$-M^T b + (M^T M)a = 0$$

$$(M^T M)a = M^T b$$

$$a = (M^T M)^{-1} M^T b \quad \text{ура!}$$

# Геометрический смысл

---

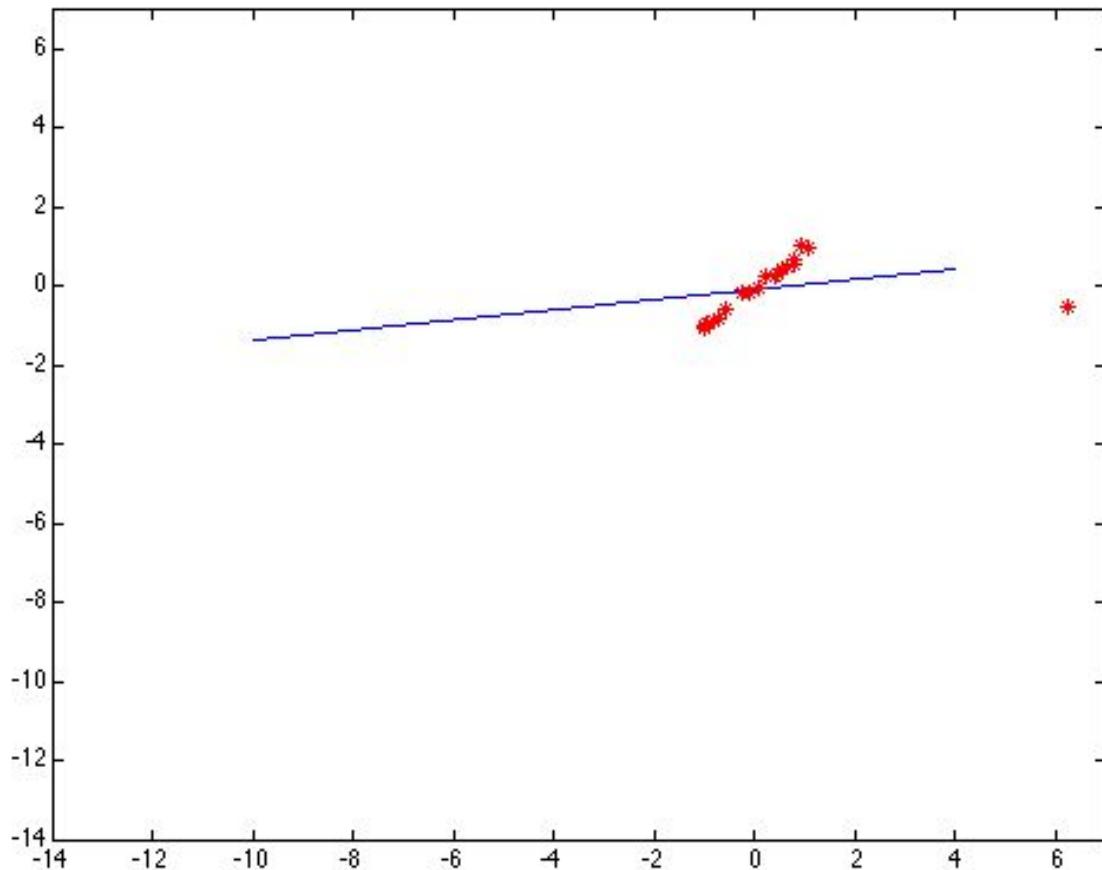


# Геометрический смысл

---

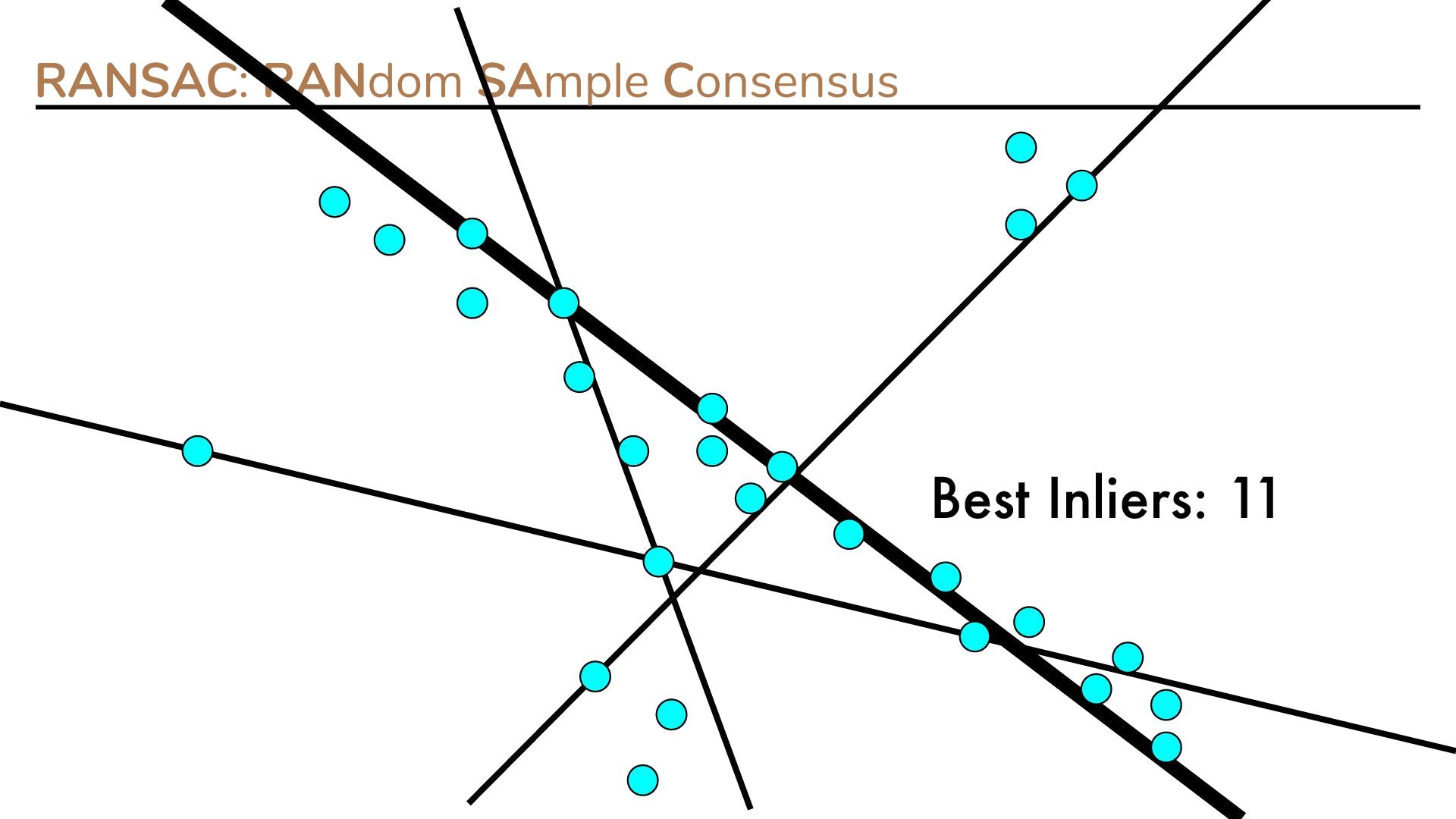
Ошибка квадратичная!

Очень плохо справляется с  
выбросами (outliers) в данных



## RANSAC: RANdom SAmple Consensus

---



# RANSAC: RANdom SAmple Consensus

---

- Параметры: данные, модель, n - количество точек для обучения, k - количество итераций, t - порог, d - порог “хорошой модели”

```
bestmodel = None
```

```
bestfit = INF
```

```
While i < k:
```

```
    sample = draw n random points from data
```

```
    Fit model to sample
```

```
    inliers = data within t of model
```

```
    if inliers > bestfit:
```

```
        Fit model to all inliers
```

```
        bestfit = inliers
```

```
        bestmodel = model
```

```
    if inliers > d:
```

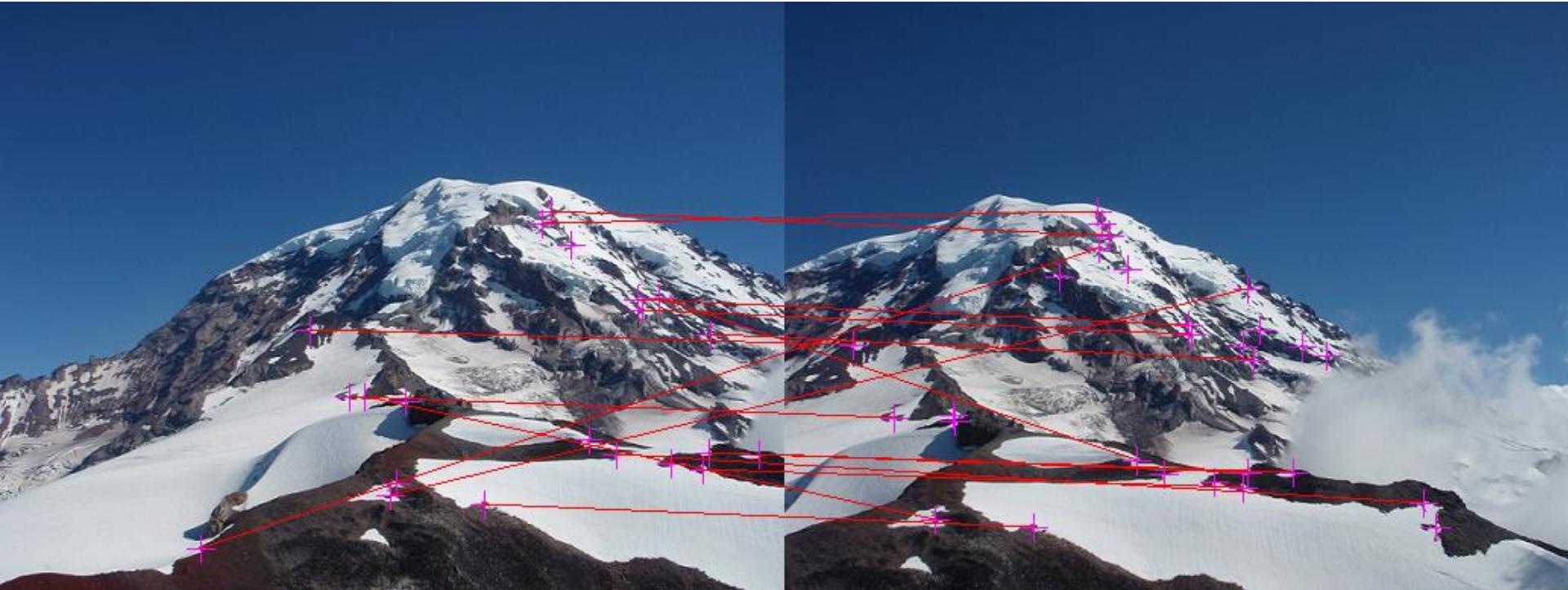
```
        return model
```

```
return bestmodel
```

## RANSAC: RANdom SAmple Consensus

---

- Хорошо работает с большим количеством шума



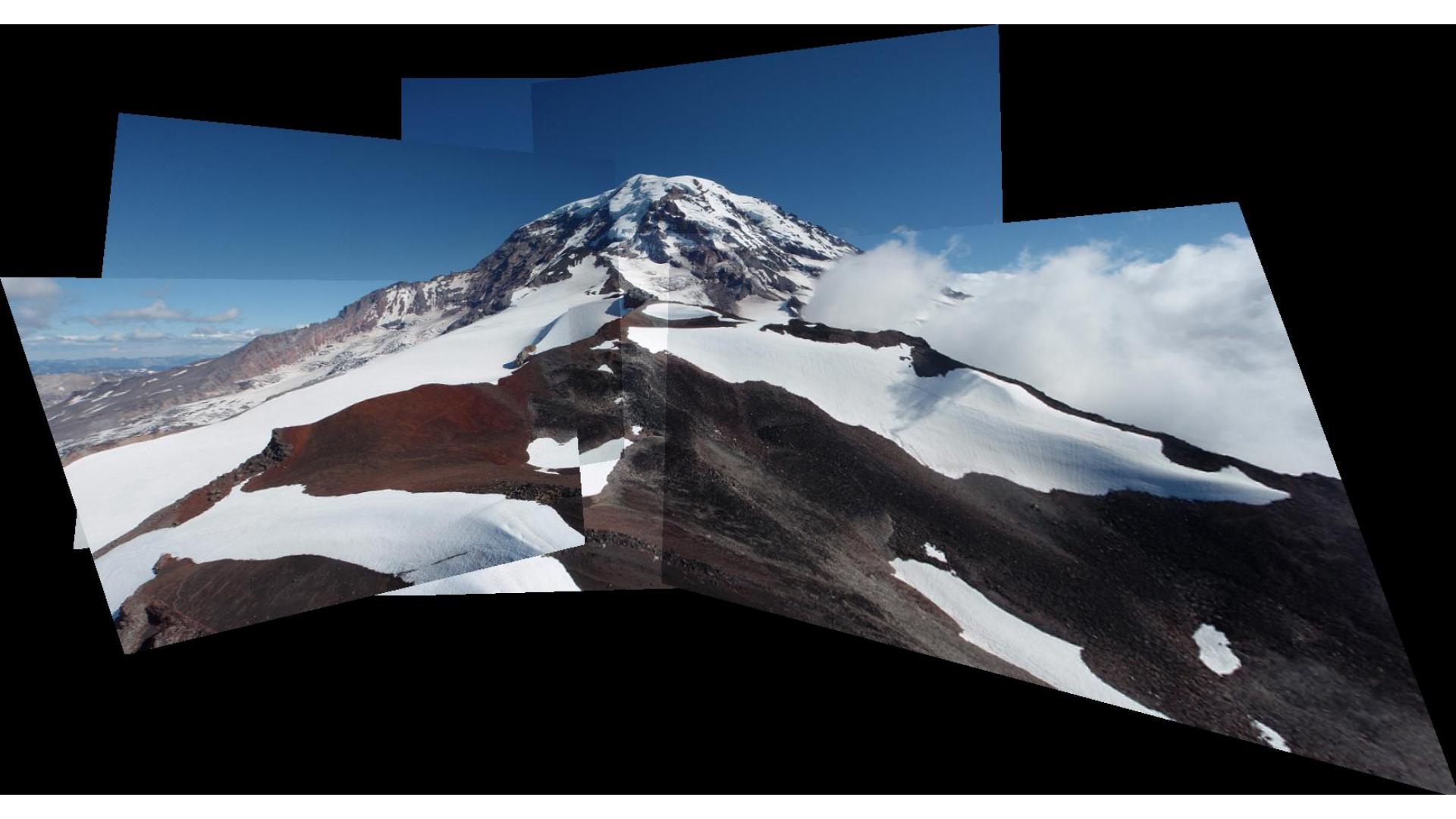
# Проективное преобразование

---

- Матричное уравнение

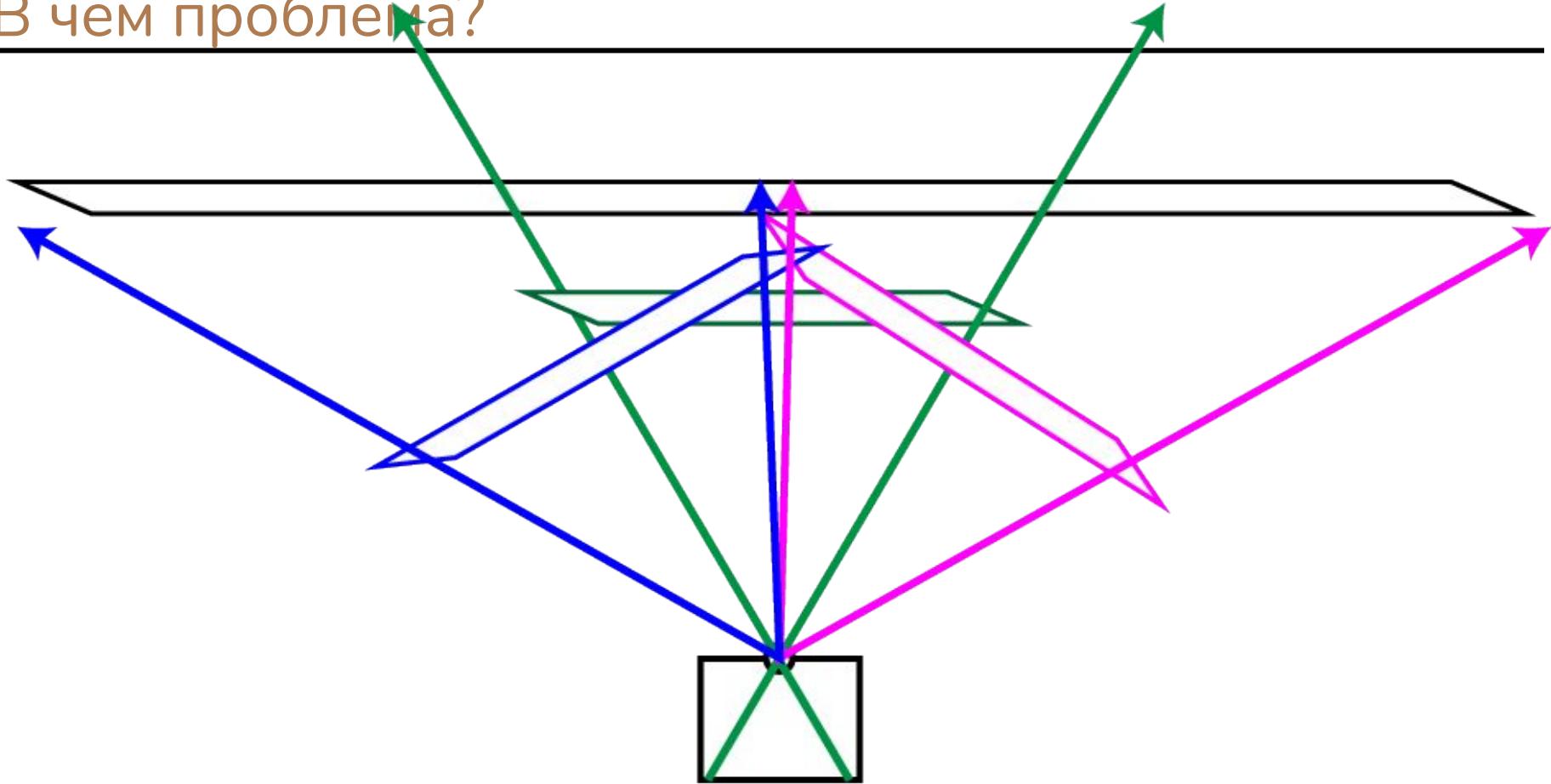
$$\mathbf{M} \begin{bmatrix} m_{x1} & m_{y1} & 1 & 0 & 0 & 0 & -m_{x1}n_{x1} & -m_{y1}n_{x1} \\ 0 & 0 & 0 & m_{x1} & m_{y1} & 1 & -m_{x1}n_{y1} & -m_{y1}n_{y1} \\ m_{x2} & m_{y2} & 1 & 0 & 0 & 0 & -m_{x2}n_{x2} & -m_{y2}n_{x2} \\ 0 & 0 & 0 & m_{x2} & m_{y2} & 1 & -m_{x2}n_{y2} & -m_{y2}n_{y2} \\ m_{x3} & m_{y3} & 1 & 0 & 0 & 0 & -m_{x3}n_{x3} & -m_{y3}n_{x3} \\ 0 & 0 & 0 & m_{x3} & m_{y3} & 1 & -m_{x3}n_{y3} & -m_{y3}n_{y3} \\ m_{x4} & m_{y4} & 1 & 0 & 0 & 0 & -m_{x4}n_{x4} & -m_{y4}n_{x4} \\ 0 & 0 & 0 & m_{x4} & m_{y4} & 1 & -m_{x4}n_{y4} & -m_{y4}n_{y4} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} n_{x1} \\ n_{y1} \\ n_{x2} \\ n_{y2} \\ n_{x3} \\ n_{y3} \\ n_{x4} \\ n_{y4} \end{bmatrix}$$

- Решаем  $\mathbf{M} \mathbf{a} = \mathbf{b}$ 
  - Точное решение если количество строк  $M = 8$
  - МНК если количество строк  $M > 8$



В чем проблема?

---



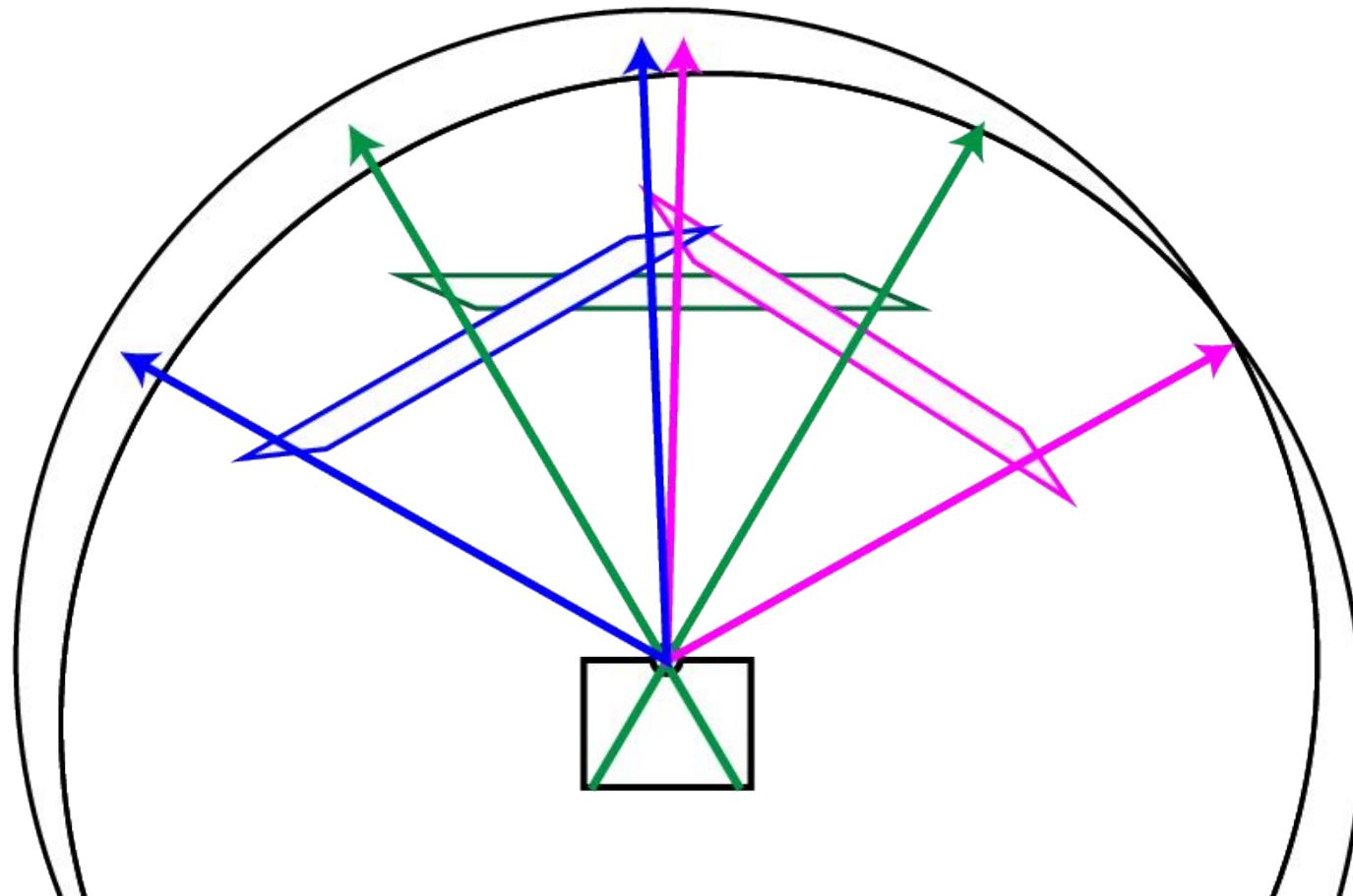
Плохо для больших панорам!

---



# Как это исправить? Цилиндры!

---



# Как это исправить? Цилиндры!

Calculate angle and height:

$$\theta = (x - x_c) / f$$

$$h = (y - y_c) / f$$

Find unit cylindrical coords:

$$X' = \sin(\theta)$$

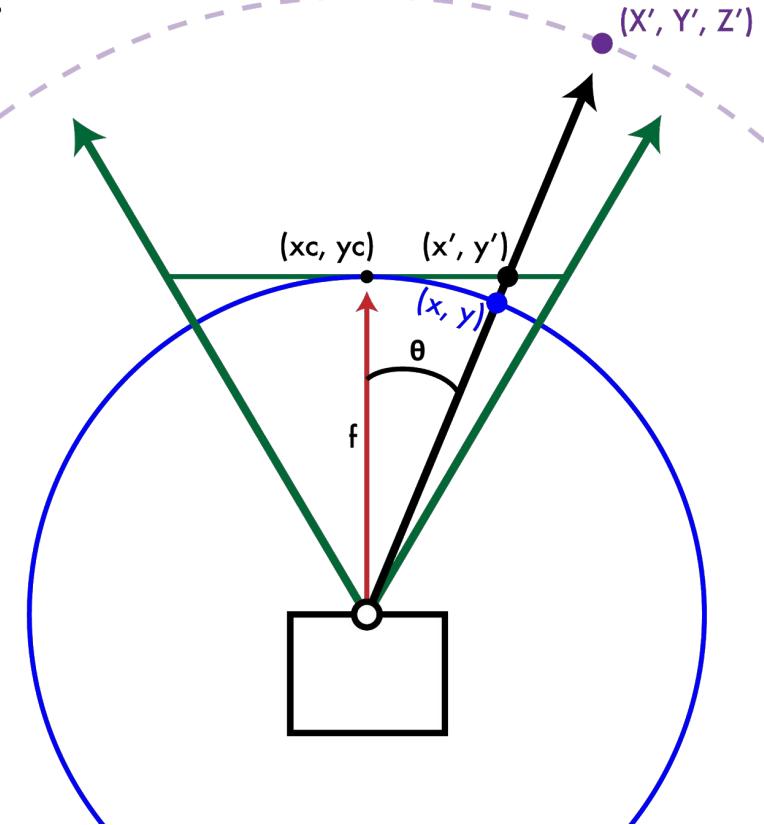
$$Y' = h$$

$$Z' = \cos(\theta)$$

Project to image plane:

$$x' = f X' / Z' + x_c$$

$$y' = f Y' / Z' + y_c$$



# Это работает?



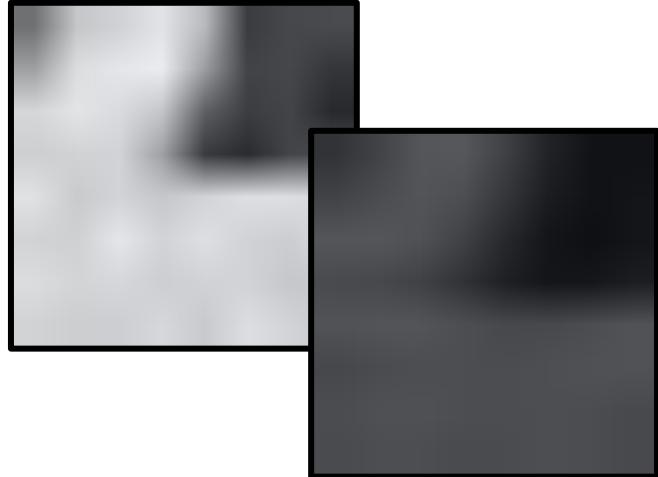
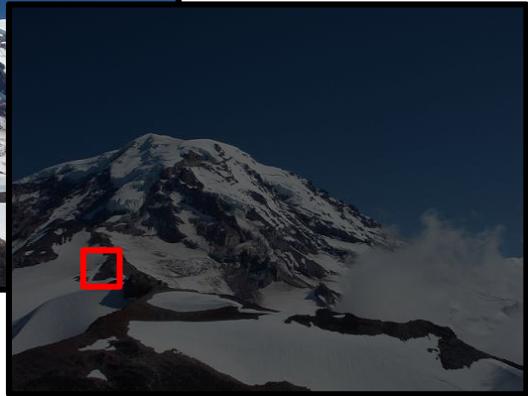
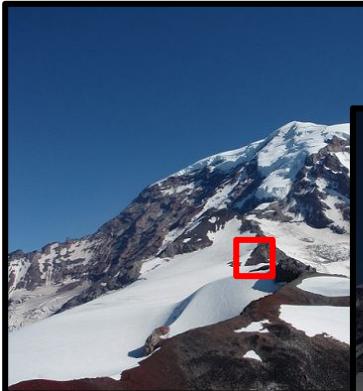
Это работает? Да!



# Все хорошо!

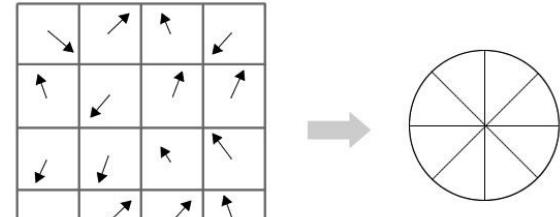
---

- Не так быстро!
- У нашего дескриптора есть проблемы:
  - Неинвариантность к изменению света
- Хотим признаки инвариантные к изменению освещения



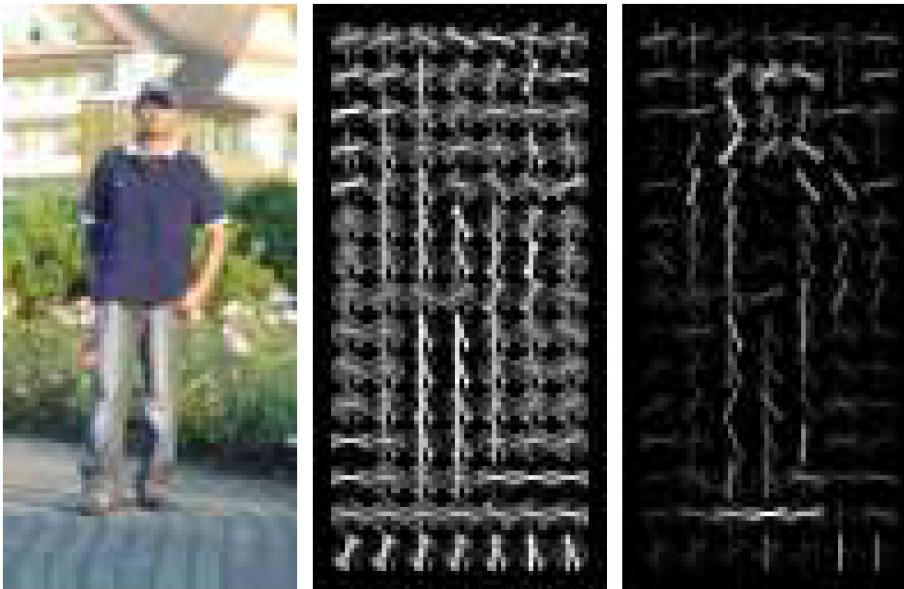
# Histogram of Oriented Gradients (HOG)

- Dalal and Triggs, 2005
- Неплохой дескриптор
  - Вычислить градиенты
  - Разбить на блоки (4x4, 16x16 cells)
  - Построить гистограммы
  - Нормализовать гистограммы
- Абсолютные значения магнитуды не важны
  - Позволяет бороться с изменением освещенности
- Обучаем SVM на распознавание людей



# Histogram of Oriented Gradients (HOG)

---

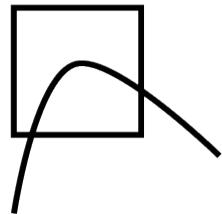


# Еще проблемы

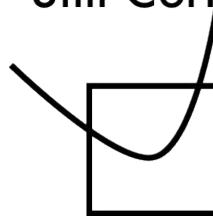
---

- Детектор Харриса:
  - Инвариантен к поворотам
  - Не инвариантен к масштабированиям
- Дескрипторы:
  - Пиксельное описание - не инвариант к поворотам
  - HOG - также не инвариант к поворотам

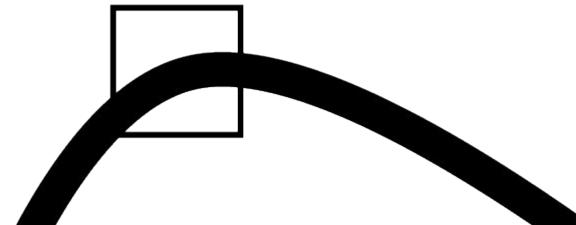
Corner



Still Corner



Not Corner



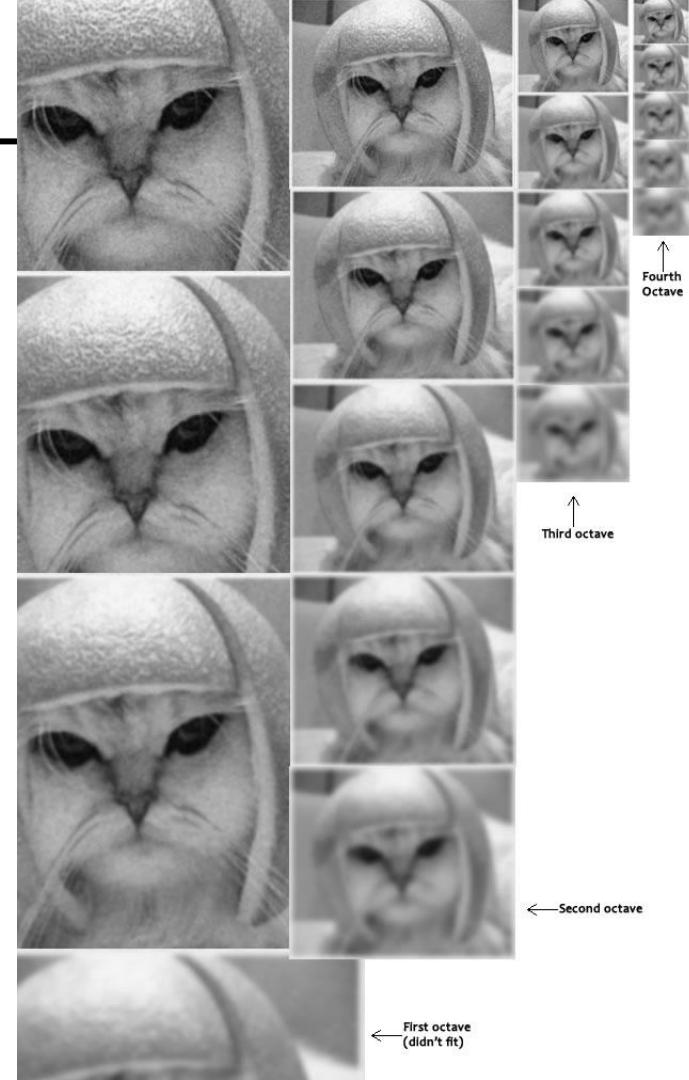
# SIFT

---

- Scale Invariant Feature Transform (SIFT)
  - Lowe et al. 2004
  - Самая цитируемая статья в CV
  - Запатентована (до сих пор)
- Построить scale space
- Найти ключевые точки
- Построить устойчивый к повороту к дескриптор
  - Нормализовать по направлению
  - Нормализовать по свету

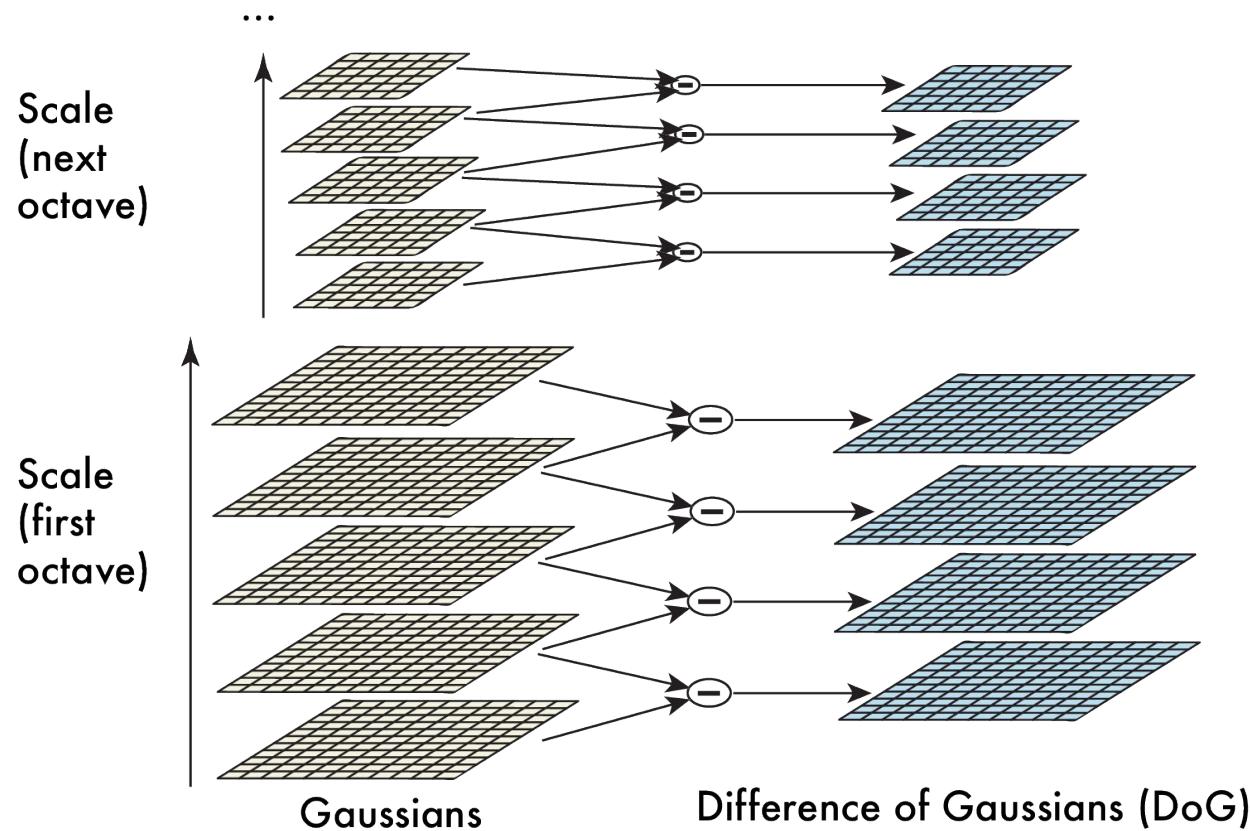
Интерактивный [тutorиал](#)

# Scale space



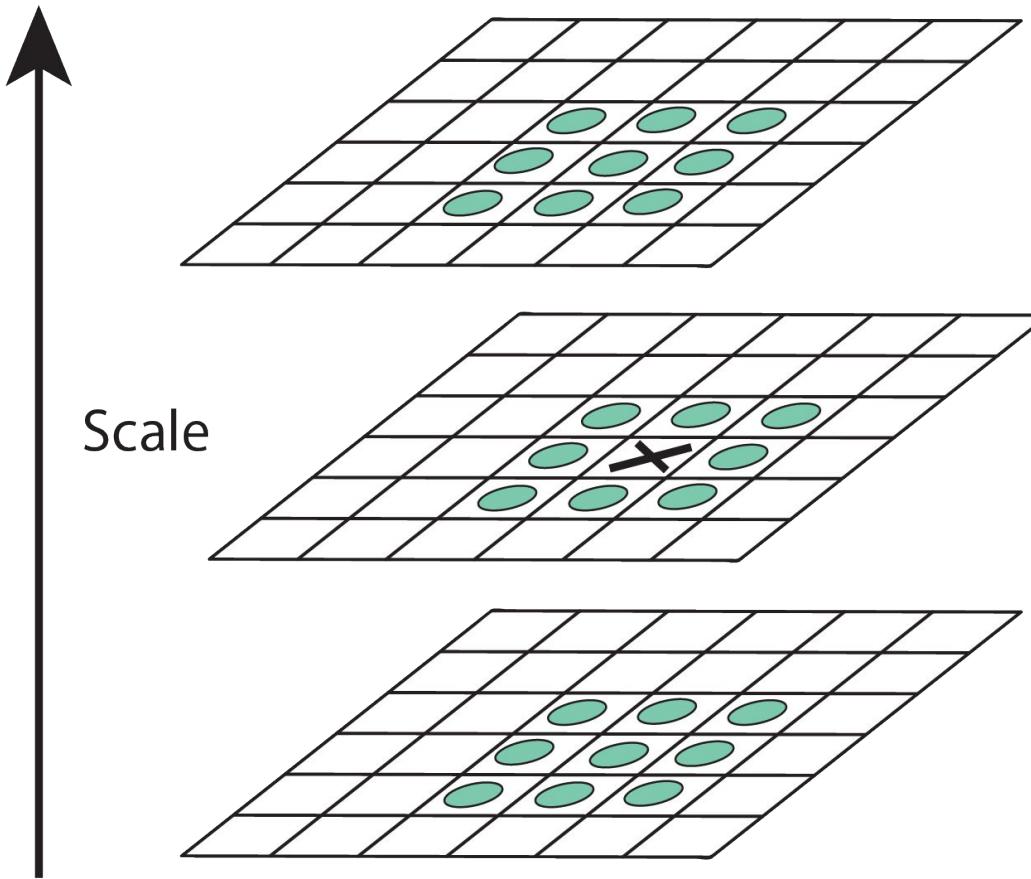
# DoG

---



# Нахождение локальных максимумов

---



# Убрать лишние точки

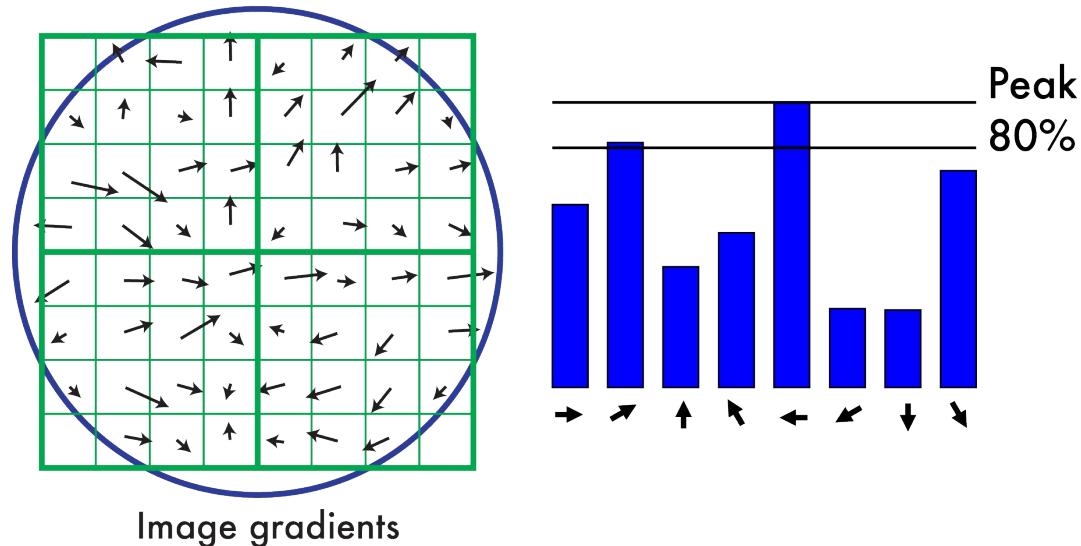
---

- Вычислить градиенты
  - Не по всему изображению, а по избранным точкам. Быстрее!
  - Убрать точки со слабым откликом
- ФУНКЦИЯ ОТКЛИКА УГЛА
  - То же самое, structure matrix, детерминант и след матрицы
  - $r$  - масштаб, на котором рассматривается точка

$$\frac{\text{Tr}(\mathbf{H})^2}{\text{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}$$

# Нахождение ориентации региона

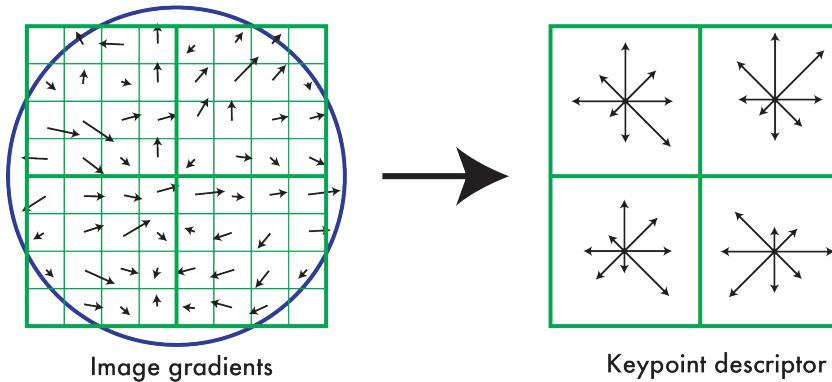
- Взвешенная гистограмма градиентов соседей
  - Любой градиент больший или равный 80 % значения пика, получает свой дескриптор
    - Один пиксель - 2 ключевых точки
  - Дескрипторы нормализуются по главному направлению



## Дескриптор (аля HOG)

---

- Разделить на подобласти ( $2 \times 2$ ,  $4 \times 4$ )
- В каждой подобласти построить гистограмму
  - Нормализовать
  - Столбцы большие чем 0.2, обрезать до 0.2
  - Снова нормализовать
  - Позволяет бороться с изменением освещения



# SIFT

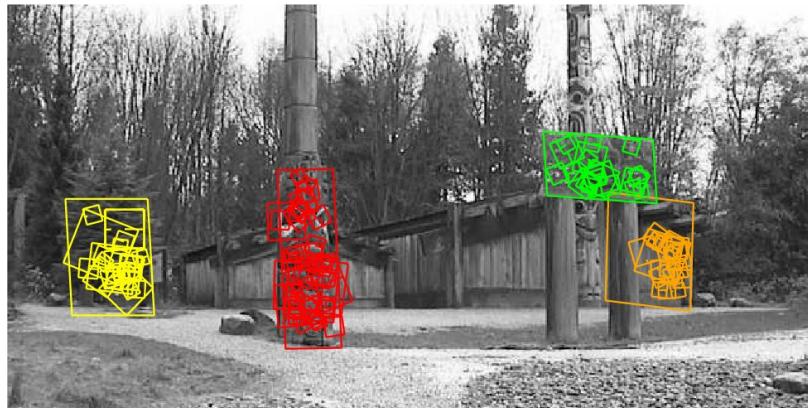
---

- Находит хорошие точки, описывает их
- Нахождение объектов, распознавание, панорамы, ...



# SIFT is great!

---



# Optical flow

## Темы

---

- Sparse\* Optical Flow (Lucas-Kanade, 1981)
- Dense Optical Flow (Farneback, 2003)

# Что такое Optical Flow?

---

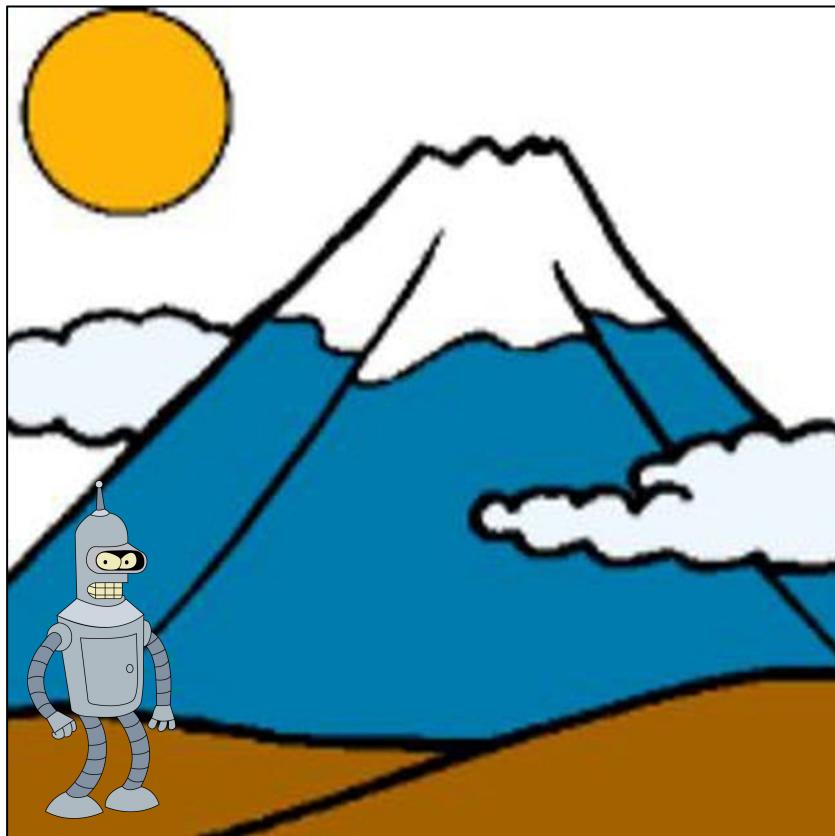
Что такое Optical Flow?

Движение!

---

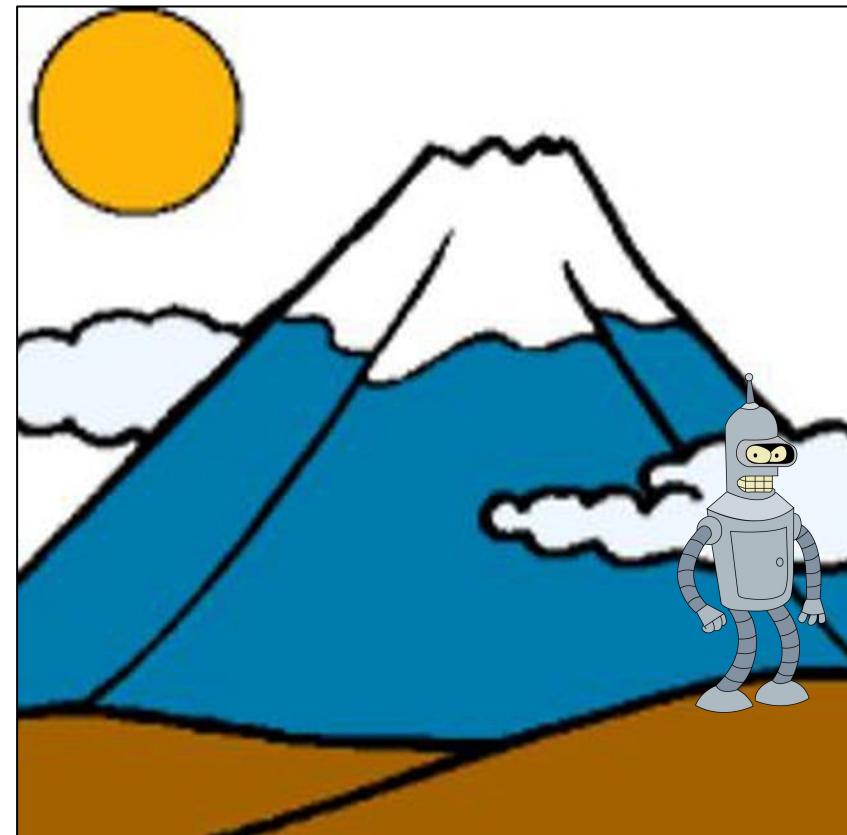
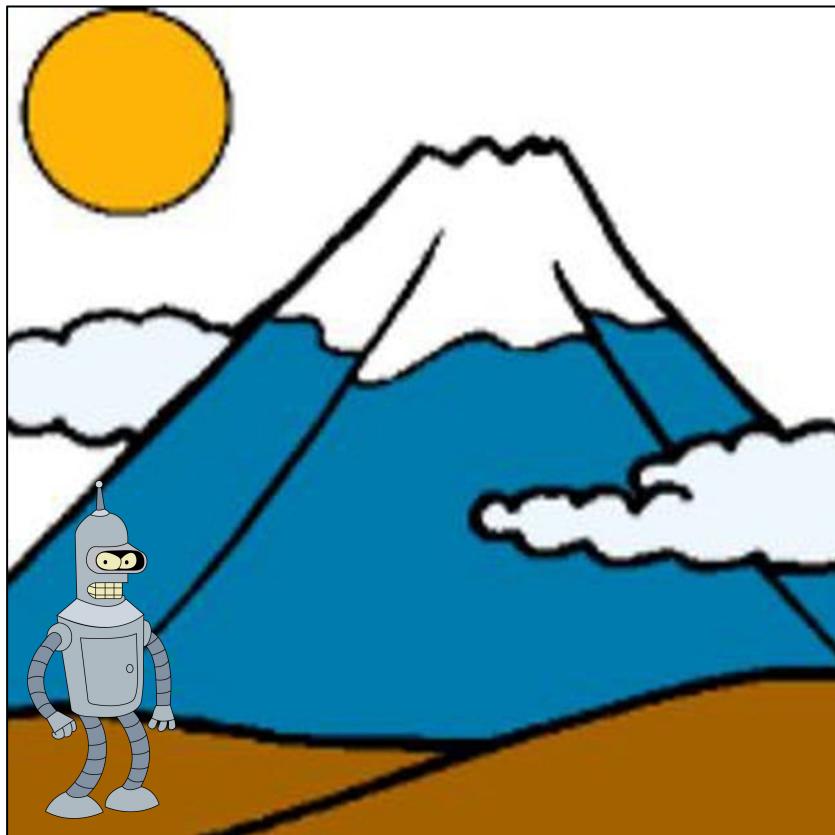
Что такое Optical Flow?

Движение!



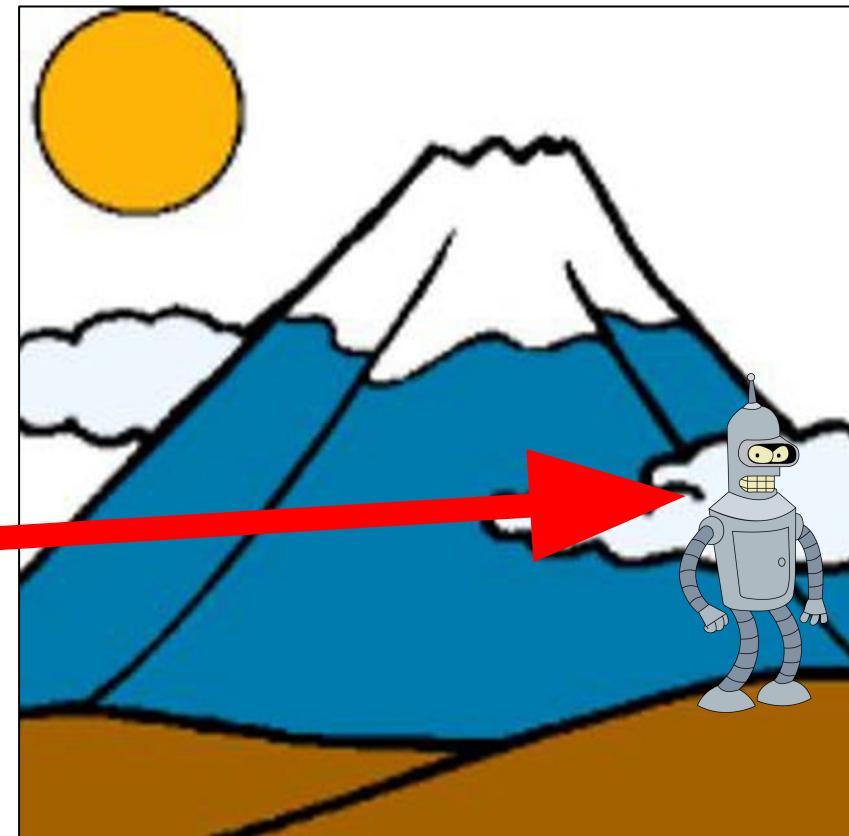
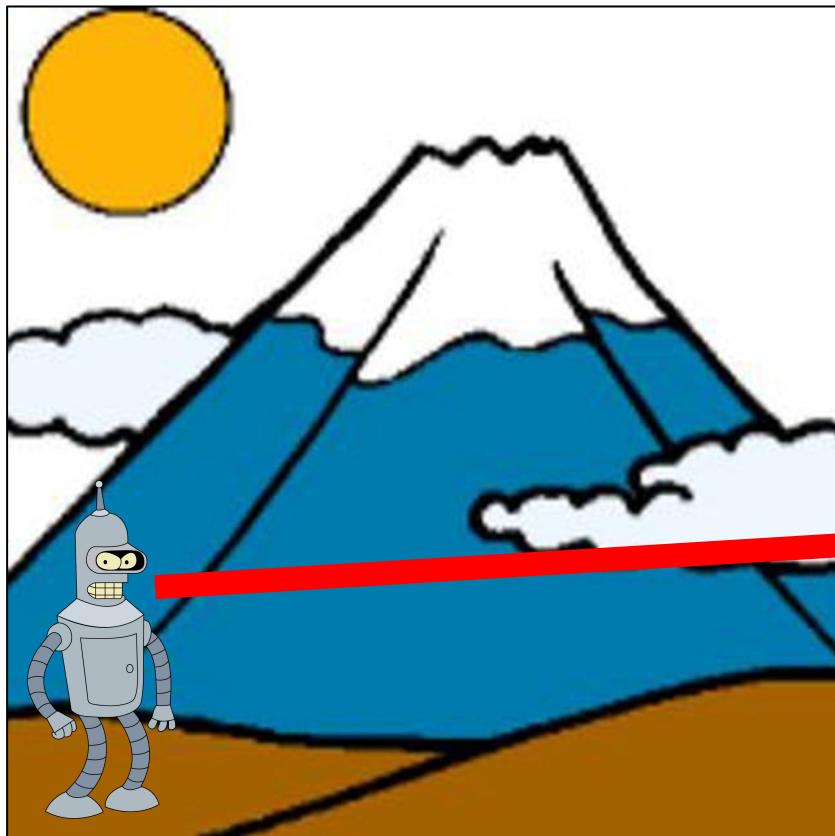
Что такое Optical Flow?

Движение!



Что такое Optical Flow?

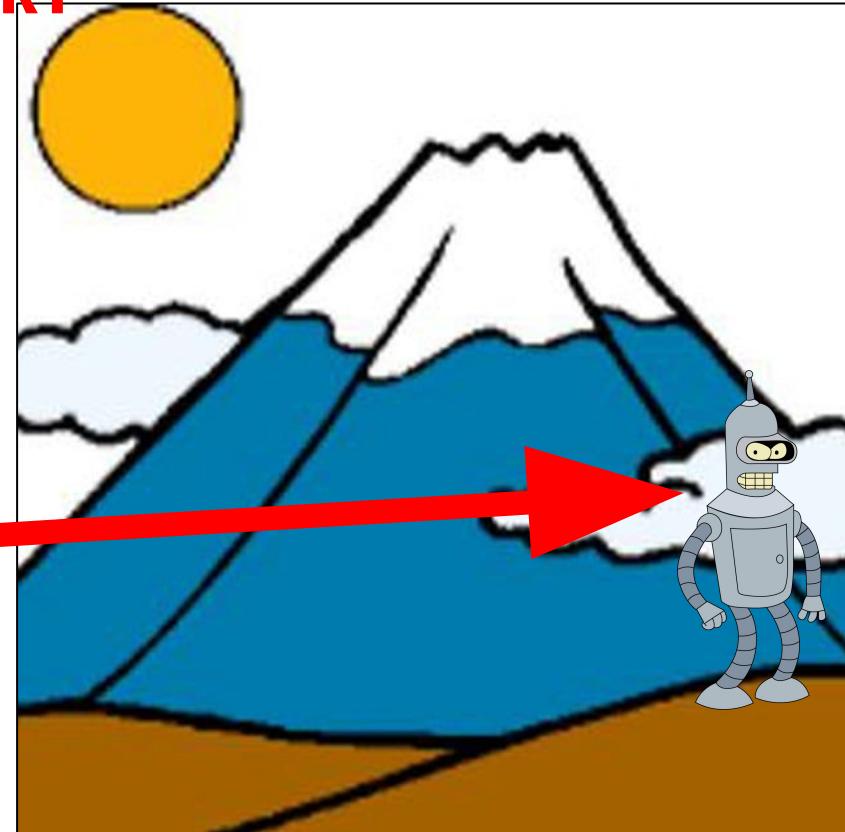
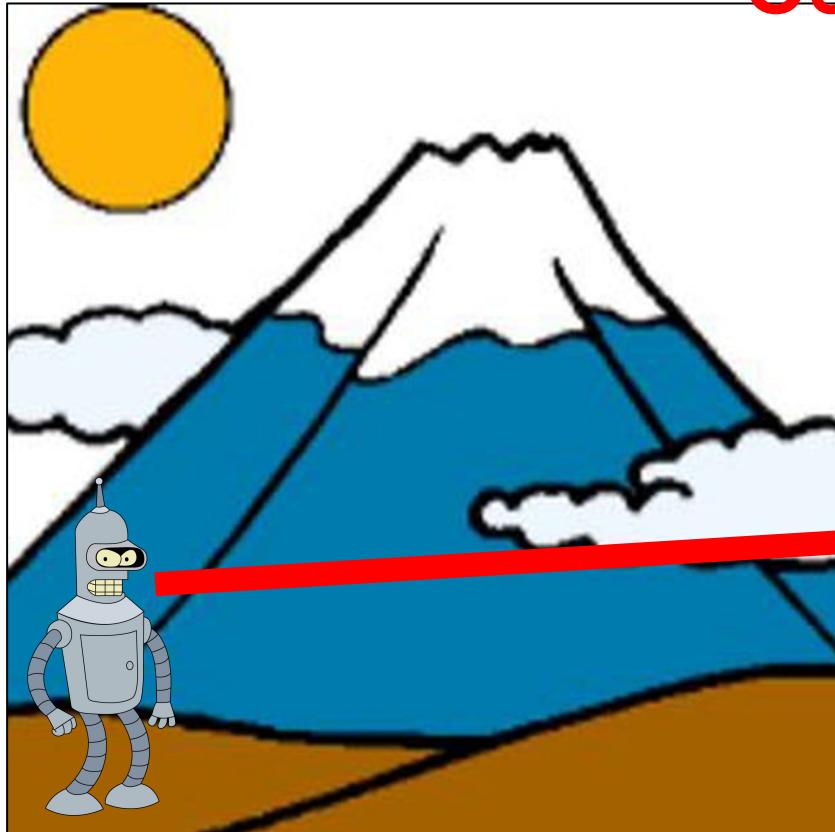
Движение!



# Что такое Optical Flow?

# Движение

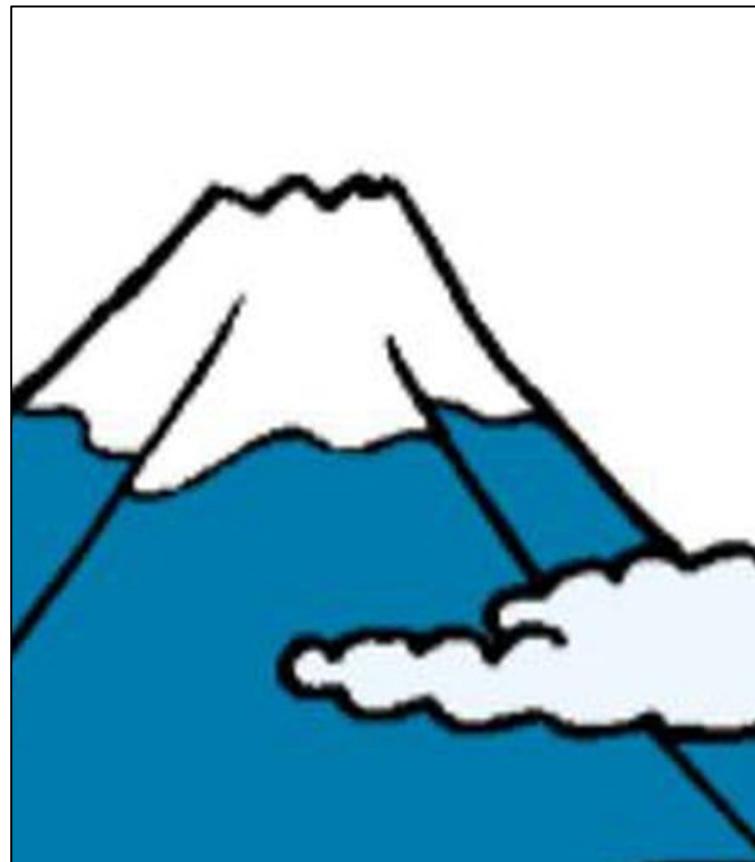
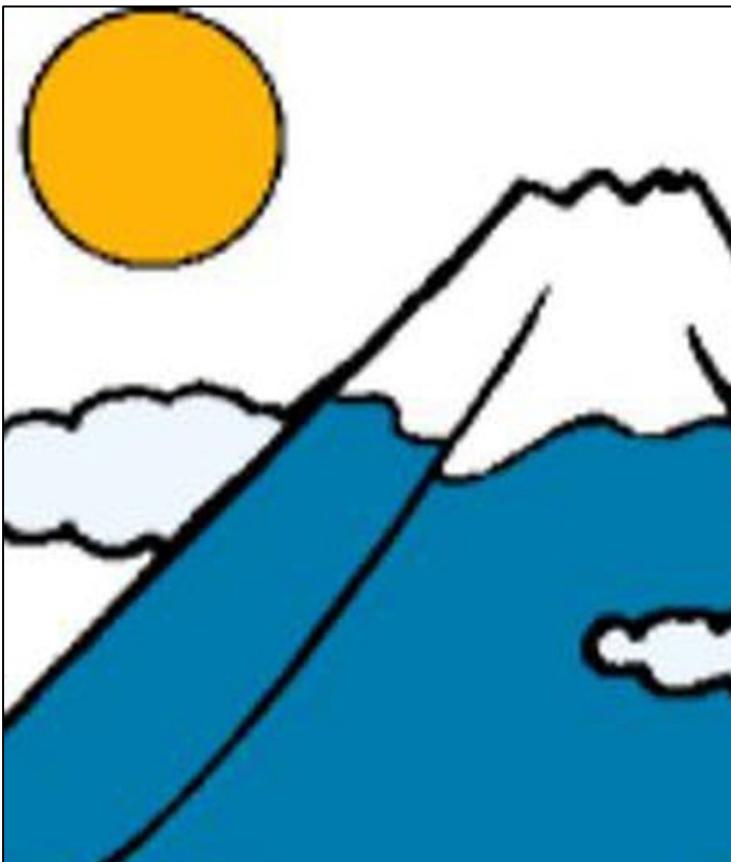
Объект



Что такое Optical Flow?

Движение

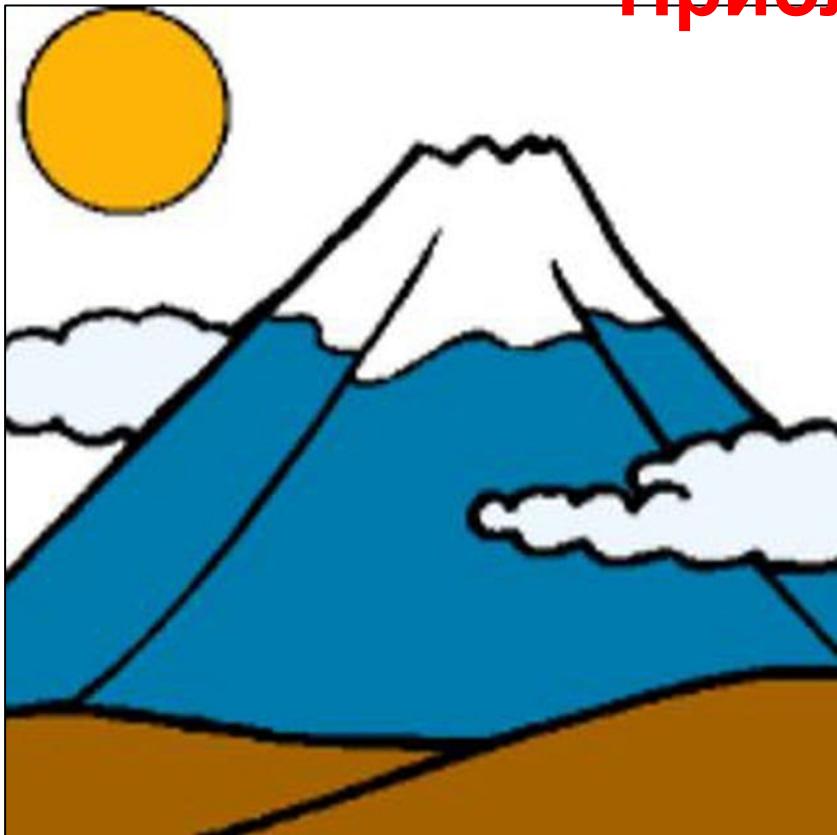
Сдвиг



Что такое Optical Flow?

Движение

Приближение



# Что такое Optical Flow?

# Движение

---



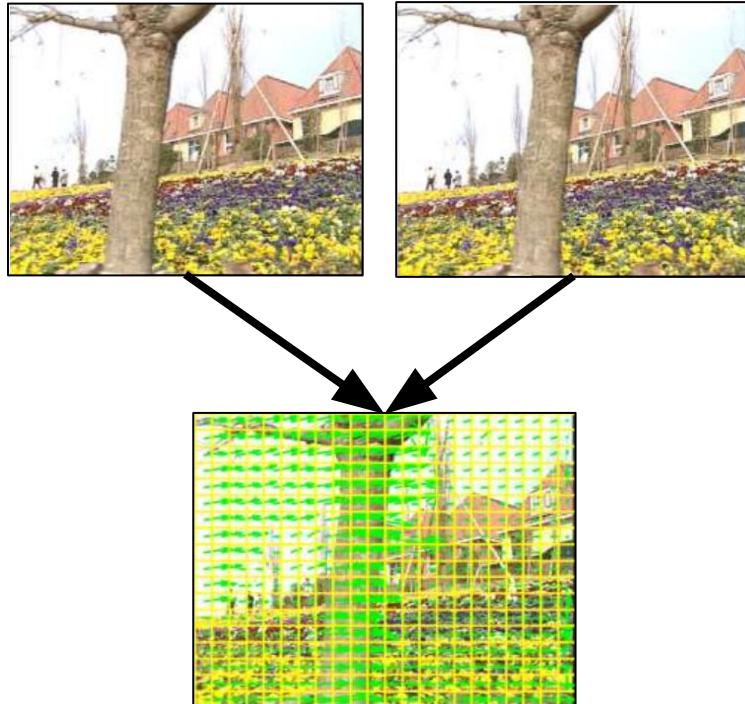
Зачем?

---

Зачем?

---

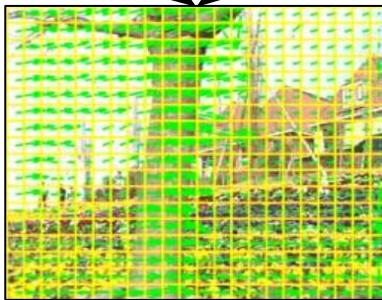
## Motion Estimation



Зачем?

---

## Motion Estimation



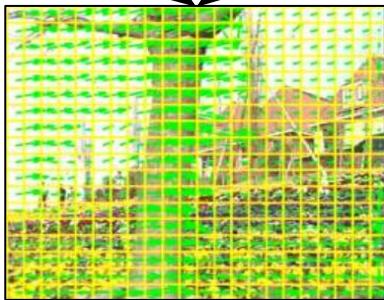
## Object Tracking



Зачем?

---

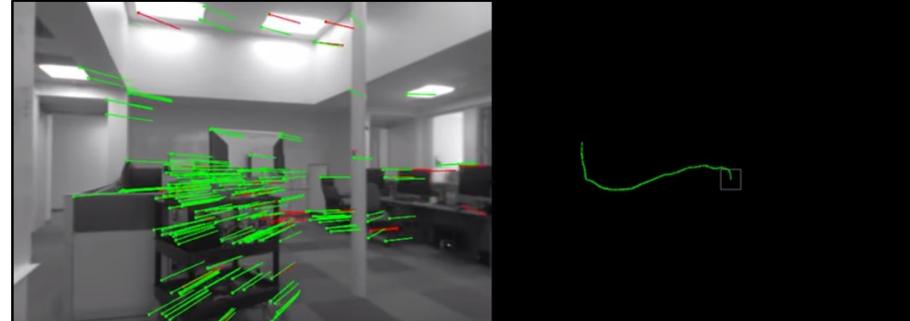
## Motion Estimation



## Object Tracking



## Visual Odometry



---

Как находить  
Optical flow?

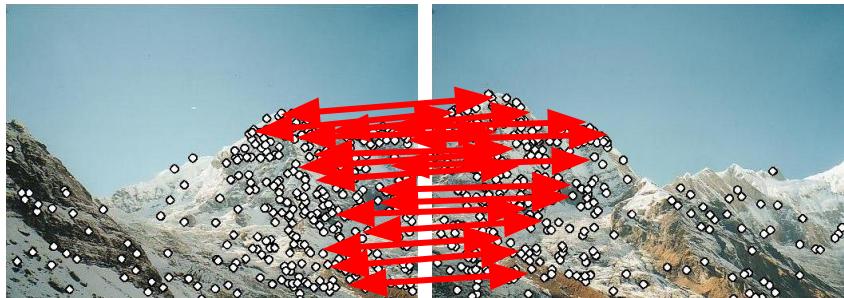
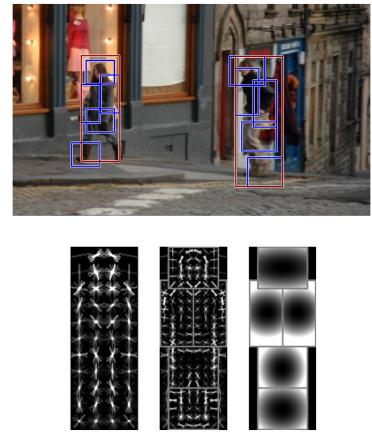
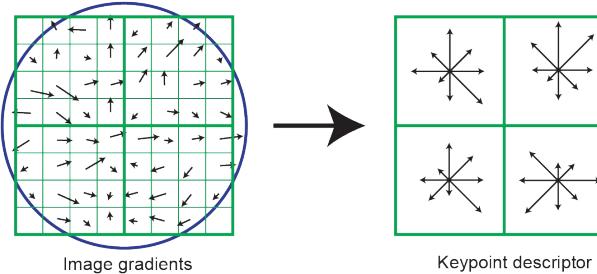
# Матчинг признаков

---

# В предыдущих сериях:

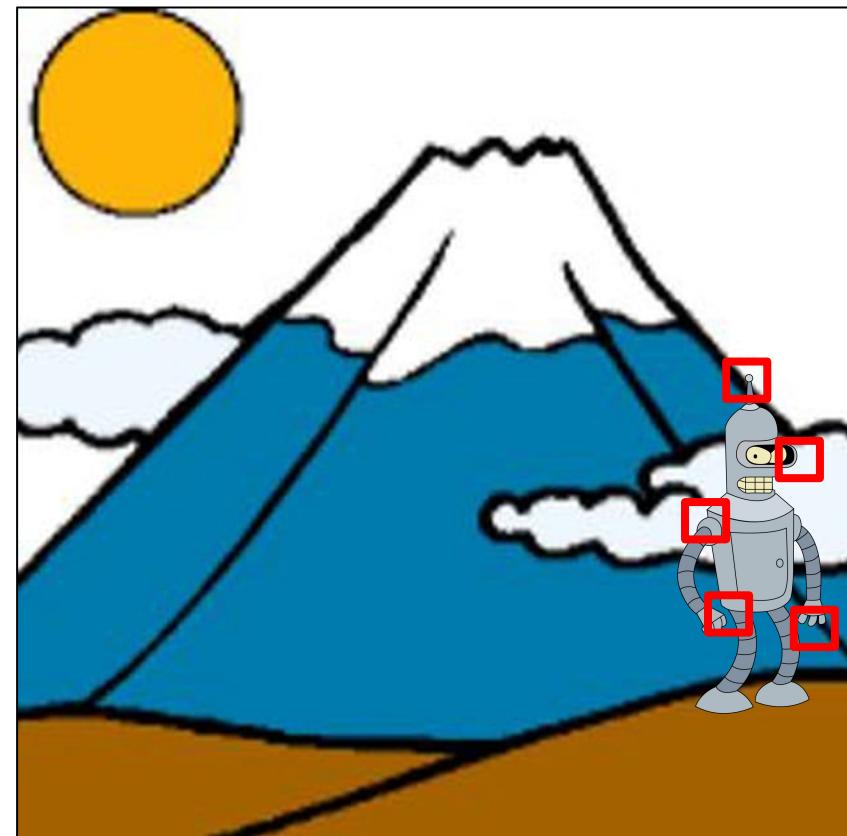
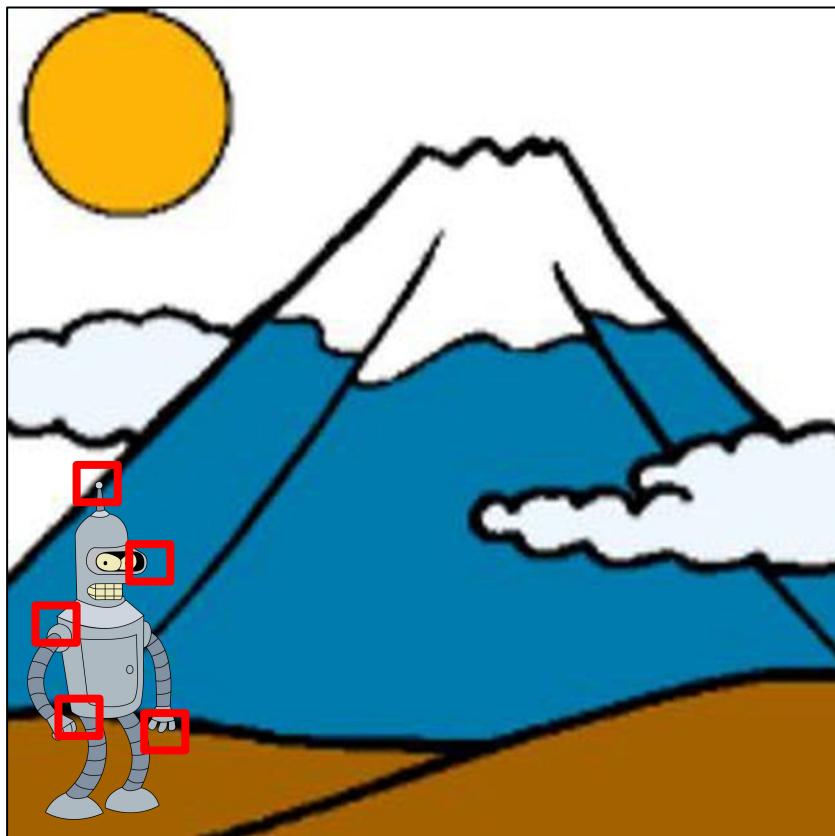
---

- Характерные регионы изображения
- Векторное описание этих регионов
- Используется в:
  - Matching
  - Recognition
  - Detection



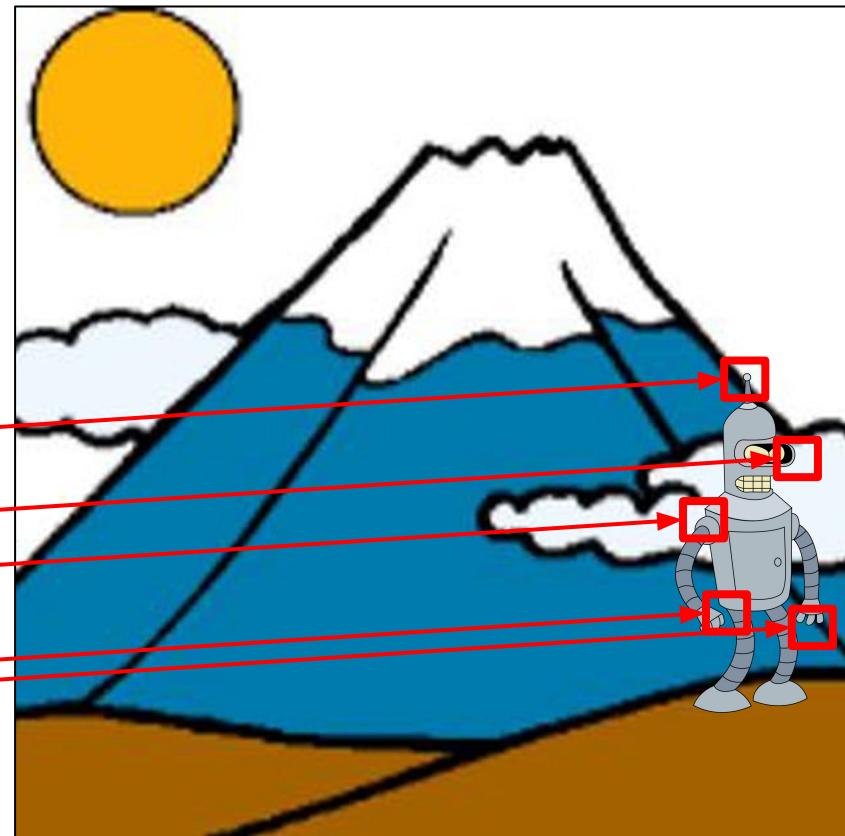
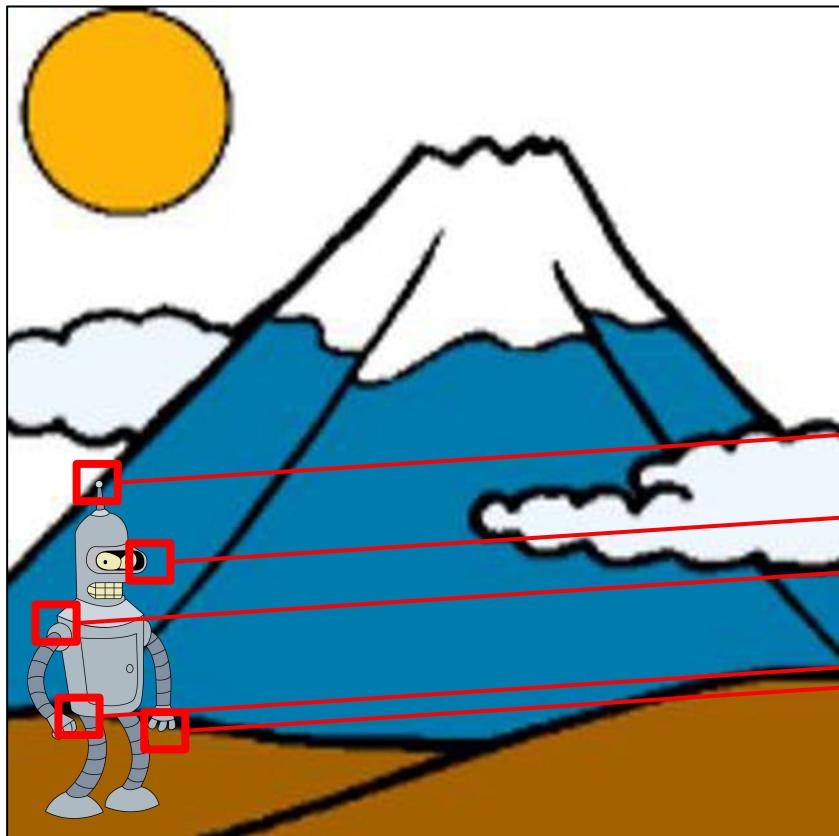
# Матчинг признаков

---



# Матчинг признаков

---



# Матчинг признаков

---

Cons:

# Матчинг признаков

---

Cons:

-Мы оцениваем движение  
для слишком маленького  
количества точек (sparse)!

# Матчинг признаков

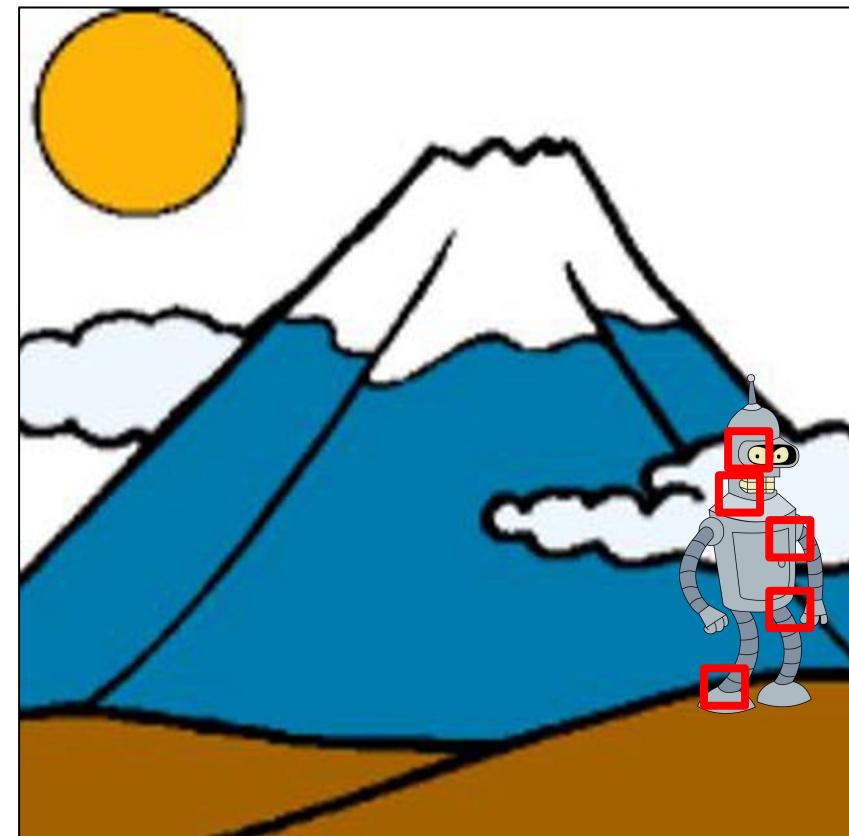
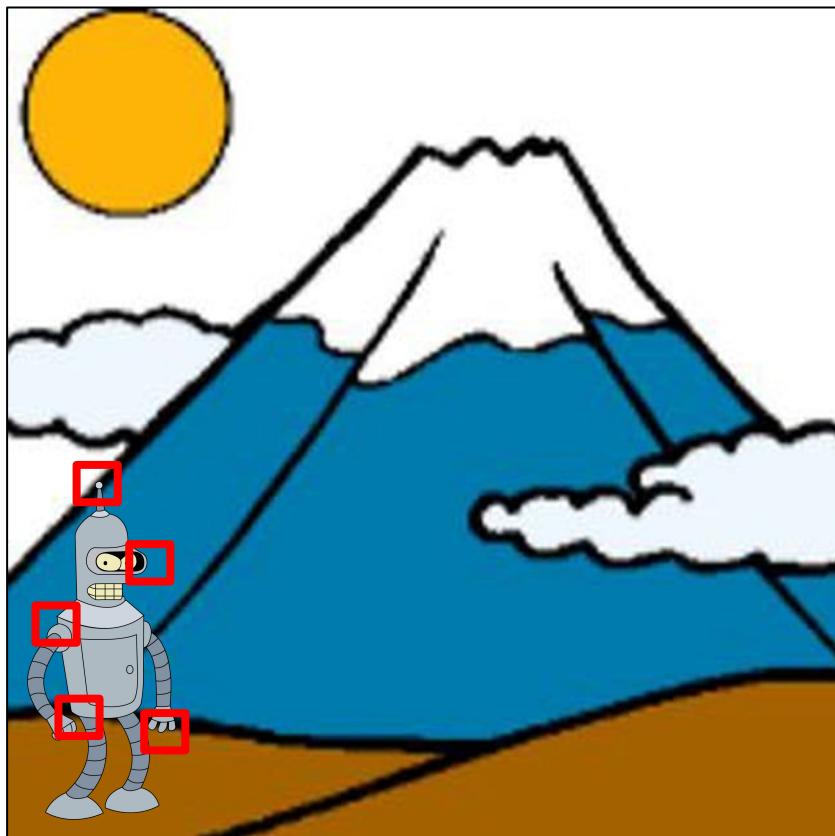
---

Cons:

- Мы оцениваем движение для слишком маленького количества точек (sparse)!
- Точки матчатся не точно

# Матчинг признаков

---



# Матчинг признаков

---

Cons:

- Мы оцениваем движение для слишком маленького количества точек (sparse)!
- Точки матчатся не точно
- Много False Positive

# Матчинг признаков

---

Cons:

-Мы оцениваем движение  
для слишком маленького  
количества точек (sparse)!

-Точки матчятся не точно  
-Много False Positive

Pros:

# Матчинг признаков

---

Cons:

- Мы оцениваем движение для слишком маленького количества точек (sparse)!
- Точки матчются не точно
- Много False Positiv

Pros:

- Инварианты к поворотам и масштабированию
- Инвариант к изменению освещенности
- Может справляться с большими перемещениями

# Feature Matching

---

Cons:

-Мы оцениваем движение для слишком маленького количества точек

-Точки матчаться

-Много False Pos

Pros:

-Инварианты к поворотам и масштабированию

изменению

(почти)

пяться с

большими  
перемещениями

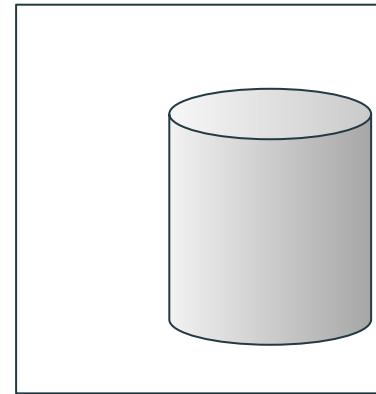
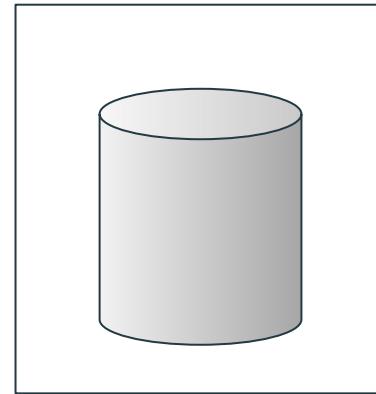
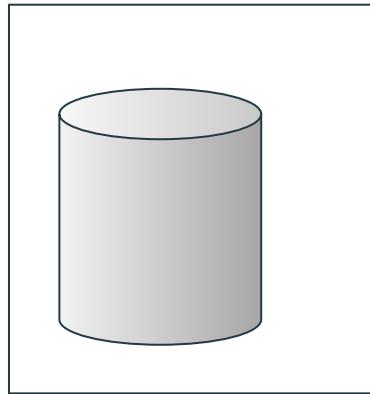
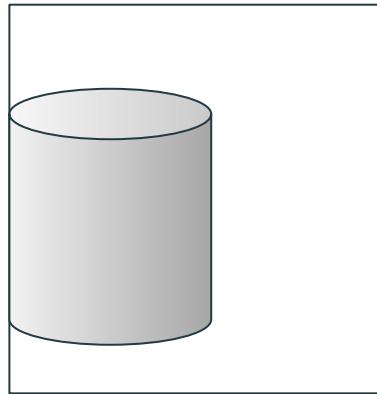
**В общем случае  
плохо работает для  
задачи Optical Flow**

---

# Что делать?

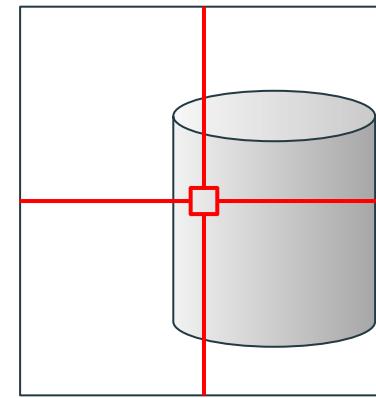
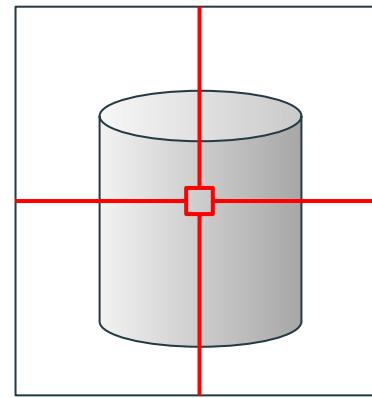
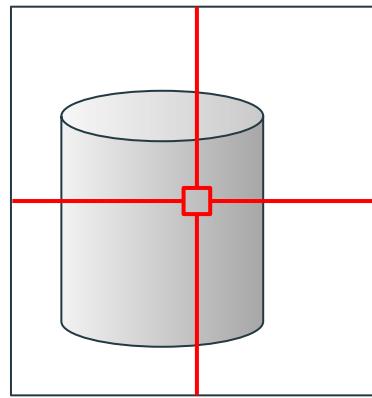
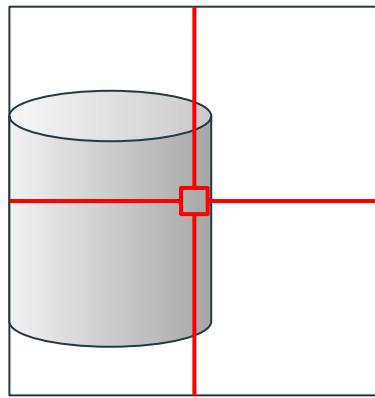
# Наблюдение...

---



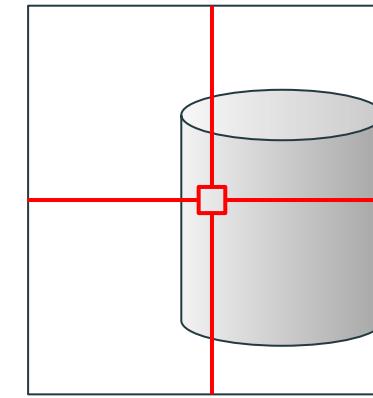
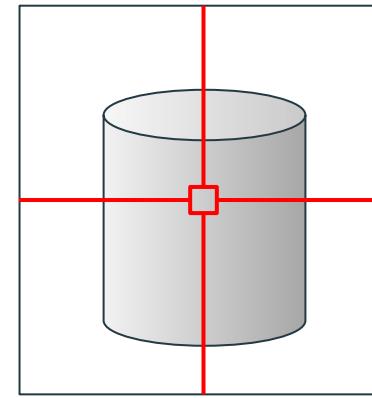
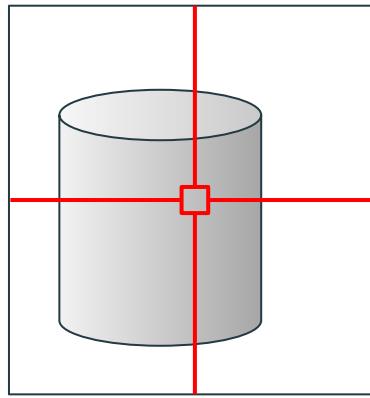
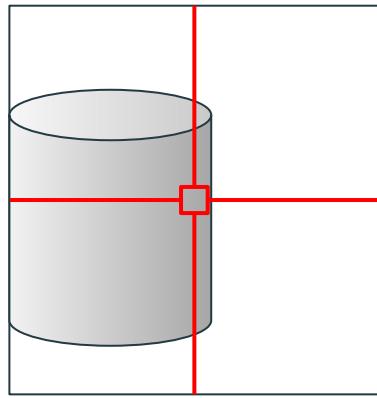
# Наблюдение...

---



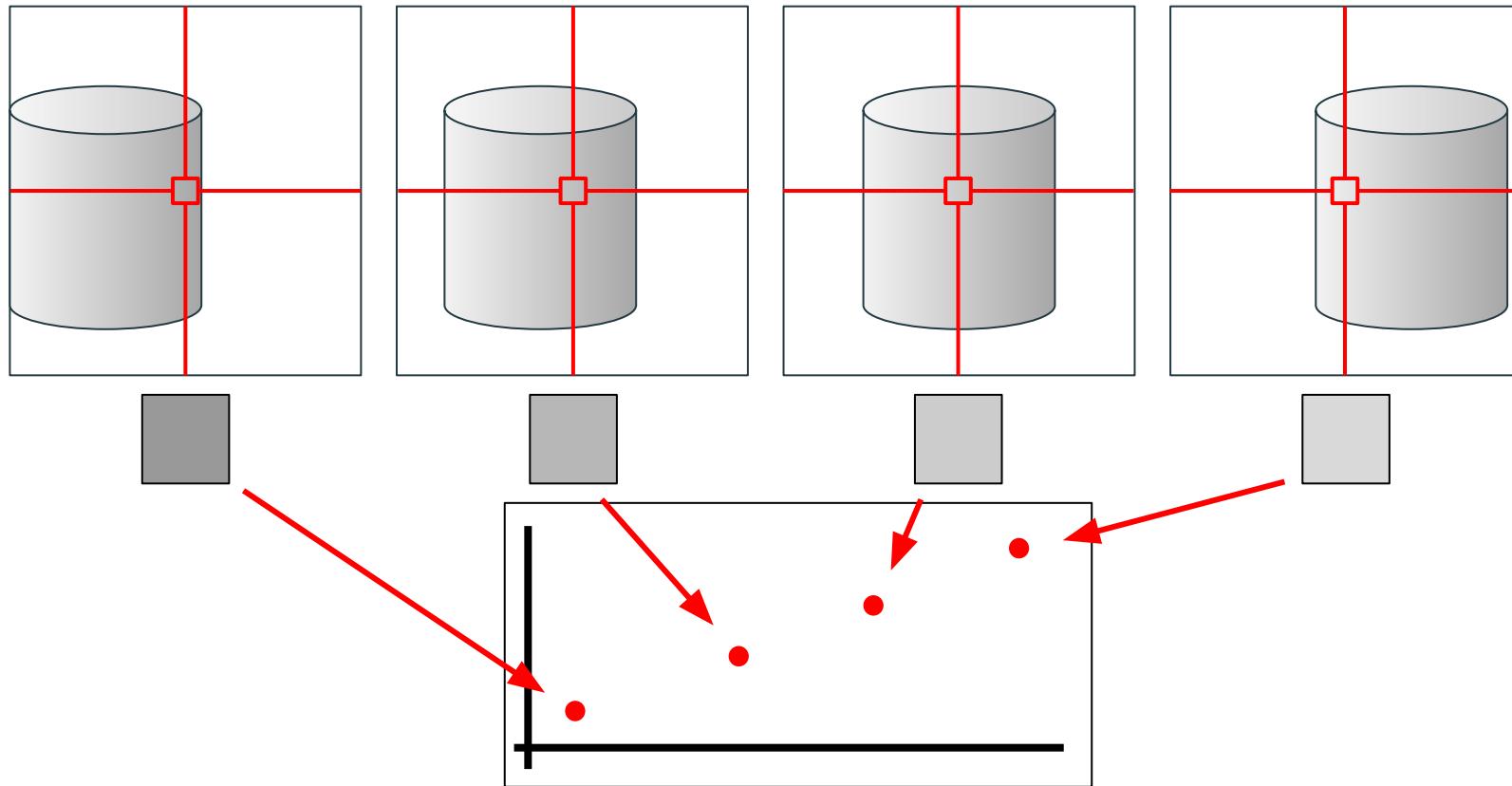
# Наблюдение...

---



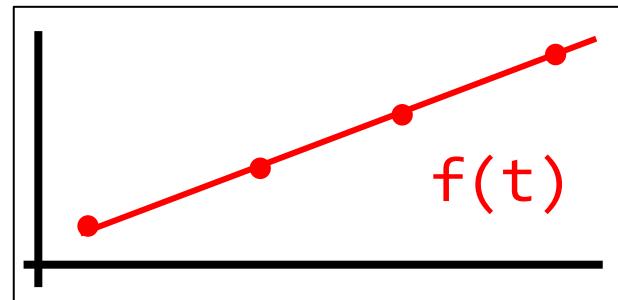
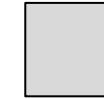
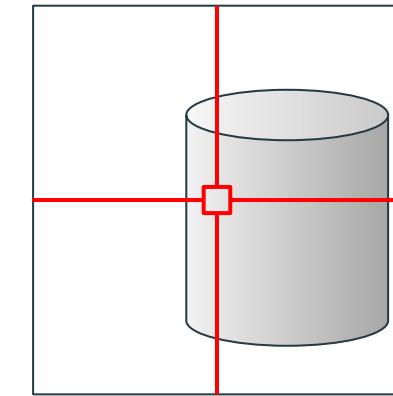
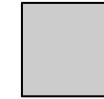
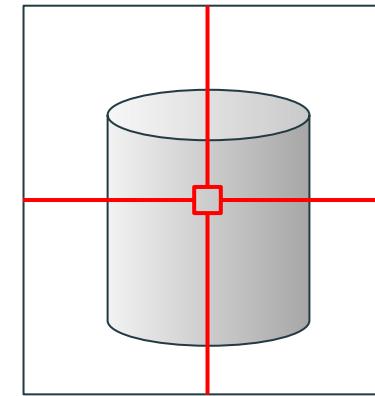
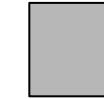
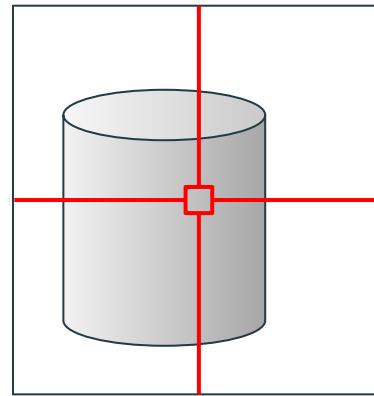
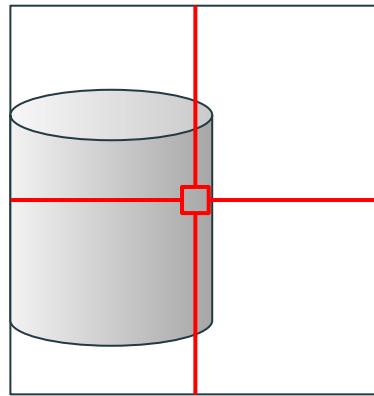
# Наблюдение...

---



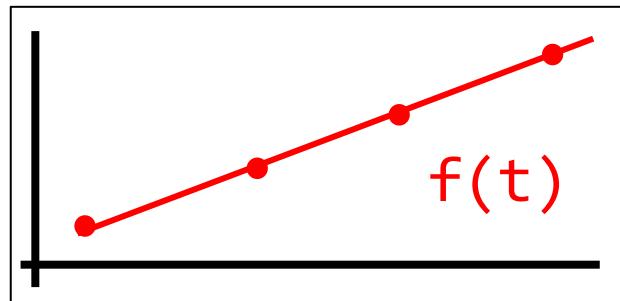
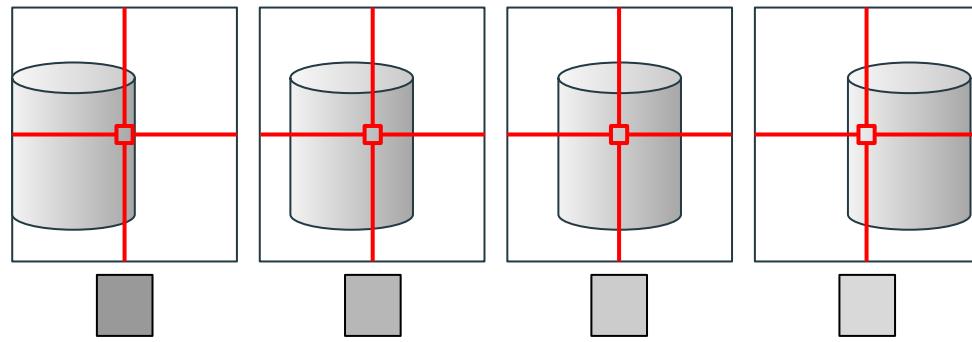
# Наблюдение...

---



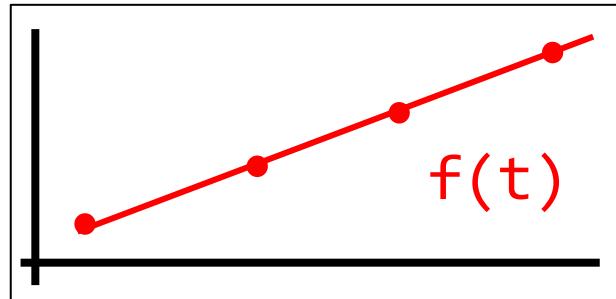
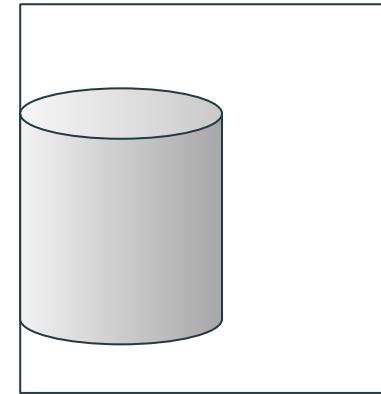
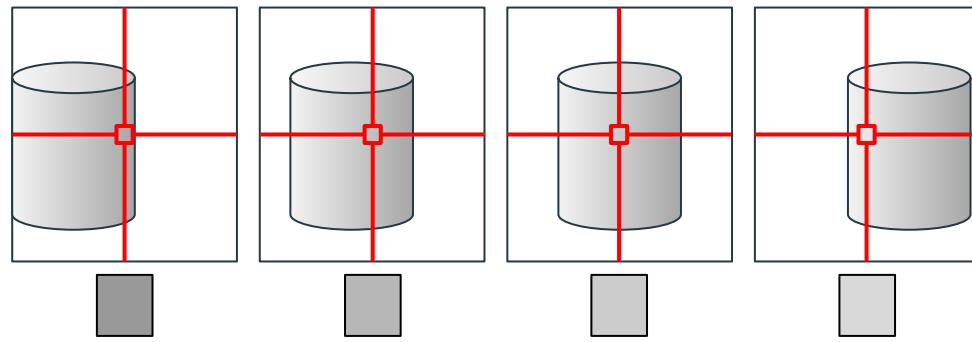
# Наблюдение...

---



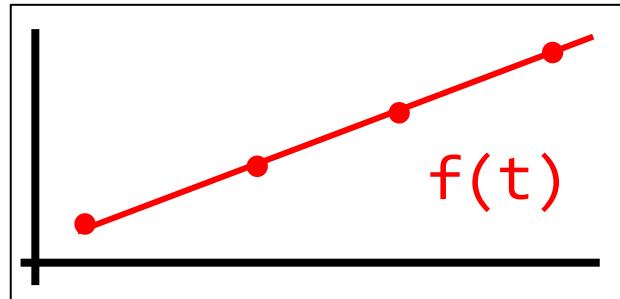
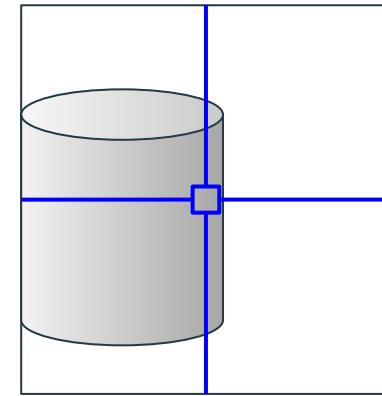
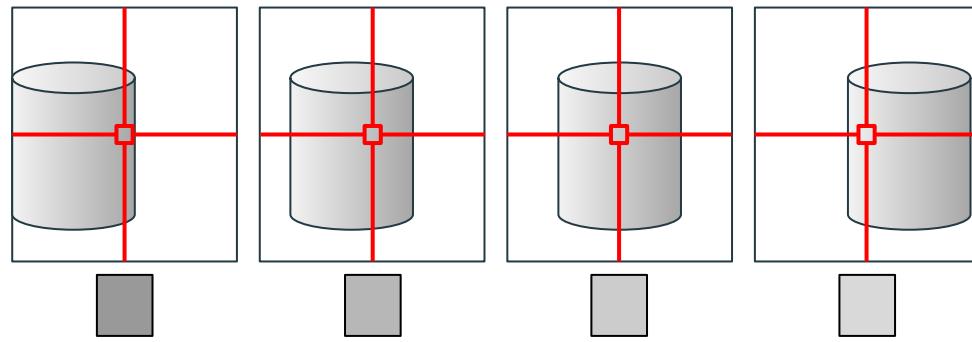
# Наблюдение...

---



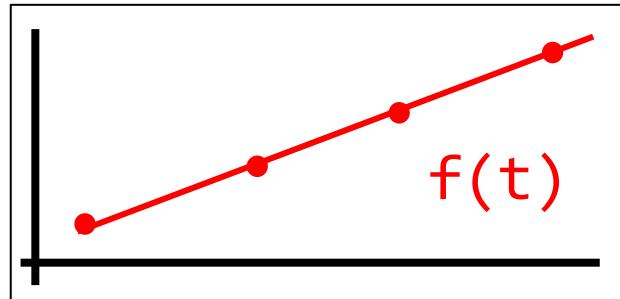
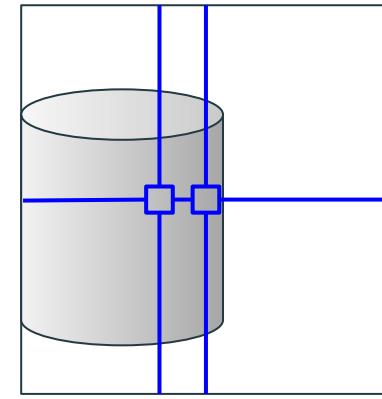
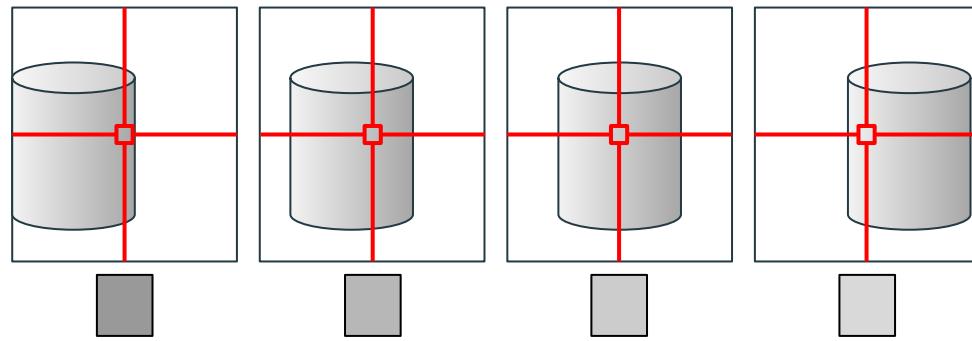
# Наблюдение...

---



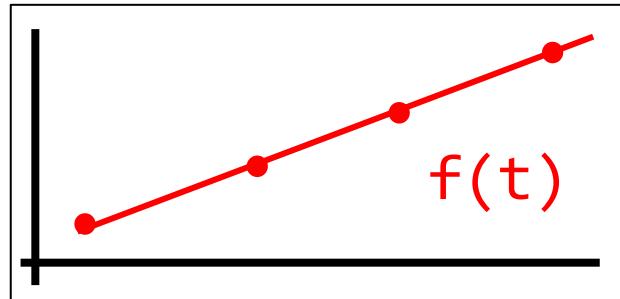
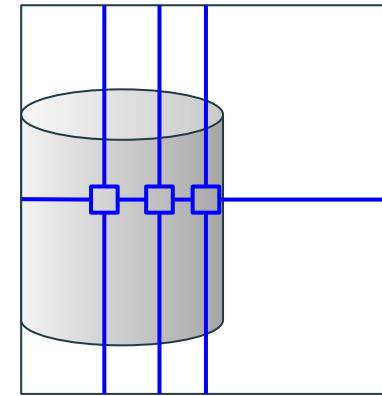
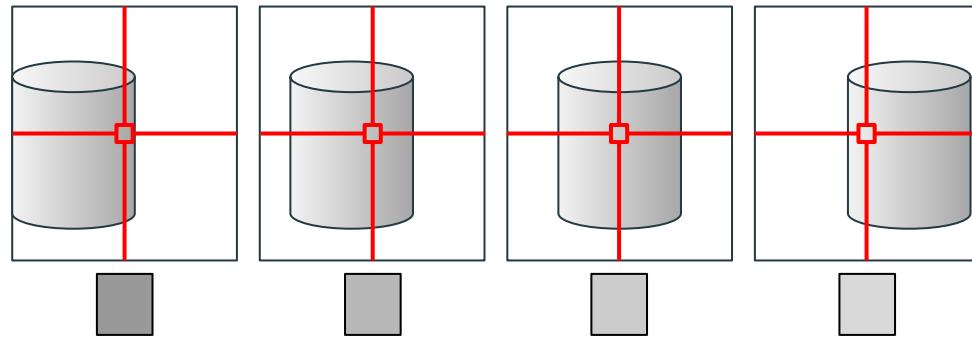
# Наблюдение...

---



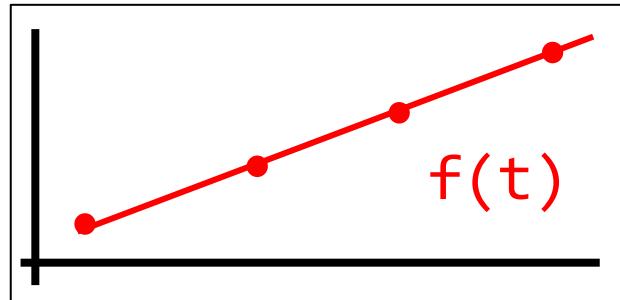
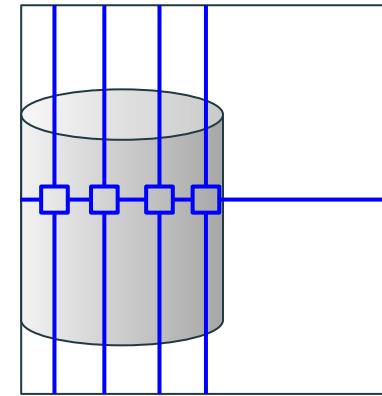
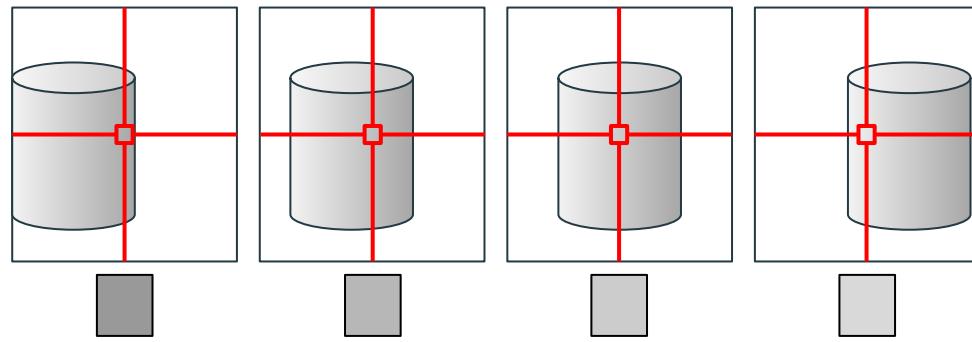
# Наблюдение...

---



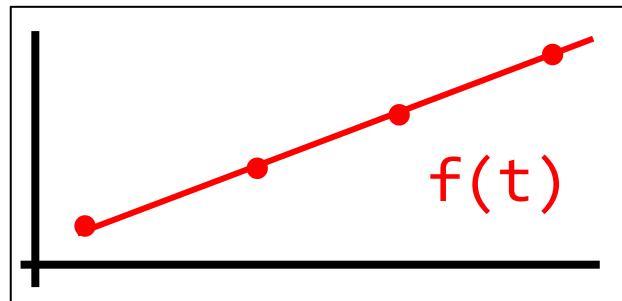
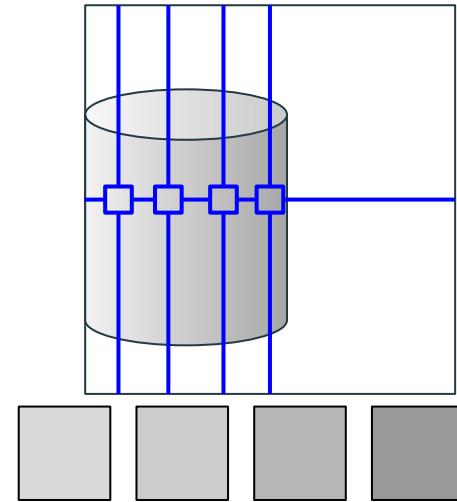
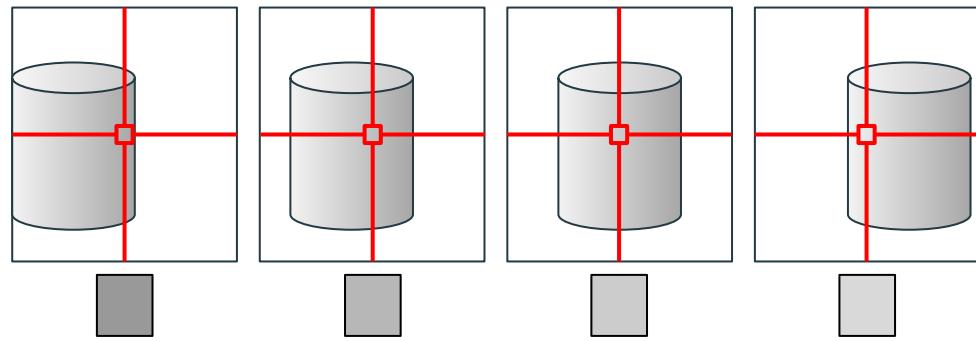
# Наблюдение...

---



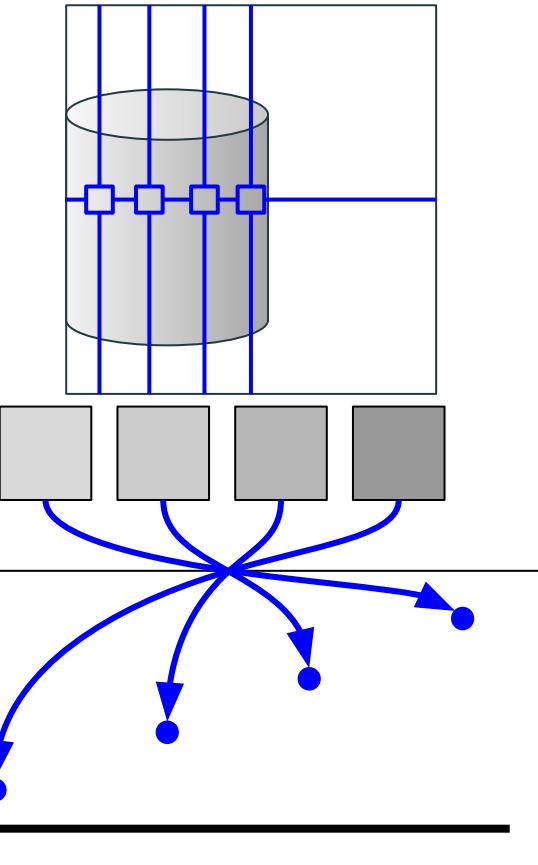
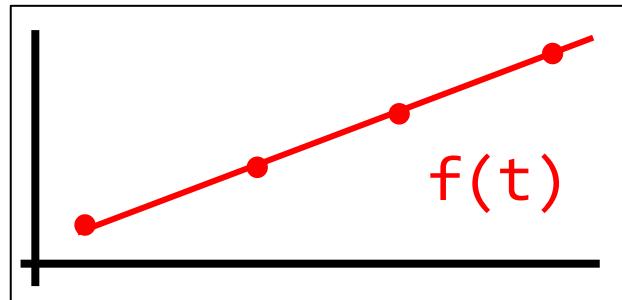
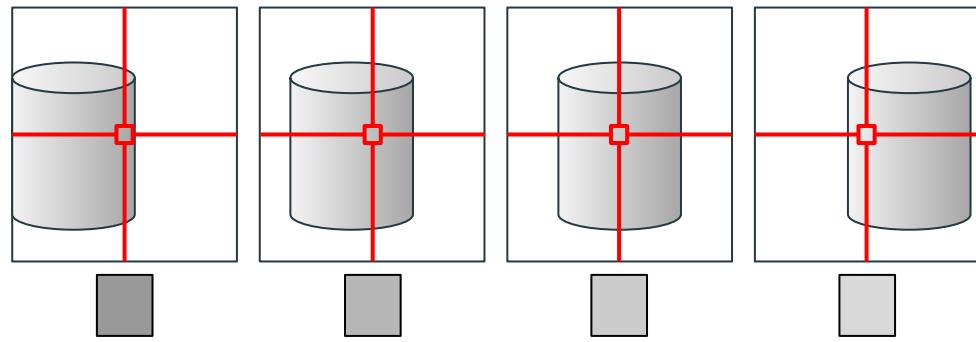
# Наблюдение...

---



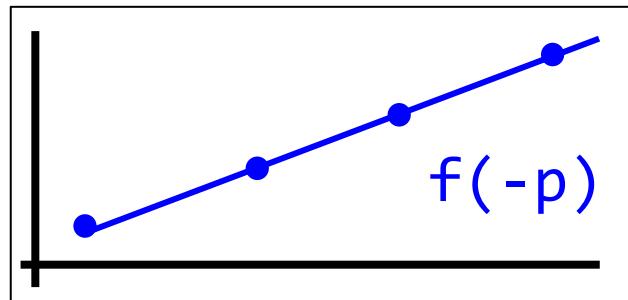
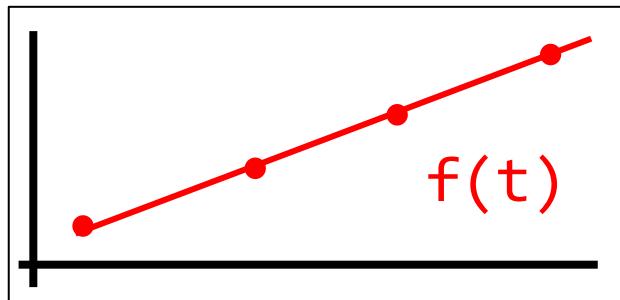
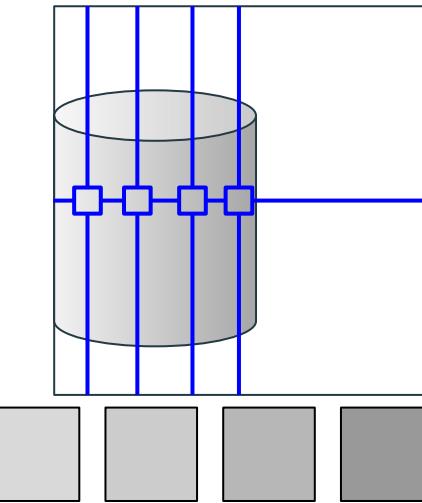
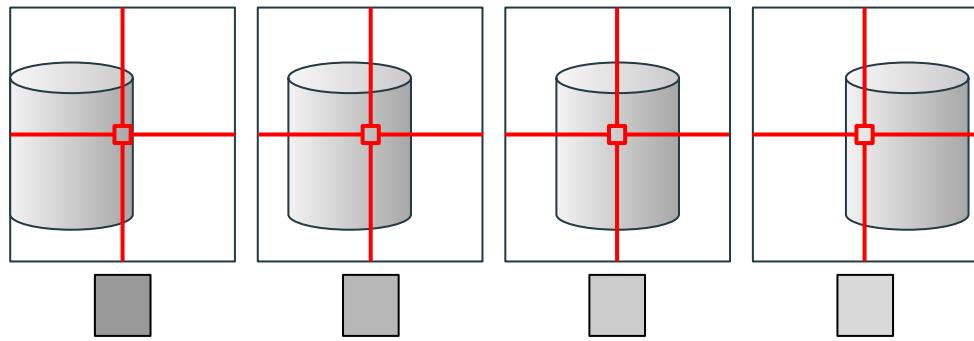
# Наблюдение...

---



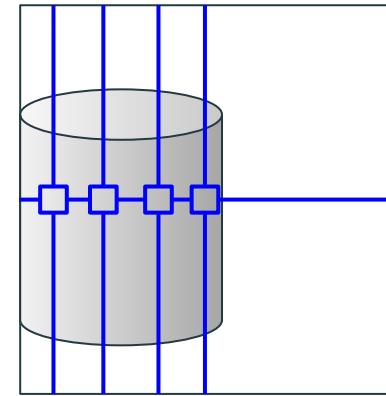
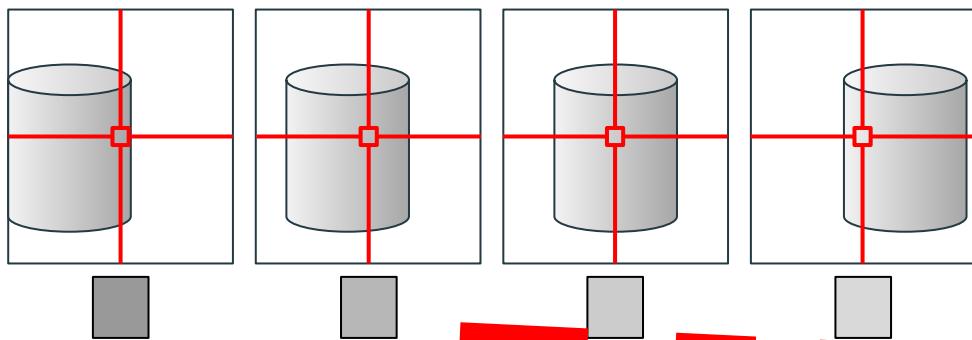
# Наблюдение...

---

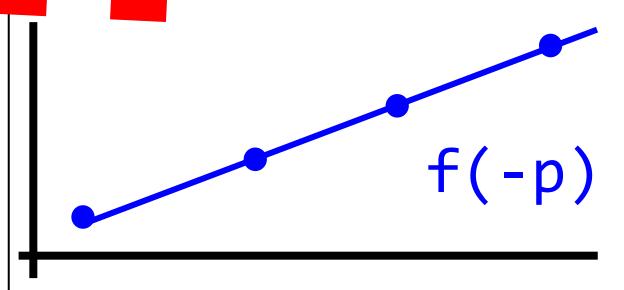
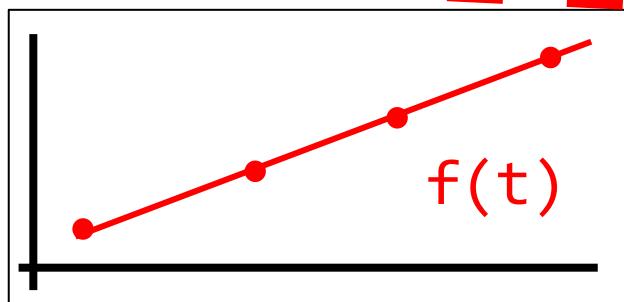
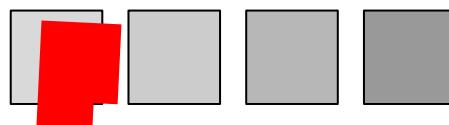


# Наблюдение...

---

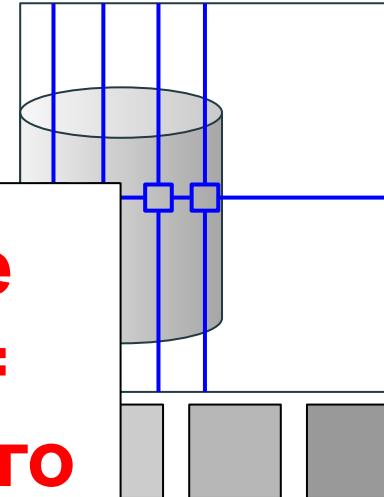
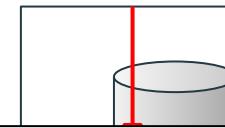
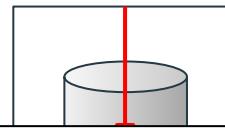
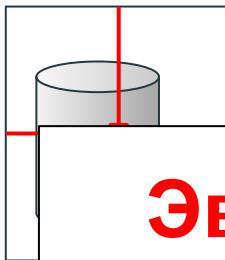
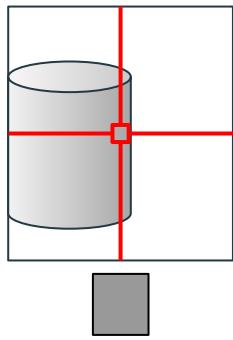


**SAME!**

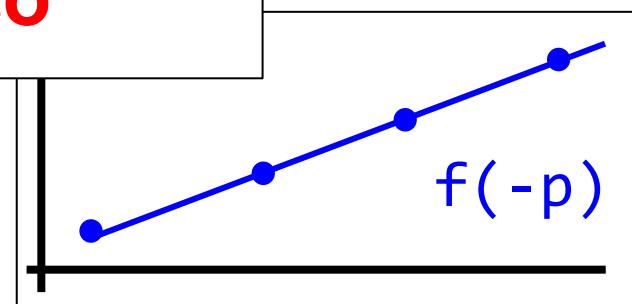
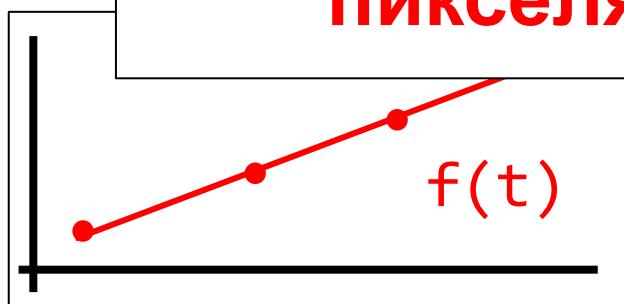


# An observation...

---



**Эвристика: Смещение  
объекта по времени =  
смещение наблюдаемого  
пикселя направо**



# Lucas-Kanade Optical Flow

---

Как использовать эту эвристику для расчета Optical flow?

# Lucas-Kanade Optical Flow

---

Как использовать эту эвристику для расчета Optical flow?

B.D. Lucas, T. Kanade, “An Image Registration Technique with an Application to Stereo Vision”, in Proceedings of Image Understanding Workshop, 1981, pp. 121-130.

## Lucas-Kanade Optical Flow

---

Предполагаем, что объект (пиксель) сместился на величину  $\Delta p$  за время  $\Delta t$

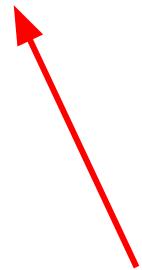
$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

## Lucas-Kanade Optical Flow

---

Предполагаем, что объект (пиксель) сместился на величину  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

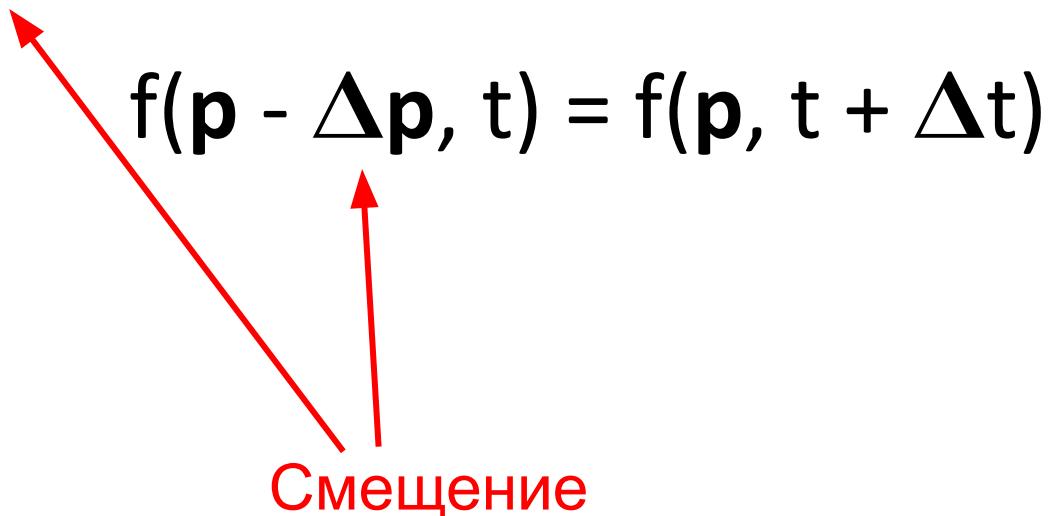


Наблюдаемый пиксель

# Lucas-Kanade Optical Flow

---

Предполагаем, что объект (пиксель) сместился на величину  $\Delta p$  за время  $\Delta t$



# Lucas-Kanade Optical Flow

---

Предполагаем, что объект (пиксель) сместился на величину  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Первое  
изображение

Второе  
изображение

## Lucas-Kanade Optical Flow

---

Предполагаем, что объект (пиксель) сместился на величину  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$



## Lucas-Kanade Optical Flow

---

Предполагаем, что объект (пиксель) сместился на величину  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Апроксимируем рядом Тейлора (до первого порядка)

## Lucas-Kanade Optical Flow

---

Предполагаем, что объект (пиксель) сместился на величину  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Апроксимируем рядом Тейлора (до первого порядка)

Что такое ряд  
Тейлора?

# Ряд Тейлора

---

Описывает сложную функцию полиномом  $n$ -ого порядка

# Ряд Тейлора

---

Описывает сложную функцию полиномом n-ого порядка

$$f(x) \approx \sum_i a_i x^i$$

# Ряд Тейлора

---

Описывает сложную функцию полиномом n-ого порядка

$$f(x) \approx \sum_i a_i x^i$$

Первый порядок:  $i=0, 1$

$$\approx a_1 x^1 + a_0 x^0$$

$$\approx mx + b$$

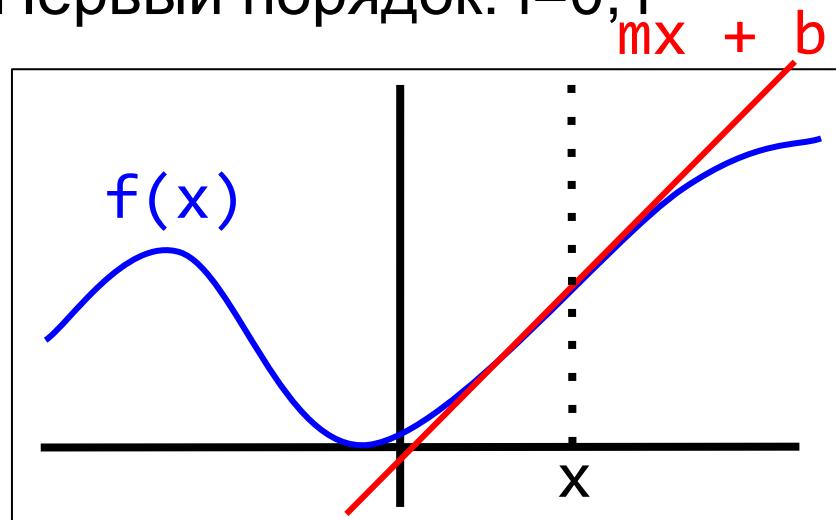
# Quick & Dirty: Taylor Expansion

---

Описывает сложную функцию полиномом n-ого порядка

$$f(x) \approx \sum_i a_i x^i$$

Первый порядок:  $i=0, 1$



$$\approx a_1 x^1 + a_0 x^0$$

$$\approx mx + b$$

# Quick & Dirty: Taylor Expansion

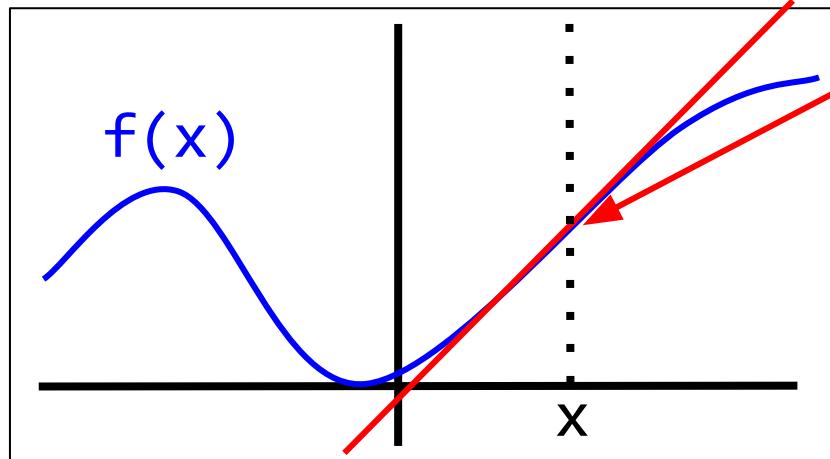
Описывает сложную функцию полиномом n-ого порядка

$$f(x) \approx \sum_i a_i x^i$$

Первый порядок:  $i=0, 1$

$$mx + b$$

Равны в  
точке  $x$



$$\approx a_1 x^1 + a_0 x^0$$

$$\approx mx + b$$

## Lucas-Kanade Optical Flow

---

Предполагаем, что объект сместился на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Раскладываем в ряд Тейлора до первого порядка

$$m(p - \Delta p) + b \approx f(p, t + \Delta t)$$

## Lucas-Kanade Optical Flow

---

Предполагаем, что объект сместился на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Раскладываем в ряд Тейлора до первого порядка

$$m(p - \Delta p) + b \approx f(p, t + \Delta t)$$

$$mp - m\Delta p + b \approx f(p, t + \Delta t)$$

## Lucas-Kanade Optical Flow

---

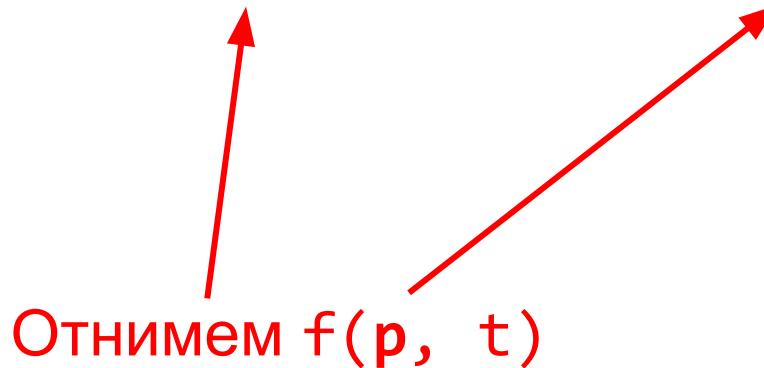
$$m\mathbf{p} - m\Delta\mathbf{p} + \mathbf{b} \approx \mathbf{f}(\mathbf{p}, t + \Delta t)$$

# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$



## Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$

# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$

Подставляем  $m\mathbf{p} + b$   
вместо  $f(\mathbf{p}, t)$ , и все  
сокращается!

## Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$

$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

## Lucas-Kanade Optical Flow

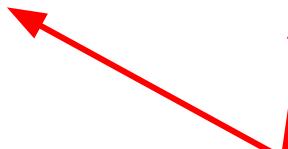
---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$

$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$



Это мы знаем

# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$

$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Это то что мы  
хотим узнать

Это мы знаем

# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$

А что это??

$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Это то что мы  
хотим узнать

Это мы знаем

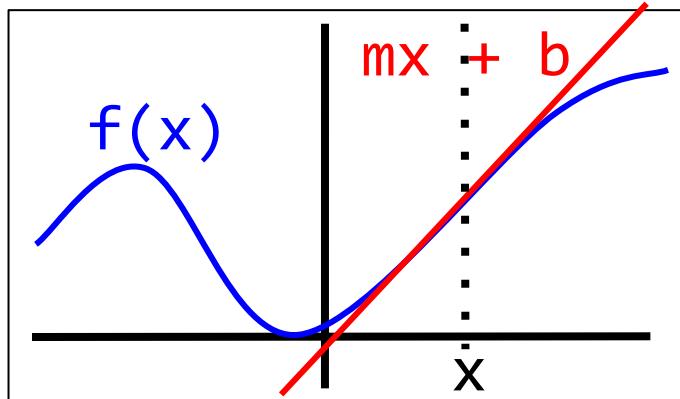
# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$



$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$

Наклон прямой

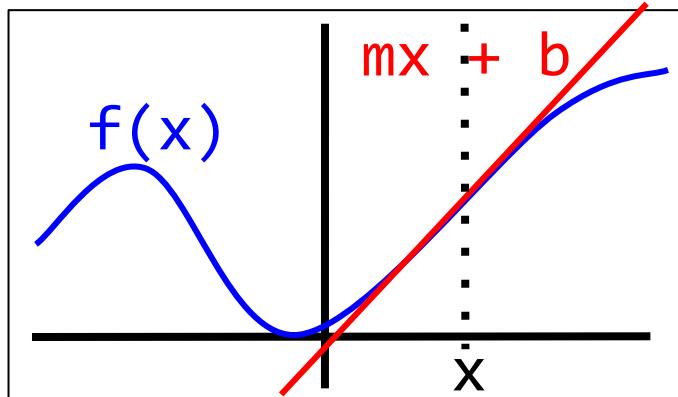
# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$



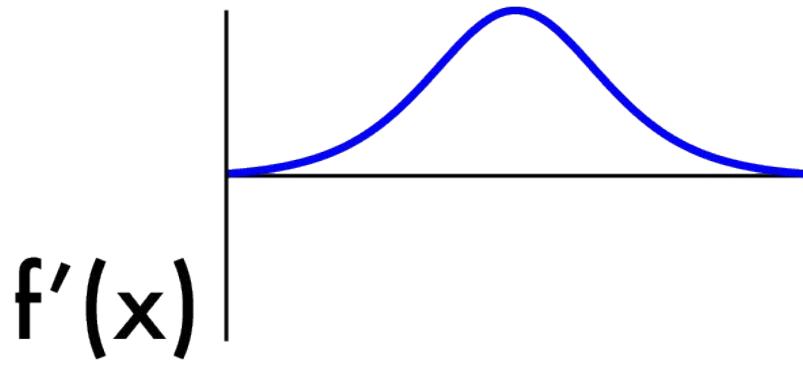
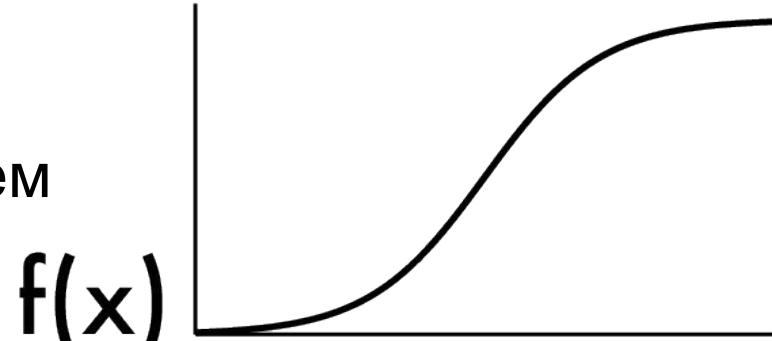
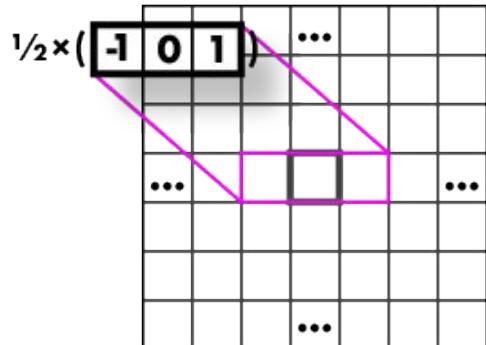
$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Наклон прямой = производная  
= градиент

# Вспомним: Image derivatives

---

- Вспомним:
  - $f'(a) = \lim_{h \rightarrow 0} \frac{f(a + h) - f(a)}{h}$ .
- У нас нет “настоящей”  
Функции  $\rightarrow$  численно оцениваем
- Давайте подставим  $h = 2$
- Как это будет выглядеть?



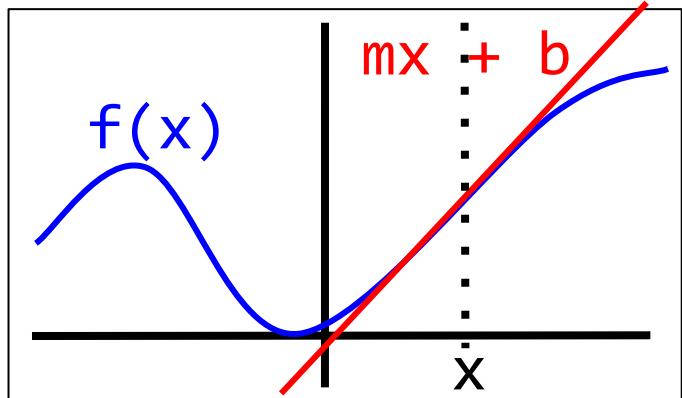
# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(x, t) = mx + b$



$$\begin{aligned} -m\Delta\mathbf{p} &\approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t) \\ m &= d\mathbf{p} = \begin{bmatrix} dx \\ dy \end{bmatrix} \end{aligned}$$

# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(\mathbf{x}, t) = m\mathbf{x} + b$

$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

$$-\mathbf{d}\mathbf{p}\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

## Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(\mathbf{x}, t) = m\mathbf{x} + b$

$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

$$-\mathbf{d}\mathbf{p}\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Избавимся от векторной нотации

# Lucas-Kanade Optical Flow

---

$$m\mathbf{p} - m\Delta\mathbf{p} + b \approx f(\mathbf{p}, t + \Delta t)$$

$$m\mathbf{p} - m\Delta\mathbf{p} + b - f(\mathbf{p}, t) \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

Вспомним, что  $f(\mathbf{x}, t) = m\mathbf{x} + b$

$$-m\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

$$-\mathbf{d}\mathbf{p}\Delta\mathbf{p} \approx f(\mathbf{p}, t + \Delta t) - f(\mathbf{p}, t)$$

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

# Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

## Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$

# Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^* \textcolor{red}{u} + dy^* \textcolor{red}{v} = I_t[x,y] - I_{t+\Delta t}[x,y]$$

# Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$

## Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$

# Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^*u + dy^*v = I_t[x,y] \rightarrow I_{t+\Delta t}[x,y]$$

## Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$

## Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^* \textcolor{red}{u} + dy^* \textcolor{red}{v} = I_t[x,y] - I_{t+\Delta t}[x,y]$$

Решим относительно  $\textcolor{red}{u}, \textcolor{red}{v}$

# Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^* \textcolor{red}{u} + dy^* \textcolor{red}{v} = I_t[x,y] - I_{t+\Delta t}[x,y]$$

Решим относительно  $\textcolor{red}{u}, \textcolor{red}{v}$

1 уравнение, 2 неизвестных

# Lucas-Kanade Optical Flow

---

$$-dx\Delta x + -dy\Delta y \approx f((x,y), t + \Delta t) - f((x,y), t)$$

Немножко изменим обозначения:

$$dx^* \textcolor{red}{u} + dy^* \textcolor{red}{v} = I_t[x,y] - I_{t+\Delta t}[x,y]$$

Решим относительно  $\textcolor{red}{u}, \textcolor{red}{v}$

1 уравнение, 2 неизвестных

:(

# Lucas-Kanade Optical Flow

---

Вернемся назад

$$dx^* \mathbf{u} + dy^* \mathbf{v} = I_t[x, y] - I_{t+\Delta t}[x, y]$$

# Lucas-Kanade Optical Flow

---

Вернемся назад

$$dx^* u + dy^* v = I_t[x, y] - I_{t+\Delta t}[x, y]$$

В чем смысл этого выражения?

## Lucas-Kanade Optical Flow

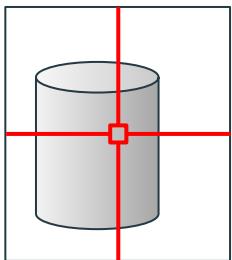
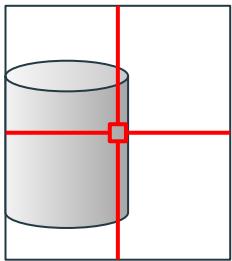
---

$$dx^* \mathbf{u} + dy^* \mathbf{v} = I_t[x, y] - I_{t+\Delta t}[x, y]$$

# Lucas-Kanade Optical Flow

---

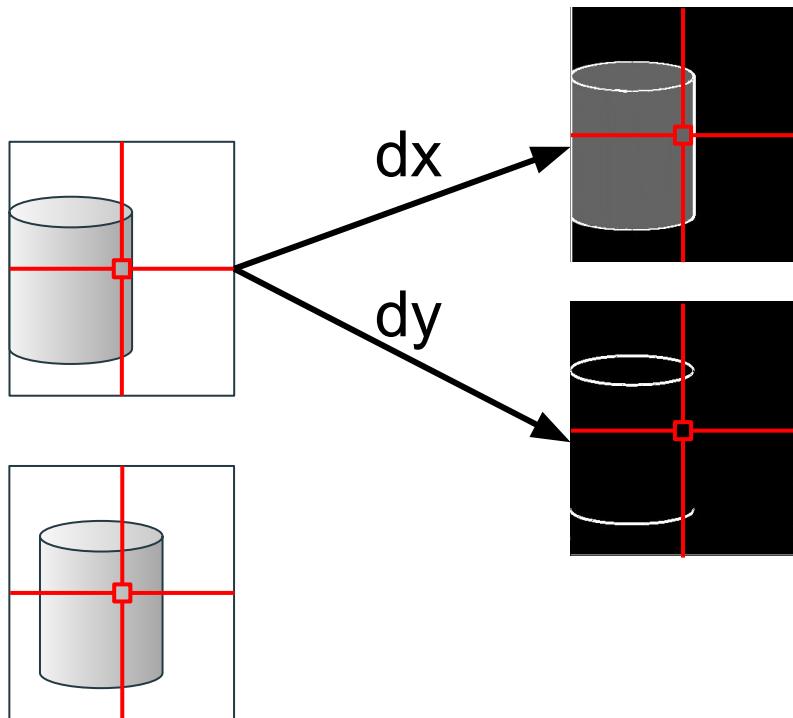
$$dx^* \mathbf{u} + dy^* \mathbf{v} = I_t[x, y] - I_{t+\Delta t}[x, y]$$



# Lucas-Kanade Optical Flow

---

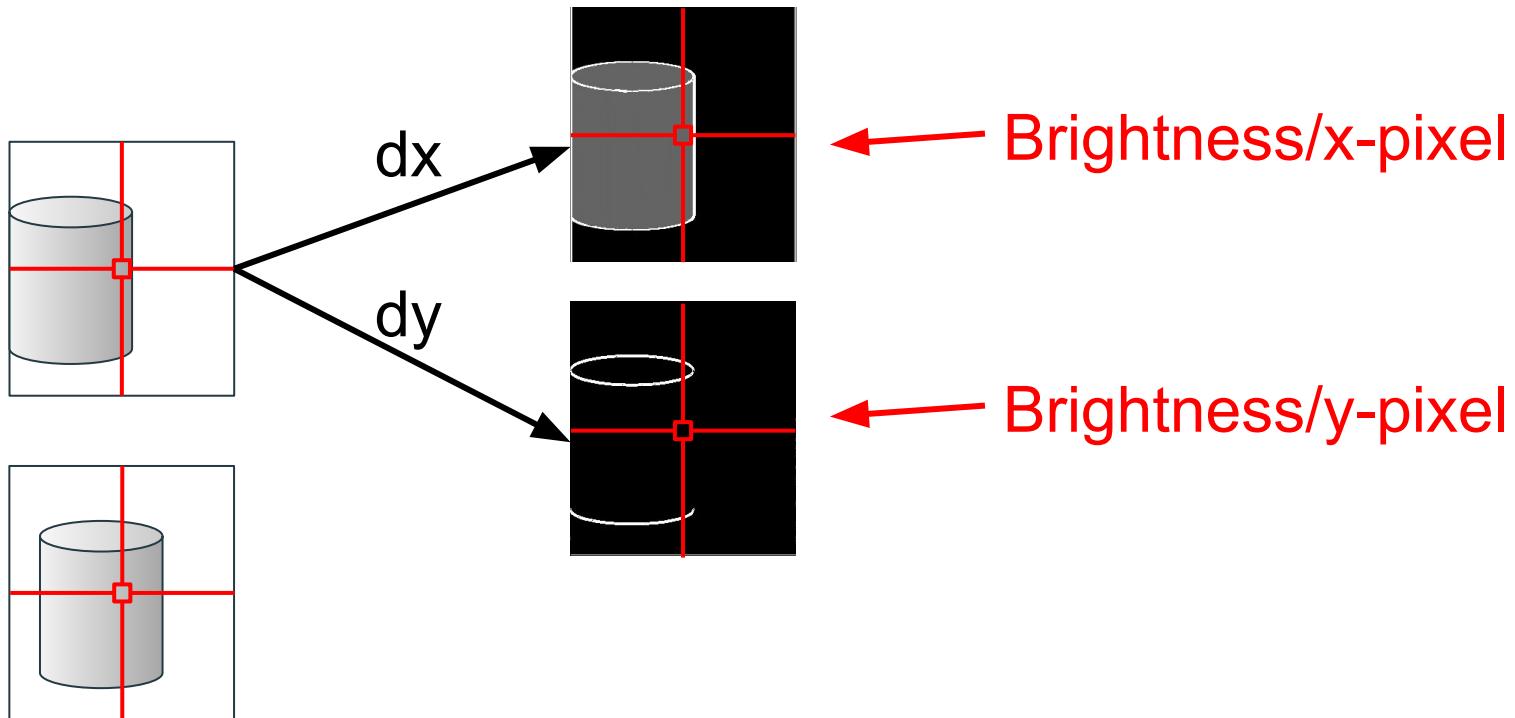
$$dx^* u + dy^* v = I_t[x, y] - I_{t+\Delta t}[x, y]$$



# Lucas-Kanade Optical Flow

---

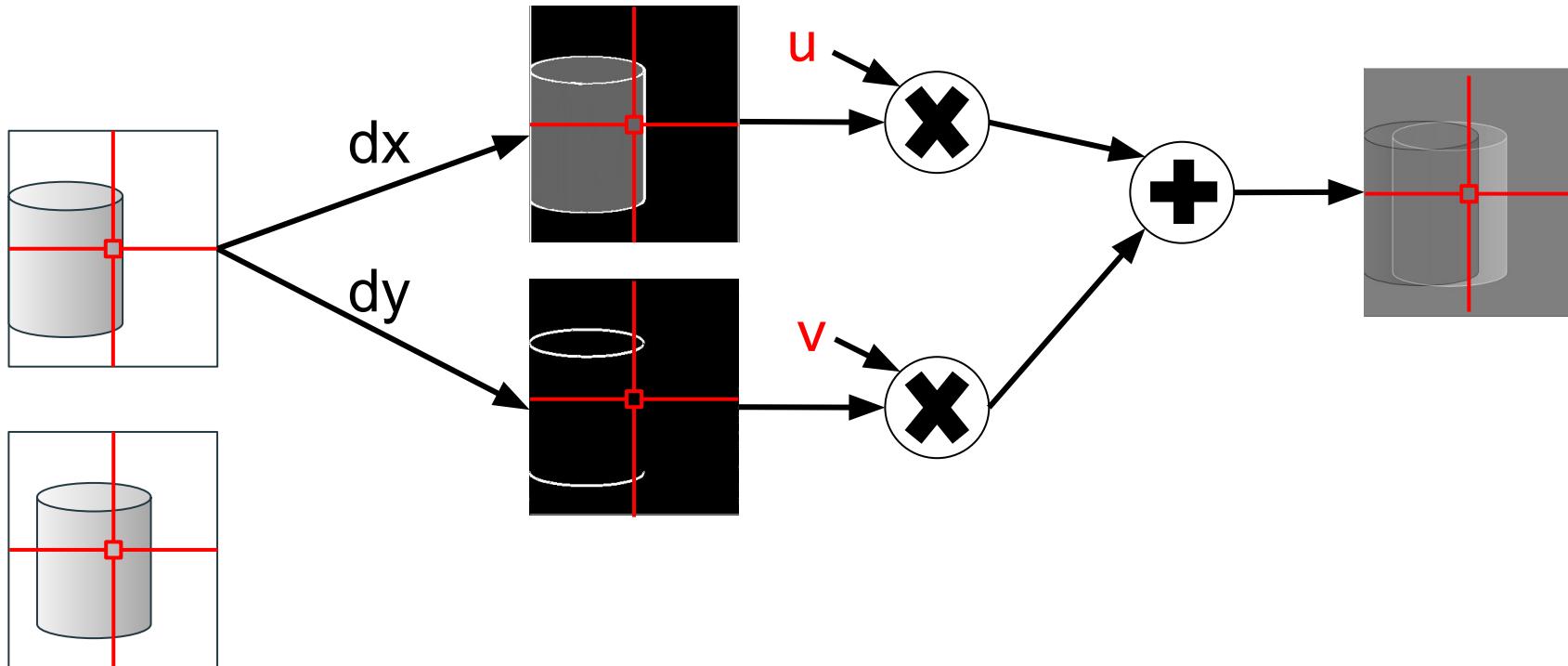
$$dx^* u + dy^* v = I_t[x, y] - I_{t+\Delta t}[x, y]$$



# Lucas-Kanade Optical Flow

---

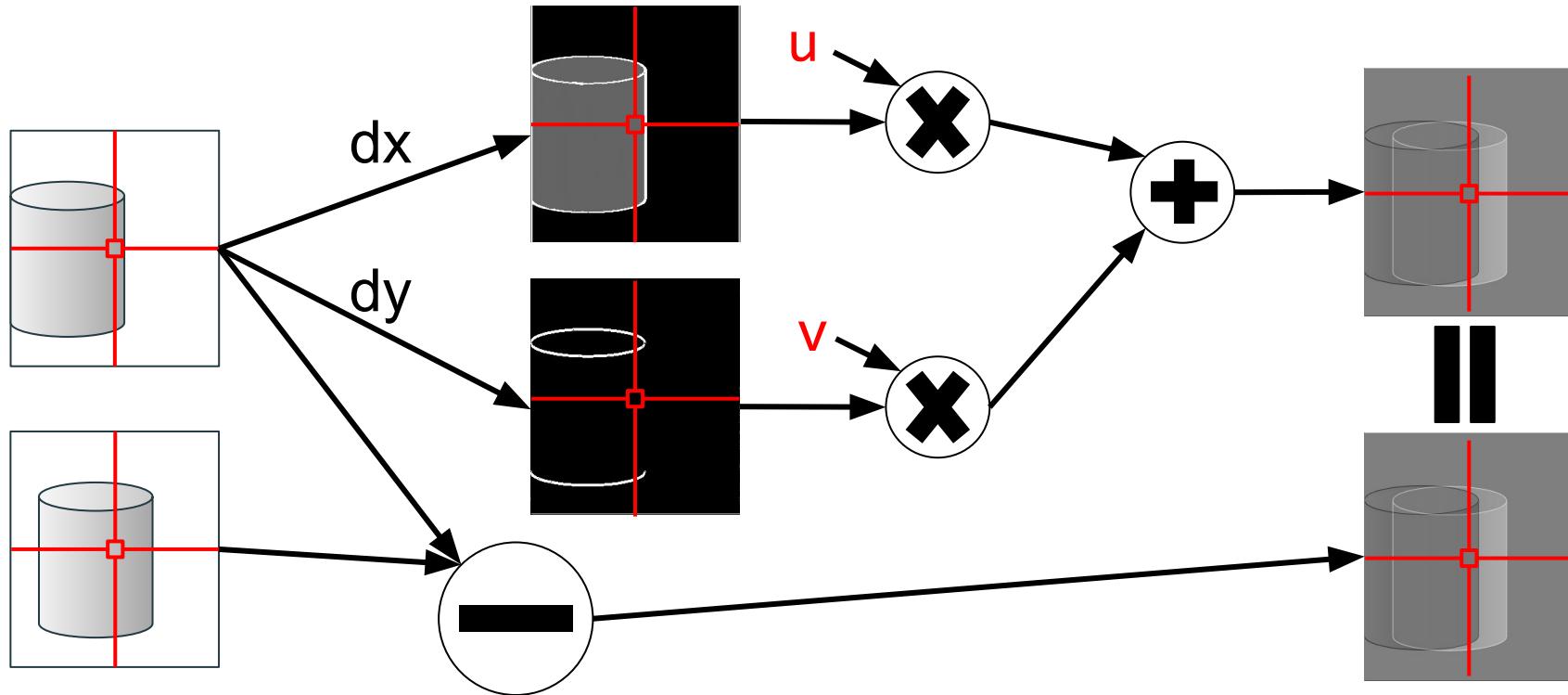
$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$



# Lucas-Kanade Optical Flow

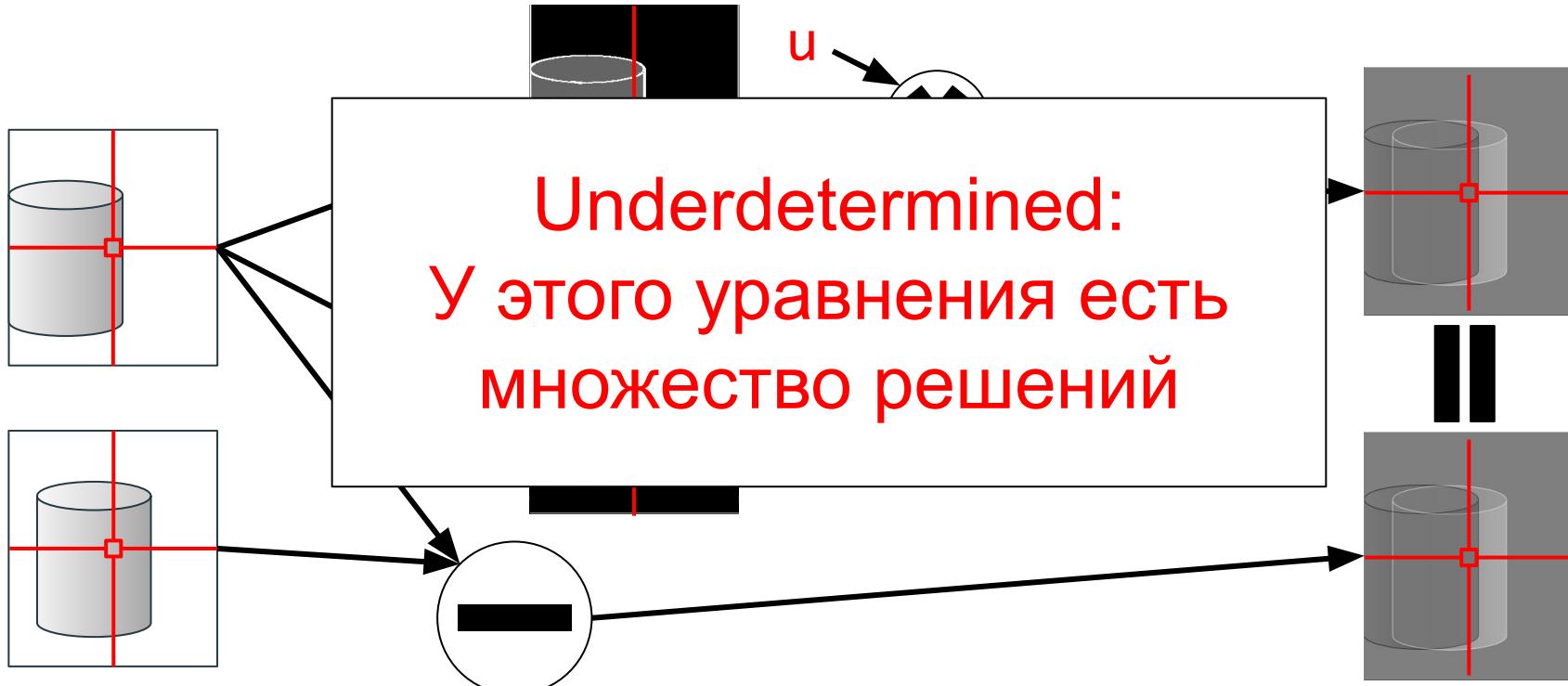
---

$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$



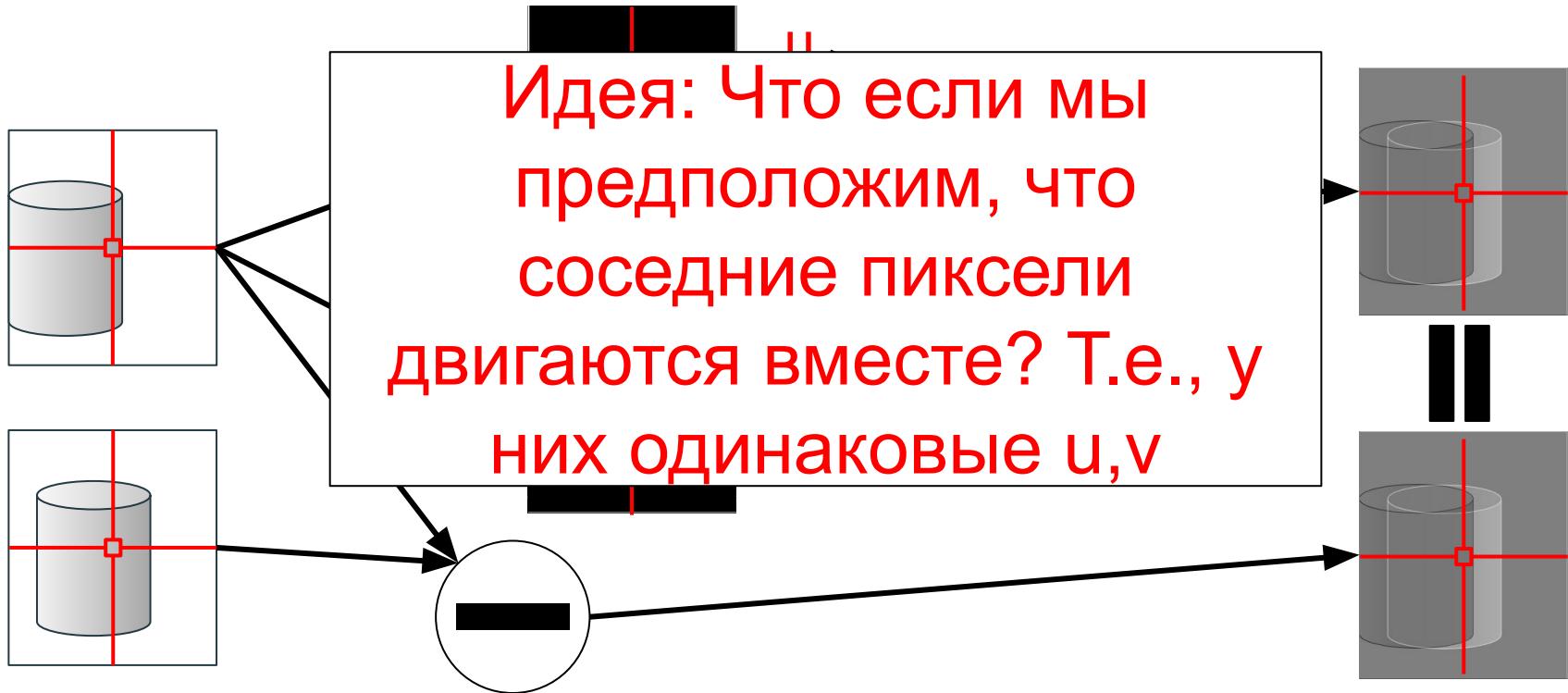
# Lucas-Kanade Optical Flow

$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$



# Lucas-Kanade Optical Flow

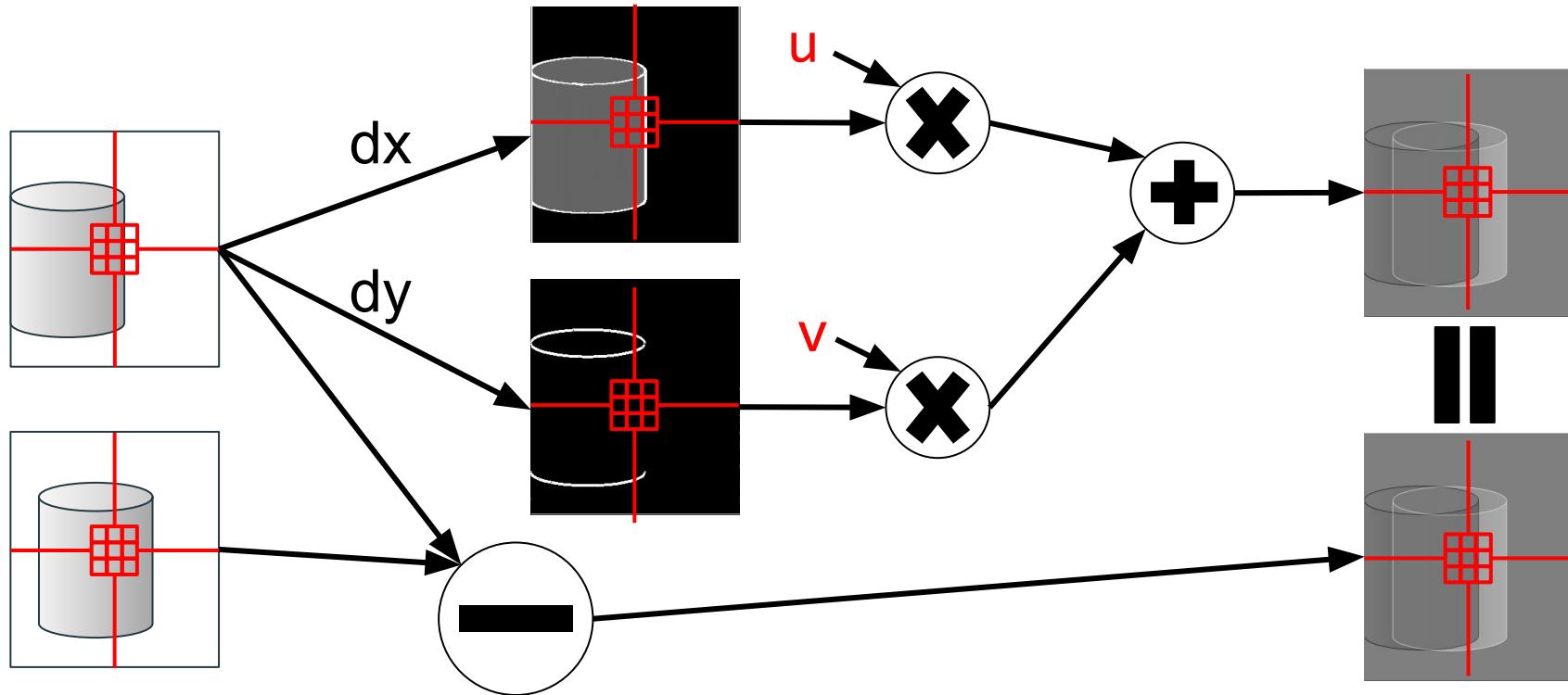
$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$



# Lucas-Kanade Optical Flow

---

$$dx^*u + dy^*v = I_t[x,y] - I_{t+\Delta t}[x,y]$$



## Lucas-Kanade Optical Flow

---

$$dx_1 * \textcolor{red}{u} + dy_1 * \textcolor{red}{v} = I_t[x_1, y_1] - I_{t+\Delta t}[x_1, y_1]$$

$$dx_2 * \textcolor{red}{u} + dy_2 * \textcolor{red}{v} = I_t[x_2, y_2] - I_{t+\Delta t}[x_2, y_2]$$

...

$$dx_9 * \textcolor{red}{u} + dy_9 * \textcolor{red}{v} = I_t[x_9, y_9] - I_{t+\Delta t}[x_9, y_9]$$

9 уравнений, 2 неизвестных

## Lucas-Kanade Optical Flow

---

$$dx_1 * \textcolor{red}{u} + dy_1 * \textcolor{red}{v} = I_t[x_1, y_1] - I_{t+\Delta t}[x_1, y_1]$$

$$dx_2 * \textcolor{red}{u} + dy_2 * \textcolor{red}{v} = I_t[x_2, y_2] - I_{t+\Delta t}[x_2, y_2]$$

...

$$dx_9 * \textcolor{red}{u} + dy_9 * \textcolor{red}{v} = I_t[x_9, y_9] - I_{t+\Delta t}[x_9, y_9]$$

9 уравнений, 2 неизвестных  $\rightarrow$  Overdetermined

# Lucas-Kanade Optical Flow

---

$$S = \begin{bmatrix} dx_1 & dy_1 \\ dx_2 & dy_2 \\ \dots \\ dx_9 & dy_9 \end{bmatrix} \quad \Delta p = \begin{bmatrix} u \\ v \end{bmatrix} \quad T = \begin{bmatrix} I_t[x_1, y_1] - I_{t+\Delta t}[x_1, y_1] \\ I_t[x_2, y_2] - I_{t+\Delta t}[x_2, y_2] \\ \dots \\ I_t[x_9, y_9] - I_{t+\Delta t}[x_9, y_9] \end{bmatrix}$$
$$S \Delta p = T$$

# Lucas-Kanade Optical Flow

---

$$S = \begin{bmatrix} dx_1 & dy_1 \\ dx_2 & dy_2 \\ \dots \\ dx_9 & dy_9 \end{bmatrix} \quad \Delta p = \begin{bmatrix} u \\ v \end{bmatrix} \quad T = \begin{bmatrix} I_t[x_1, y_1] - I_{t+\Delta t}[x_1, y_1] \\ I_t[x_2, y_2] - I_{t+\Delta t}[x_2, y_2] \\ \dots \\ I_t[x_9, y_9] - I_{t+\Delta t}[x_9, y_9] \end{bmatrix}$$

$$S\Delta p = T$$

Least-squares solution

$$\| S\Delta p - T \| ^2 = 0$$

$$\Delta p = (S^T S)^{-1} S^T T$$

# Lucas-Kanade Optical Flow

---

$$S = \begin{bmatrix} dx_1 & dy_1 \\ dx_2 & dy_2 \\ \dots \\ dx_9 & dy_9 \end{bmatrix}$$

$$\Delta p = \begin{bmatrix} u \\ v \end{bmatrix}$$

$$T = \begin{bmatrix} I_t[x_1, y_1] - I_{t+\Delta t}[x_1, y_1] \\ I_t[x_2, y_2] - I_{t+\Delta t}[x_2, y_2] \\ \dots \\ I_t[x_9, y_9] - I_{t+\Delta t}[x_9, y_9] \end{bmatrix}$$

$$S\Delta p = T$$

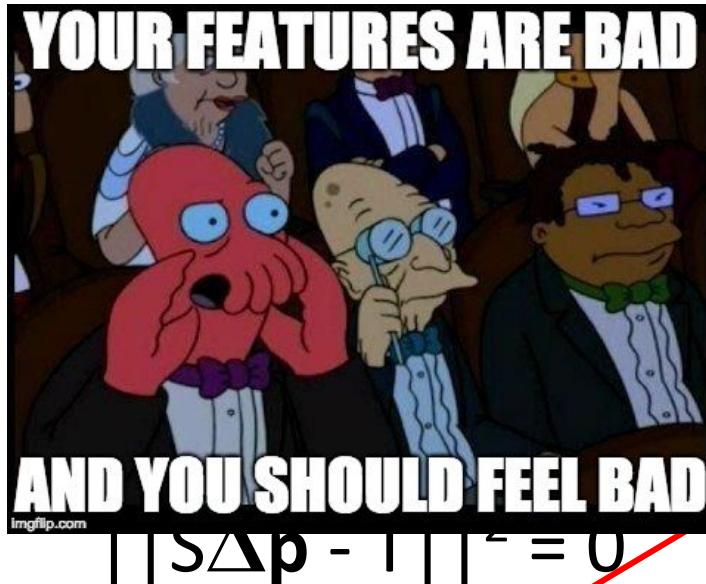
Что если это  
матрица  
необратима?

$$||S\Delta p - T||^2 = 0$$

$$\Delta p = (S^T S)^{-1} S^T T$$

# Lucas-Kanade Optical Flow

$$S = \begin{bmatrix} dx_1 & dy_1 \\ dx_2 & dy_2 \\ \dots \\ dx_9 & dy_9 \end{bmatrix}$$



$$\|S\Delta p - T\|^2 = 0$$

$$\Delta p = (S^T S)^{-1} S^T T$$

$$\begin{aligned} & - I_{t+\Delta t}[x_1, y_1] \\ & - I_{t+\Delta t}[x_2, y_2] \\ & \dots \\ & - I_{t+\Delta t}[x_9, y_9] \end{aligned}$$

Что если это

матрица

необратима?

нету структуры  
вокруг пикселя  
(x,y)

## Lucas-Kanade Optical Flow

---

Проверяем на обратимость:  $\Delta p = (S^T S)^{-1} S^T T$

## Lucas-Kanade Optical Flow

---

Проверяем на обратимость:  $\Delta p = (S^T S)^{-1} S^T T$

$S^T S$  - симметрична, а значит можно разложить как:

$$S^T S = U \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} U^T$$

## Lucas-Kanade Optical Flow

---

Проверяем на обратимость:  $\Delta p = (S^T S)^{-1} S^T T$

$S^T S$  - симметрична, а значит можно разложить как:

$$S^T S = U \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} U^T$$

$$\det(S^T S - \lambda I) = 0$$

# Lucas-Kanade Optical Flow

---

Проверяем на обратимость:  $\Delta p = (S^T S)^{-1} S^T T$

$S^T S$  - симметрична, а значит можно разложить как:

$$S^T S = U \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} U^T$$

$$\det(S^T S - \lambda I) = 0$$

$$\begin{vmatrix} \sum_i (dx_i)^2 - \lambda & \sum_i (dx_i)(dy_i) \\ \sum_i (dx_i)(dy_i) & \sum_i (dy_i)^2 - \lambda \end{vmatrix} = 0$$

## Lucas-Kanade Optical Flow

---

Проверяем на обратимость:  $\Delta p = (S^T S)^{-1} S^T T$

$S^T S$  - симметрична, а значит можно разложить как:

$$S^T S = U \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} U^T$$

$$\det(S^T S - \lambda I) = 0$$

$$\begin{vmatrix} \sum_i (dx_i)^2 - \lambda & \sum_i (dx_i)(dy_i) \\ \sum_i (dx_i)(dy_i) & \sum_i (dy_i)^2 - \lambda \end{vmatrix} = 0$$

Если либо  $\lambda_1$  или  $\lambda_2 = 0$ , значит  $S$  необратима и решения нету!

## Lucas-Kanade Optical Flow

---

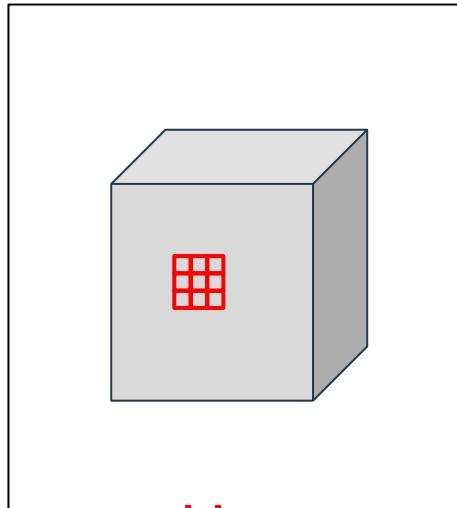
Как выбрать  $(x, y)$ ? Что значит быть “good feature”?

# Lucas-Kanade Optical Flow

---

Как выбрать  $(x,y)$ ? Что значит быть “хорошим признаком”?

Плохо



Нет

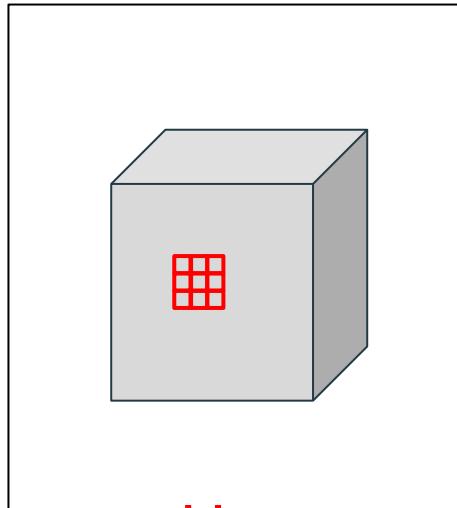
градиента

# Lucas-Kanade Optical Flow

---

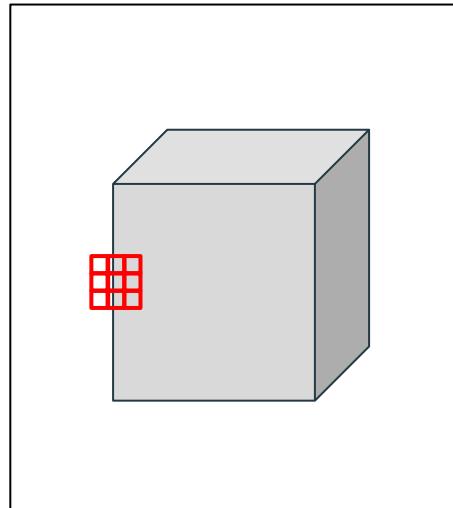
Как выбрать  $(x,y)$ ? Что значит быть “хорошим признаком”?

Плохо



Нет  
градиента

Получше

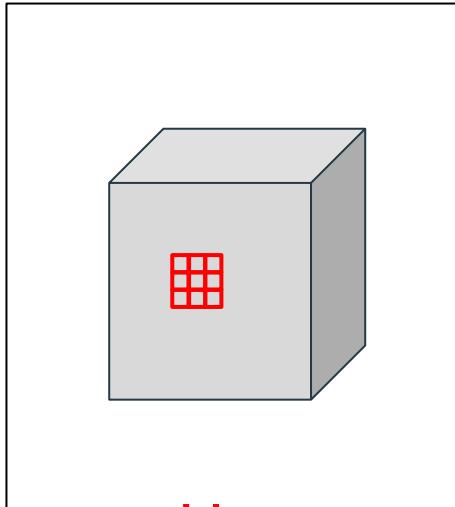


Только горизонтальное  
движение

# Lucas-Kanade Optical Flow

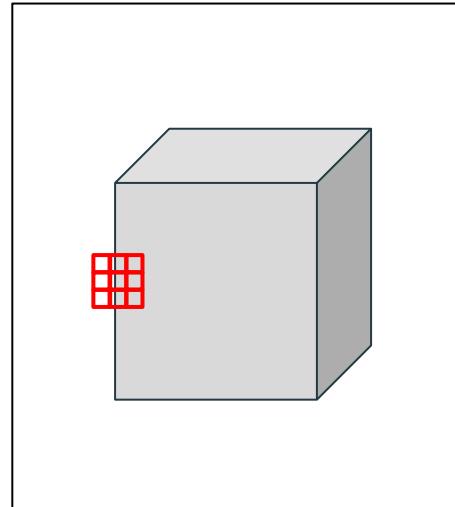
Как выбрать  $(x,y)$ ? Что значит быть “хорошим признаком”?

Плохо



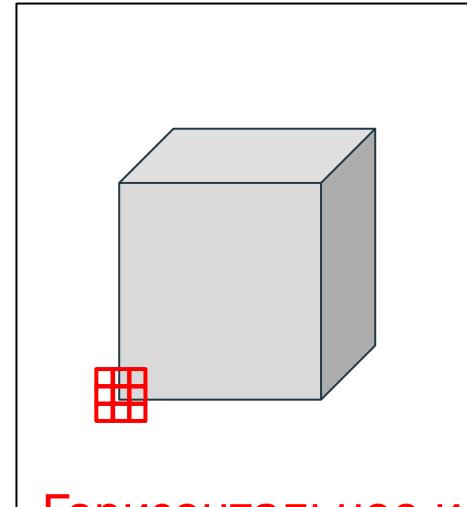
Нет  
градиента

Получше



Только горизонтальное  
движение

Супер



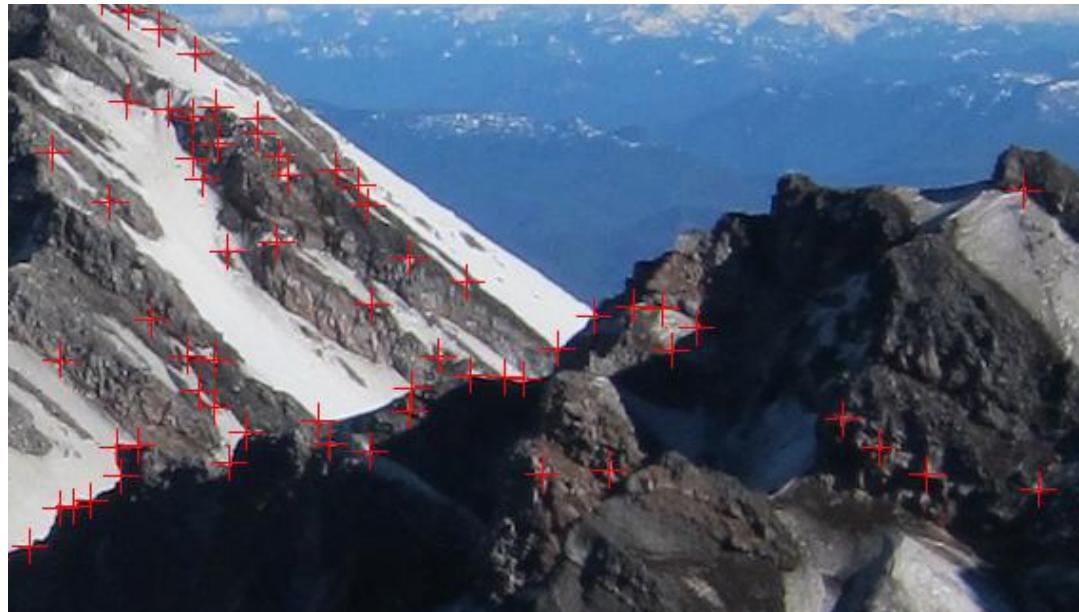
Горизонтальное и  
вертикальное  
движение

# Lucas-Kanade Optical Flow

---

Как выбрать  $(x, y)$ ? Что значит быть “good feature”?

УГЛЫ



# Lucas-Kanade Optical Flow

---

Когда алгоритм фейлится?

# Lucas-Kanade Optical Flow

---

Когда алгоритм фейлится?

-Изменения освещенности

-Большие движения

-Нет хороших фичей

-Проблема апертуры

# Lucas-Kanade Optical Flow

---

Когда алгоритм фейлится?

-Изменения освещенности

-Большие движения

-Нет хороших фичей

-Проблема апертуры

# Lucas-Kanade Optical Flow

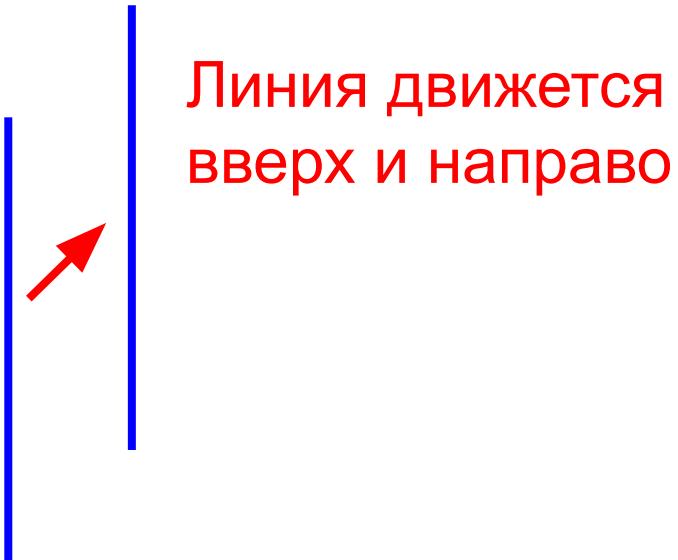
---

Проблема апертуры

# Lucas-Kanade Optical Flow

---

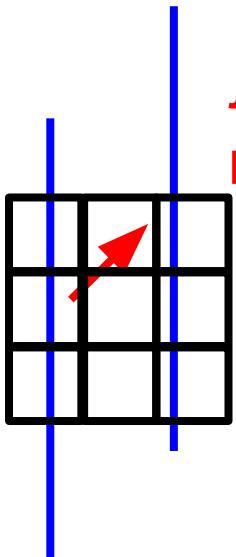
## Проблема апертуры



# Lucas-Kanade Optical Flow

---

## Проблема апертуры



Линия движется  
вверх и направо

# Lucas-Kanade Optical Flow

---

## Проблема апертуры



# Lucas-Kanade Optical Flow

---

Проблема апертуры - Barber pole illusion

## Lucas-Kanade Optical Flow

---

Проблема апертуры - Barber pole illusion



# Lucas-Kanade Optical Flow

---

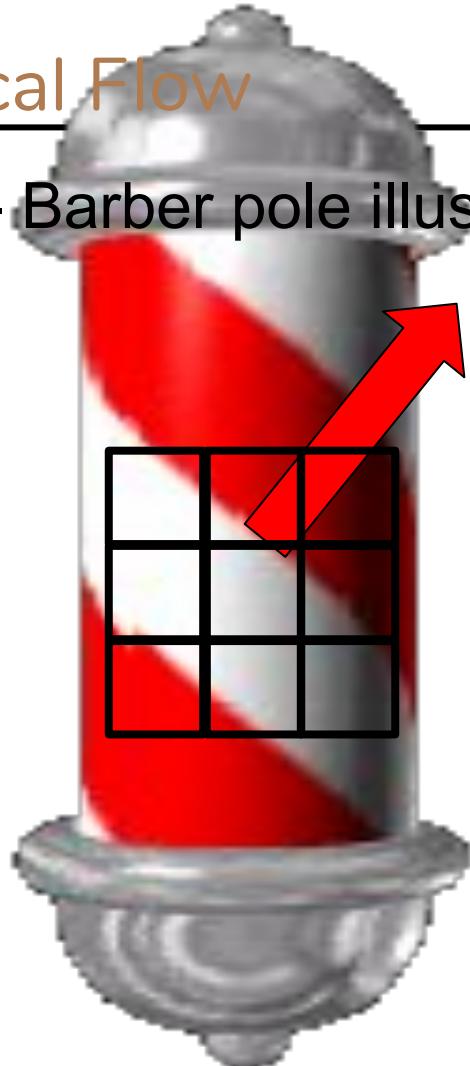
Проблема апертуры - Barber pole illusion



# Lucas-Kanade Optical Flow

---

Проблема апертуры - Barber pole illusion



Смотрим через  
окошко - движется  
вверх и направо

## Улучшения: Iterative LK

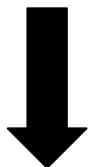
---

$$dx_1 * u + dy_1 * v = I_t[x,y] - I_{t+\Delta t}[x,y]$$

## Улучшения: Iterative LK

---

$$dx_1 * u + dy_1 * v = I_t[x, y] - I_{t+\Delta t}[x, y]$$



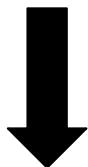
Оцениваем один раз  $dx_1$ ,  $dy_1$   
подставляем и снова переоцениваем

$$dx_2 * u + dy_2 * v = I_t[x, y] - I_{t+\Delta t}[x + dx_1, y + dy_1]$$

## Улучшения: Iterative LK

---

$$dx_1 * u + dy_1 * v = I_t[x, y] - I_{t+\Delta t}[x, y]$$



Оцениваем один раз  $dx_1$ ,  $dy_1$   
подставляем и снова переоцениваем

$$dx_2 * u + dy_2 * v = I_t[x, y] - I_{t+\Delta t}[x + dx_1, y + dy_1]$$



Оцениваем один раз  $dx_2$ ,  $dy_n$   
подставляем и снова переоцениваем

...

$$dx_n * u + dy_n * v = I_t[x, y] - I_{t+\Delta t}[x + dx_{n-1}, y + dy_{n-1}]$$

## Улучшения: Image Pyramids

---

# Улучшения: Image Pyramids

---

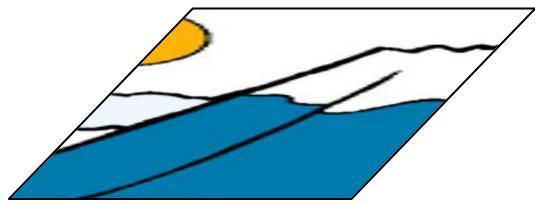


Image 1

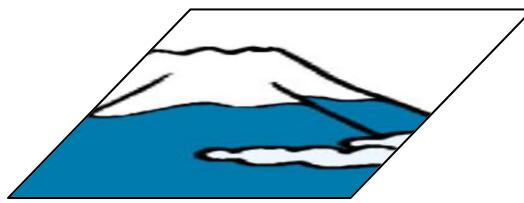


Image 2

# Улучшения: Image Pyramids

---

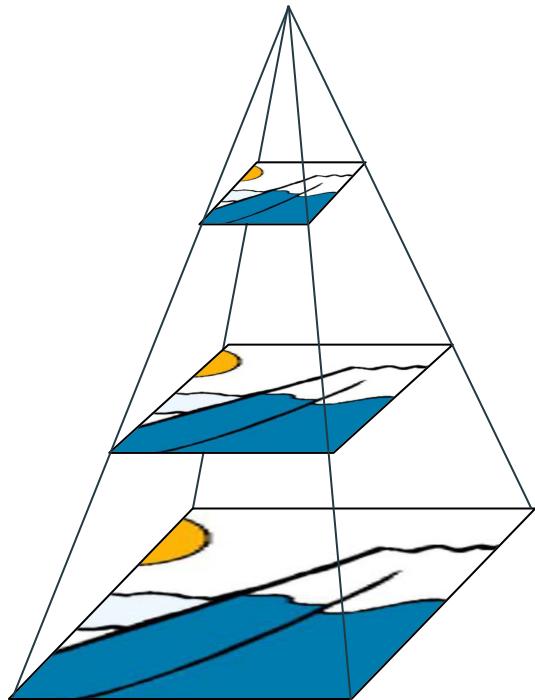


Image 1

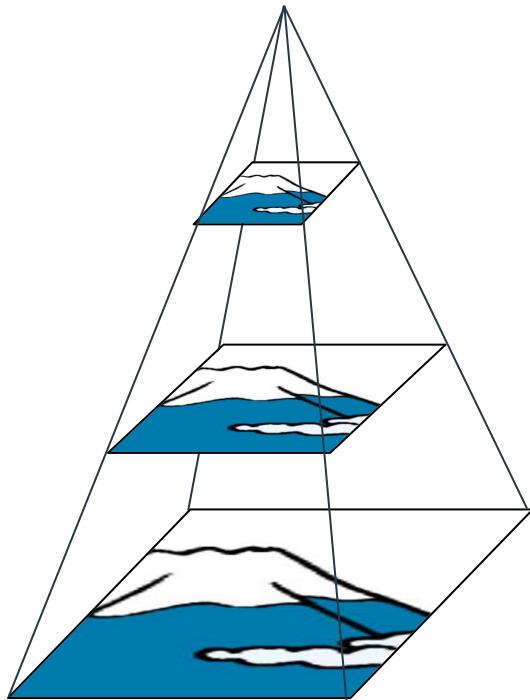
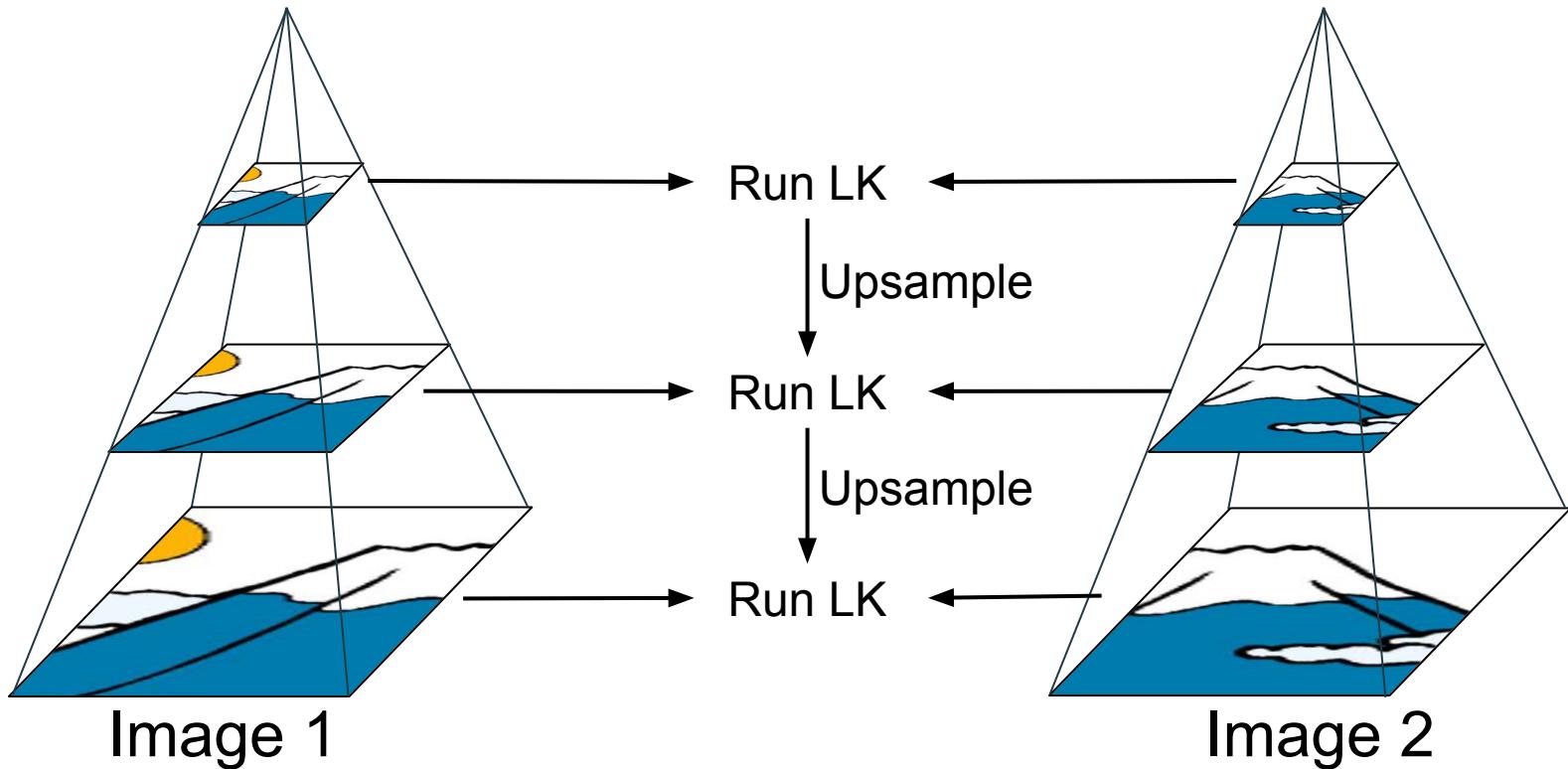


Image 2

# Улучшения: Image Pyramids

---



# Optical Flow

---

# Optical Flow

## Sparse vs Dense

## Sparse vs Dense

Считает OF для  
конкретных точек-  
фичей

Считает OF для  
всех пикселей

## Sparse vs Dense

Считает OF для  
конкретных точек-  
фичей

Считает OF для  
всех пикселей

Lucas-Kanade - теоретически Dense, но  
на практике работает только на хороших  
точках-фичах, т.е., он sparse.

# Как вычислить Dense Optical flow

# Farnebäck Optical Flow

---

Farnebäck, Gunnar. "Two-frame motion estimation based on polynomial expansion." *Scandinavian conference on Image analysis.* Springer, Berlin, Heidelberg, 2003.

# Farnebäck Optical Flow

---

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

# Farnebäck Optical Flow

---

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$



в LK-методе мы  
аппроксимировали это рядом  
Тейлора первого порядка

# Farnebäck Optical Flow

---

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Вместо этого, давайте аппроксимировать рядом Тейлора второго порядка

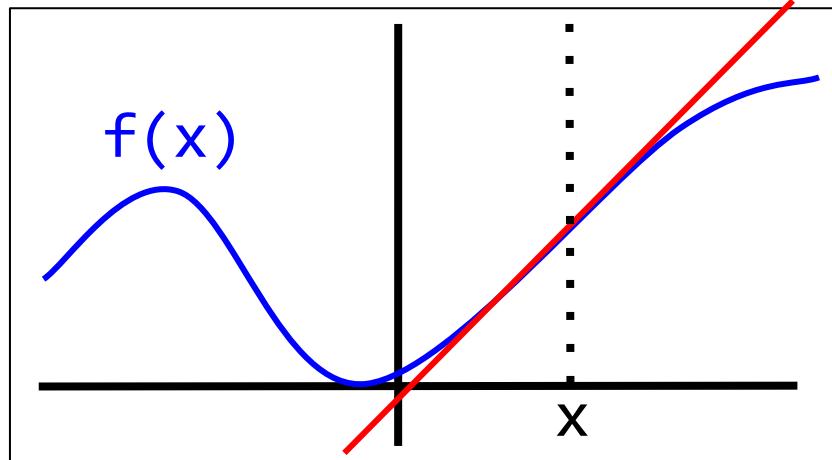
# Ряды Тейлора

---

Сложная функция  $\approx$  полином n-ого порядка

$$f(x) \approx \sum_i a_i x^i$$

Первый порядок:  $i=1$   $mx + b$



$$\approx a_1 x^1 + a_0 x^0$$

$$\approx mx + b$$

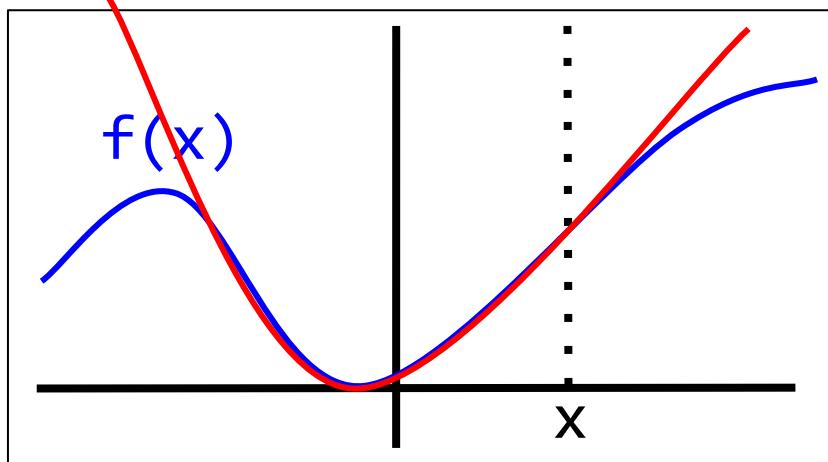
# Ряды Тейлора

---

Сложная функция  $\approx$  полином n-ого порядка

$$f(x) \approx \sum_i a_i x^i$$

Второй порядок  $i=2$   $ax^2 + bx + c$



$$\approx a_1 x^2 + a_1 x^1 + a_0 x^0$$

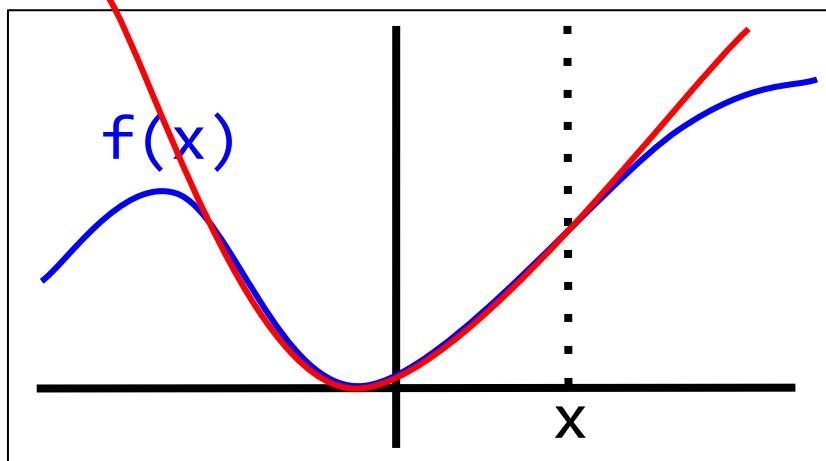
$$\approx ax^2 + bx + c$$

# Ряды Тейлора

Сложная функция  $\approx$  полином n-ого порядка

$$f(x) \approx \sum_i a_i x^i$$

Второй порядок  $i=2$   $ax^2 + bx + c$



$$\approx a_1 x^2 + a_1 x^1 + a_0 x^0$$

$$\approx ax^2 + bx + c$$

Vector notation

$$\approx \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b} \mathbf{x} + \mathbf{c}$$

# Farnebäck Optical Flow

---

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Вместо этого, давайте аппроксимировать рядом Тейлора 2 порядка

# Farnebäck Optical Flow

---

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Вместо этого, давайте аппроксимировать рядом Тейлора 2 порядка

$$(p - \Delta p)^T A_t (p - \Delta p) + b_t (p - \Delta p) + c_t = p^T A_{t+\Delta t} p + b_{t+\Delta t} p + c_{t+\Delta t}$$

# Farnebäck Optical Flow

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Вместо этого, давайте аппроксимировать рядом Тейлора 2 порядка

$$(p - \Delta p)^T A_t (p - \Delta p) + b_t (p - \Delta p) + c_t = p^T A_{t+\Delta t} p + b_{t+\Delta t} p + c_{t+\Delta t}$$

Параметры полинома из  
первого изображения

Параметры полинома из  
второго изображения

# Farnebäck Optical Flow

---

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Вместо этого, давайте аппроксимировать рядом Тейлора 2 порядка

$$(p - \Delta p)^T A_t (p - \Delta p) + b_t (p - \Delta p) + c_t = p^T A_{t+\Delta t} p + b_{t+\Delta t} p + c_{t+\Delta t}$$

$$p^T A_t p + (b_t - 2A_t \Delta p)^T p + \Delta p^T A_t \Delta p - b_t \Delta p + c_t = p^T A_{t+\Delta t} p + p b_{t+\Delta t} + c_{t+\Delta t}$$

# Farnebäck Optical Flow

Мы предположили, объект смещается на  $\Delta p$  за время  $\Delta t$

$$f(p - \Delta p, t) = f(p, t + \Delta t)$$

Вместо этого, давайте аппроксимировать рядом Тейлора 2 порядка

$$(p - \Delta p)^T A_t (p - \Delta p) + b_t (p - \Delta p) + c_t = p^T A_{t+\Delta t} p + b_{t+\Delta t} p + c_{t+\Delta t}$$

$$p^T A_t p + (b_t - 2A_t \Delta p)^T p + \Delta p^T A_t \Delta p - b_t \Delta p + c_t = p^T A_{t+\Delta t} p + p b_{t+\Delta t} + c_{t+\Delta t}$$

$$A_t = A_{t+\Delta t}$$

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

$$c_{t+\Delta t} = \Delta p^T A_t \Delta p - b_t \Delta p + c_t$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Решение для  $\Delta p$

$$\Delta p = -\frac{1}{2}(A_t)^{-1}(b_{t+\Delta t} - b_t)$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Решение для  $\Delta p$

$$\Delta p = -\frac{1}{2}(A_t)^{-1}(b_{t+\Delta t} - b_t)$$

$A$ ,  $b$ , и  $c$  могут быть найдены поточечно используя соседние пиксели как и в предыдущем алгоритма

# Farnebäck Optical Flow

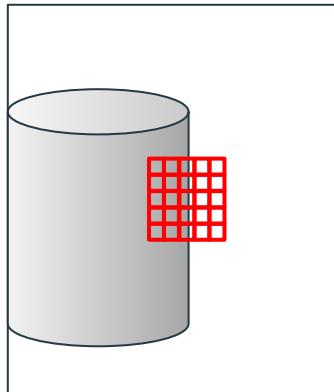
---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Решение для  $\Delta p$

$$\Delta p = -\frac{1}{2}(A_t)^{-1}(b_{t+\Delta t} - b_t)$$

$A$ ,  $b$ , и  $c$  могут быть найдены поточечно используя соседние пиксели как и в предыдущем алгоритма



# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Решение для  $\Delta p$

$$\Delta p = -\frac{1}{2}(A_t)^{-1}(b_{t+\Delta t} - b_t)$$

$A$ ,  $b$ , и  $c$  могут быть найдены поточечно используя соседние пиксели как и в предыдущем алгоритма

$$S = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \dots \\ x_k & y_k \end{bmatrix} \quad T = \begin{bmatrix} I[x_1, y_1] \\ I[x_2, y_2] \\ \dots \\ I[x_k, y_k] \end{bmatrix}$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Решение для  $\Delta p$

$$\Delta p = -\frac{1}{2}(A_t)^{-1}(b_{t+\Delta t} - b_t)$$

$A$ ,  $b$ , и  $c$  могут быть найдены поточечно используя соседние пиксели как и в предыдущем алгоритма

$$S = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \dots \\ x_k & y_k \end{bmatrix} \quad T = \begin{bmatrix} I[x_1, y_1] \\ I[x_2, y_2] \\ \dots \\ I[x_k, y_k] \end{bmatrix}$$

$$SAS^T + bS^T + c = T$$
$$\|SAS^T + bS^T + c - T\|^2 = 0$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Решение для  $\Delta p$

$$\Delta p = -\frac{1}{2}(A_t)^{-1}(b_{t+\Delta t} - b_t)$$

$A$ ,  $b$ , и  $c$  могут быть найдены поточечно используя соседние пиксели как и в предыдущем алгоритма

$$S = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \cdots \\ x_k & y_k \end{bmatrix} \quad T = \begin{bmatrix} I[x_1, y_1] \\ I[x_2, y_2] \\ \cdots \\ I[x_k, y_k] \end{bmatrix}$$

$$\begin{aligned} SAS^T + bS^T + c &= T \\ ||SAS^T + bS^T + c - T||^2 &= 0 \end{aligned}$$

Quadratic Regression

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Решение для  $\Delta p$

$$\Delta p = -\frac{1}{2}(A_t)^{-1}(b_{t+\Delta t} - b_t)$$

Такая же проблема как  
раньше: что если матрица  
необратима?

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A_t \Delta p = -\frac{1}{2}(b_{t+\Delta t} - b_t)$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A_t = A_{t+\Delta t}$$

$$A_t \Delta p = -\frac{1}{2}(b_{t+\Delta t} - b_t)$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A_t = A_{t+\Delta t}$$

$$A_t \Delta p = -\frac{1}{2}(b_{t+\Delta t} - b_t)$$

В реальности не  
равны

$$A_t \neq A_{t+\Delta t}$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A_t = A_{t+\Delta t}$$

$$A_t \Delta p = -\frac{1}{2}(b_{t+\Delta t} - b_t)$$

В реальности не  
равны

$$A_t \neq A_{t+\Delta t}$$

Усредняем

$$A = \frac{1}{2}(A_t + A_{t+\Delta t})$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A_t \Delta p = -\frac{1}{2}(b_{t+\Delta t} - b_t)$$

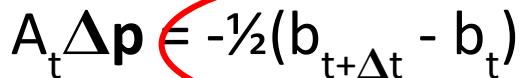
# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A_t \Delta p = -\frac{1}{2}(b_{t+\Delta t} - b_t)$$



$\Delta b = -\frac{1}{2}(b_{t+\Delta t} - b_t)$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A \Delta p = \Delta b$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A \Delta p = \Delta b$$

Предполагаем, что соседи двигаются вместе

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A \Delta p = \Delta b$$

Предполагаем, что соседи двигаются вместе

$$A_i \Delta p = \Delta b_i$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A \Delta p = \Delta b$$

Предполагаем, что соседи двигаются вместе

$$A_i \Delta p = \Delta b_i$$

МНК

$$\sum_i w_i ||A_i \Delta p - \Delta b_i||^2$$

# Farnebäck Optical Flow

---

$$b_{t+\Delta t} = b_t - 2A_t \Delta p$$

Попробуем другой подход:

$$A \Delta p = \Delta b$$

Предполагаем, что соседи двигаются вместе

$$A_i \Delta p = \Delta b_i$$

МНК

$$\sum_i w_i ||A_i \Delta p - \Delta b_i||^2$$



Гауссовые веса (мы больше учитываем центральные пиксели чем краевые)

# Farnebäck Optical Flow

---

Все еще предполагаем, что  $\Delta p$  одинаковая для всех соседних пикселей, хотя на самом деле это не так...

# Farnebäck Optical Flow

---

Все еще предполагаем, что  $\Delta p$  одинаковая для всех соседних пикселей, хотя на самом деле это не так...

ПРЕДПОЛОЖИМ: что  $\Delta p$  - аффинное преобразование всех соседних пикселей

# Farnebäck Optical Flow

---

Все еще предполагаем, что  $\Delta p$  одинаковая для всех соседних пикселей, хотя на самом деле это не так...

ПРЕДПОЛОЖИМ: что  $\Delta p$  - аффинное преобразование всех соседних пикселей

$$\Delta p = \begin{bmatrix} a_1 + a_2x + a_3y + a_7x^2 + a_8xy \\ a_4 + a_5x + a_6y + a_7xy + a_8y^2 \end{bmatrix}$$

# Farnebäck Optical Flow

---

Все еще предполагаем, что  $\Delta p$  одинаковая для всех соседних пикселей, хотя на самом деле это не так...

ПРЕДПОЛОЖИМ: что  $\Delta p$  - преобразование которое выглядит как:

$$\Delta p = \begin{bmatrix} a_1 + a_2x + a_3y + a_7x^2 + a_8xy \\ a_4 + a_5x + a_6y + a_7xy + a_8y^2 \end{bmatrix}$$

$$S = \begin{bmatrix} 1 & x & y & 0 & 0 & 0 & x^2 & xy \\ 0 & 0 & 0 & 1 & x & y & xy & y^2 \end{bmatrix}$$

$\Delta p = Sd$

$$d = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix}$$

# Farnebäck Optical Flow

---

$$\sum_i w_i ||A_i \Delta p - \Delta b_i||^2$$

# Farnebäck Optical Flow

---

$$\sum_i w_i \| A_i \Delta p - \Delta b_i \|^2$$

↓  
Подставляем  $Sd$   
вместо  $\Delta p$ .

$$\sum_i w_i \| A_i Sd - \Delta b_i \|^2$$

# Farnebäck Optical Flow

---

$$\sum_i w_i \| A_i \Delta p - \Delta b_i \| ^2$$

Подставляем  $Sd$   
вместо  $\Delta p$ .

$$\sum_i w_i \| A_i Sd - \Delta b_i \| ^2$$

Решаем относительно  $d$

$$d = \left( \sum_i w_i (S_i)^T (A_i)^T S_i \right)^{-1} \sum_i w_i (S_i)^T (A_i)^T \Delta b_i$$

# Farnebäck Optical Flow

---

$$\sum_i w_i \| A_i \Delta p - \Delta b_i \| ^2$$

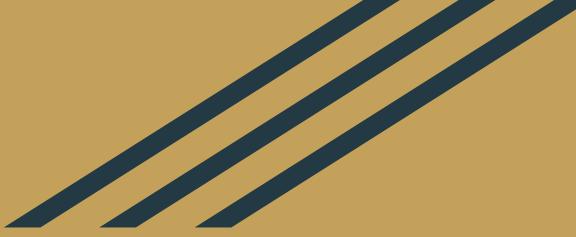
Подставляем  $Sd$   
вместо  $\Delta p$ .

$$\sum_i w_i \| A_i Sd - \Delta b_i \| ^2$$

Решаем относительно  $d$

$$d = \left( \sum_i w_i (S_i)^T (A_i)^T S_i \right)^{-1} \sum_i w_i (S_i)^T (A_i)^T \Delta b_i$$

Такая же проблема  
обратимости, но на практике  
проблем меньше



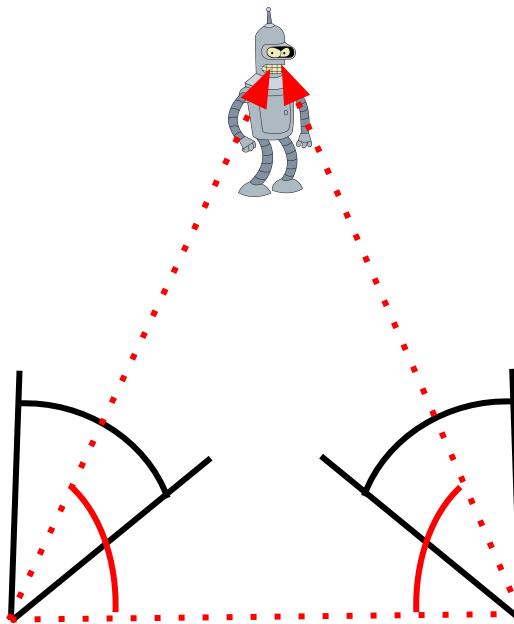
# 3D зрение

# Как мы ощущаем глубину?

# Как мы ощущаем глубину?

---

-Бинокулярное схождение

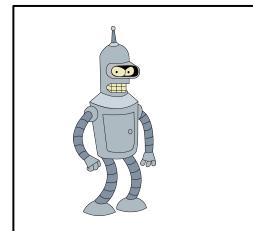
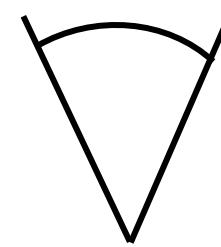
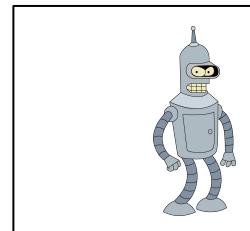
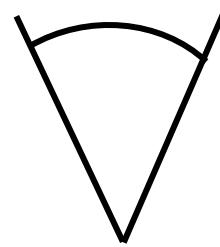


# Как мы ощущаем глубину?

---

-Бинокулярное схождение

-Бинокулярный параллакс



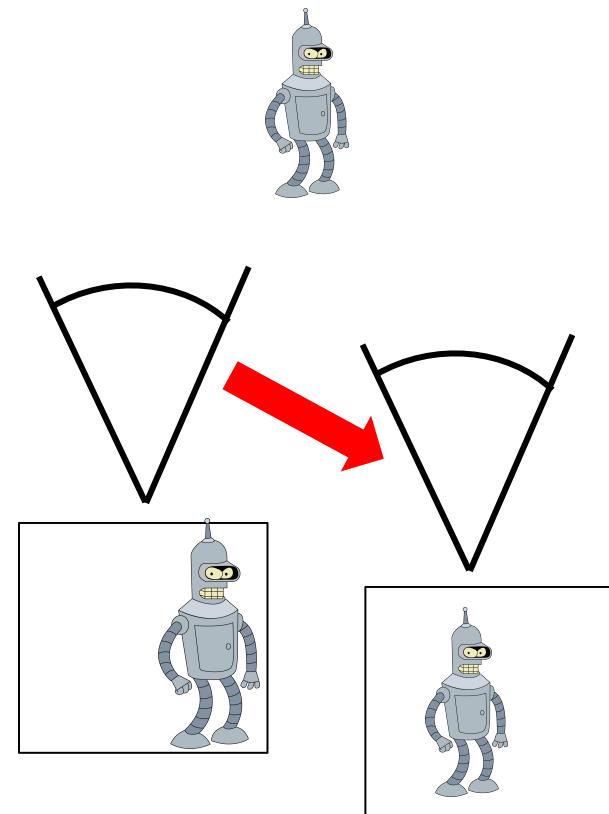
# Как мы ощущаем глубину?

---

-Бинокулярное схождение

-Бинокулярный параллакс

-Монокулярный параллакс



# Как мы ощущаем глубину?

-Бинокулярное схождение

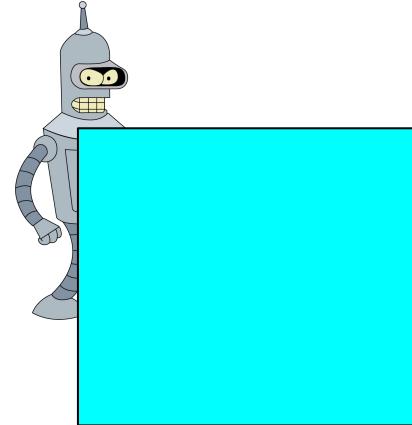
-Бинокулярный параллакс

-Монокулярный параллакс

-”Подсказки”

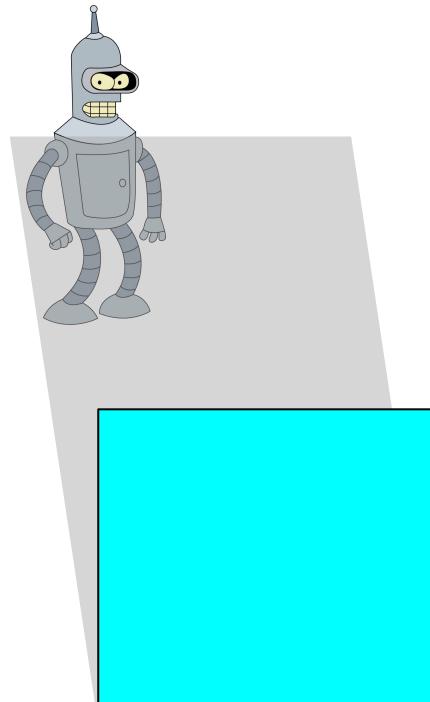
# Как мы ощущаем глубину?

- Бинокулярное схождение
- Бинокулярный параллакс
- Монокулярный параллакс
- ”Подсказки”
  - Перекрытие

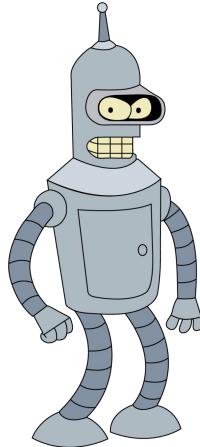


# Как мы ощущаем глубину?

- Бинокулярное схождение
- Бинокулярный параллакс
- Монокулярный параллакс
- ”Подсказки”
  - Перекрытие
  - Тени

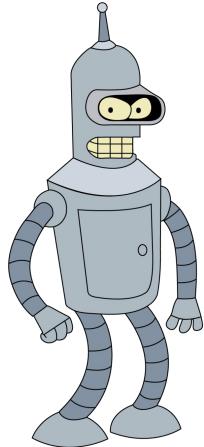


# Интерактивная демонстрация



# Интерактивная демонстрация

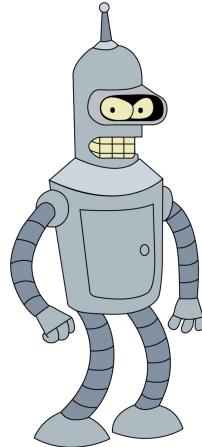
Закройте один глаз



# Интерактивная демонстрация

Закройте один глаз

Закройте Бендера большим пальцем

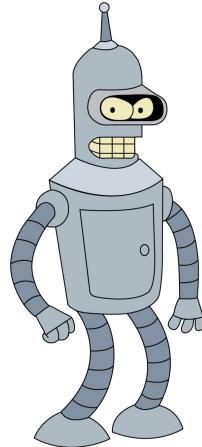


# Интерактивная демонстрация

Закройте один глаз

Закройте Бендера большим пальцем

Закройте один глаз, откройте другой



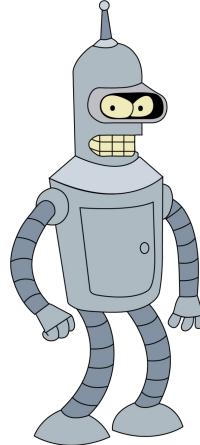
# Интерактивная демонстрация

Закройте один глаз

Закройте Бендера большим пальцем

Закройте один глаз, откройте другой

Бендер двигается!



# Как компьютеры понимают глубину?

---

# Как компьютеры понимают глубину?

---

-Стерео пара

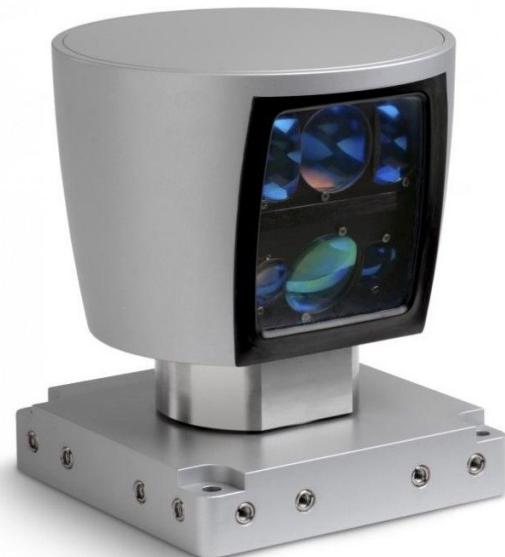


# Как компьютеры понимают глубину?

---

-Стерео пара

-Lidar



# Как компьютеры понимают глубину?

---

-Стерео пара

-Lidar

-Structured light



# Как компьютеры понимают глубину?

---

-Стерео пара

-Lidar

-Structured light

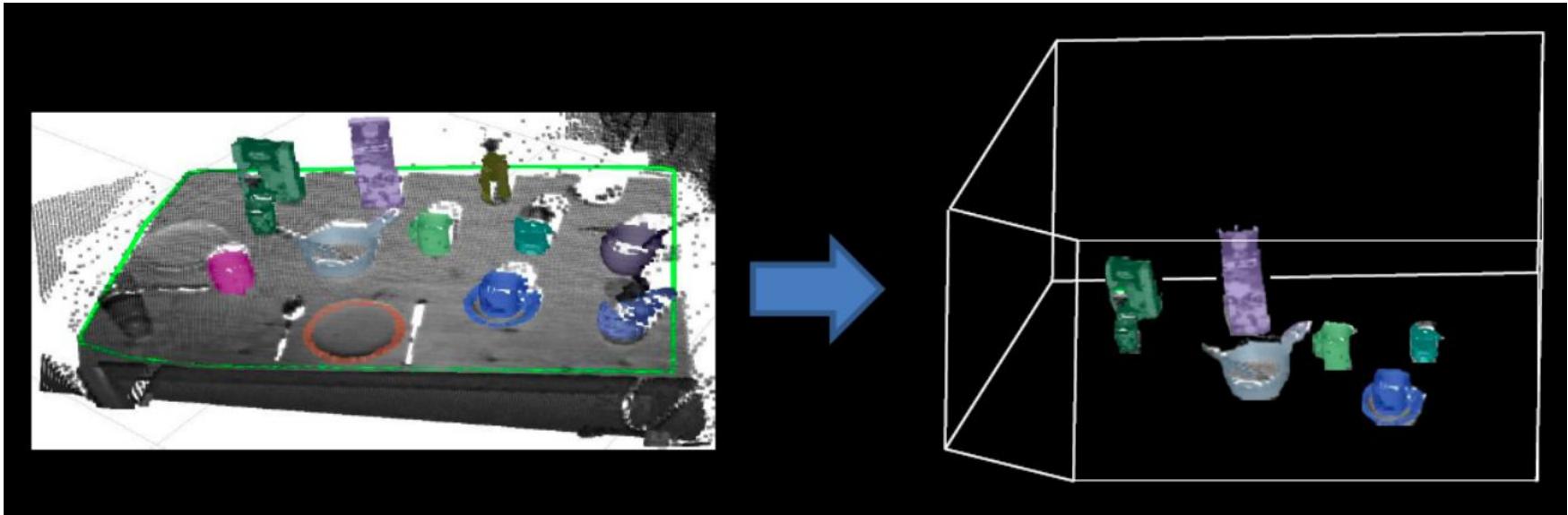
-Время возвращения



# Зачем нужно знать глубину?

# Зачем нужно знать глубину?

-Сегментация объектов



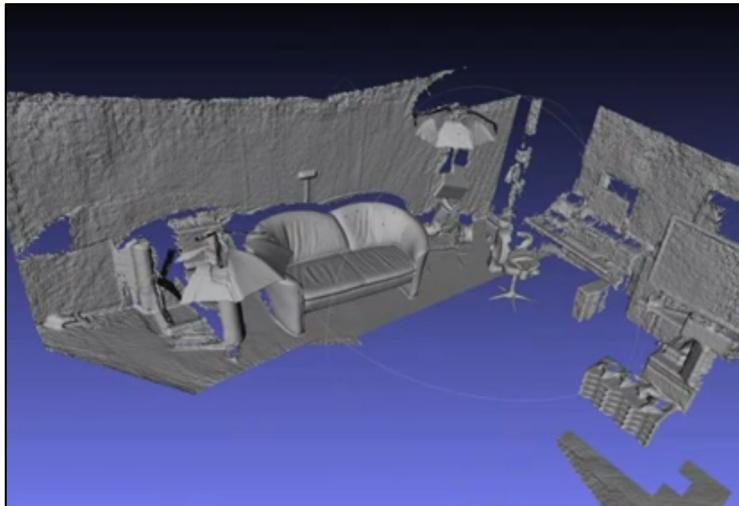
# Зачем нужно знать глубину?

---

-Сегментация объектов

-3D reconstruction

KinectFusion



# Зачем

-Сегментация

-3D rec



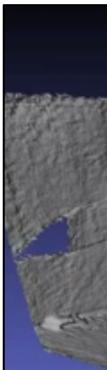
Live Input Depth Map



Live Model Output



Live RGB Image (unused)



Canonical Model Reconstruction



Warped Model

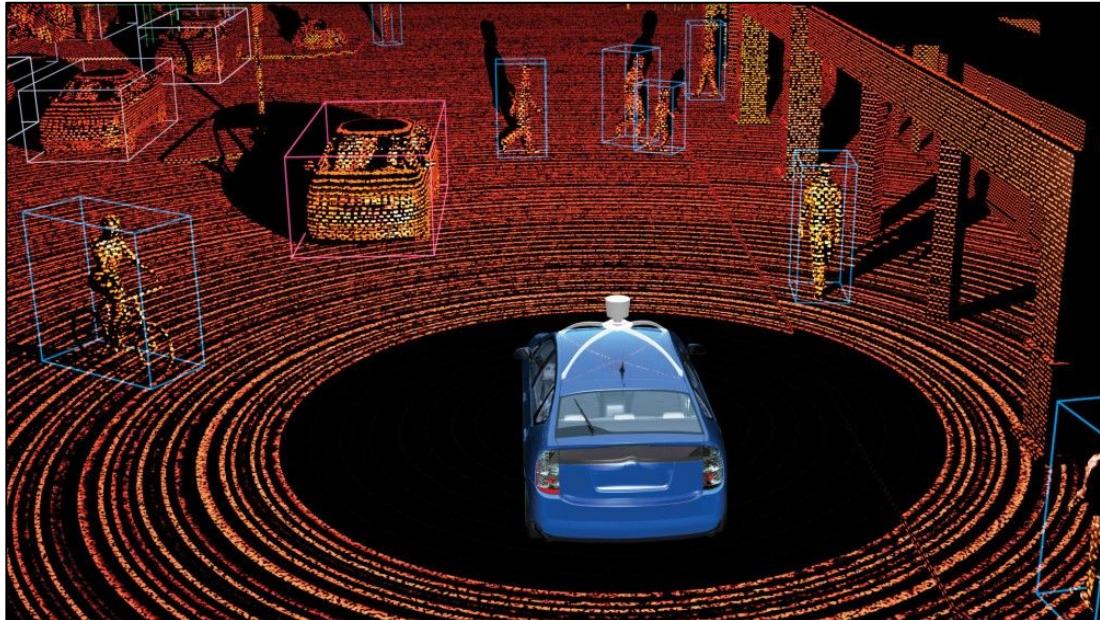
# Зачем нужно знать глубину?

---

-Сегментация объектов

-3D reconstruction

-Навигация



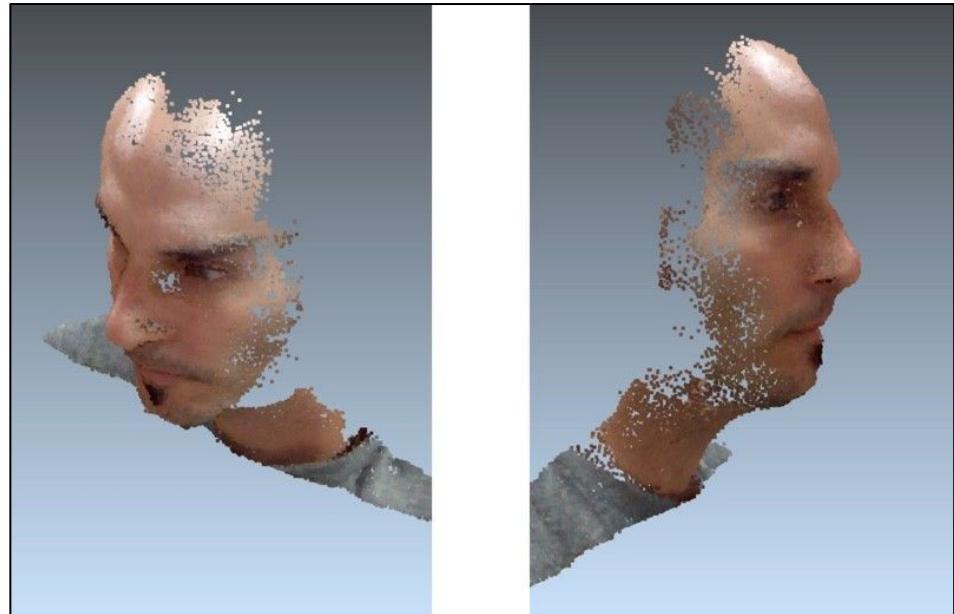
# Зачем нужно знать глубину?

-Сегментация объектов

-3D reconstruction

-Навигация

-Распознавание лиц



# Зачем нужно знать глубину?

- Сегментация объектов
- 3D reconstruction
- Навигация
- Распознавание лиц
- Распознавание позиций

# Зачем

-Сегмент



-3D reco

-Навига

-Распоз

-Распоз



- The full objective function is given by:

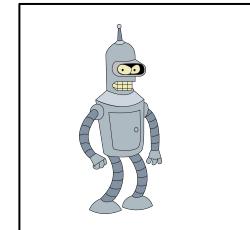
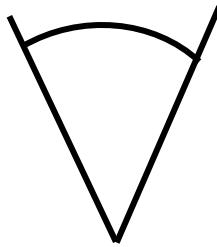
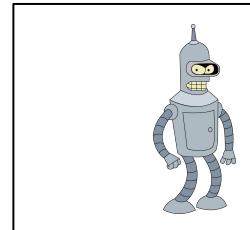
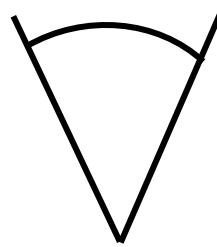
$$\hat{\theta} = \arg \min_{\theta} \sum_{u \in U} SDF_{md}(\mathbf{x}_u; \theta)^2 + \lambda \sum_{u \in D} SDF_{ms}(\hat{\mathbf{x}}_u(\theta); D)^2$$

Стерео

# Стерео

---

Бинокулярный  
параллакс



# Стерео

---



# Стерео



Смещение обратно пропорционально  
расстоянию от камеры

# Стерео



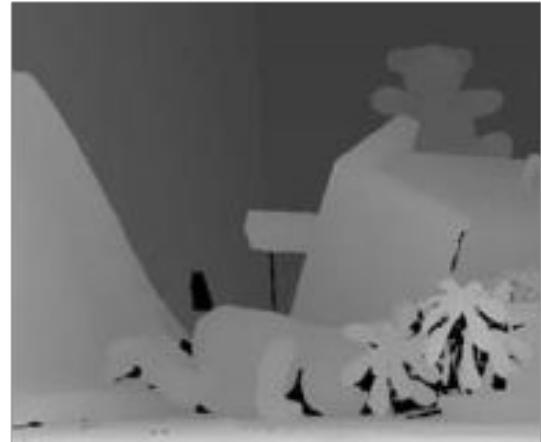
Дальше

Ближе

# Стерео

---

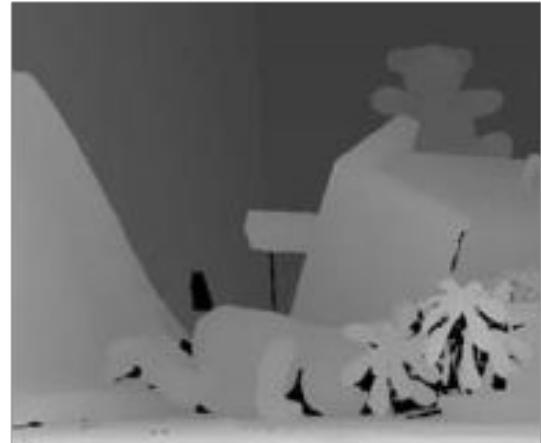
Как сопоставить различные части изображения?



# Стерео

---

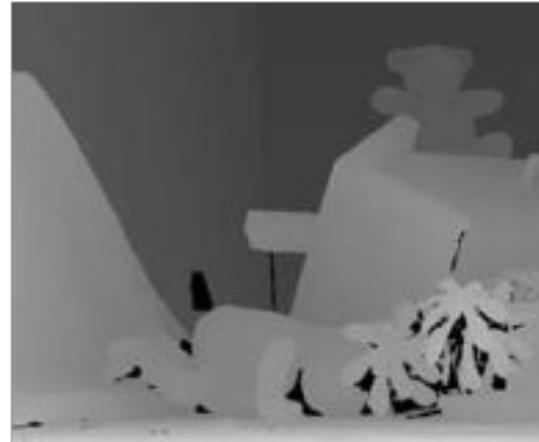
Матчинг признаков!



# Стерео

---

Как матчить? RANSAC? ...



# Стерео

---

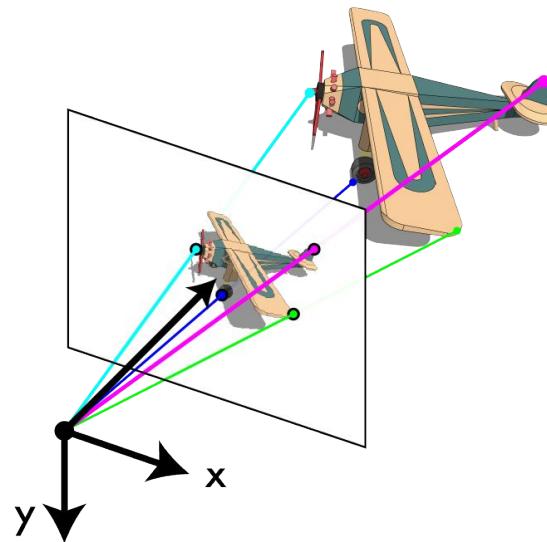
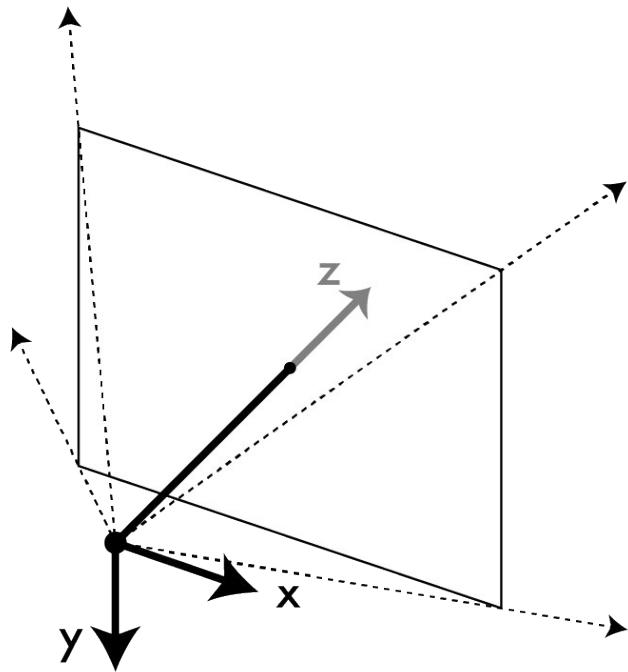
Как матчить? RANSAC? ...



Вычислительно долго, матчить можно только ключевые точки  
Возможно ли воспользоваться геометрией снимков?

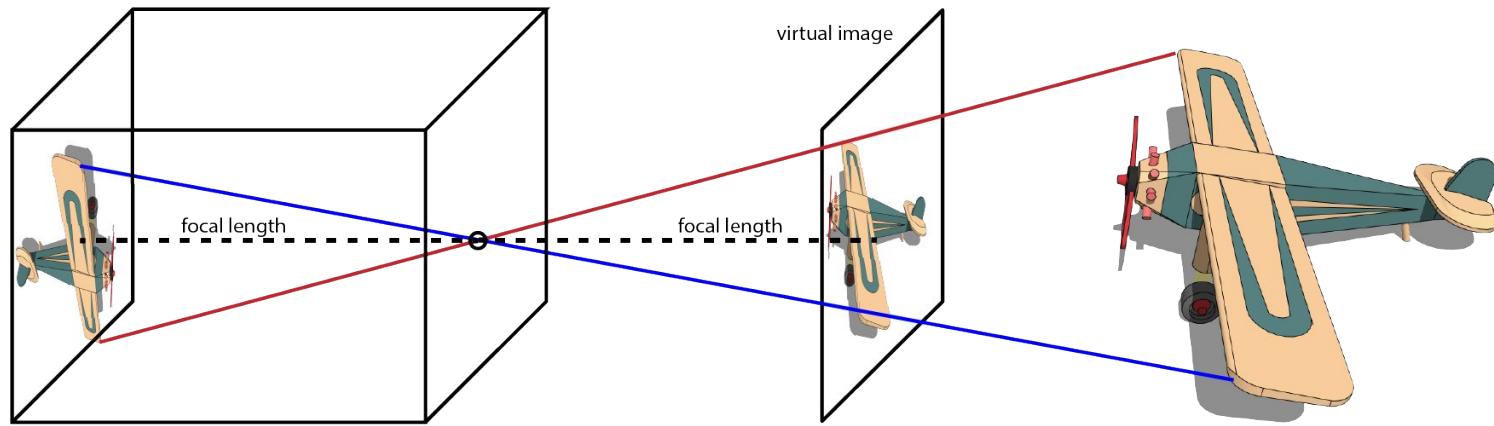
# Проекция изображения

---



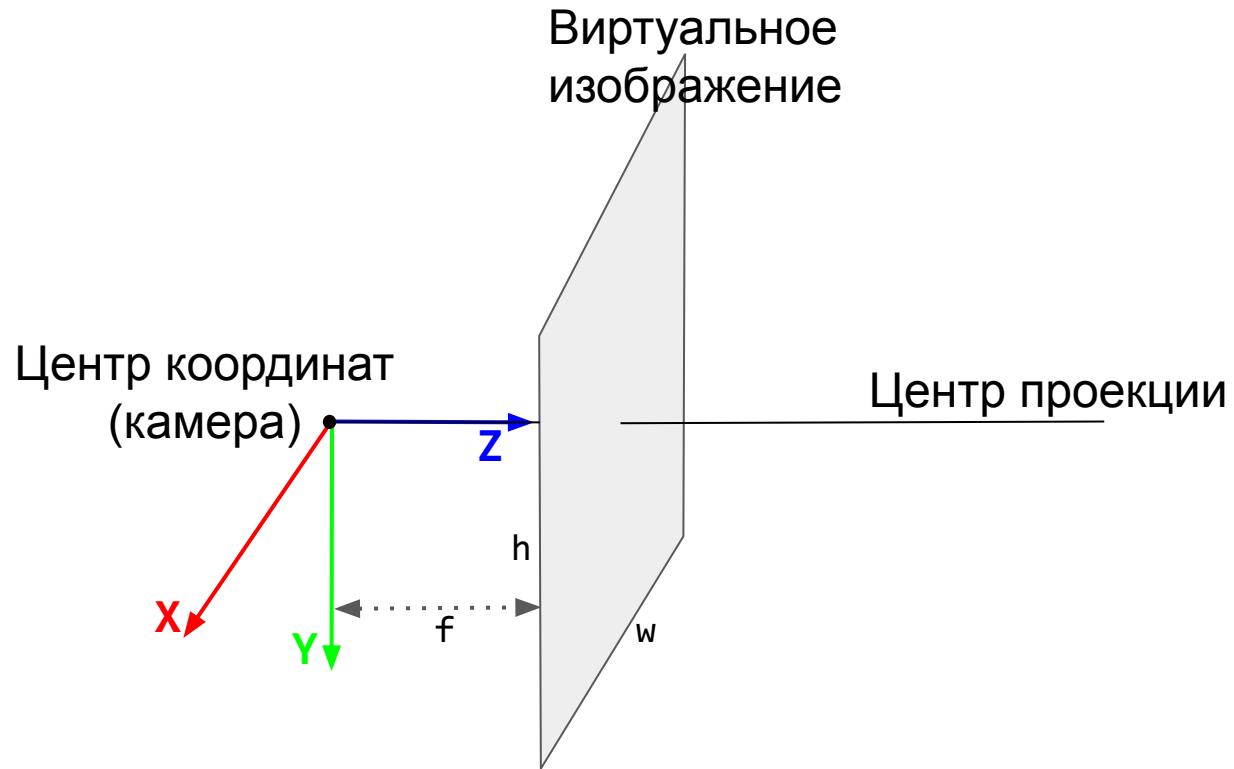
# Проекция изображения

---



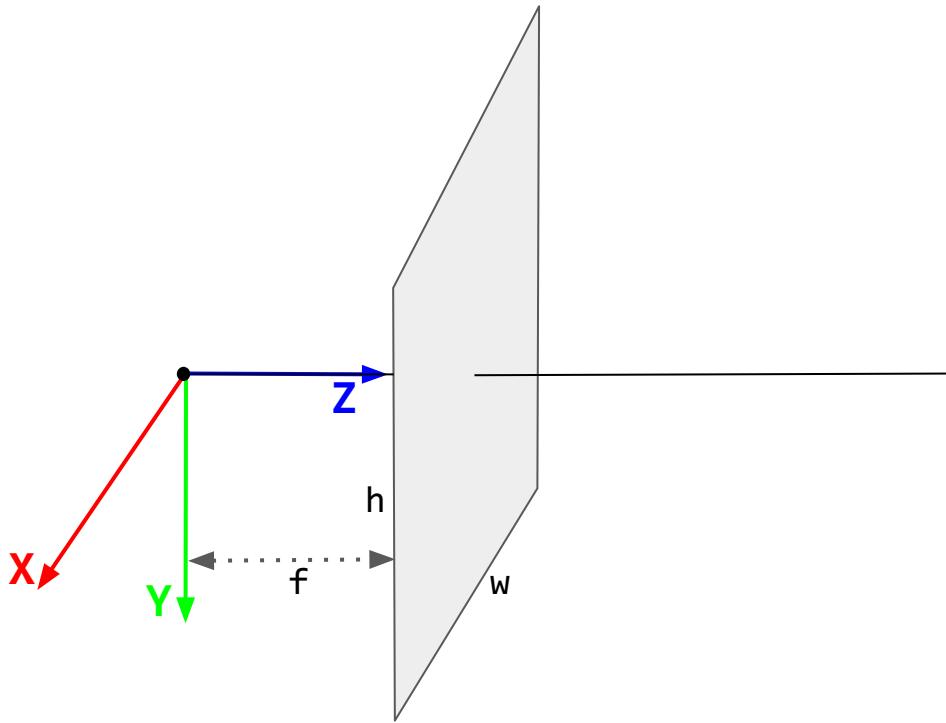
# Проекция изображения

---



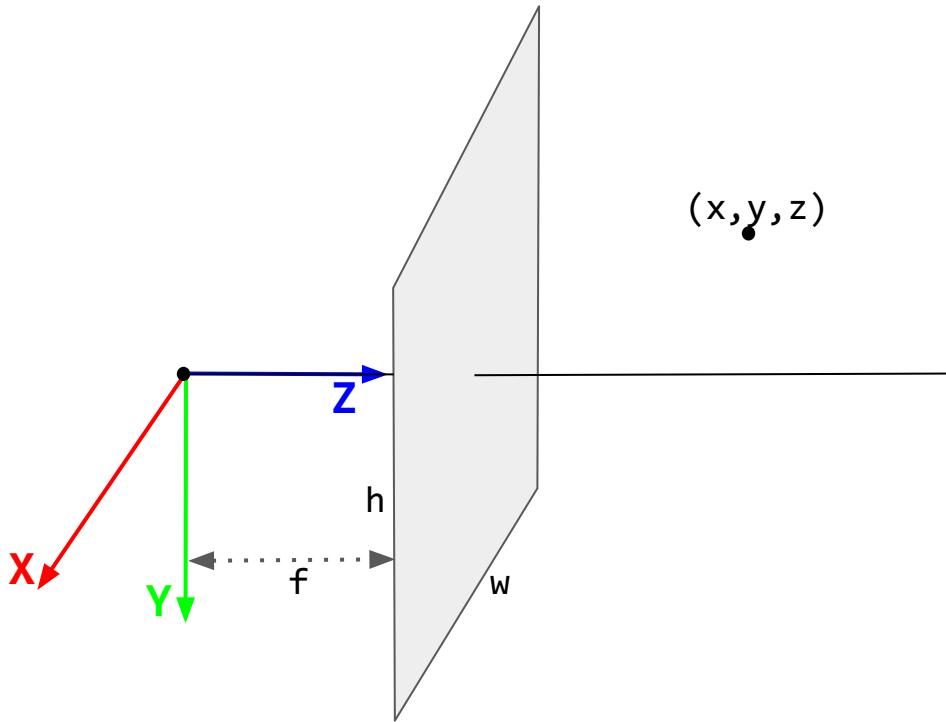
# Проекция изображения

---



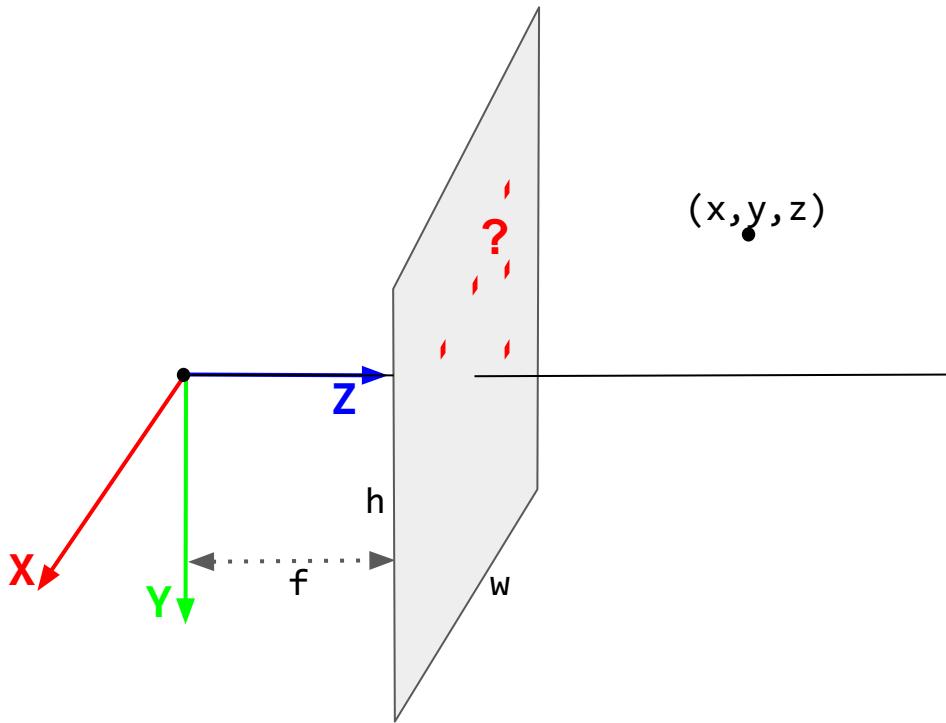
# Проекция изображения

---



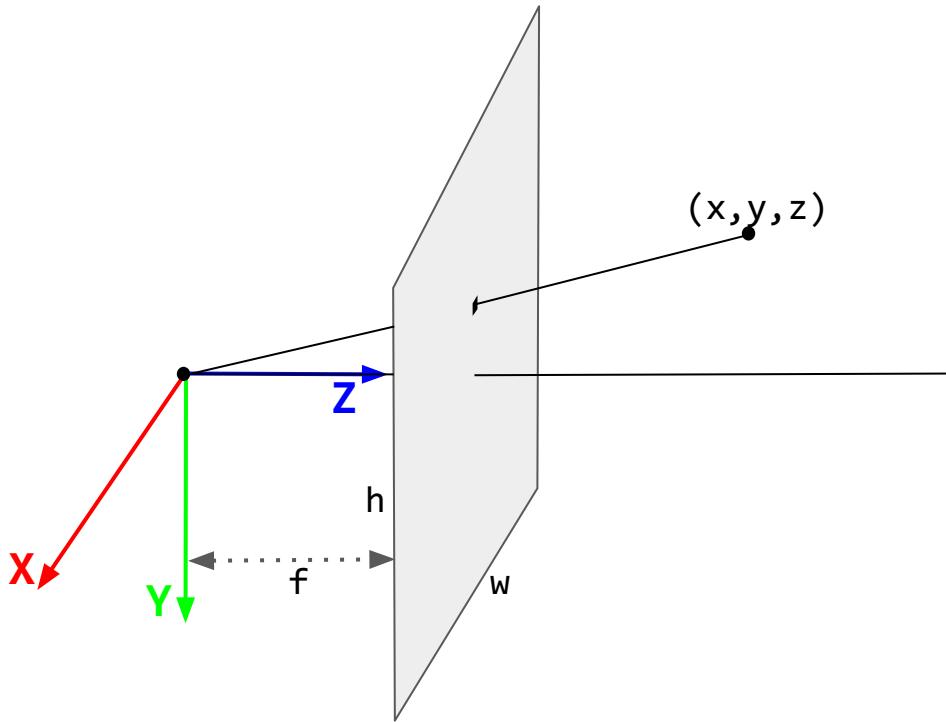
# Проекция изображения

---



# Проекция изображения

---



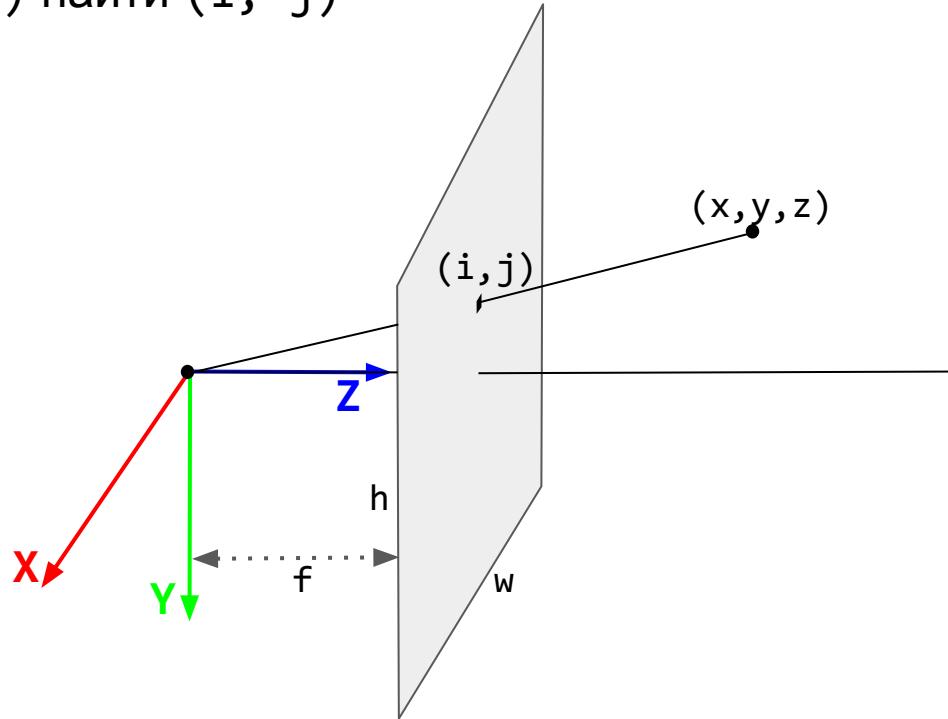
# Проекция изображения

---

Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$$i = ?$$

$$j = ?$$

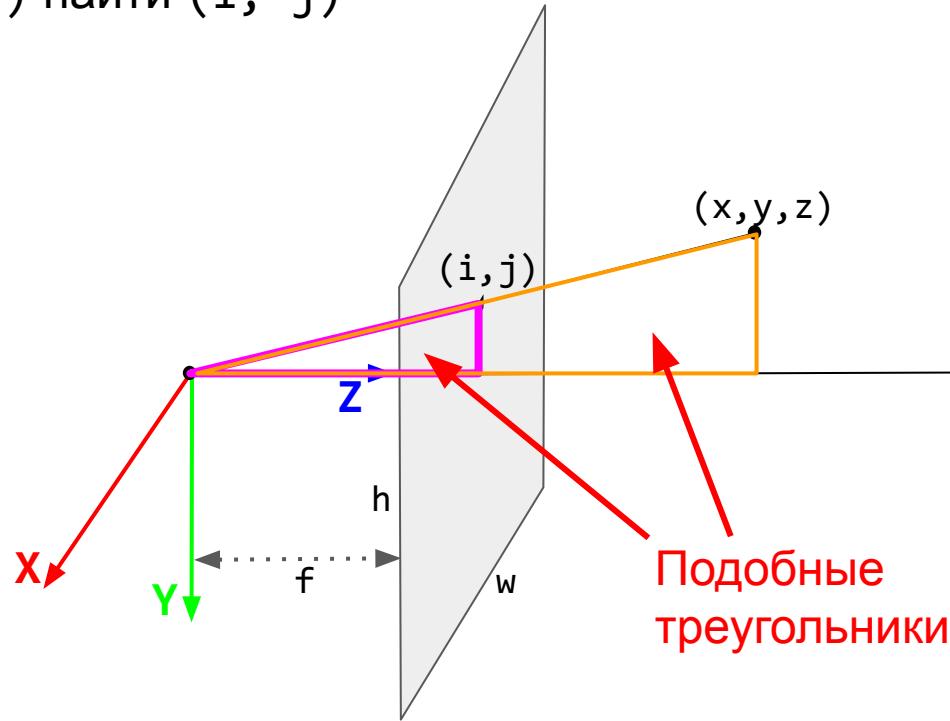


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$i = ?$

$j = ?$

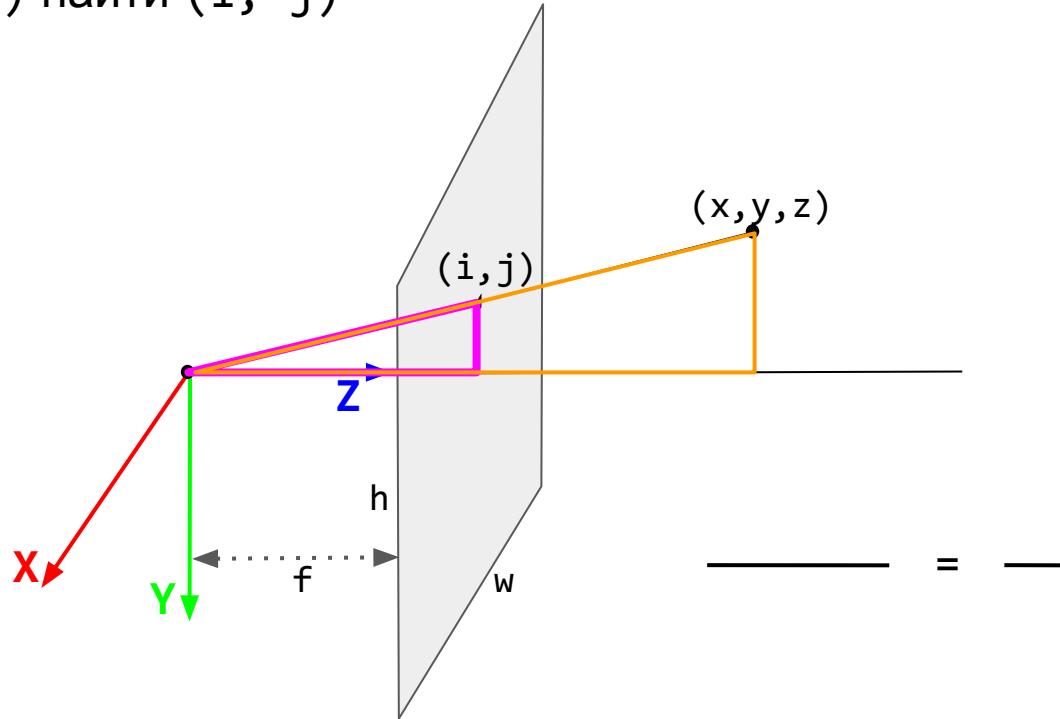


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$$i = ?$$

$$j = ?$$

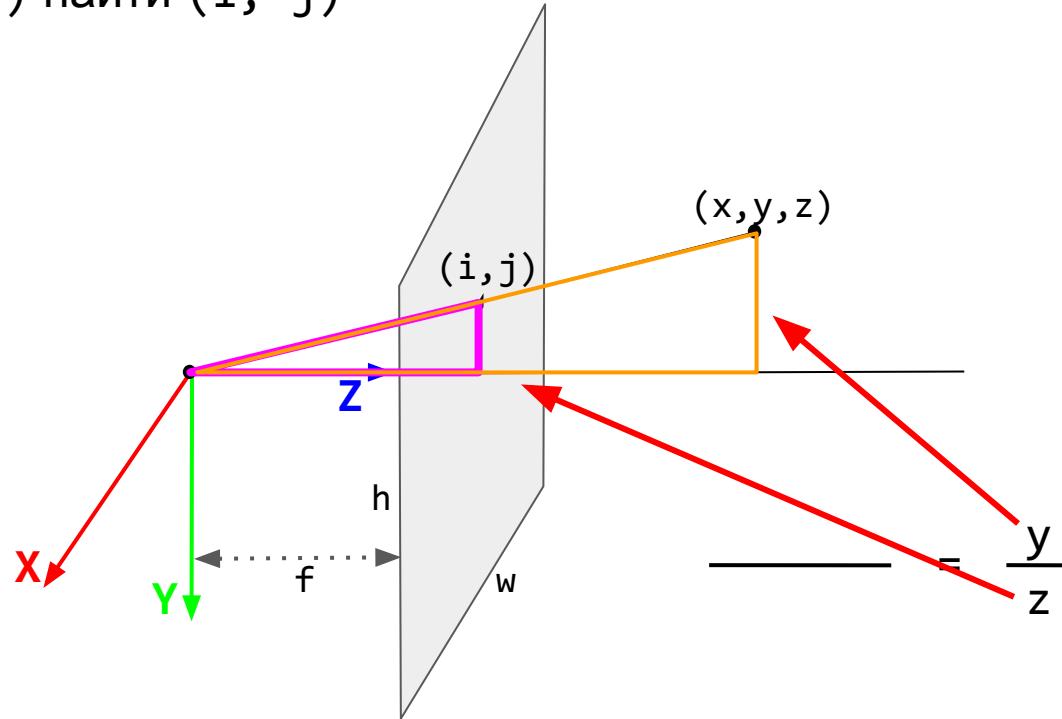


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$$i = ?$$

$$j = ?$$

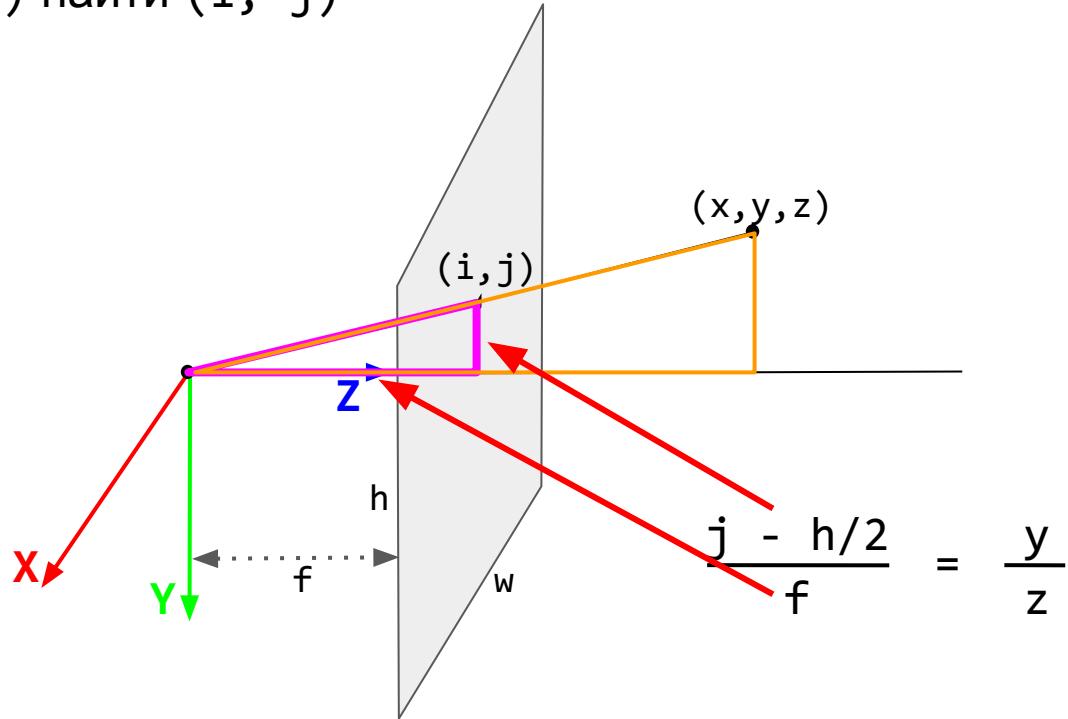


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$$i = ?$$

$$j = ?$$

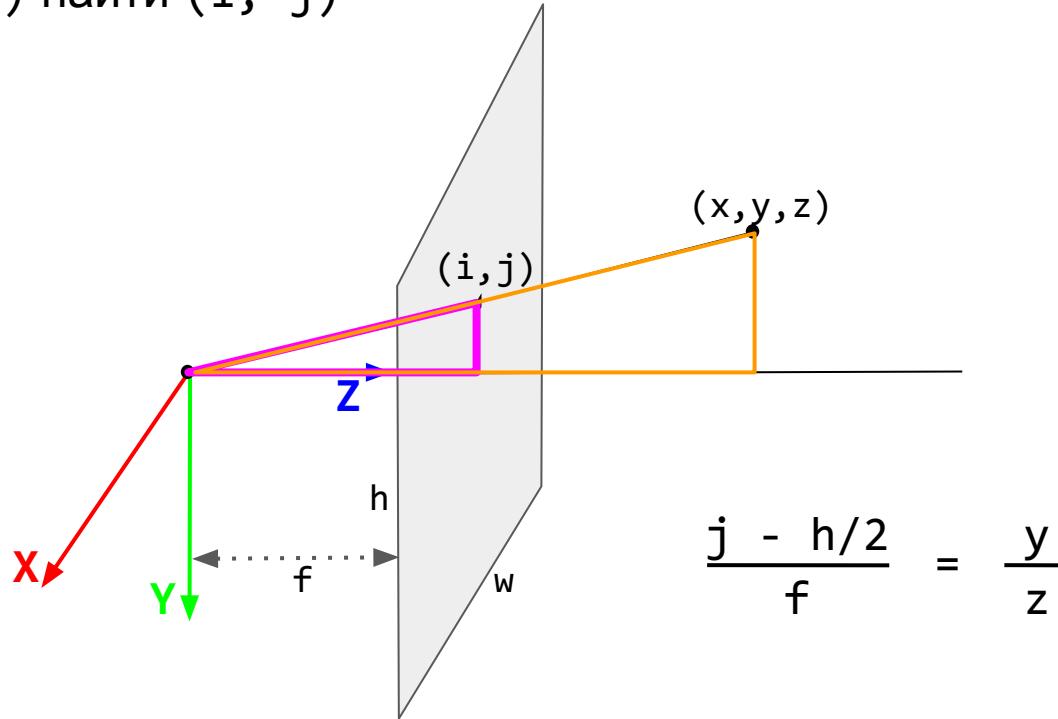


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$$i = ?$$

$$j = ?$$

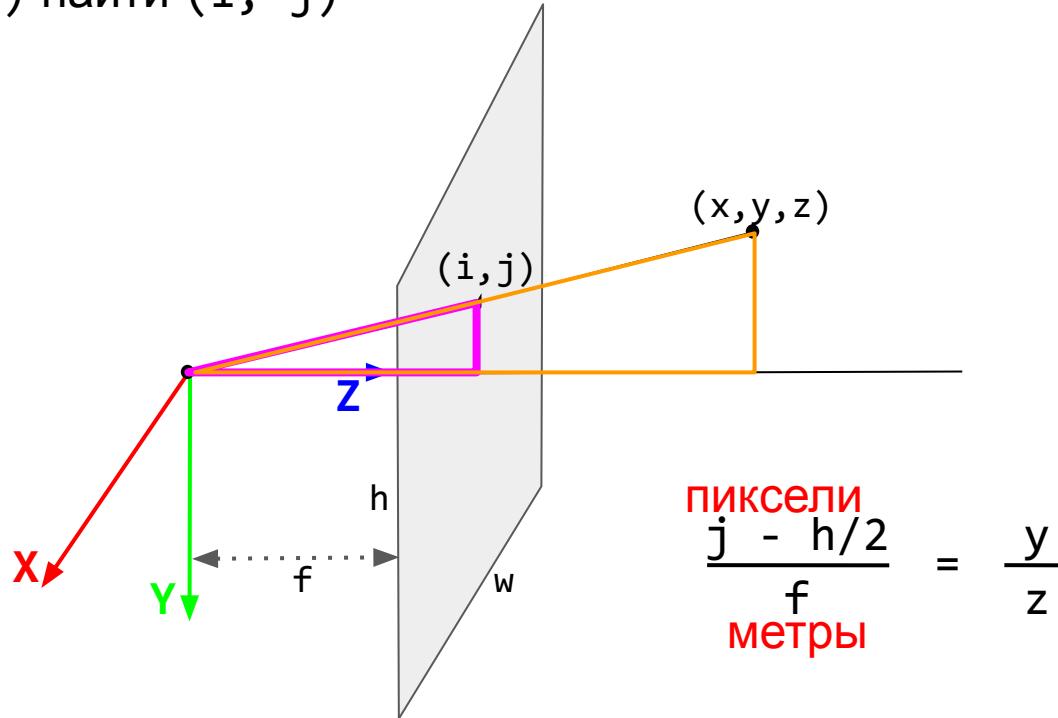


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$i = ?$

$j = ?$



# Проекция изображения

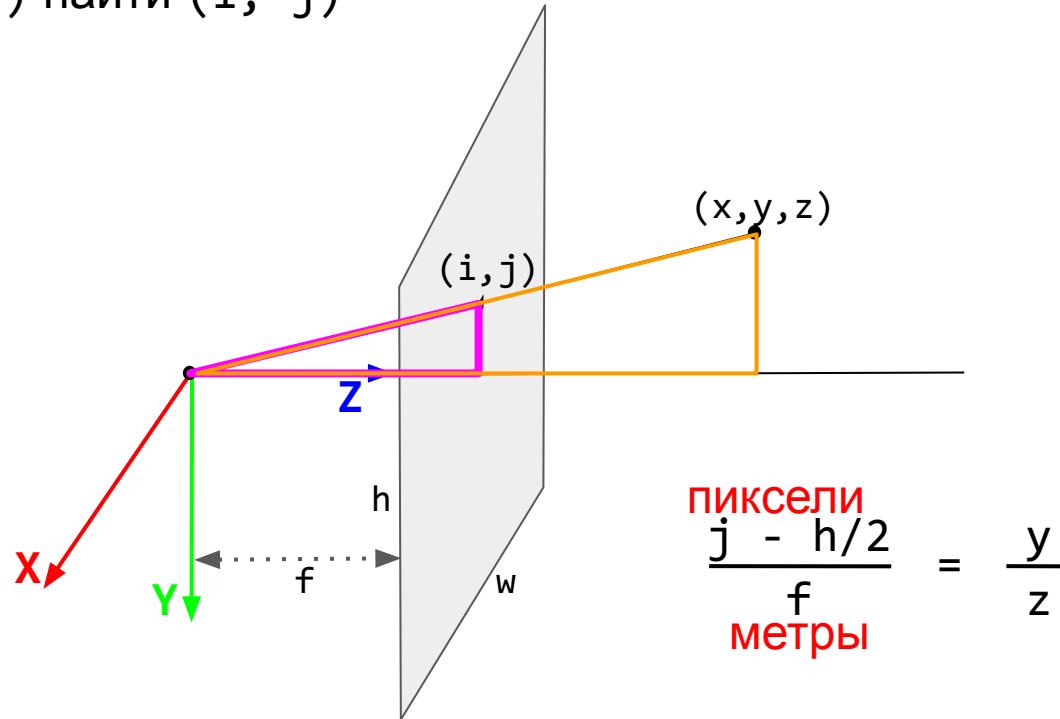
Имея  $(x, y, z)$  и  $(f, w, h)$  найти  $(i, j)$

$i = ?$

$j = ?$

Pixel Pitch:

$$p = \frac{w \text{ (пиксели)}}{w \text{ (метры)}}$$

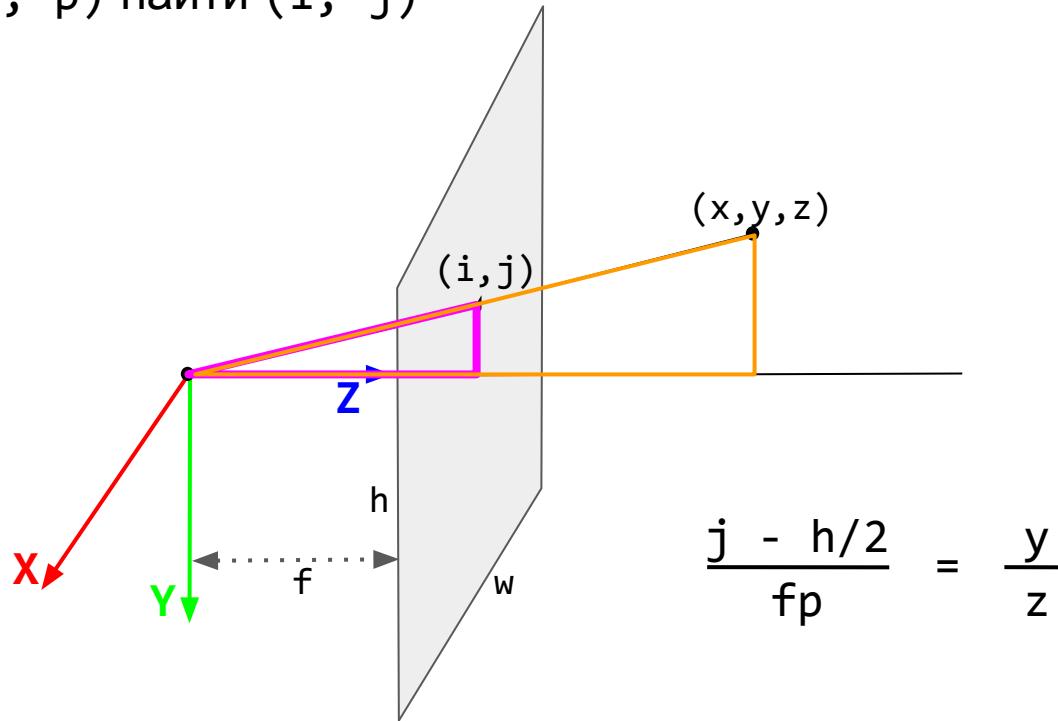


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

$$i = ?$$

$$j = \frac{fp y}{z} + \frac{h}{2}$$

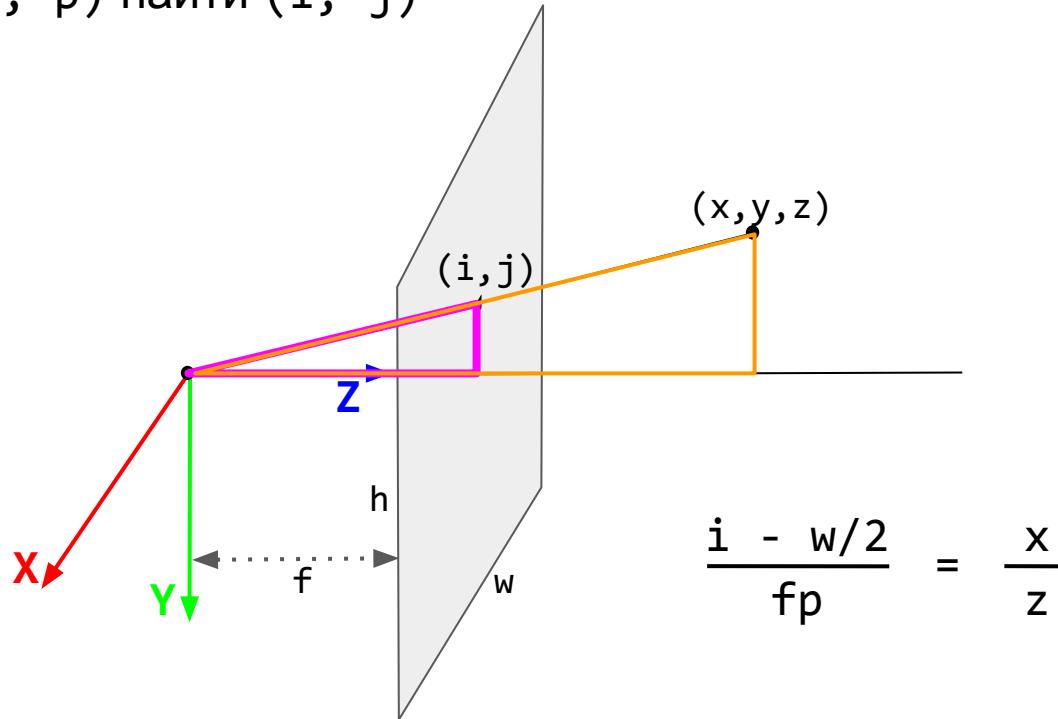


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

$$i = \frac{fp \cdot x}{z} + \frac{w}{2}$$

$$j = \frac{fp \cdot y}{z} + \frac{h}{2}$$



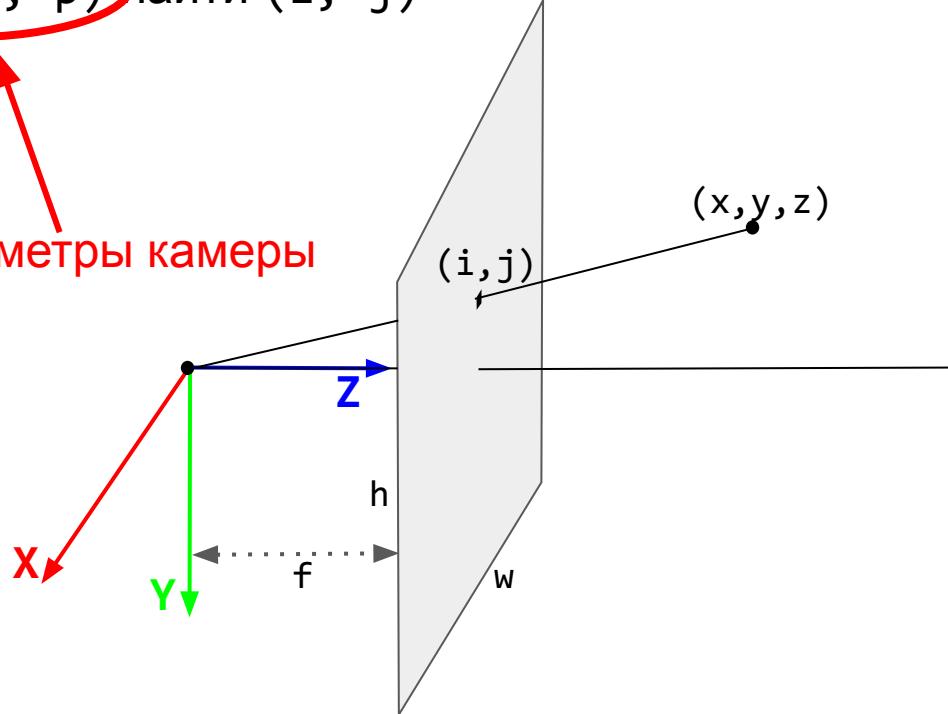
# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

$$i = \frac{fx}{z} + \frac{w}{2}$$

$$j = \frac{fy}{z} + \frac{h}{2}$$

Параметры камеры

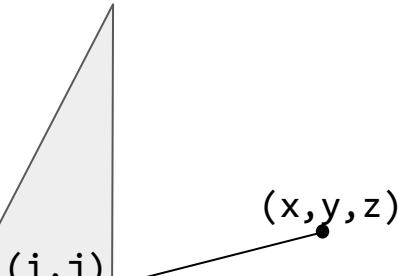


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

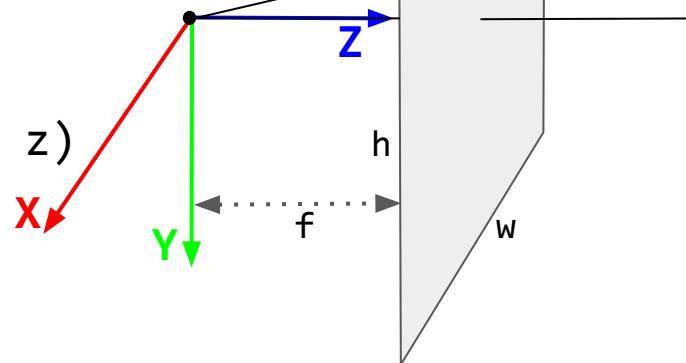
$$i = \frac{fx}{z} + \frac{w}{2}$$

$$j = \frac{fy}{z} + \frac{h}{2}$$



Имея  $(i, j)$  и  
 $(f, w, h, p)$  найти  $(x, y, z)$

$$x = ?$$



$$y = ?$$

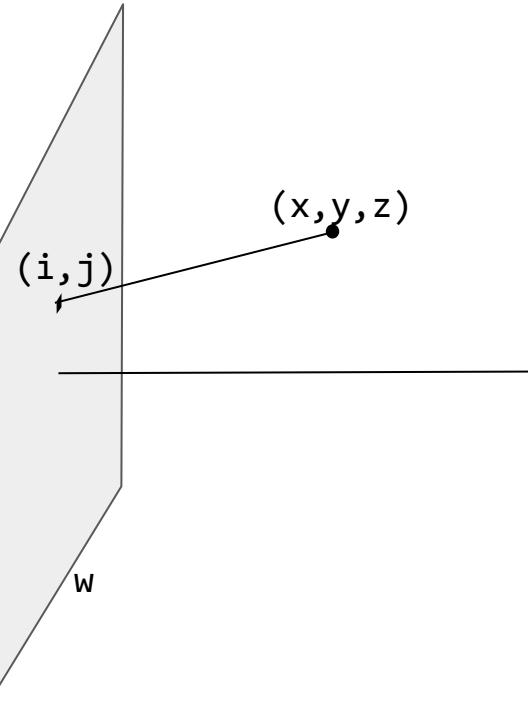
$$z = ?$$

# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

$$i = \frac{fx}{z} + \frac{w}{2}$$

$$j = \frac{fy}{z} + \frac{h}{2}$$



Имея  $(i, j)$  и

$(f, w, h, p)$  найти  $(x, y, z)$

$$x = ?$$



$$x$$

$$y = ?$$

$$z = ?$$

$$h$$

$$f$$

$$w$$

# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

$$i = \frac{fx}{z} + \frac{w}{2}$$

$$j = \frac{fy}{z} + \frac{h}{2}$$

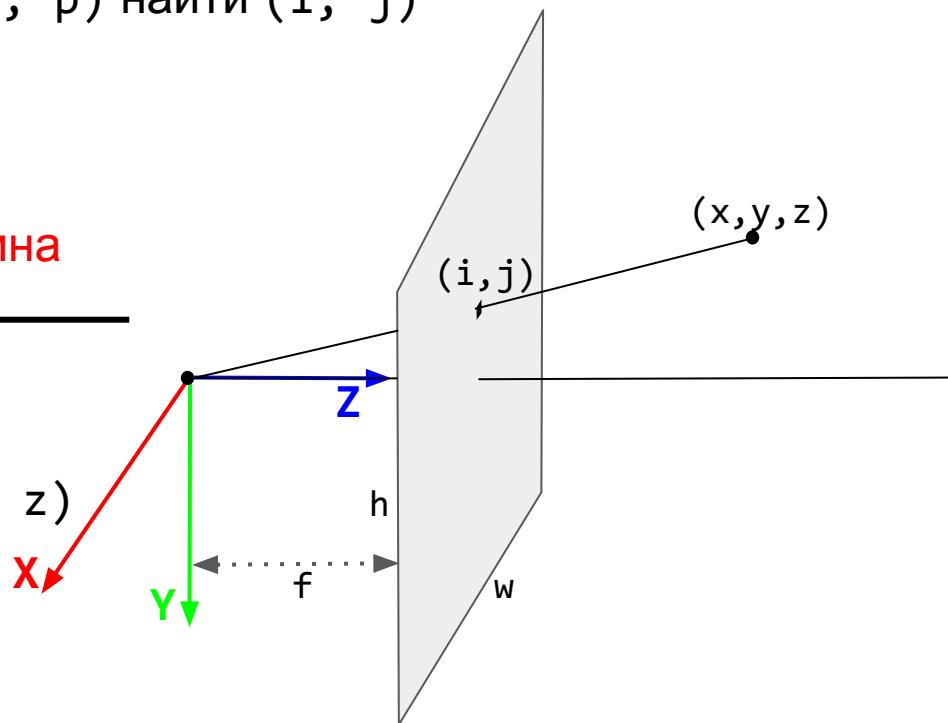
Глубина

Имея  $(i, j)$  и  
 $(f, w, h, p)$  найти  $(x, y, z)$

$$x = ?$$

$$y = ?$$

$$z = ?$$

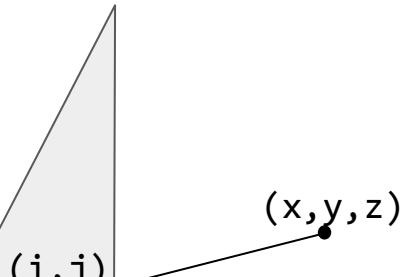


# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

$$i = \frac{fx}{z} + \frac{w}{2}$$

$$j = \frac{fy}{z} + \frac{h}{2}$$



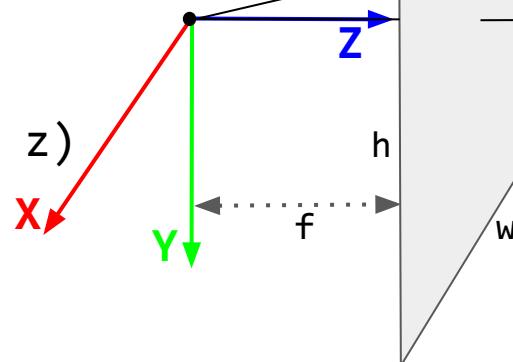
Имея  $(i, j, d)$  и

$(f, w, h, p)$  найти  $(x, y, z)$

$$x = ?$$

$$y = ?$$

$$z = d$$



# Проекция изображения

Имея  $(x, y, z)$  и  $(f, w, h, p)$  найти  $(i, j)$

$$i = \frac{fp \cdot x}{z} + \frac{w}{2}$$

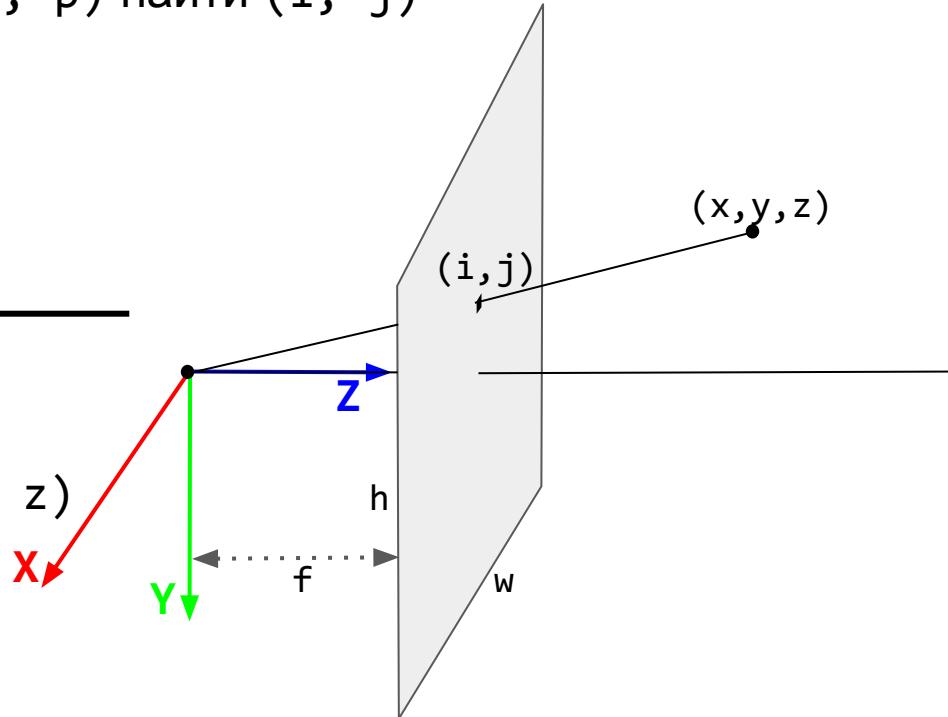
$$j = \frac{fp \cdot y}{z} + \frac{h}{2}$$

Имея  $(i, j, d)$  и  
 $(f, w, h, p)$  найти  $(x, y, z)$

$$x = \frac{(i - w/2)d}{fp}$$

$$y = \frac{(j - h/2)d}{fp}$$

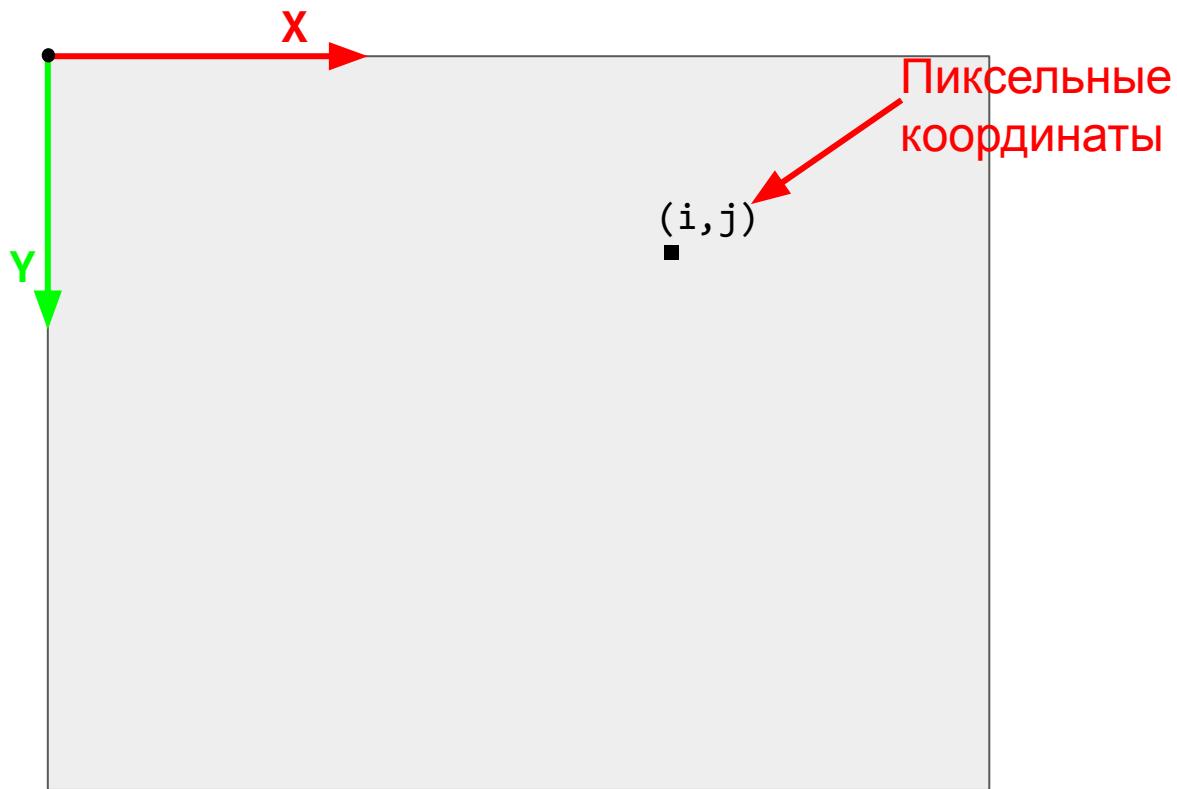
$$z = d$$



# Координаты изображения

# Координаты изображения

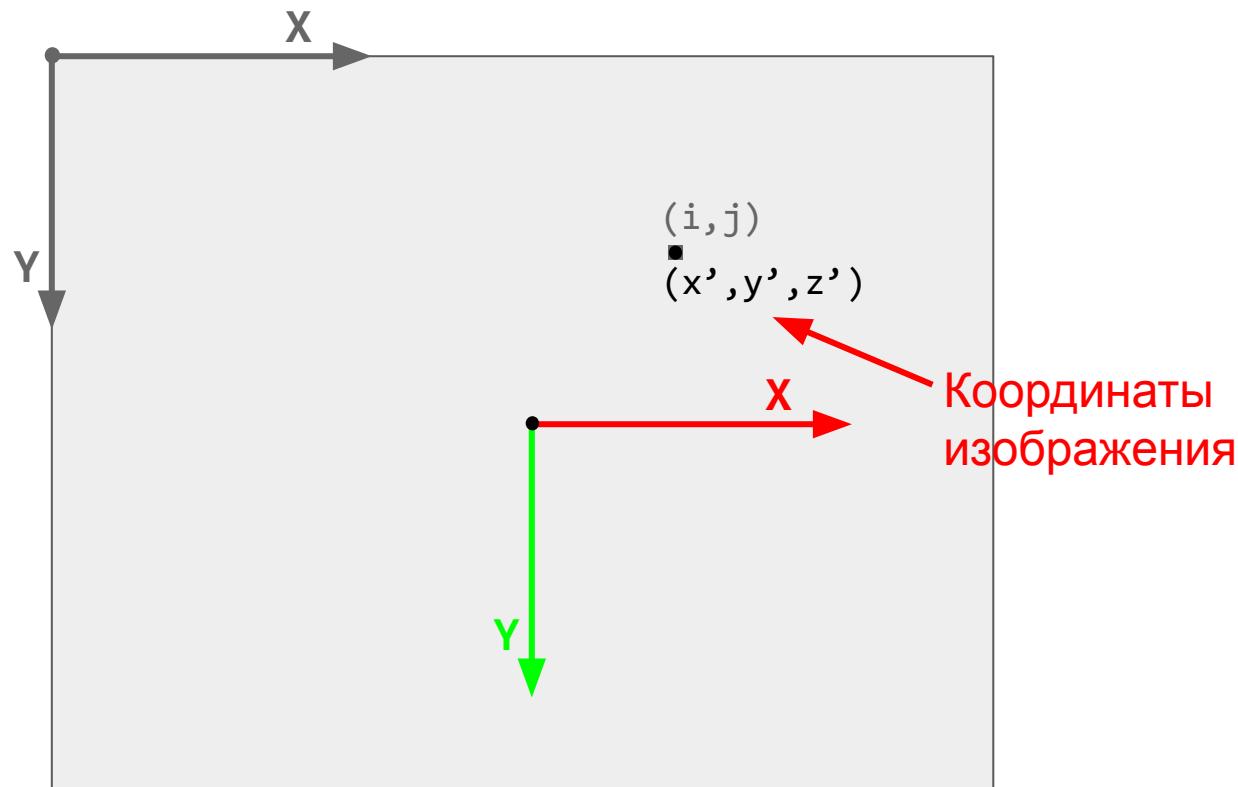
---



# Координаты изображения

3D координаты  
пикселя  
относительно  
камеры

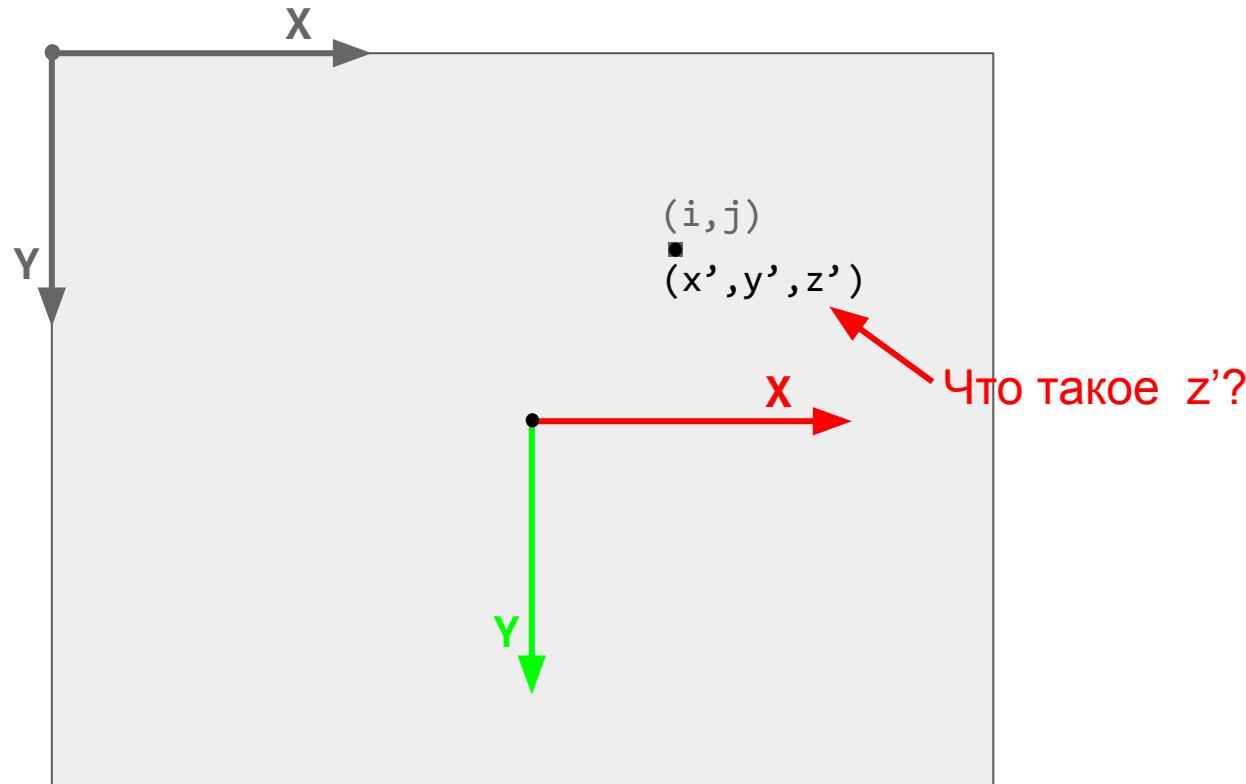
$x'$ ,  $y'$ ,  $z'$  в  
метрах



# Координаты изображения

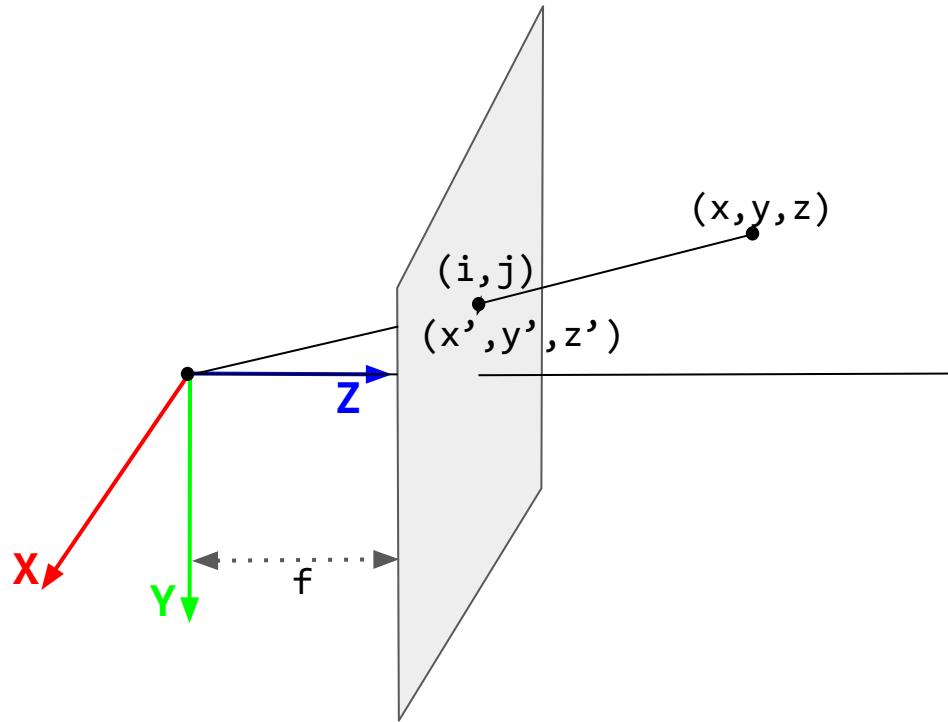
3D координаты  
пикселя  
относительно  
камеры

$x'$ ,  $y'$ ,  $z'$  в  
метрах



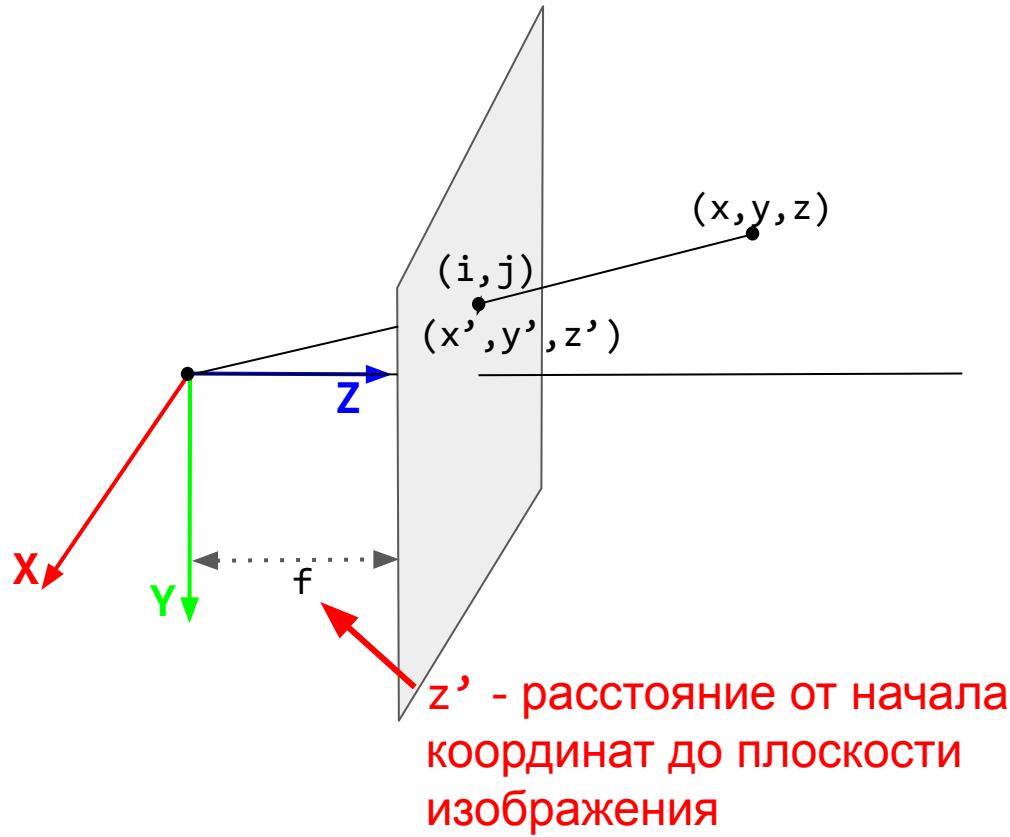
# Координаты изображения

---



# Координаты изображения

---



# Координаты изображения

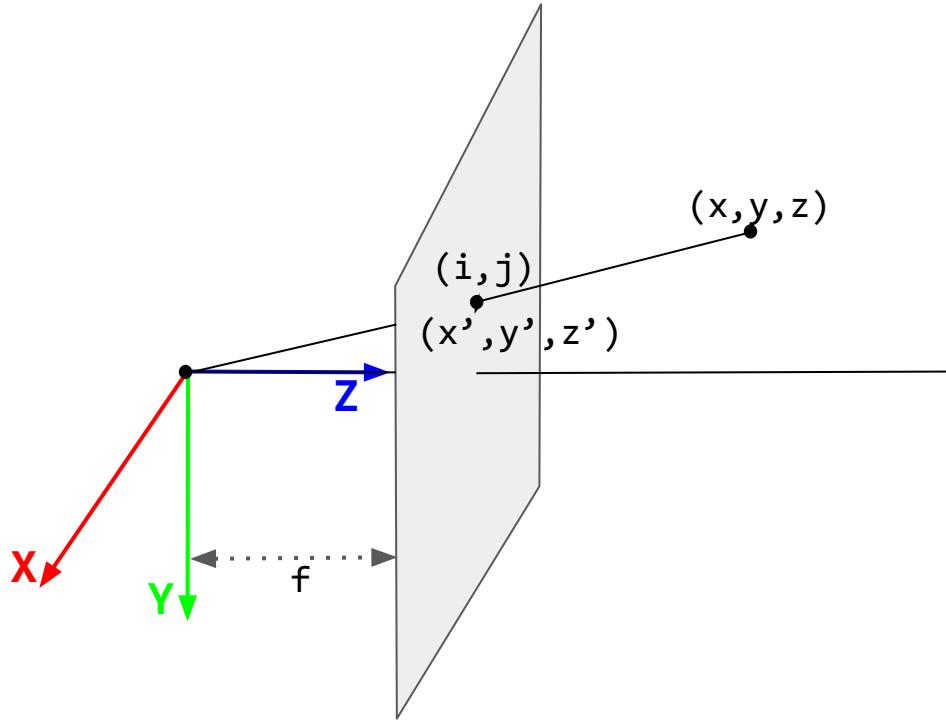
---

$$i = px' + \frac{w}{2}$$
$$j = py' + \frac{h}{2}$$

$$x' = \frac{fx}{z}$$

$$y' = \frac{fy}{z}$$

$$z' = f$$



# Координаты изображения

(метры -> пиксели)

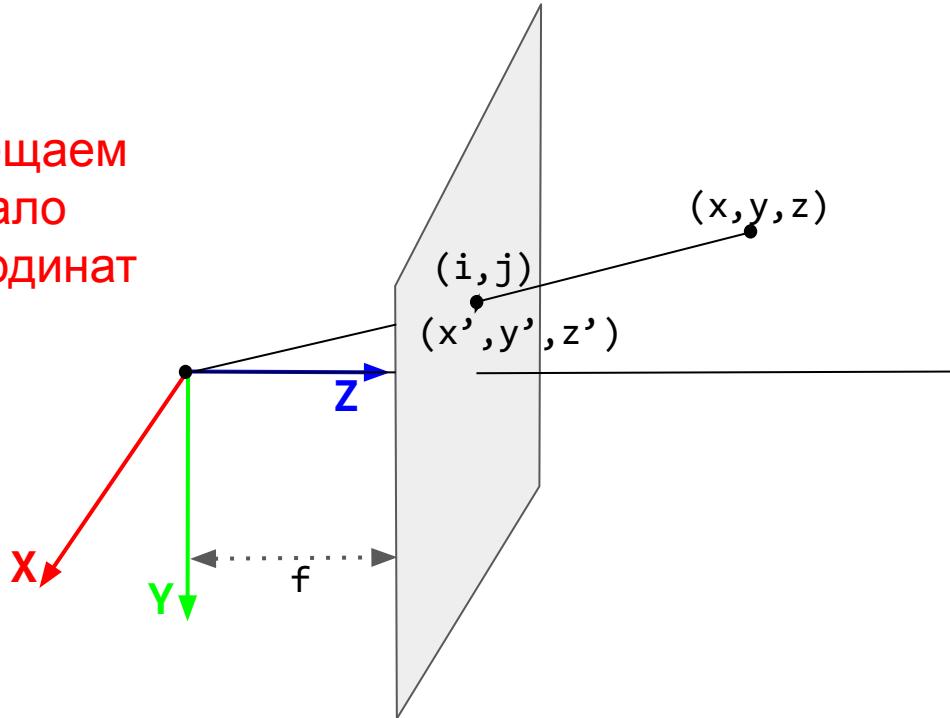
$$\begin{aligned} i &= px' + \frac{w}{2} \\ j &= py' + \frac{h}{2} \end{aligned}$$

Смещаем  
начало  
координат

$$x' = \frac{fx}{z}$$

$$y' = \frac{fy}{z}$$

$$z' = f$$



# Нормализованные координаты изображения

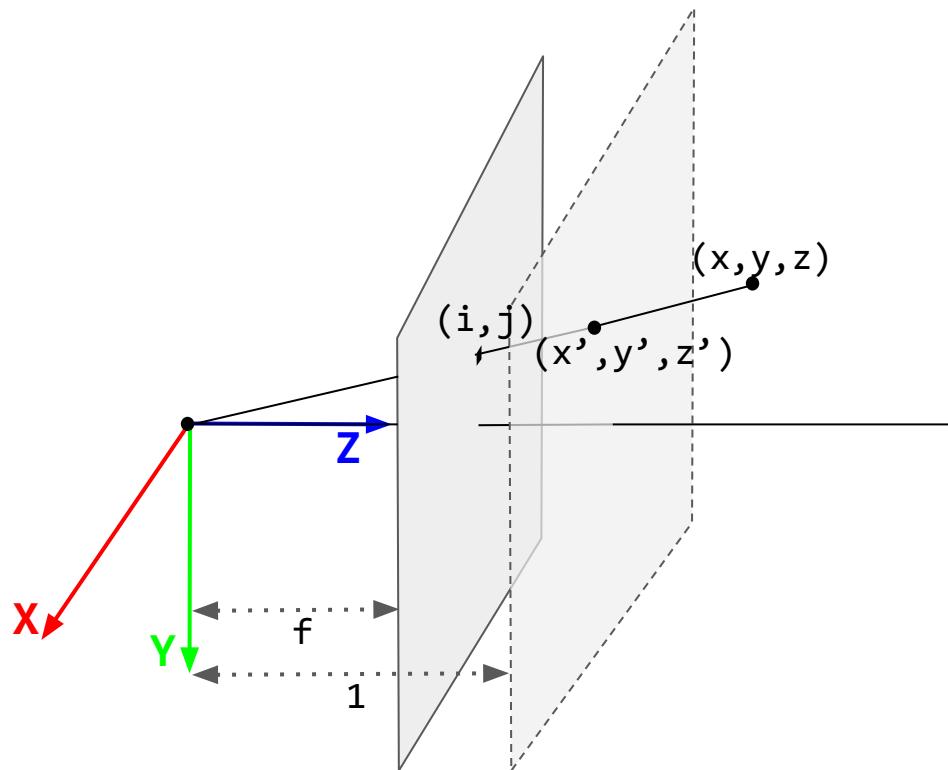
---

$$i = \textcolor{red}{f} p x' + \frac{w}{2}$$
$$j = \textcolor{red}{f} p y' + \frac{h}{2}$$

$$x' = \frac{x}{z}$$

$$y' = \frac{y}{z}$$

$$z' = 1$$



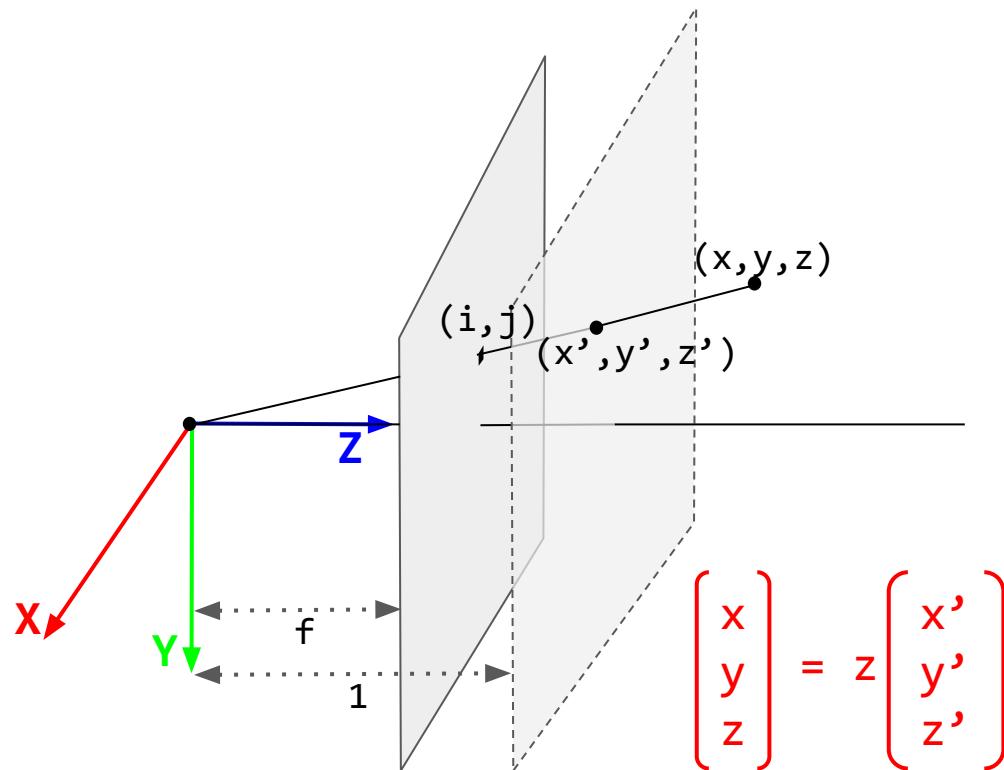
# Нормализованные координаты изображения

$$i = fpx' + \frac{w}{2}$$
$$j = fpy' + \frac{h}{2}$$

$$x' = \frac{x}{z}$$

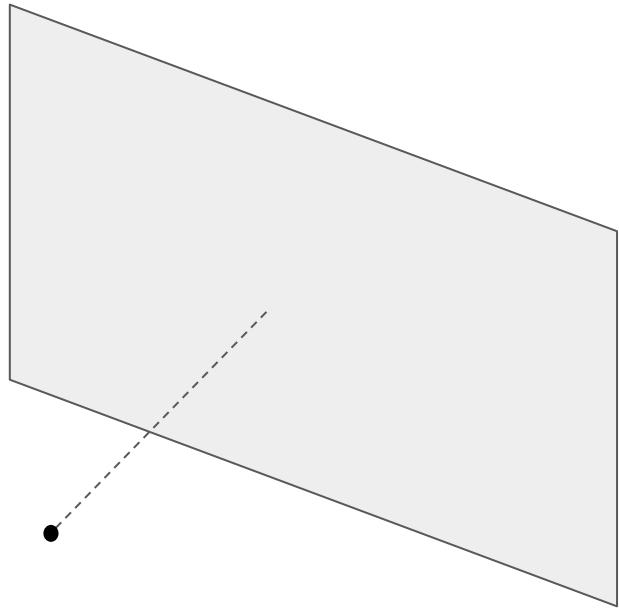
$$y' = \frac{y}{z}$$

$$z' = 1$$

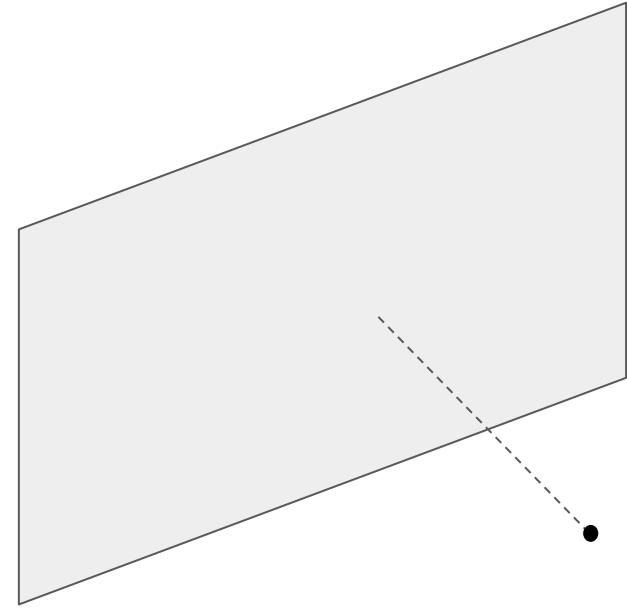


# Стерео

---



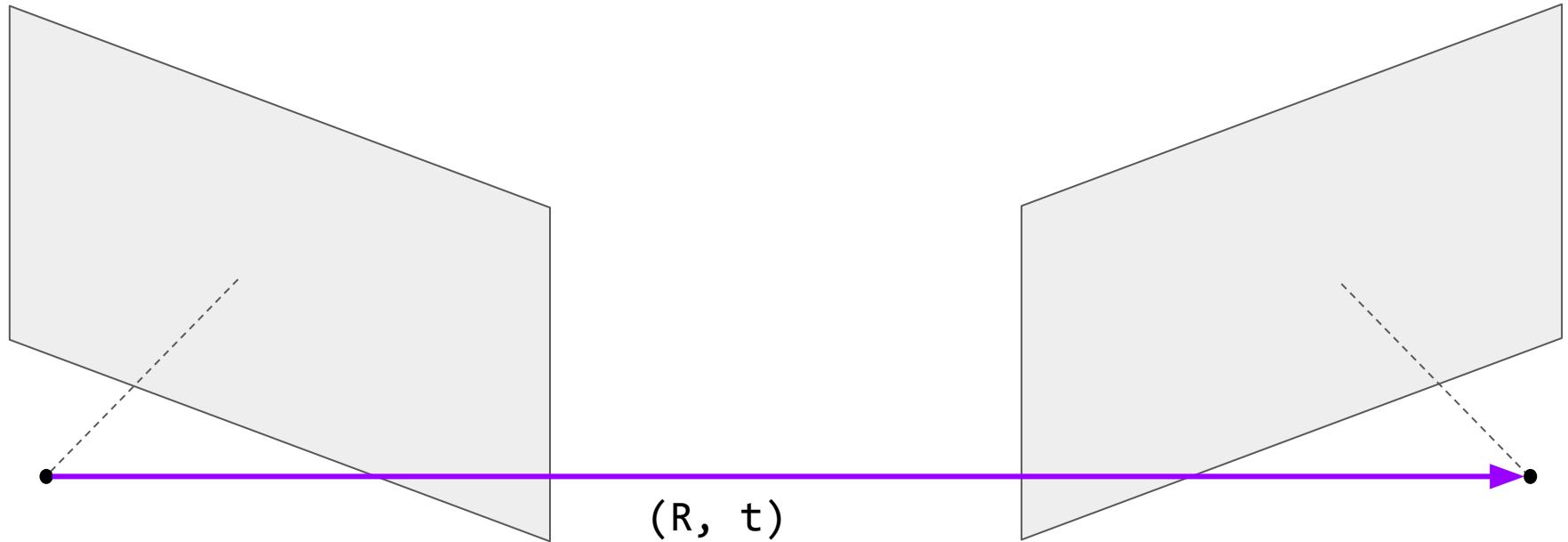
Камера 1



Камера 2

# Стерео

---



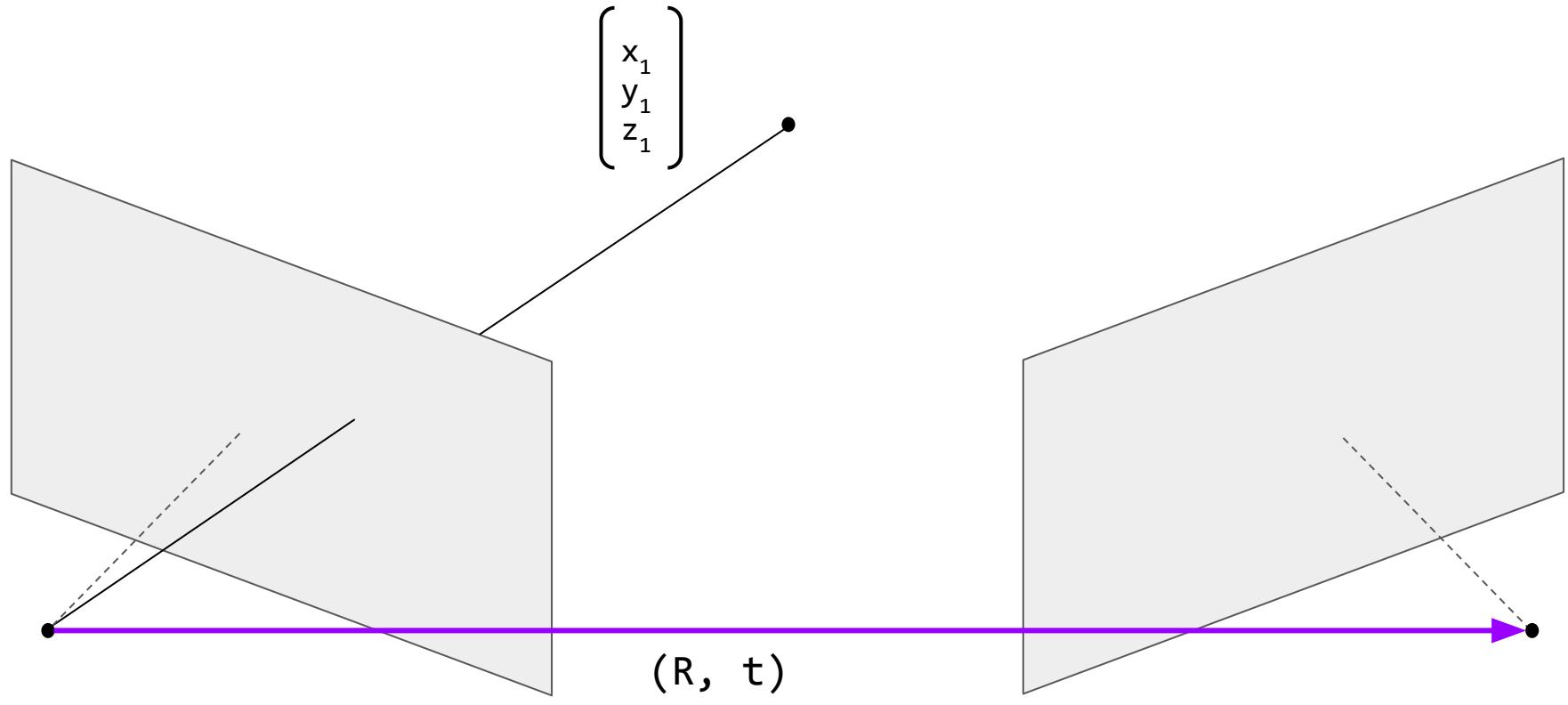
Камера 1

$(R, t)$

Камера 2

# Стерео

---

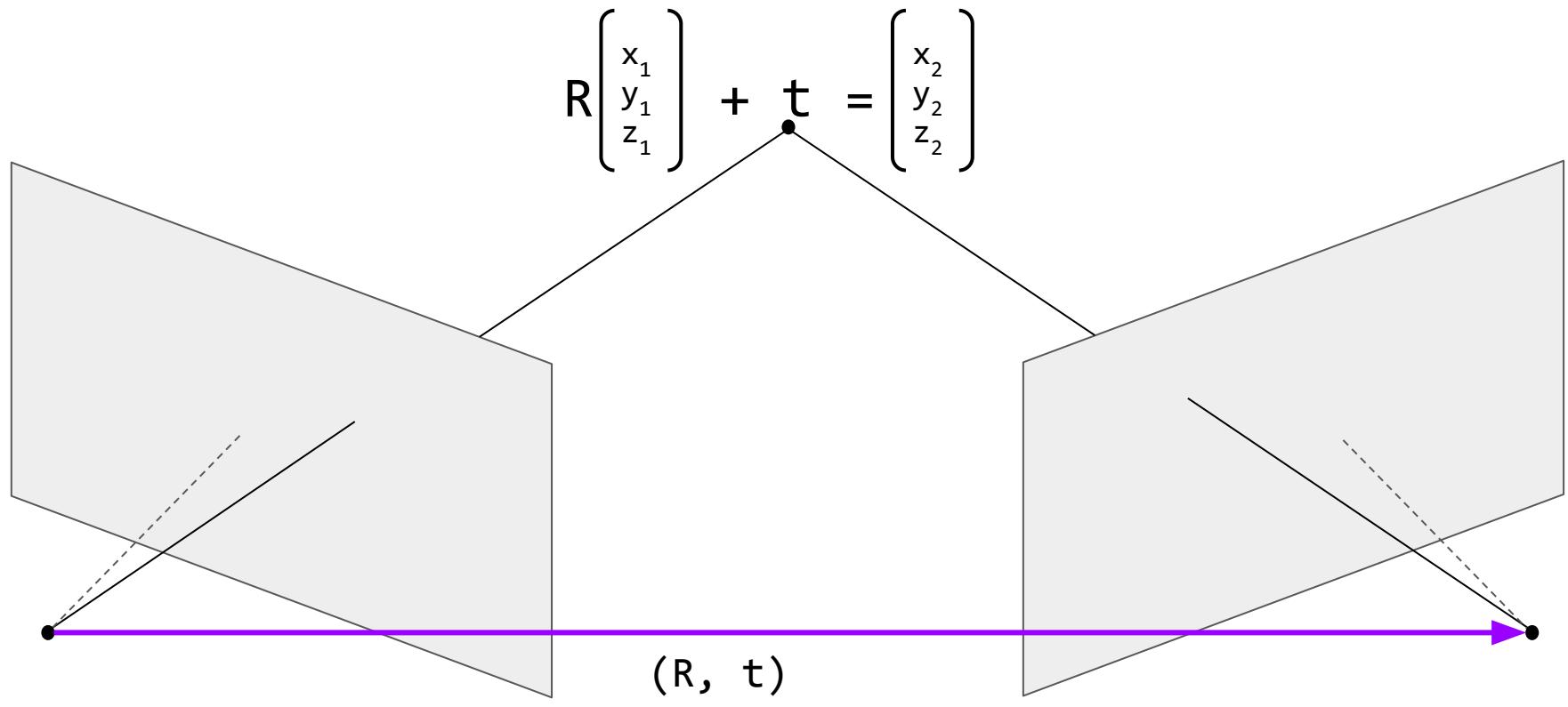


Камера 1

Камера 2

# Стерео

---

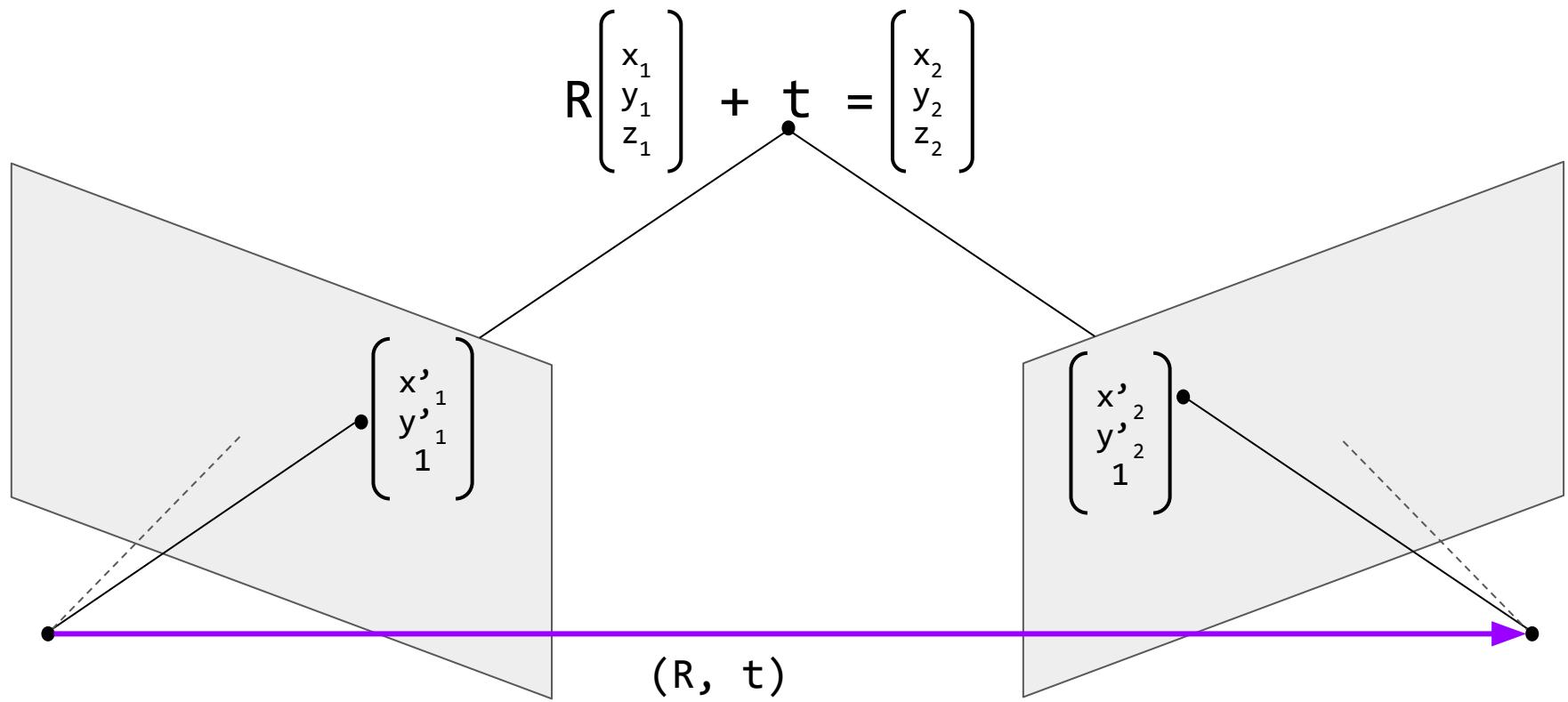


Камера 1

Камера 2

# Стерео

---

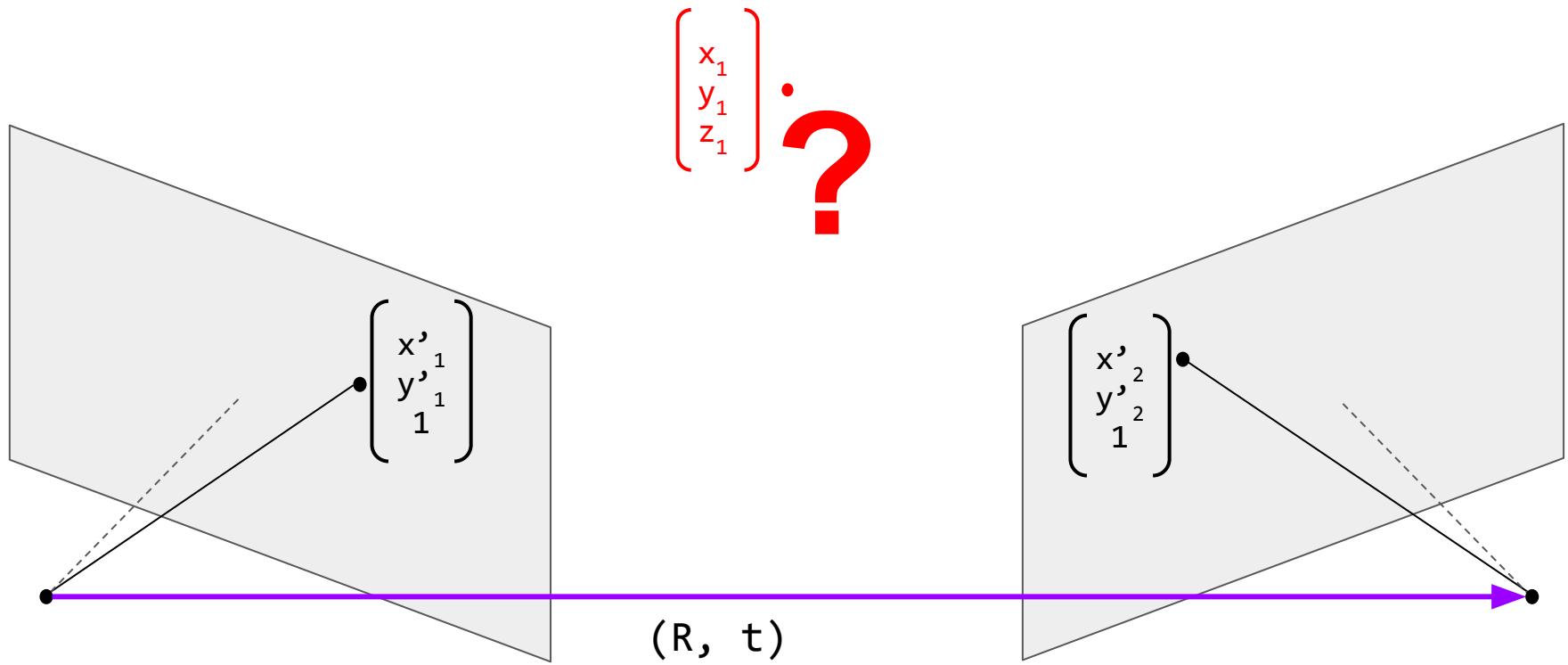


Камера 1

Камера 2

# Стерео

---



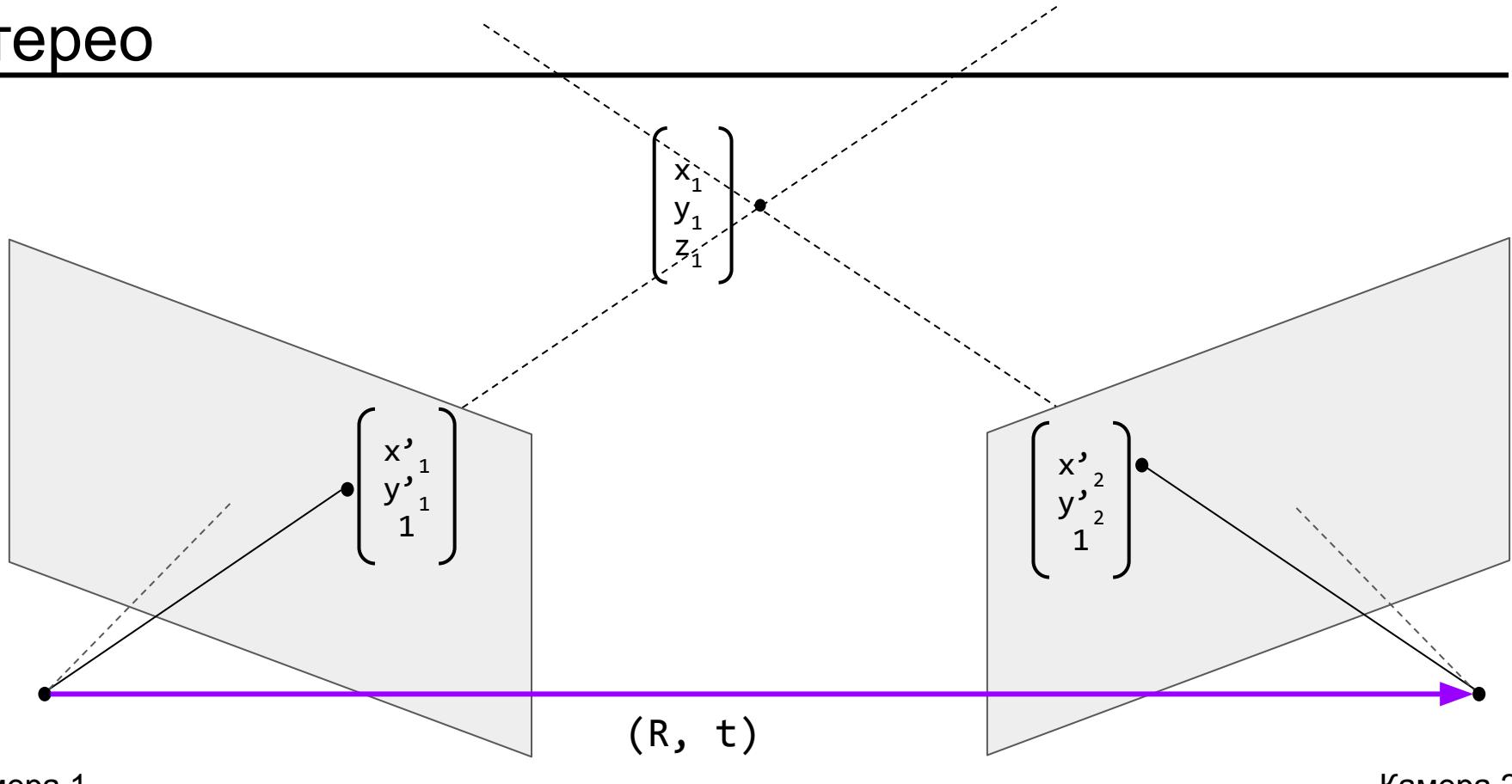
Камера 1

$(R, t)$

Камера 2

# Стерео

---



Камера 1

Камера 2

# Стерео

Линия в 3D может быть представлена как:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} + s \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix}$$

Некоторая  
точка на линии

Параметр

Наклон  
линии

Камера

a 2

# Стерео

Линия в 3D может быть представлена как:

Линия 1

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

Линия 2

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + S_2 \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \xrightarrow{\text{Преобразование к координатам Камеры 1}} -t + R^{-1}S_2 \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

Камера

а 2

# Стерео

Линия в 3D может быть представлена как:

Линия 1

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + S_1 \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \quad \begin{pmatrix} x \\ y \\ z \end{pmatrix} = z \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$$

Линия 2

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + S_2 \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \xrightarrow{\text{Преобразование к координатам Камеры 1}} -t + R^{-1} S_2 \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}$$

Камера

a 2

# Стерео

Линия в 3D может быть представлена как:

Линия 1

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + z_1 \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix}$$

Линия 2

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + z_2 \begin{pmatrix} x'_2 \\ y'_2 \\ 1 \end{pmatrix}$$

Преобразование к  
координатам Камеры 1

$$-t + R^{-1}z_2 \begin{pmatrix} x'_2 \\ y'_2 \\ 1 \end{pmatrix}$$

Камера

а 2

# Стерео

Линия в 3D может быть представлена как:

Линия 1

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + z_1 \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix}$$

Линия 2

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + z_2 \begin{pmatrix} x'_2 \\ y'_2 \\ 1 \end{pmatrix}$$

Преобразование к  
координатам Камеры 1

$\rightarrow -t + R^{-1}z_2 \begin{pmatrix} x'_2 \\ y'_2 \\ 1 \end{pmatrix}$

- Решаем системы линейных  
уравнений

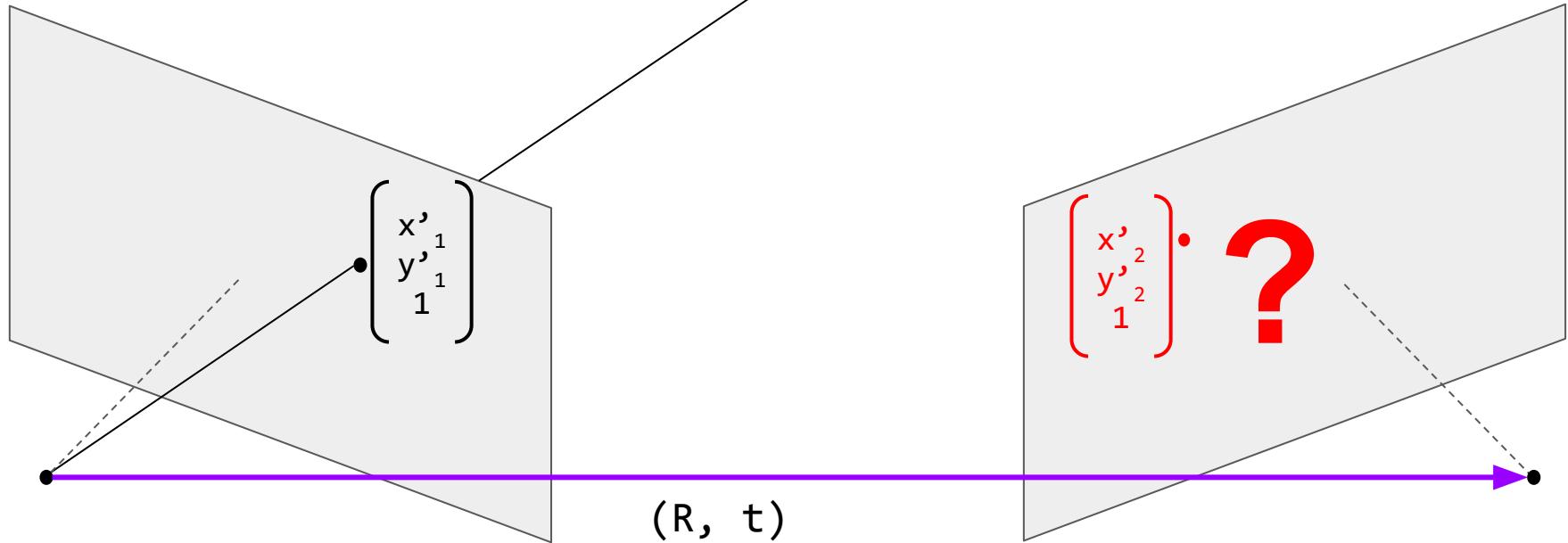
$$z_1 \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix} = -t + R^{-1}z_2 \begin{pmatrix} x'_2 \\ y'_2 \\ 1 \end{pmatrix}$$

Камера

a 2

# Стерео

---



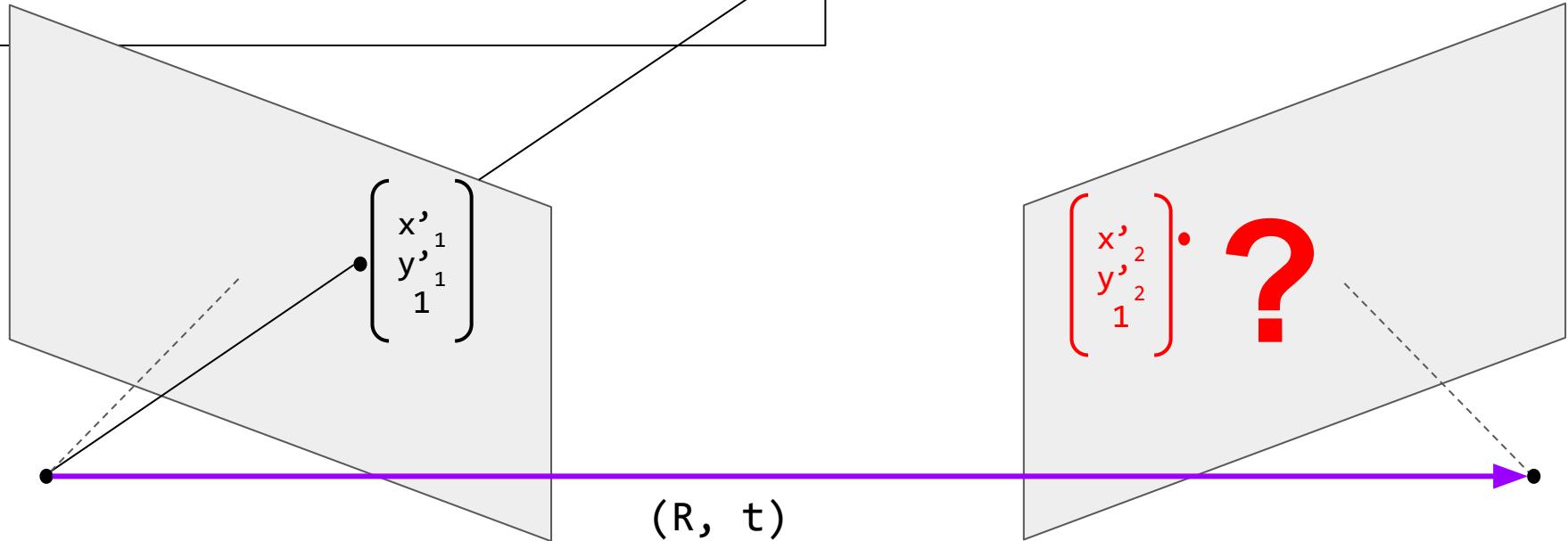
Камера 1

$(R, t)$

Камера 2

# Стерео

Матчинг признаков работает не очень хорошо. Как еще геометрия может помочь?

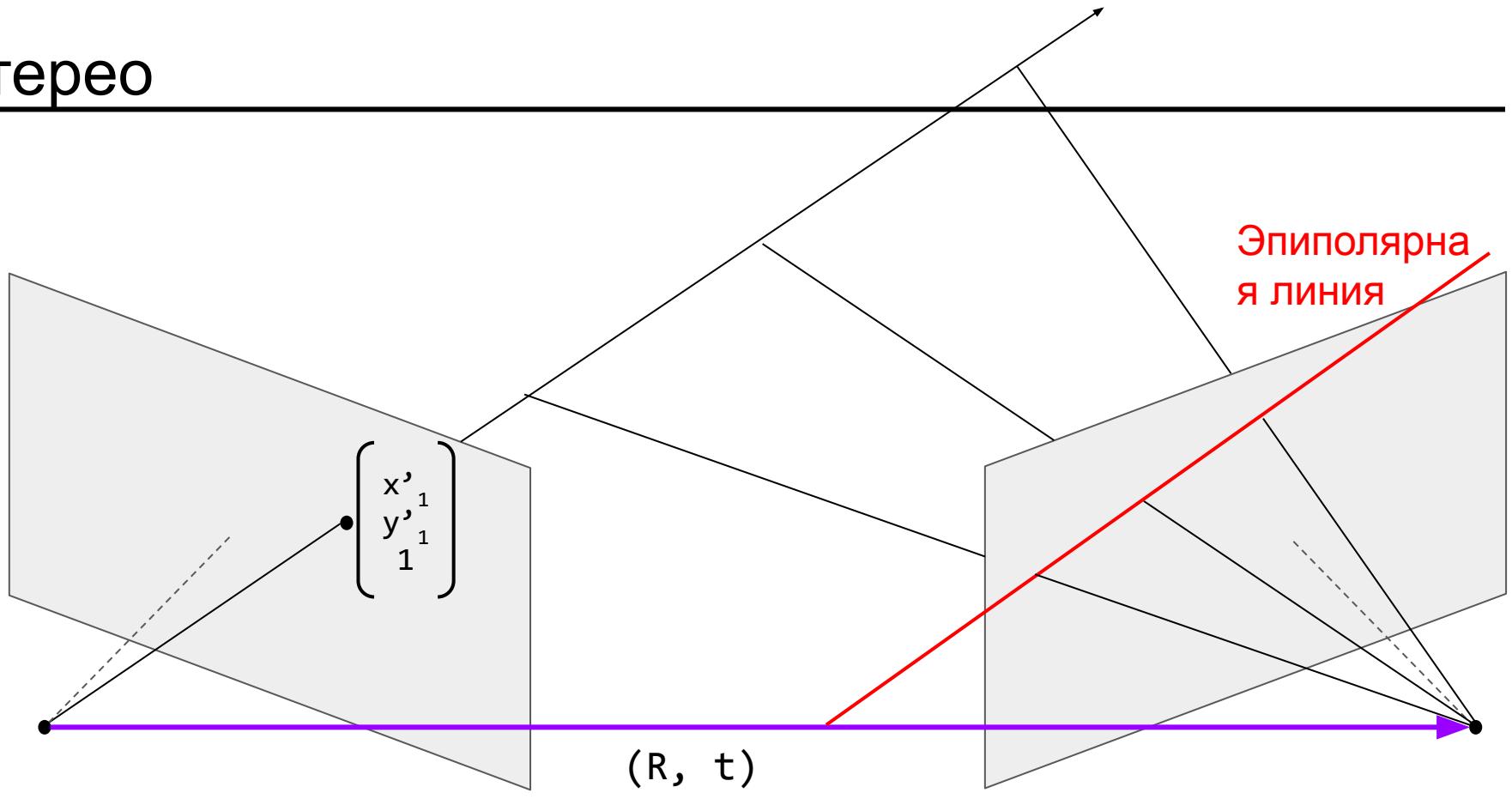


Камера 1

$(R, t)$

Камера 2

# Стерео



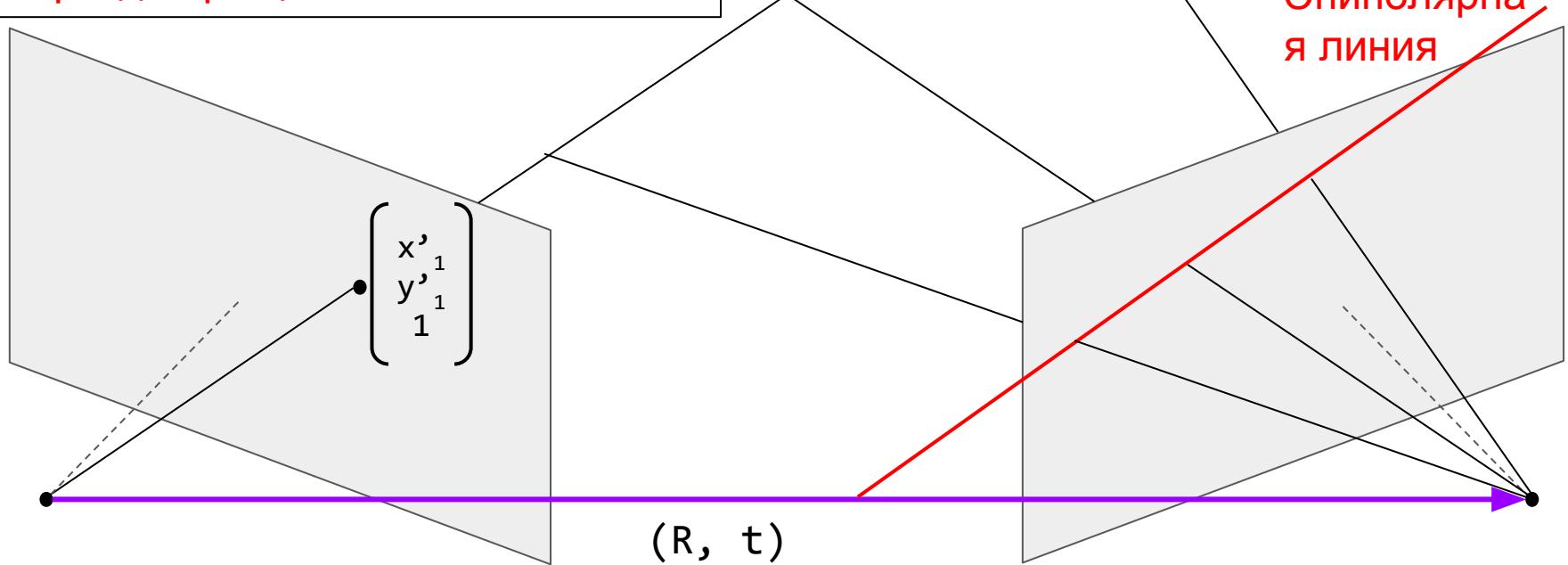
Камера 1

$(R, t)$

Камера 2

# Стерео

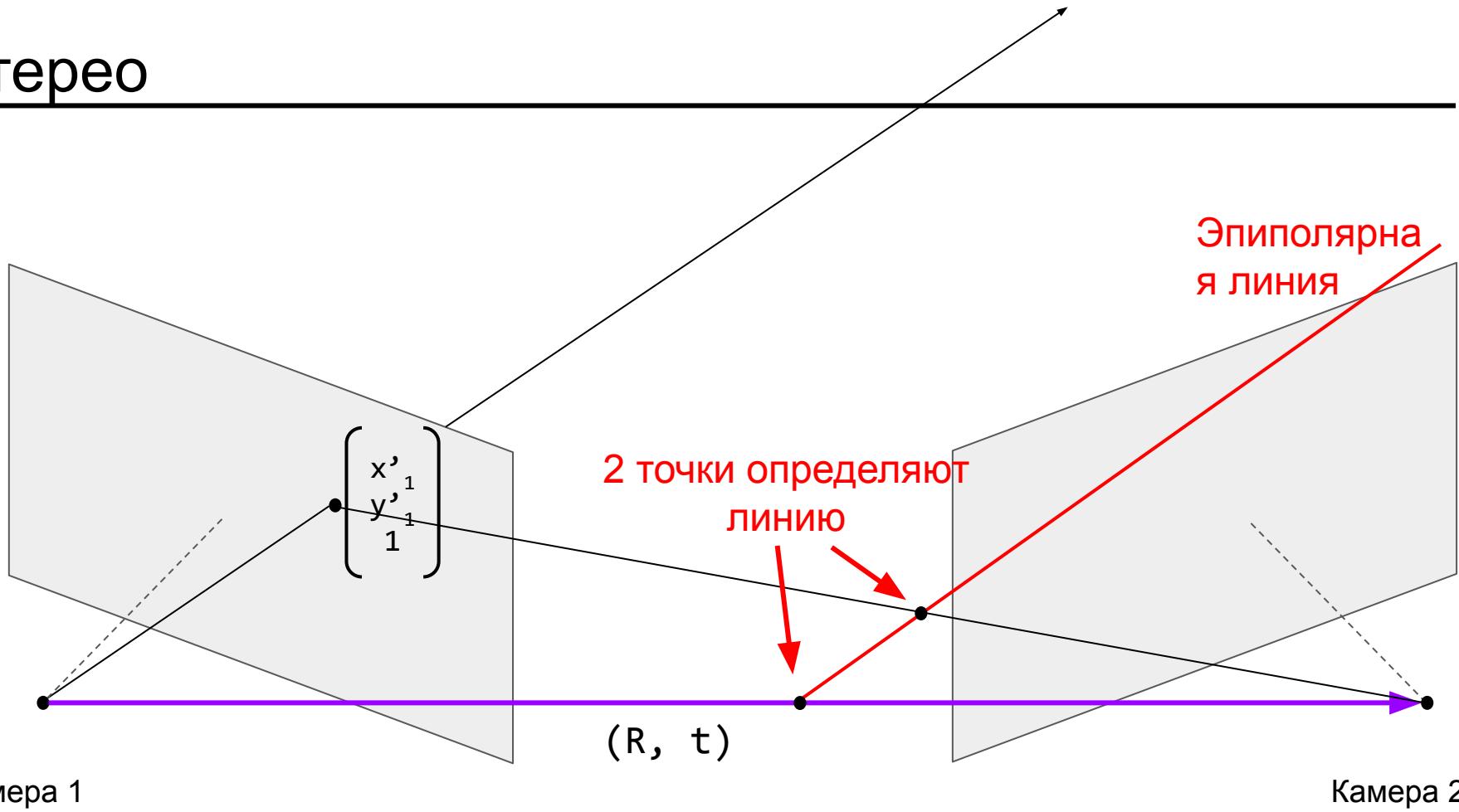
Поиск матчей вдоль этой линии  
гораздо проще



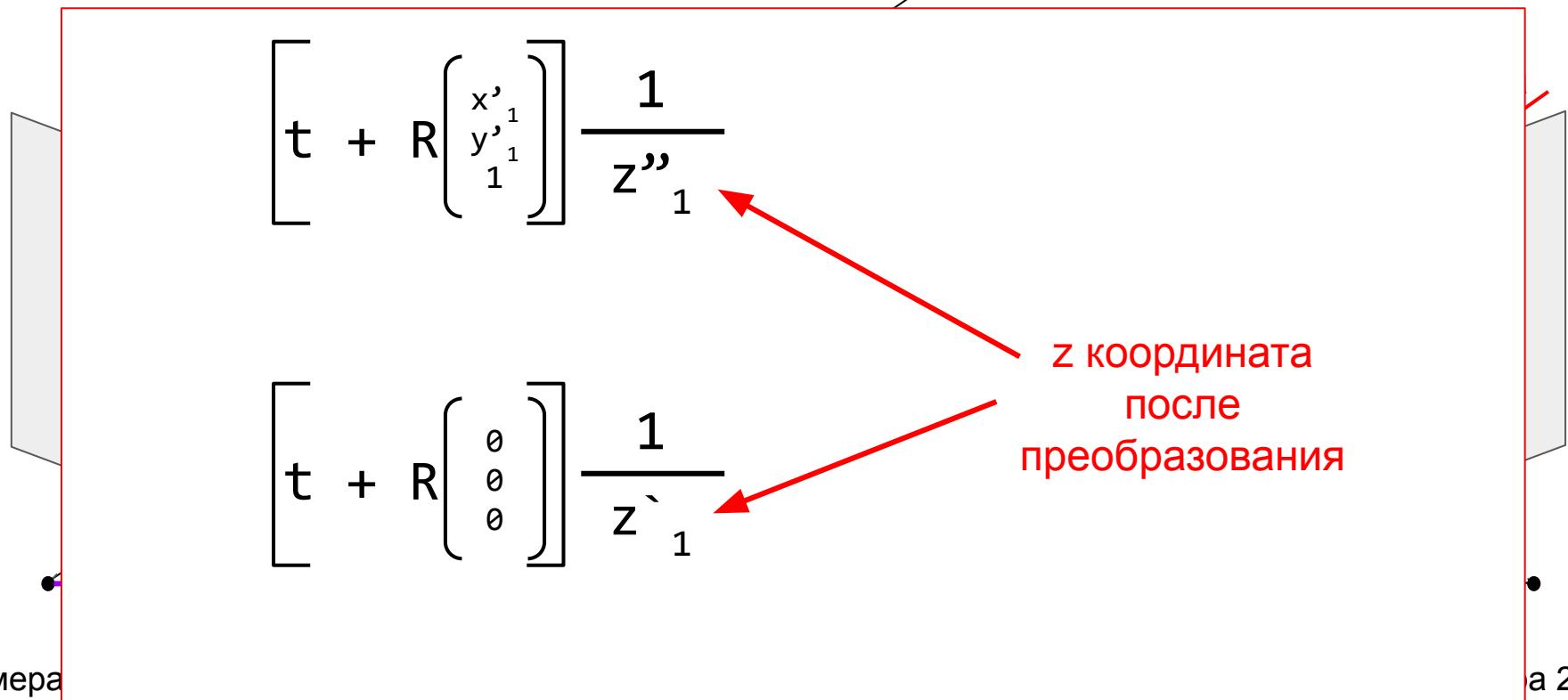
Камера 1

Камера 2

# Стерео



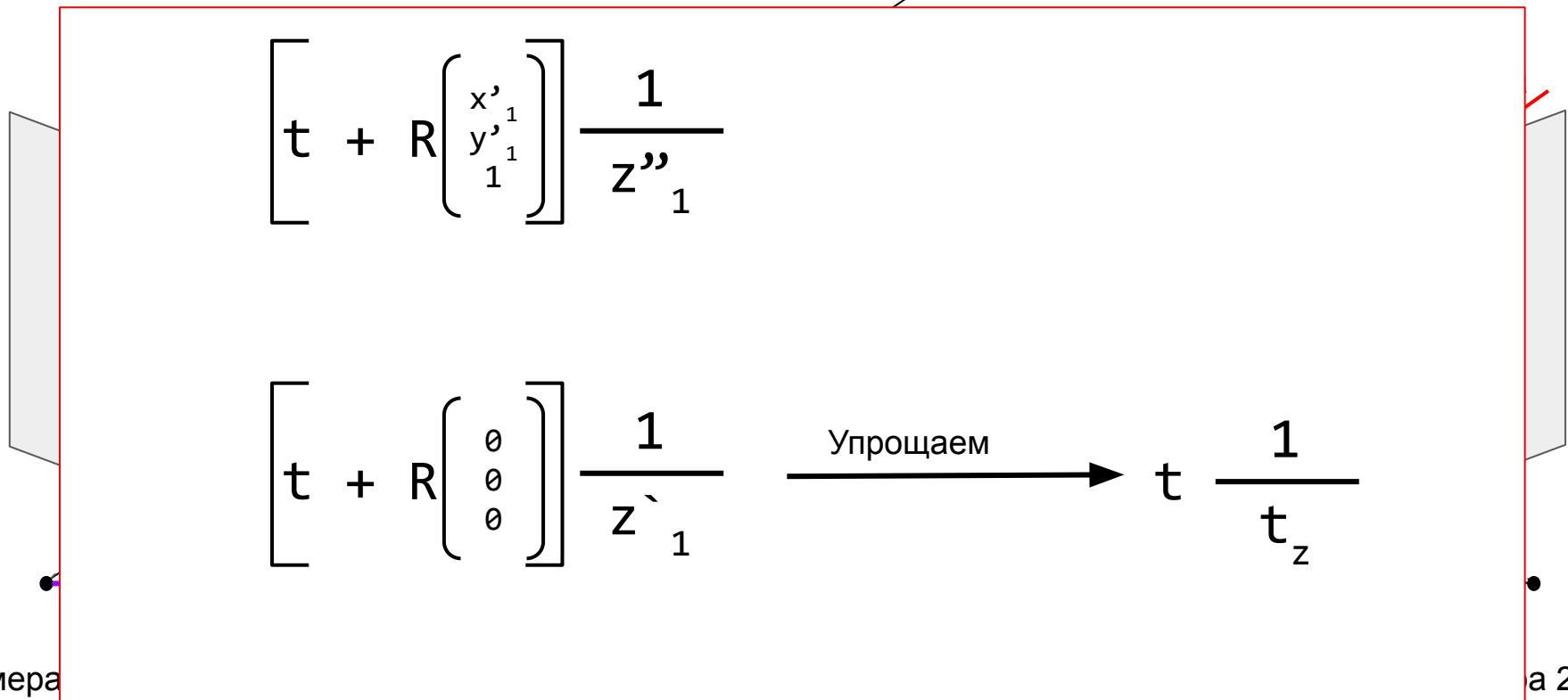
# Стерео



Камера

a 2

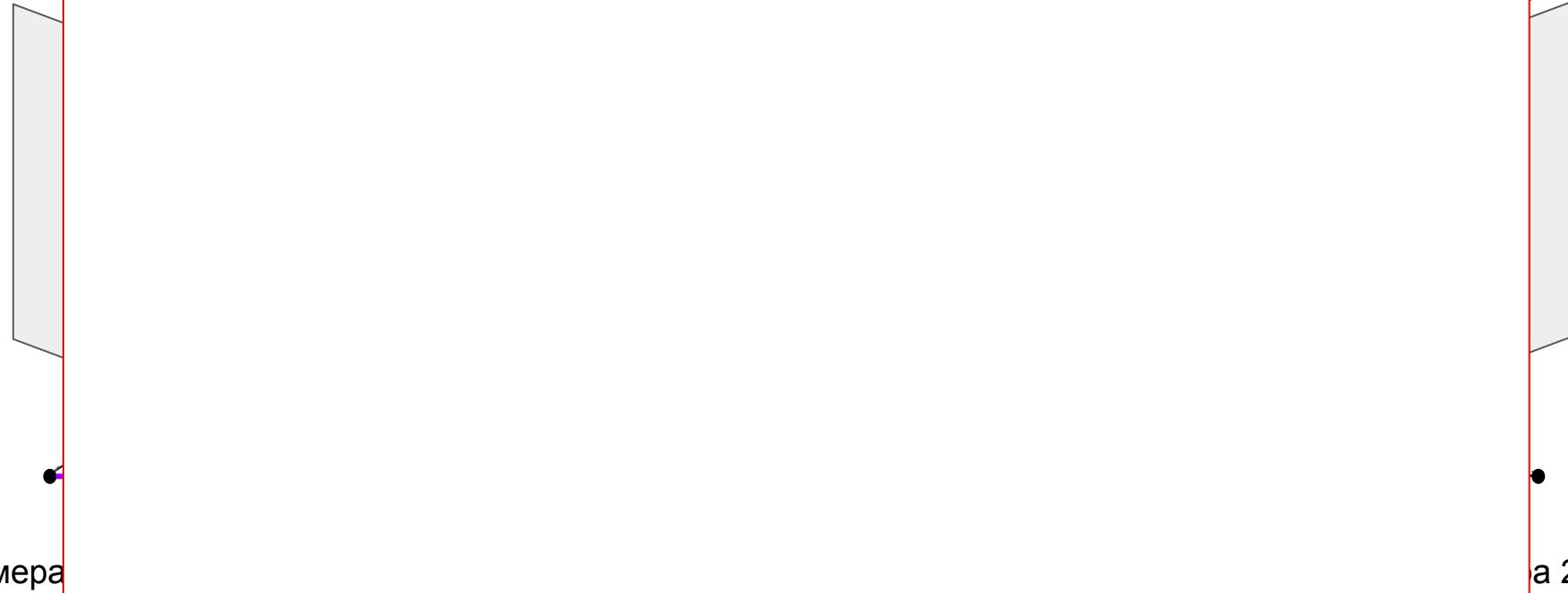
# Стерео



# Стерео

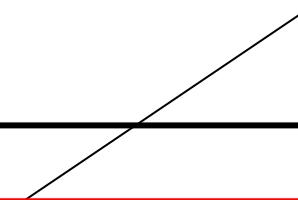
---

Как определить линию на плоскости?



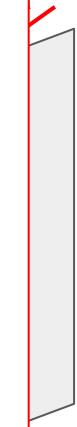
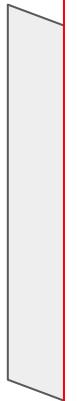
# Стерео

---



Как определить линию на плоскости?

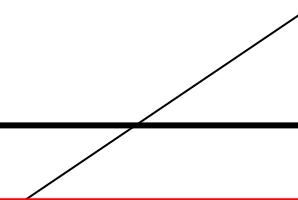
$$y = mx + b$$



Камера

a 2

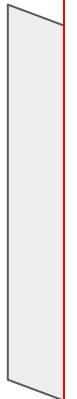
# Стерео



Как определить линию на плоскости?:

$$y = mx + b$$

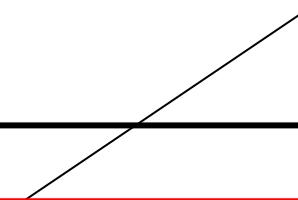
Нельзя  
определить  
вертикальную  
линию



Камера

a 2

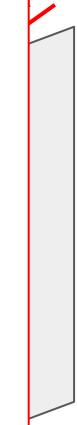
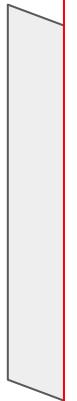
# Стерео



Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

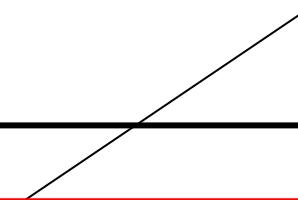
$$ax + by + c = 0$$



Камера

a 2

# Стерео



Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

$$ax + by + c = 0$$

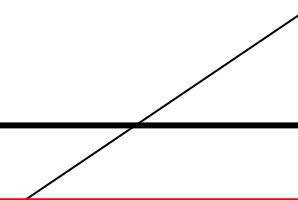
$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$



Камера

a 2

# Стерео



Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

$$ax + by + c = 0$$

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\left[ t + R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \right] \frac{1}{z''_1}$$

Обе спроектированные точки выглядят как:

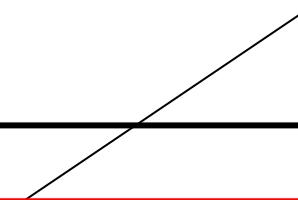
- Предполагаем что это 2D точки

Камера

$$t - \frac{1}{t_z} \bullet a_2$$

a 2

# Стерео



Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

$$ax + by + c = 0$$

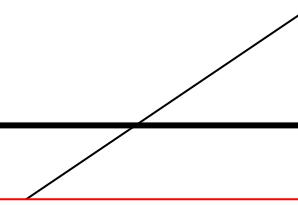
$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \left[ t + R \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} \right] \frac{1}{z'} = 0$$

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot t - \frac{1}{t_z} = 0$$

Камера

a 2

# Стерео



Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

$$ax + by + c = 0$$

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \left[ t + R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \right] \frac{1}{z'} = 0$$

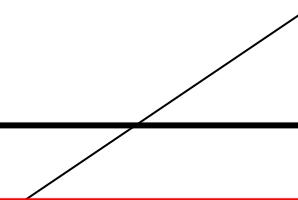
$\begin{pmatrix} a \\ b \\ c \end{pmatrix}$  - Ортогонально  
обоим  
векторам

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot t \frac{1}{t_z} = 0$$

Камера

a 2

# Стерео



Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

$$ax + by + c = 0$$

$$t - \frac{1}{t_z} \times \left[ t + R \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \right] \frac{1}{z''_1} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

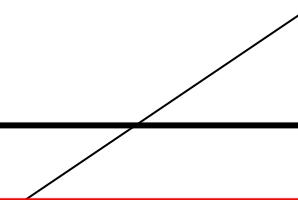


Камера



a 2

# Стерео

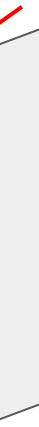
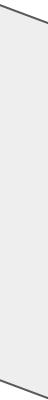


Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

$$ax + by + c = 0$$

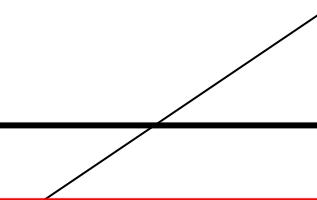
$$t - \frac{1}{t_z} \times \left[ t + R \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \right] \frac{1}{z''_1} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$



Камера

a 2

# Стерео



Как определить линию на плоскости?:

$$\cancel{y = mx + b}$$

$$ax + by + c = 0$$

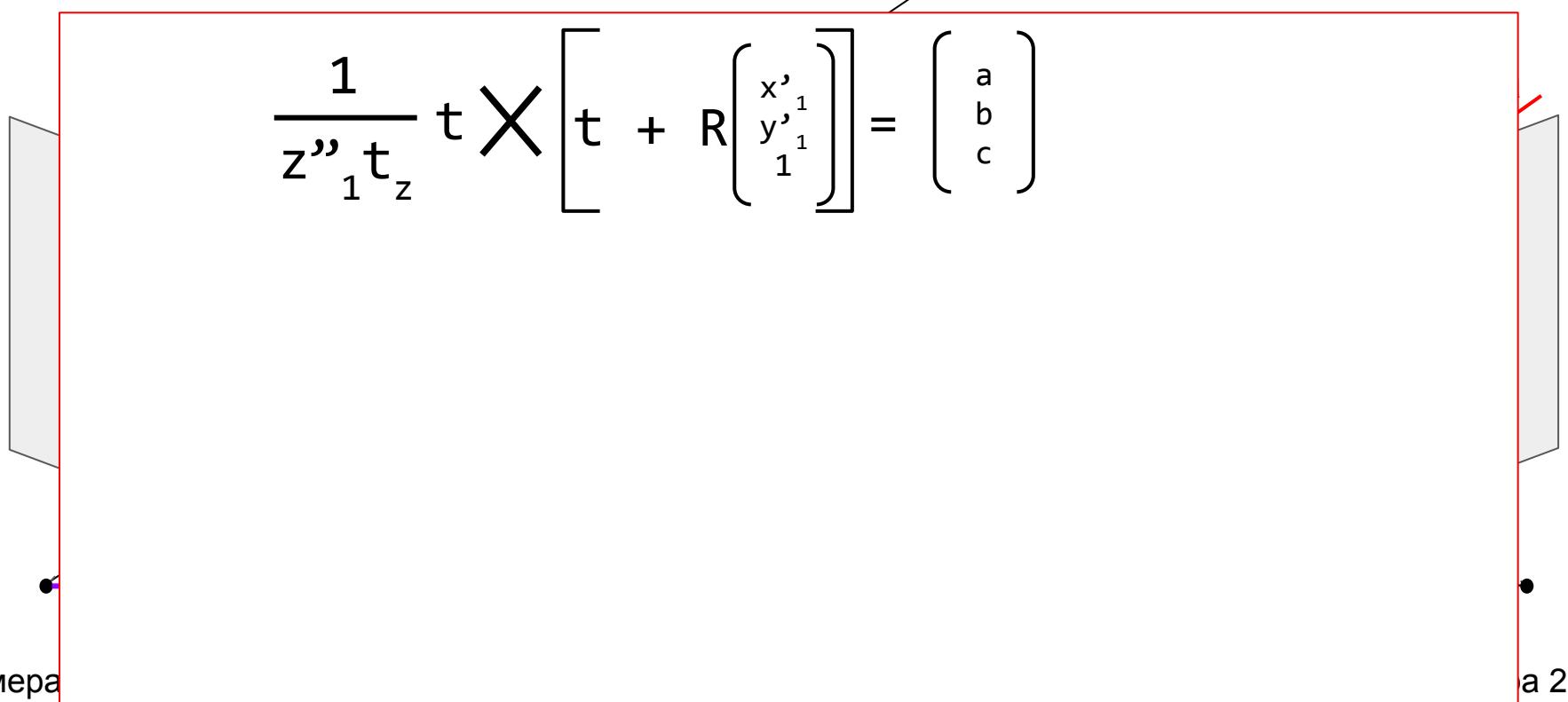
$$t - \frac{1}{t_z} \times \left[ t + R \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \right] - \frac{1}{z''_1} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$\frac{1}{z''_1 t_z} t \times \left[ t + R \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \right] = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

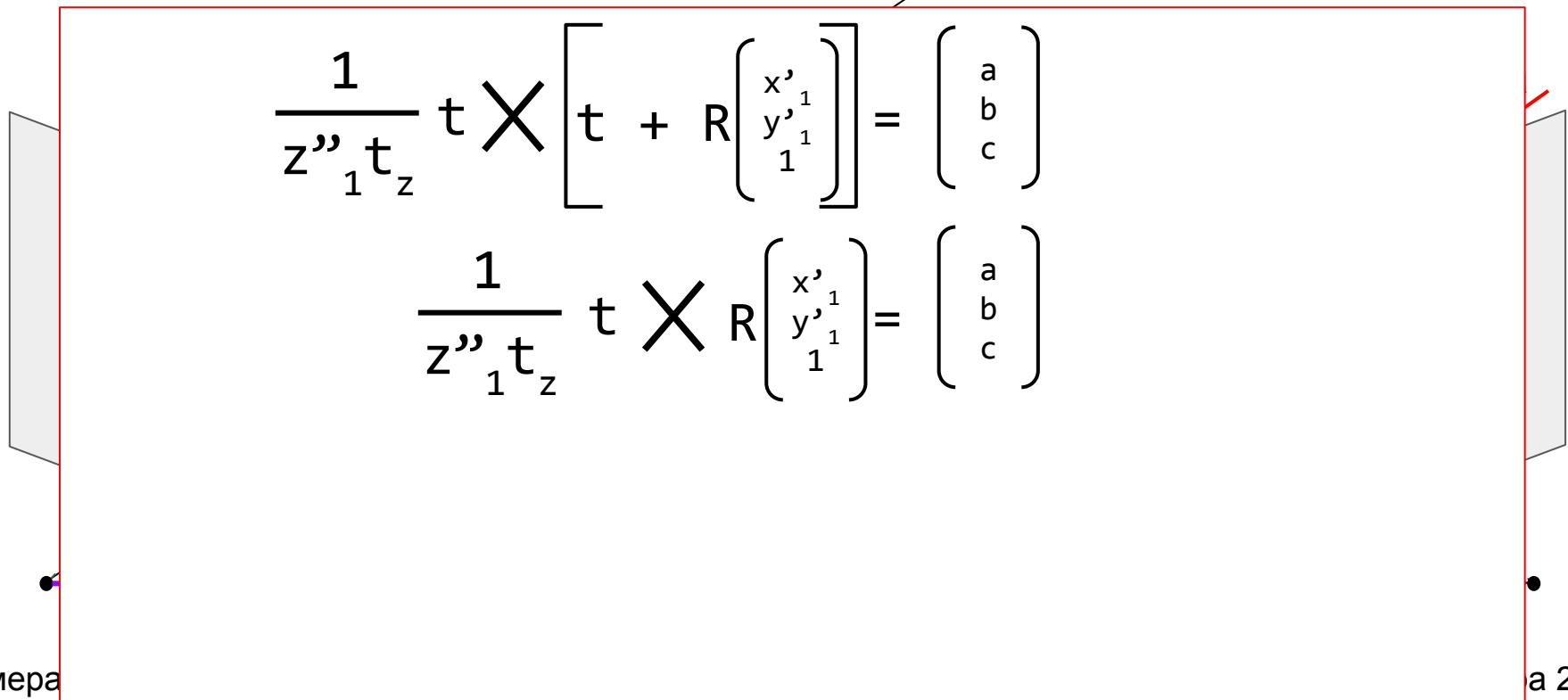
Камера

a 2

# Стерео



# Стерео



# Стерео

$$\frac{1}{z''_1 t_z} t \times \left[ t + R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \right] = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$\frac{1}{z''_1 t_z} t \times R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$ax + by + c = 0$$

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

Камера

a 2

# Стерео

$$\frac{1}{z''_1 t_z} t \times \left[ t + R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \right] = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$\frac{1}{z''_1 t_z} t \times R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$sax + sby + sc = 0$$

$$s \begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

Камера

a 2

# Стерео

$$\frac{1}{z''_1 t_z} t \times \left[ t + R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \right] = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

~~$$\frac{1}{z''_1 t_z} t \times R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$~~

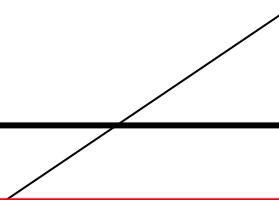
$$sax + sby + sc = 0$$

$$s \begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

Камера

a 2

# Стерео



$$\frac{1}{z''_1 t_z} t \times \left[ t + R \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix} \right] = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$\frac{1}{z''_1 t_z} t \times R \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$t \times R \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$



Камера



a 2

# Стерео

$$\frac{1}{z''_1 t_z} t \times \left[ t + R \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix} \right] = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$\frac{1}{z''_1 t_z} t \times R \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

$$t \times R \begin{pmatrix} x'_1 \\ y'_1 \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

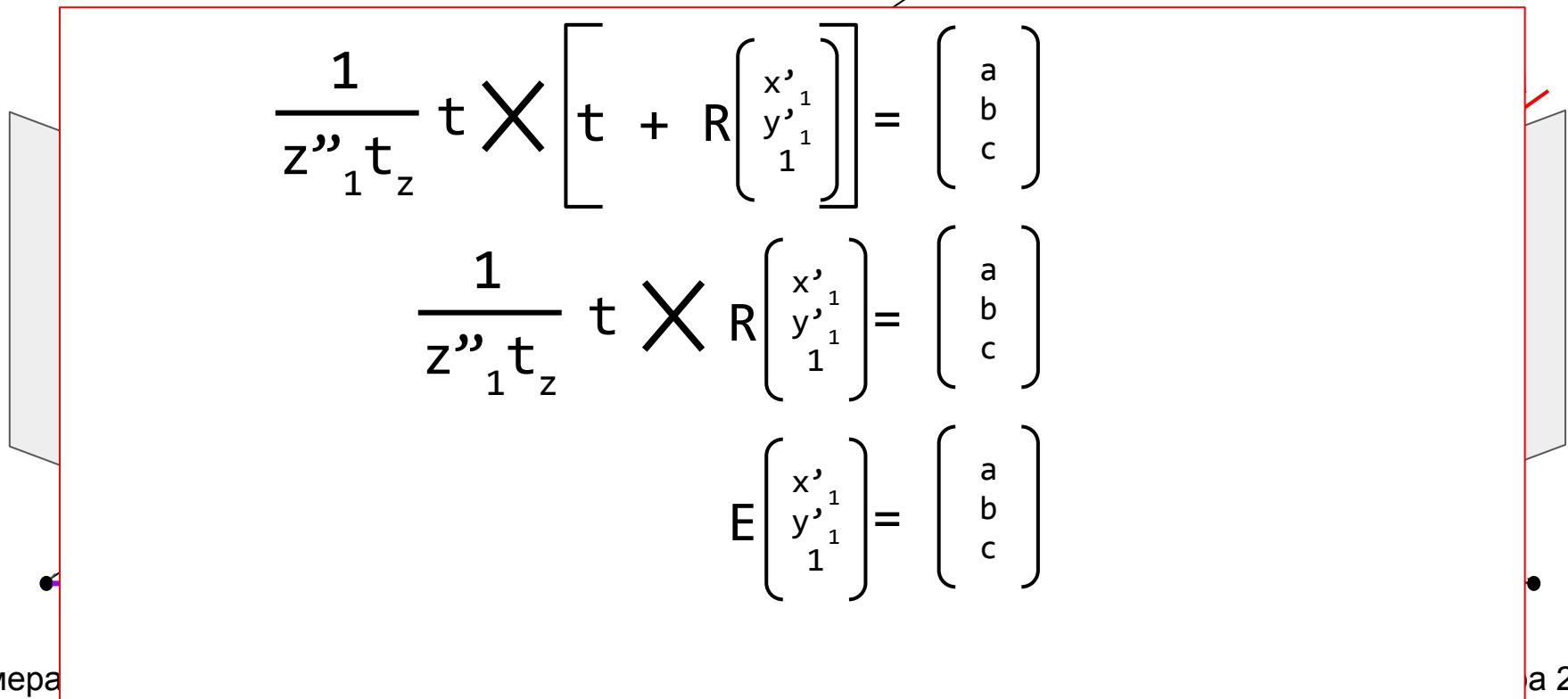
E →

essential matrix  
(Longuet-Higgins, 1981)

Камера

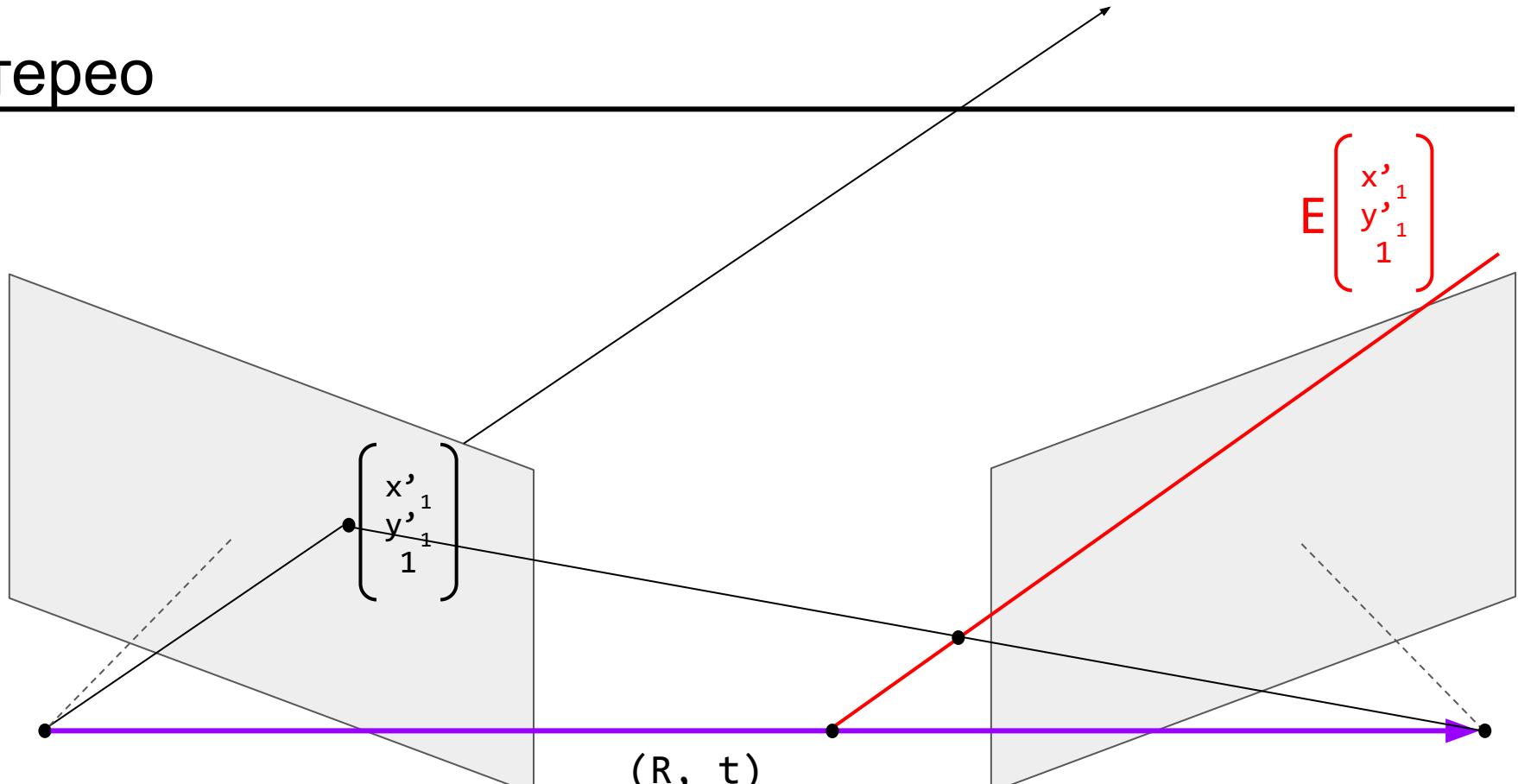
a 2

# Стерео



# Стерео

---



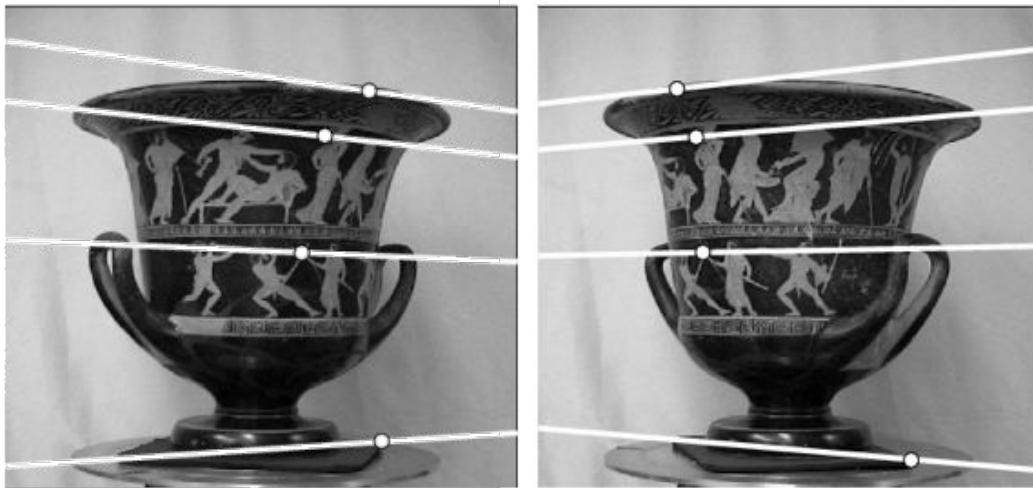
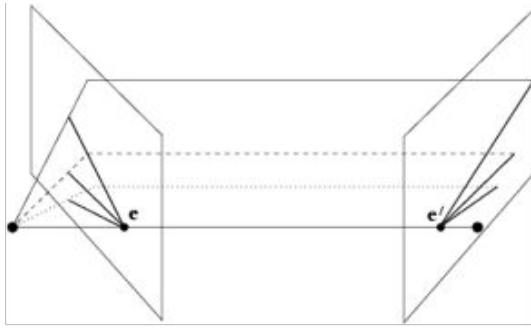
Камера 1

$(R, t)$

Камера 2

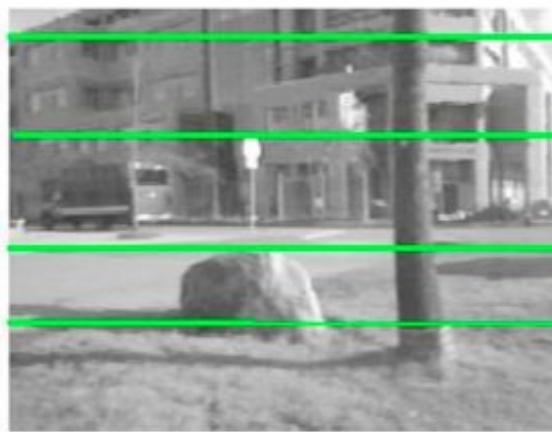
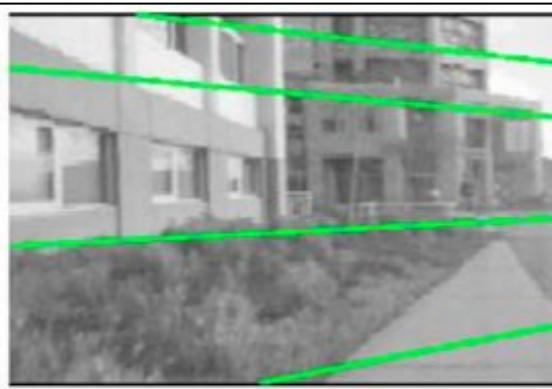
# Стерео

---

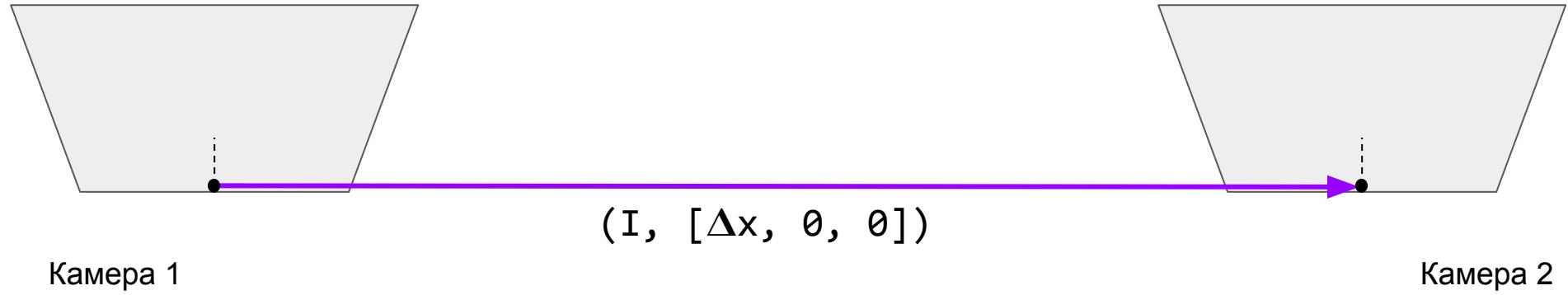


# Стерео

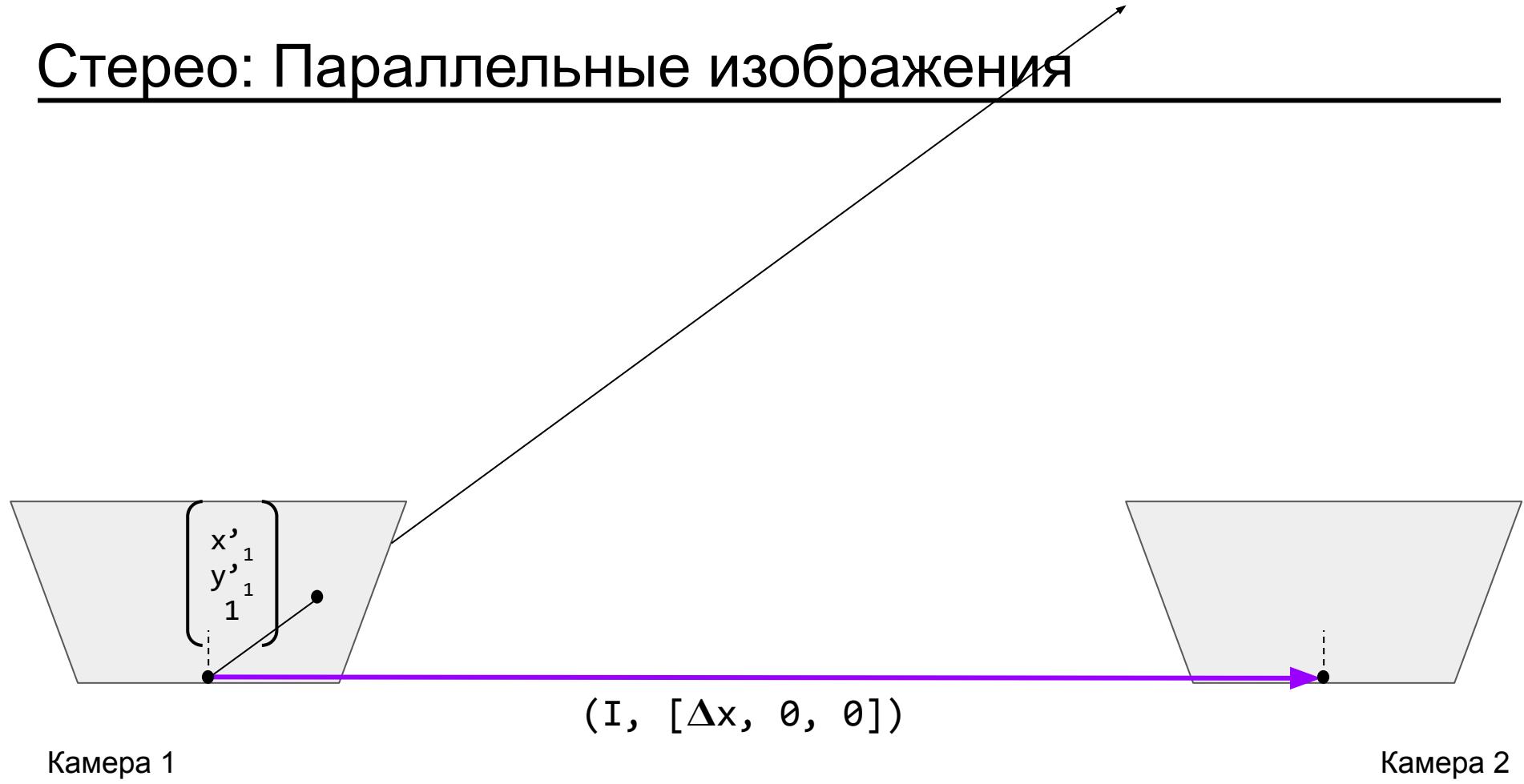
---



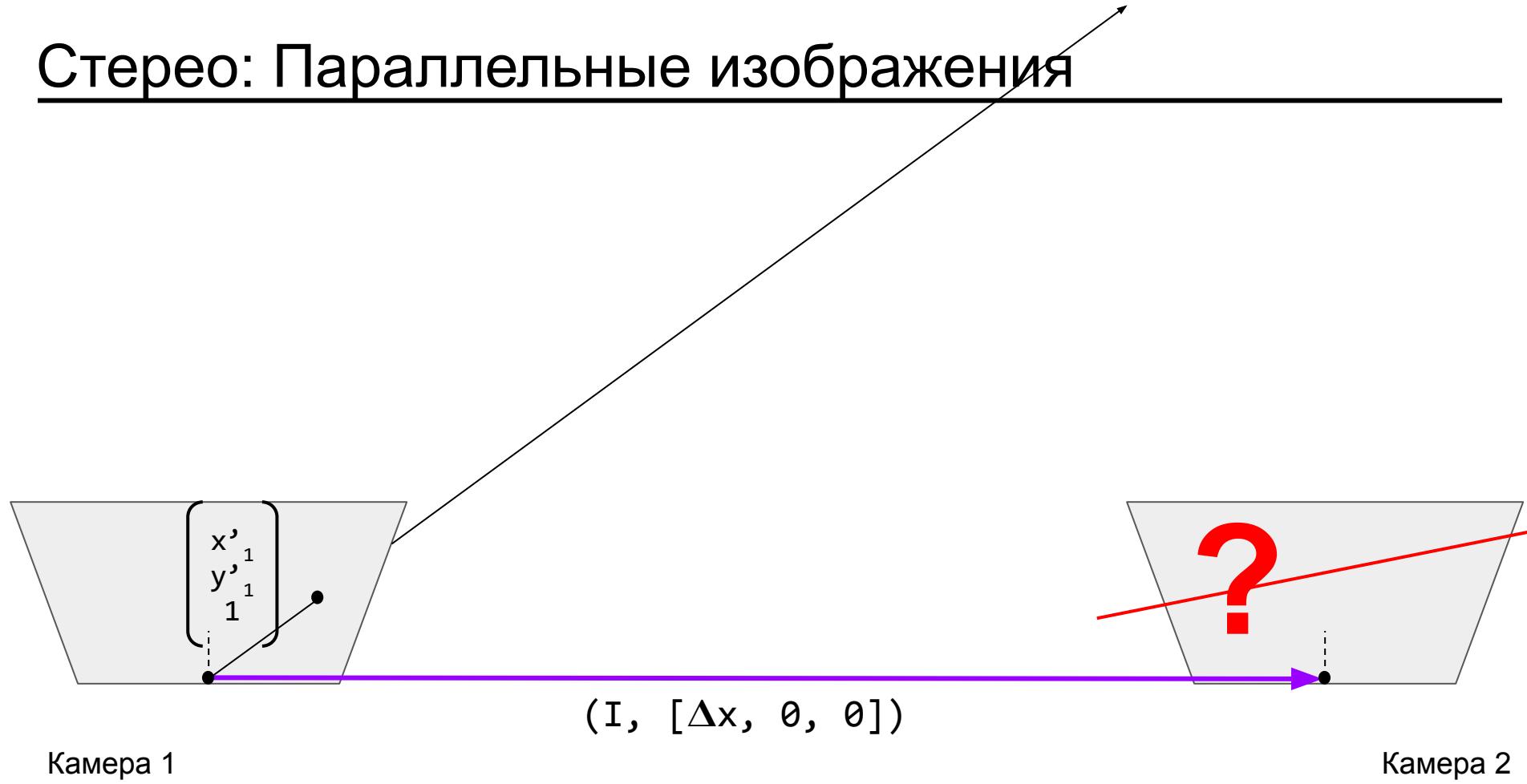
# Стерео: Параллельные изображения



# Стерео: Параллельные изображения

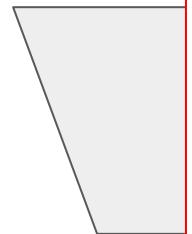


# Стерео: Параллельные изображения



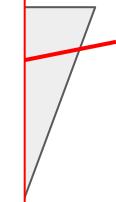
# Стерео: Параллельные изображения

$$t \times R \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$



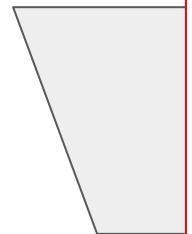
Камера

a 2



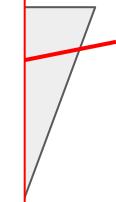
# Стерео: Параллельные изображения

$$\begin{bmatrix} \Delta x \\ 0 \\ 0 \end{bmatrix} \times I \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

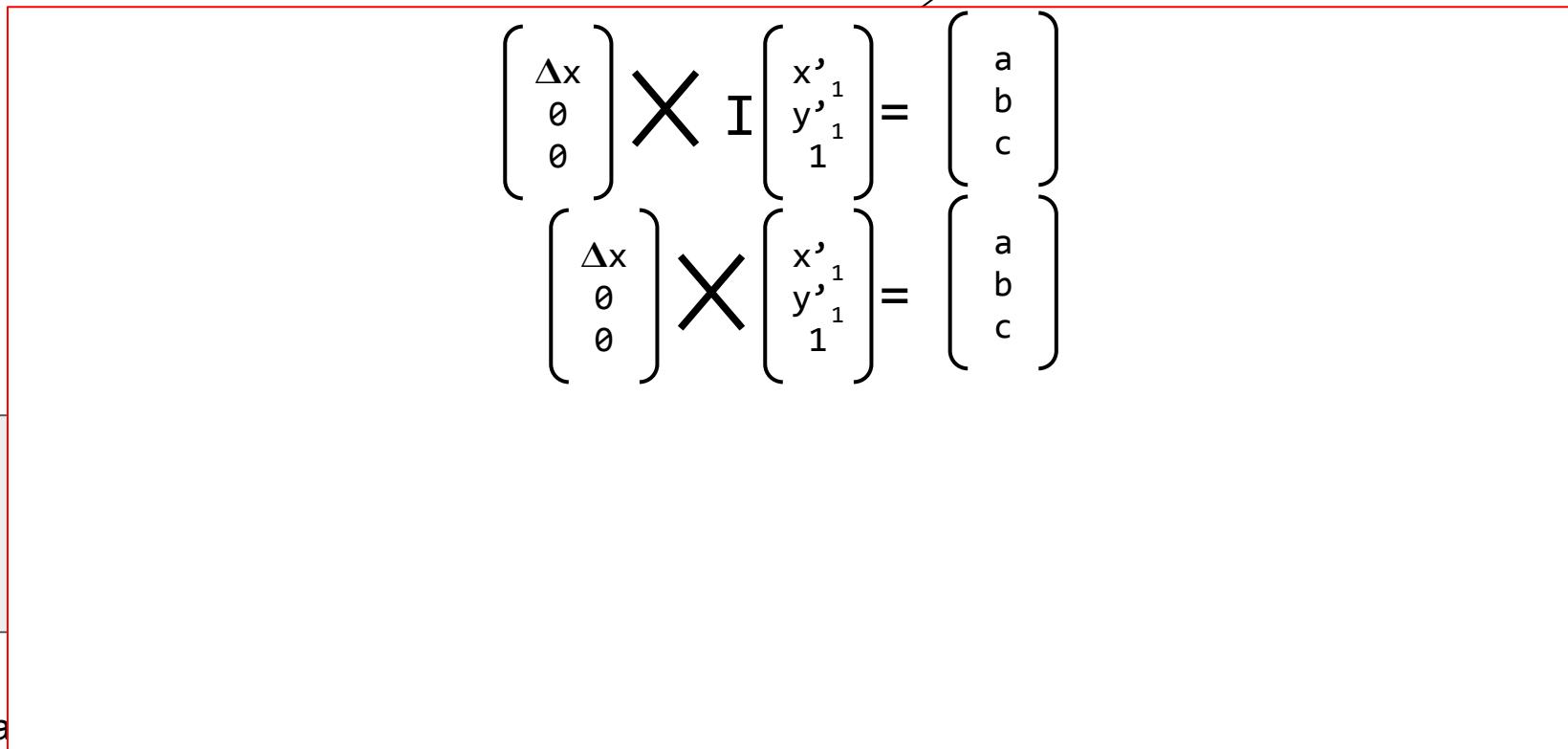


Камера

a 2



# Стерео: Параллельные изображения



The diagram illustrates two cameras, labeled "Камера" (Camera) and "a 2", positioned at the bottom left and right respectively. They are represented by gray trapezoids. A red rectangular frame encloses the two cameras and the mathematical equations. An arrow points from the top right towards the frame.

$$\begin{pmatrix} \Delta x \\ 0 \\ 0 \end{pmatrix} \times I \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$
$$\begin{pmatrix} \Delta x \\ 0 \\ 0 \end{pmatrix} \times \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

Камера

a 2

# Стерео: Параллельные изображения

$$\begin{pmatrix} \Delta x \\ 0 \\ 0 \end{pmatrix} \times I \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$
$$\begin{pmatrix} \Delta x \\ 0 \\ 0 \end{pmatrix} \times \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$
$$\begin{pmatrix} 0 \\ -\Delta x \\ \Delta xy' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

Камера

a 2

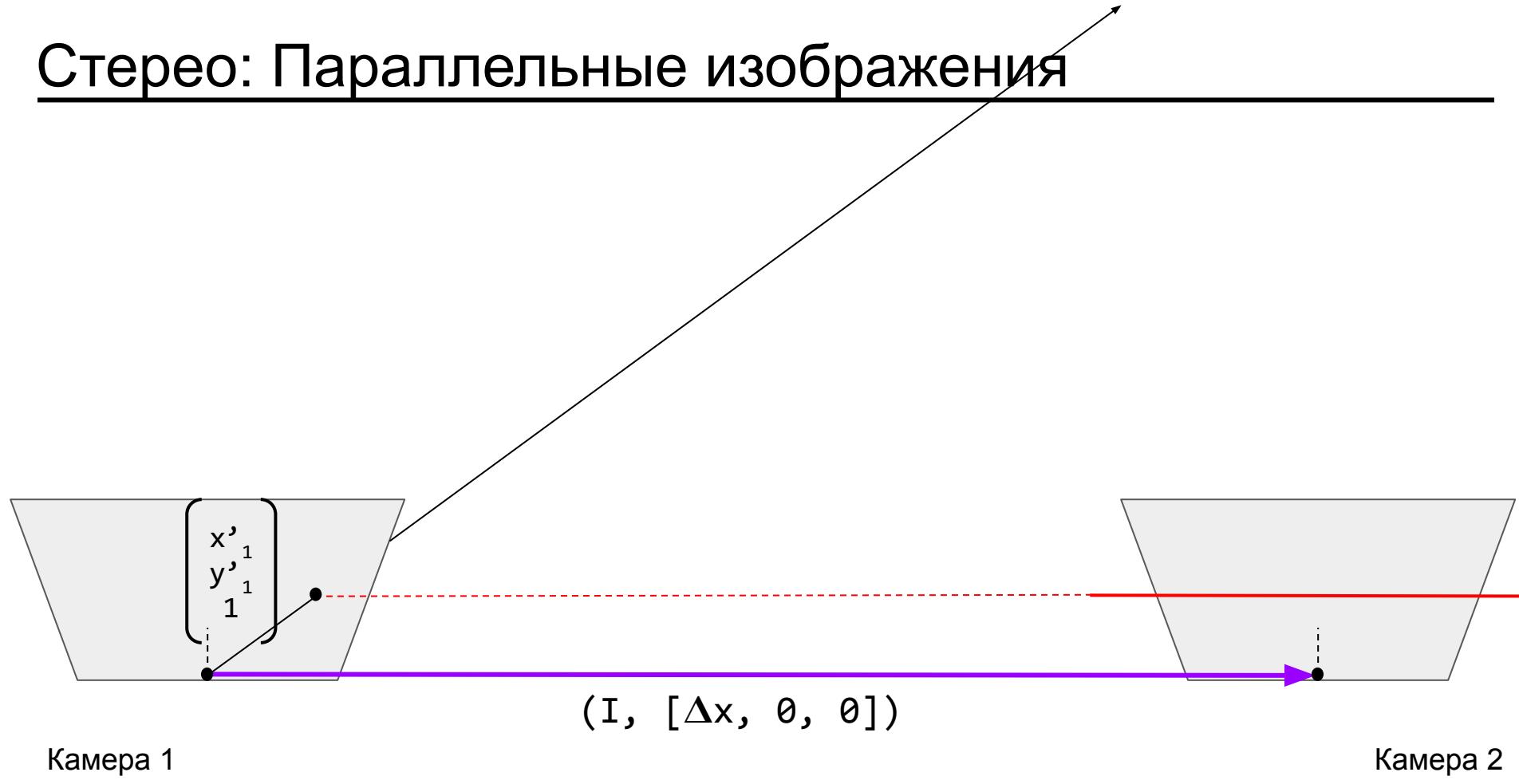
# Стерео: Параллельные изображения

$$\begin{pmatrix} \Delta x \\ 0 \\ 0 \end{pmatrix} \times I \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$
$$\begin{pmatrix} \Delta x \\ 0 \\ 0 \end{pmatrix} \times \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$
$$\begin{pmatrix} 0 \\ -\Delta x \\ \Delta xy' \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$
$$(0)x + (-\Delta x)y + (\Delta xy')_1 = 0 \longrightarrow y = y'_1$$

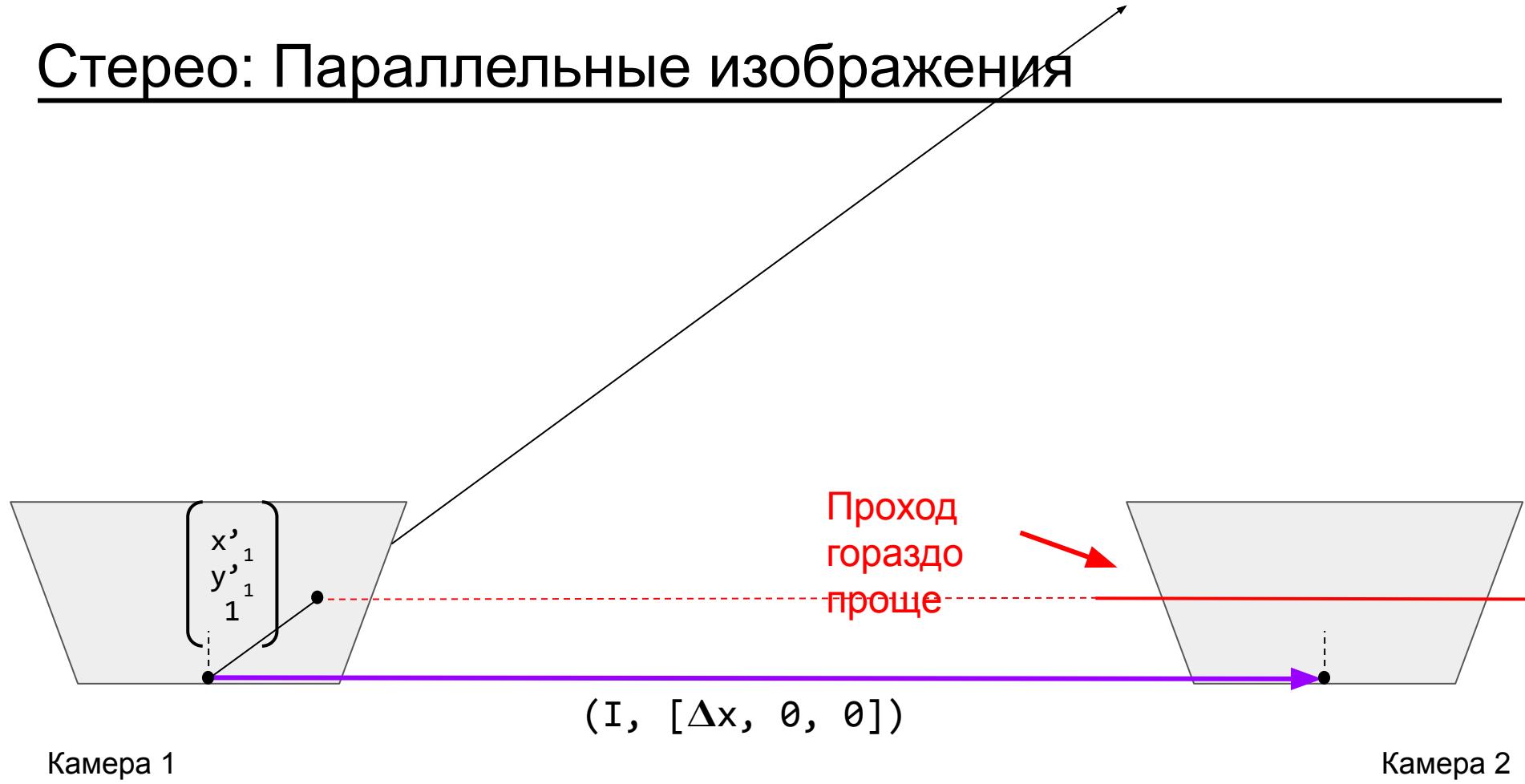
Камера

a 2

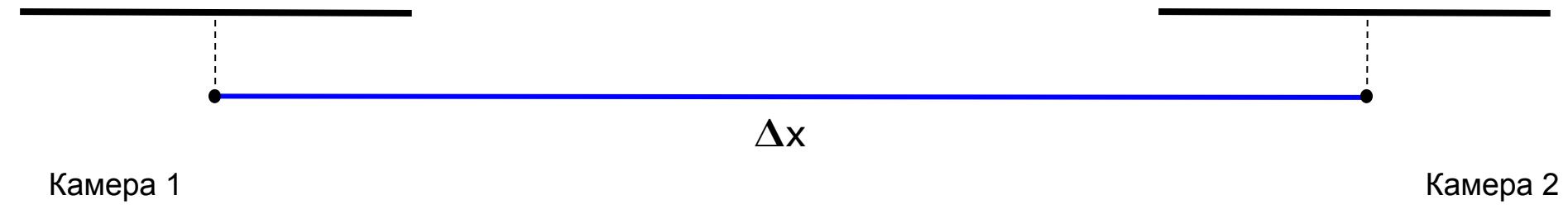
# Стерео: Параллельные изображения



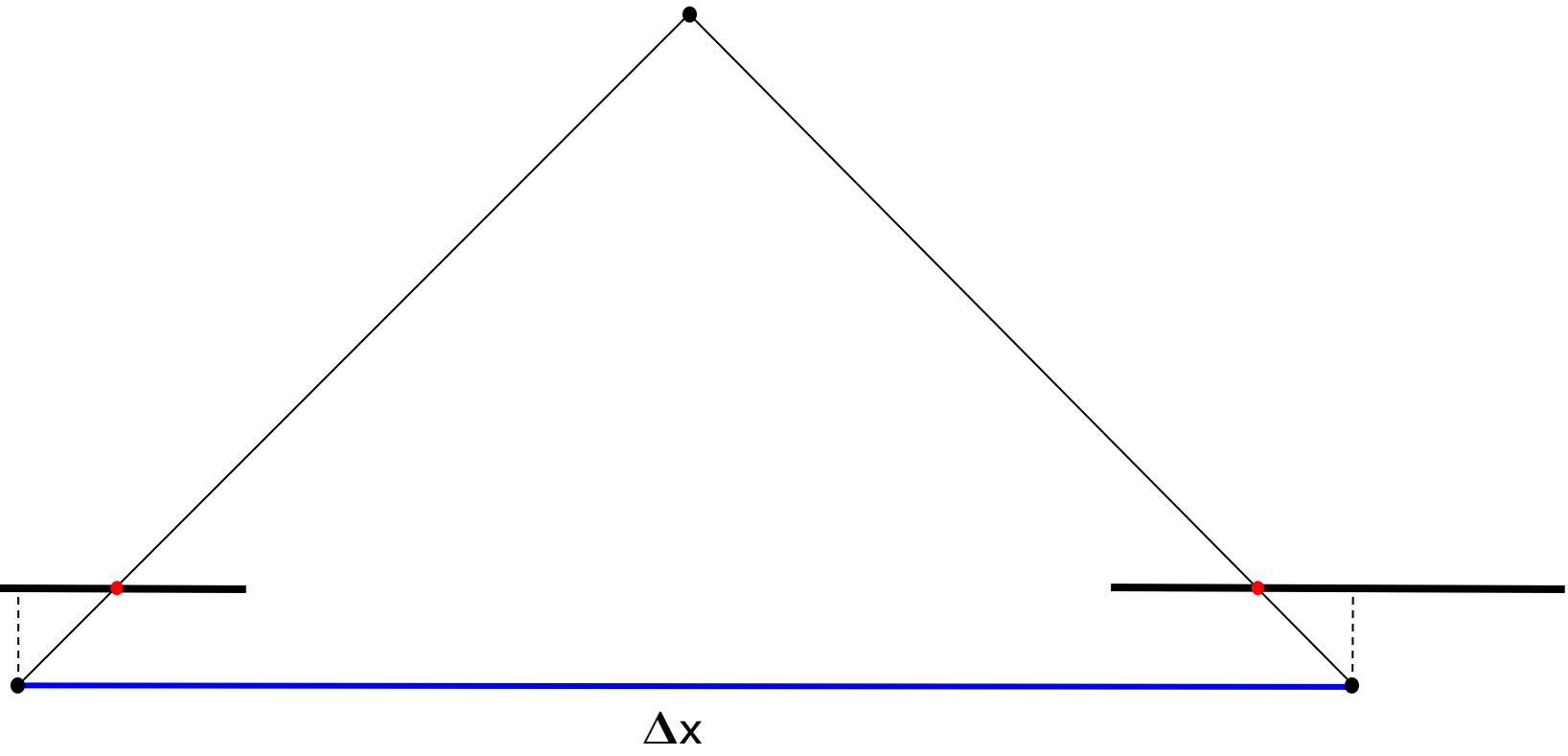
# Стерео: Параллельные изображения



# Стерео: Параллельные изображения



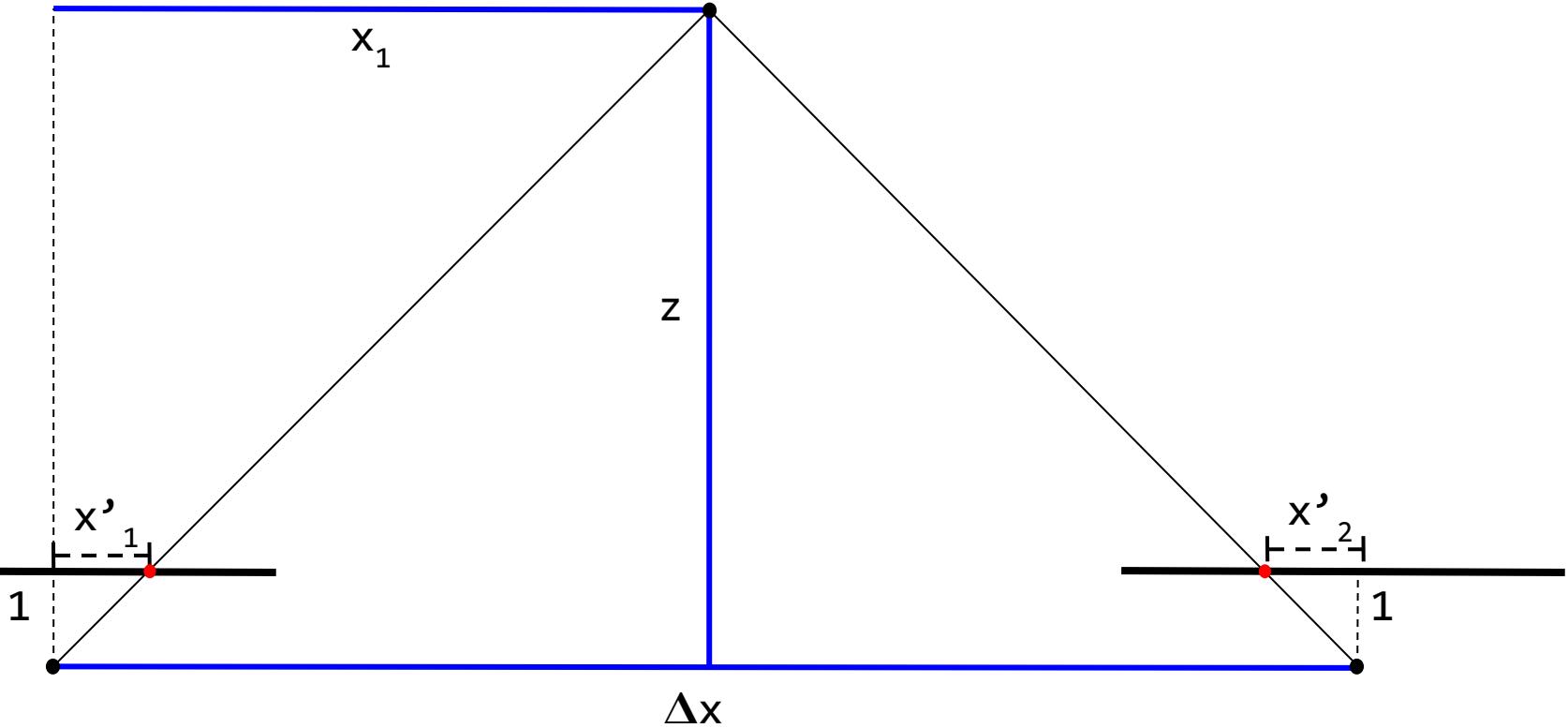
# Стерео: Параллельные изображения



Камера 1

Камера 2

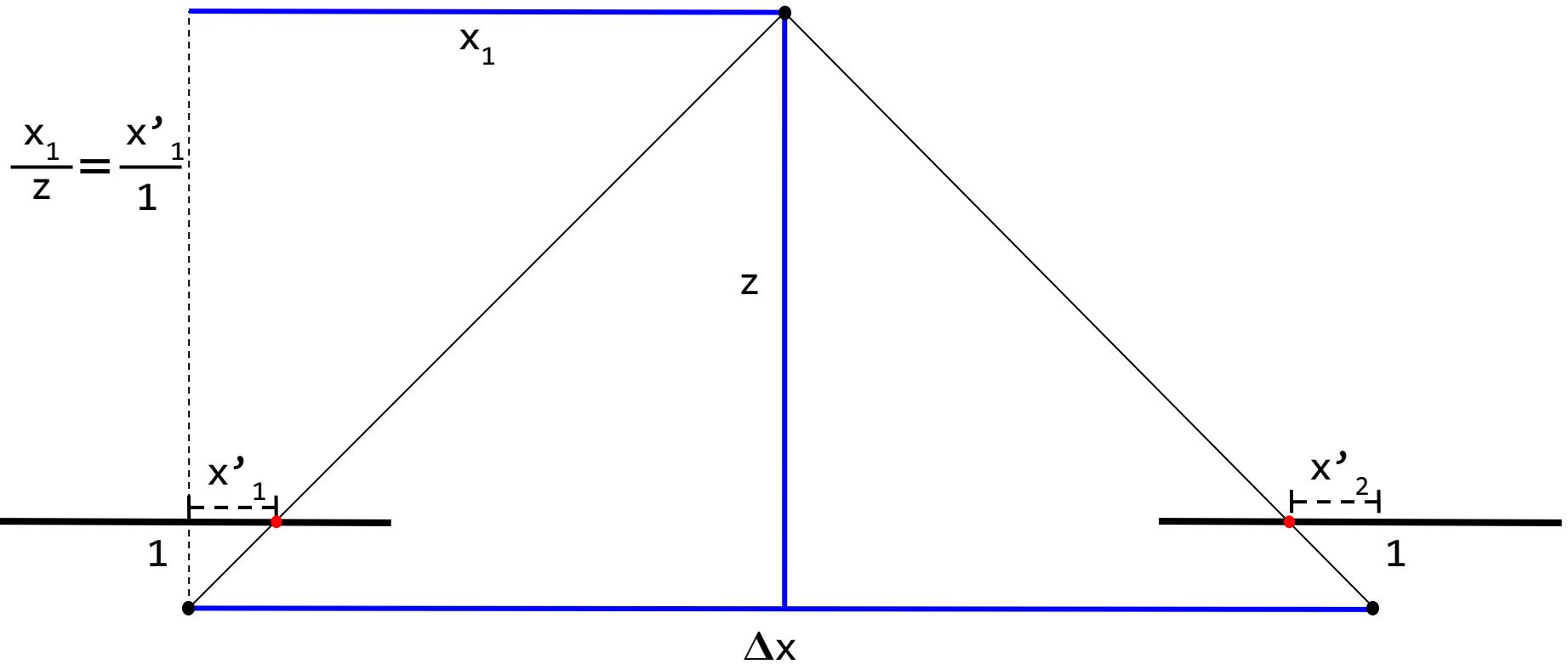
# Стерео: Параллельные изображения



Камера 1

Камера 2

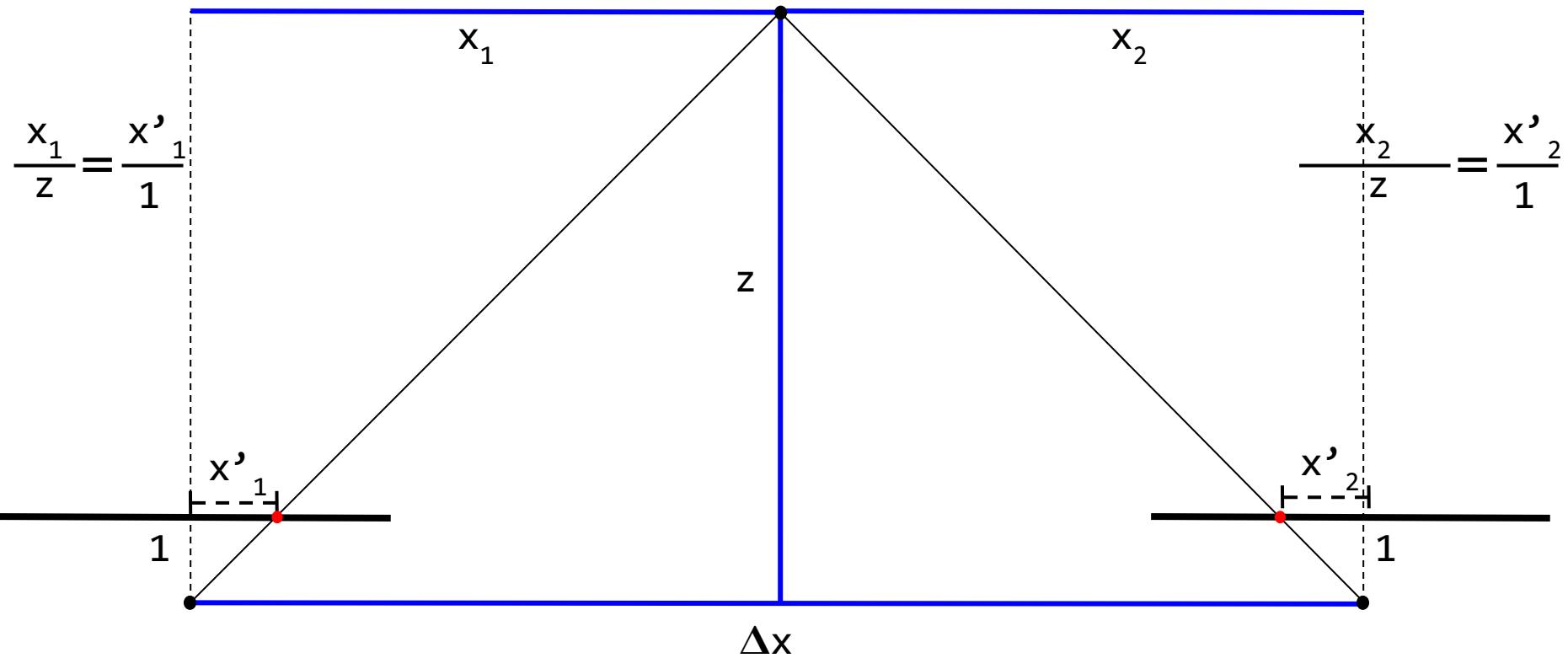
# Стерео: Параллельные изображения



Камера 1

Камера 2

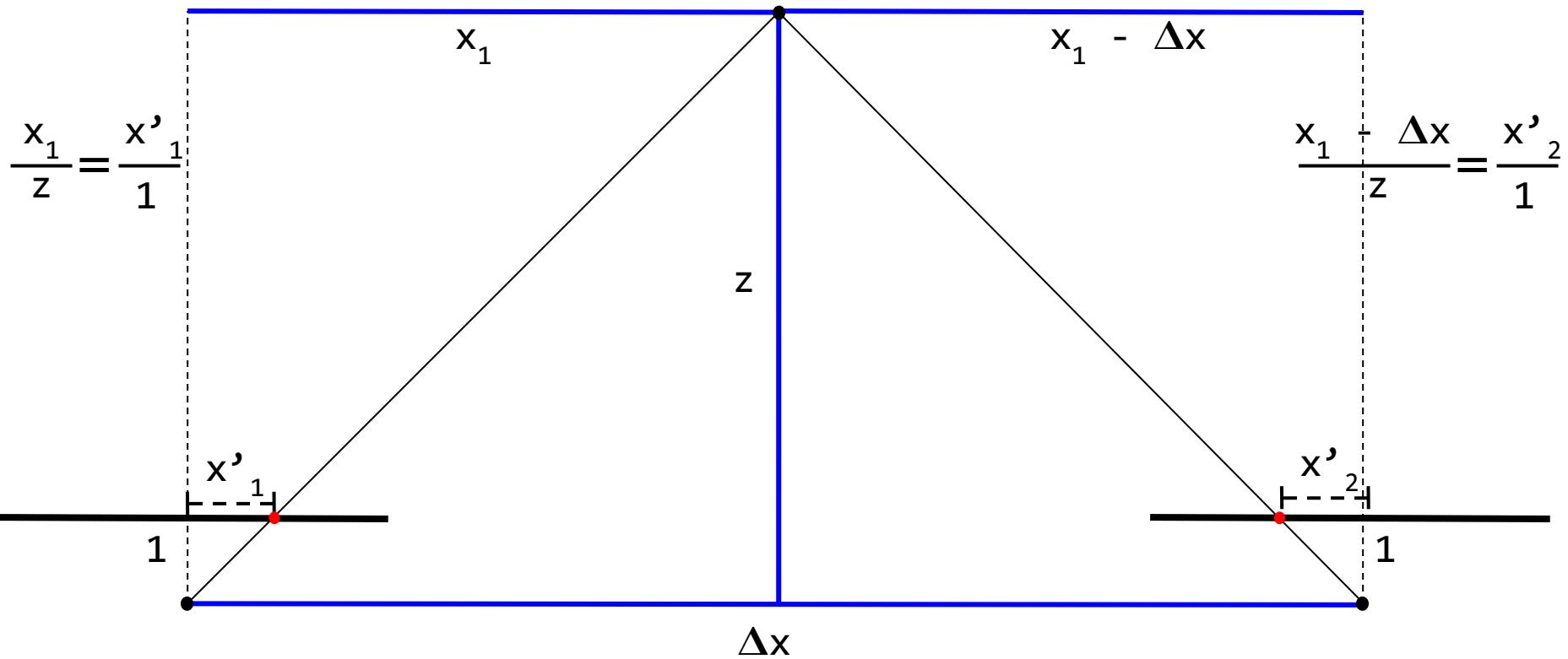
# Стерео: Параллельные изображения



Камера 1

Камера 2

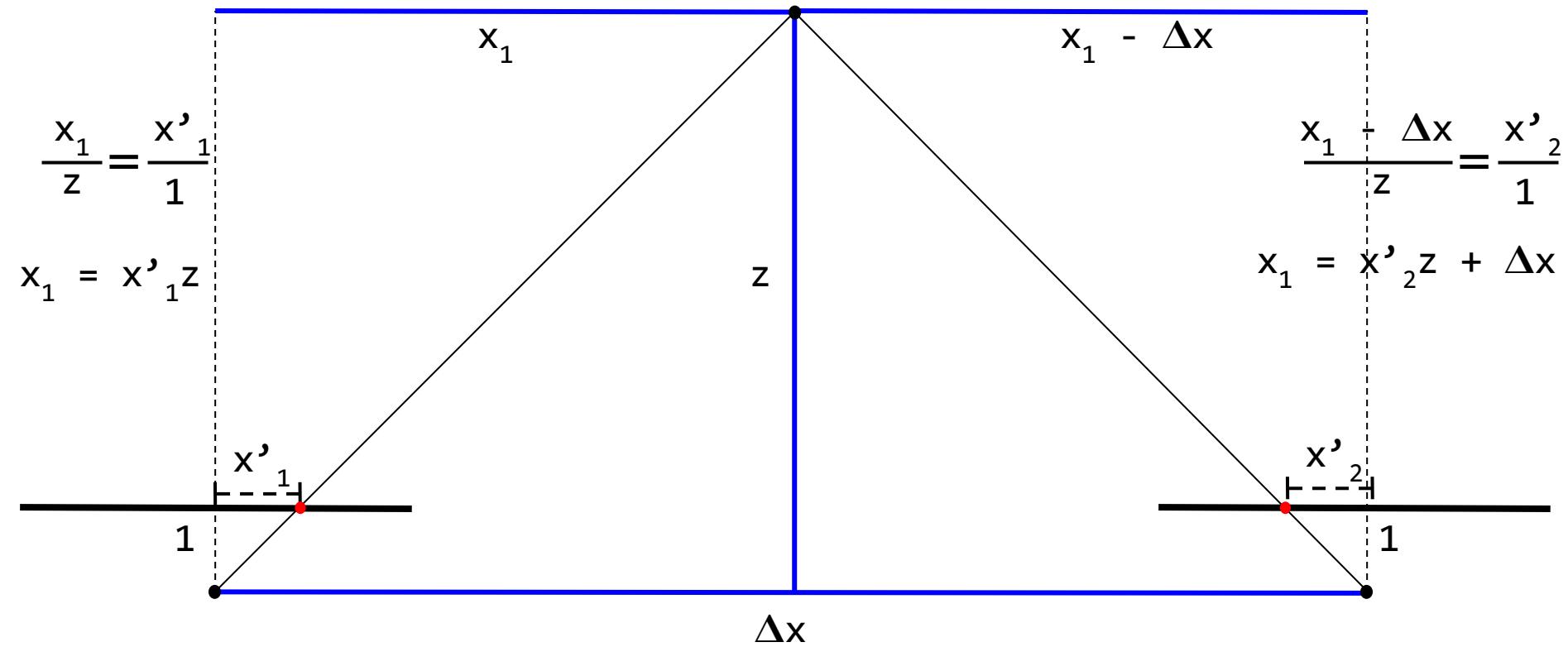
# Стерео: Параллельные изображения



Камера 1

Камера 2

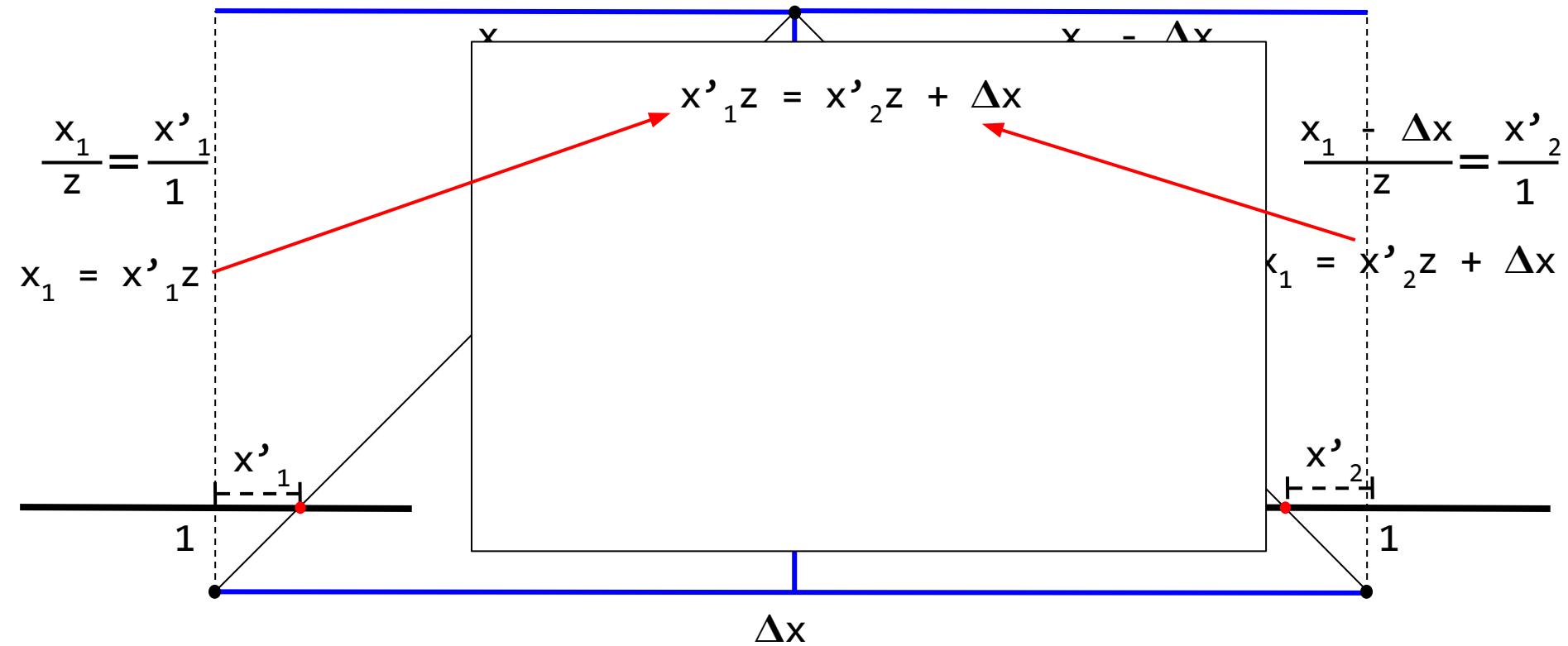
# Стерео: Параллельные изображения



Камера 1

Камера 2

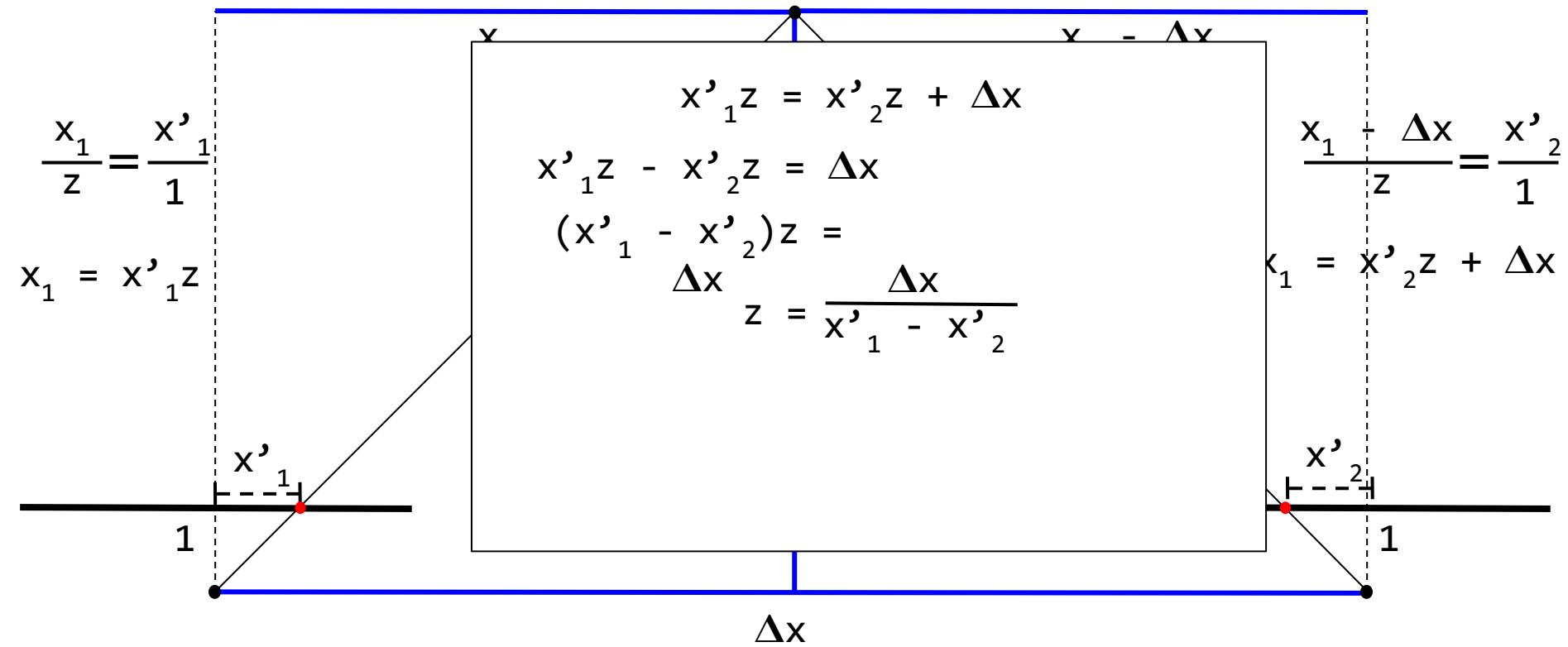
# Стерео: Параллельные изображения



Камера 1

Камера 2

# Стерео: Параллельные изображения



Камера 1

Камера 2

# Стерео: Параллельные изображения

Алгоритм (одинаковые камеры)

# Стерео: Параллельные изображения

Алгоритм (одинаковые камеры)

compute\_depth( $I_1, I_2, \Delta x, f, p$ ):

# Стерео: Параллельные изображения

Алгоритм (одинаковые камеры)

```
compute_depth( $I_1$ ,  $I_2$ ,  $\Delta x$ ,  $f$ ,  $p$ ):  
    depth <- new_image_like( $I_1$ )
```

# Стерео: Параллельные изображения

---

Алгоритм (одинаковые камеры)

```
compute_depth( $I_1$ ,  $I_2$ ,  $\Delta x$ ,  $f$ ,  $p$ ):  
    depth <- new_image_like( $I_1$ )  
    For every pixel  $i, j$  in  $I_1$ :  
         $f_{ij}$  = get_feature_descriptor( $I_1$ ,  $i$ ,  $j$ )
```

# Стерео: Параллельные изображения

---

## Алгоритм (одинаковые камеры)

```
compute_depth( $I_1$ ,  $I_2$ ,  $\Delta x$ ,  $f$ ,  $p$ ):  
    depth <- new_image_like( $I_1$ )  
    For every pixel  $i, j$  in  $I_1$ :  
         $f_{ij}$  = get_feature_descriptor( $I_1$ ,  $i$ ,  $j$ )  
        best_score <- INFINITY  
        best_i <- NaN  
        For  $i'$  in columns( $I_2$ ):
```

# Стерео: Параллельные изображения

---

## Алгоритм (одинаковые камеры)

```
compute_depth( $I_1$ ,  $I_2$ ,  $\Delta x$ ,  $f$ ,  $p$ ):  
    depth <- new_image_like( $I_1$ )  
    For every pixel  $i, j$  in  $I_1$ :  
         $f_{ij}$  = get_feature_descriptor( $I_1$ ,  $i$ ,  $j$ )  
        best_score <- INFINITY  
        best_i <- NaN  
        For  $i'$  in columns( $I_2$ ):  
             $f_{i'j}$  = get_feature_descriptor( $I_2$ ,  $i'$ ,  $j$ )  
            score = score_feature_match( $f_{ij}$ ,  $f_{i'j}$ )  
            If score < best_score:  
                best_score = score  
                best_i =  $i'$ 
```

# Стерео: Параллельные изображения

---

## Алгоритм (одинаковые камеры)

```
compute_depth( $I_1$ ,  $I_2$ ,  $\Delta x$ ,  $f$ ,  $p$ ):  
    depth <- new_image_like( $I_1$ )  
    For every pixel  $i, j$  in  $I_1$ :  
         $f_{ij}$  = get_feature_descriptor( $I_1$ ,  $i$ ,  $j$ )  
        best_score <- INFINITY  
        best_i <- NaN  
        For  $i'$  in columns( $I_2$ ):  
             $f_{i'j}$  = get_feature_descriptor( $I_2$ ,  $i'$ ,  $j$ )  
            score = score_feature_match( $f_{ij}$ ,  $f_{i'j}$ )  
            If score < best_score:  
                best_score = score  
                best_i =  $i'$   
        depth[i, j] =  $f p \Delta x / (i - best_i)$   
    Return depth
```

# Стерео: Параллельные изображения

---

## Алгоритм (одинаковые камеры)

```
compute_depth( $I_1$ ,  $I_2$ ,  $\Delta x$ ,  $f$ ,  $p$ ):  
    depth <- new_image_like( $I_1$ )  
    For every pixel  $i, j$  in  $I_1$ :  
         $f_{ij}$  = get_feature_descriptor( $I_1$ ,  $i$ ,  $j$ )  
        best_score <- INFINITY  
        best_i <- NaN  
        For  $i'$  in columns( $I_2$ ):  
             $f_{i'j}$  = get_feature_descriptor( $I_2$ ,  $i'$ ,  $j$ )  
            score = score_feature_match( $f_{ij}$ ,  $f_{i'j}$ )  
            If score < best_score:  
                best_score = score  
                best_i =  $i'$   
        depth[i, j] =  $f \Delta x / (i - best_i)$   
    Return depth
```

С параллельными изображениями все просто!

# Стерео: Параллельные изображения

---

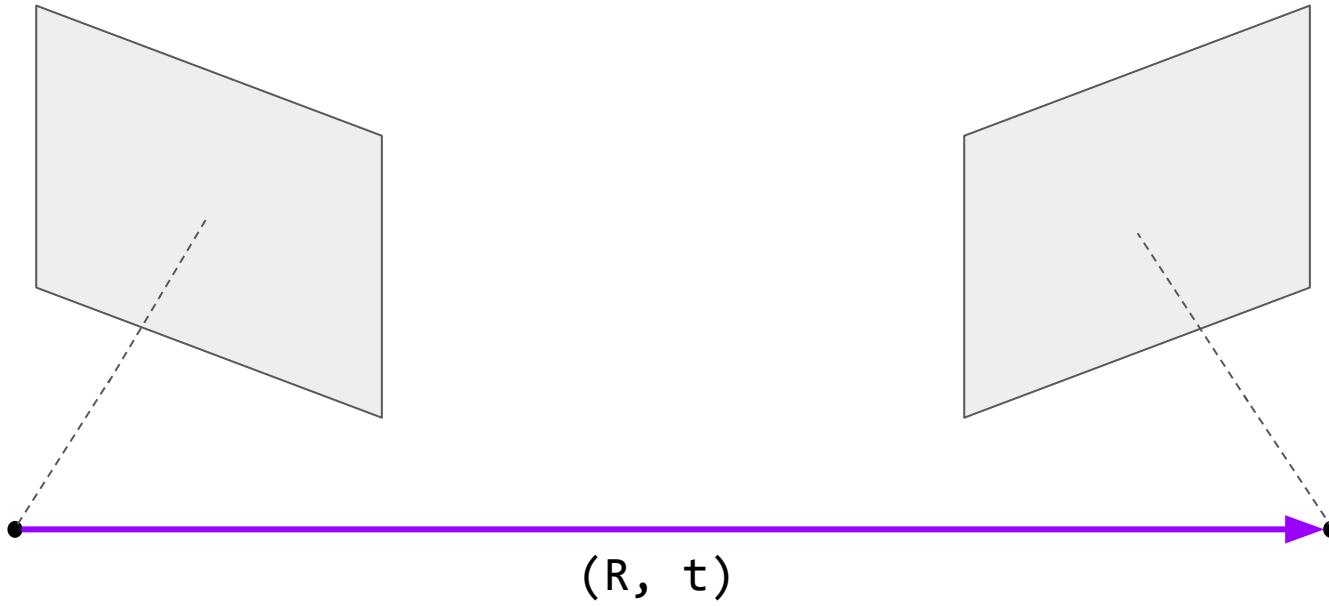
Алгоритм (одинаковые камеры)

```
compute_depth( $I_1$ ,  $I_2$ ,  $\Delta x$ ,  $f$ ,  $p$ ):  
    depth <- new_image_like( $I_1$ )  
    For every pixel  $i, j$  in  $I_1$ :  
         $f_{ij}$  = get_feature_descriptor( $I_1$ ,  $i$ ,  $j$ )  
        best_score <- INFINITY  
        best_i <- NaN  
        For  $i'$  in columns( $I_2$ ):  
             $f_{i'j}$  = get_feature_descriptor( $I_2$ ,  $i'$ ,  $j$ )  
            score = score_feature_match( $f_{ij}$ ,  $f_{i'j}$ )  
            If score < best_score:  
                best_score = score  
                best_i =  $i'$   
        depth[i, j] =  $f p \Delta x / (i - best_i)$   
    Return depth
```

ЧТО ЭТО ТАКОЕ?

# Стерео: вычисление гомографий

---



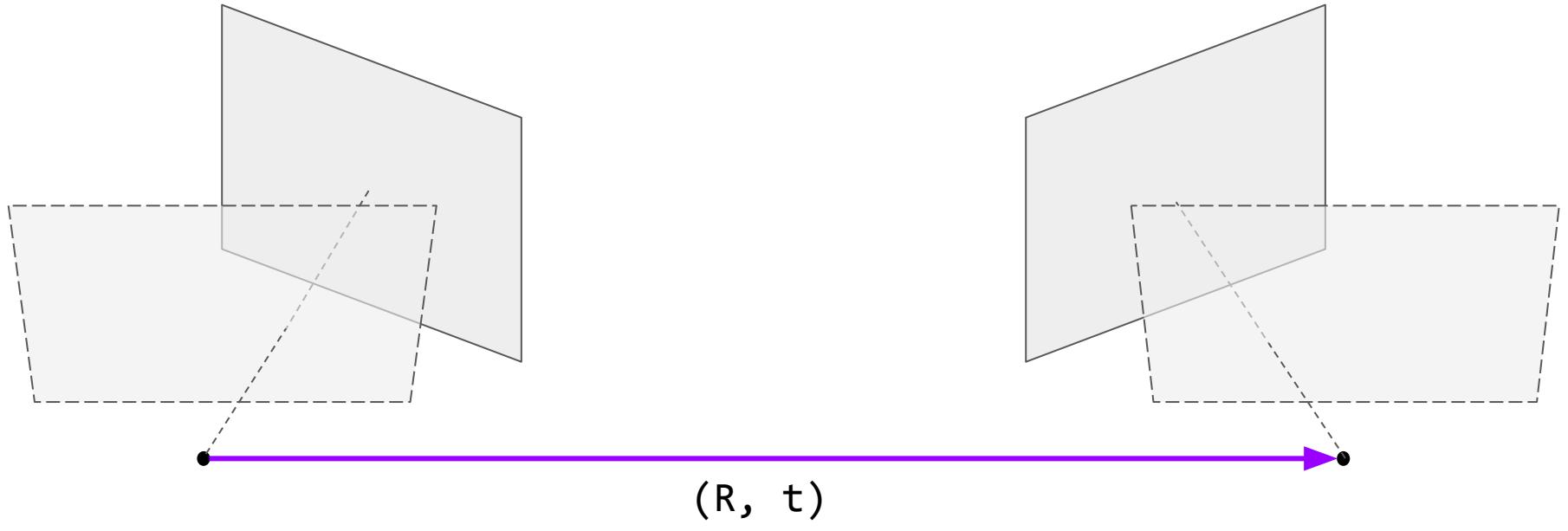
Камера 1

$(R, t)$

Камера 2

# Стерео: вычисление гомографий

---



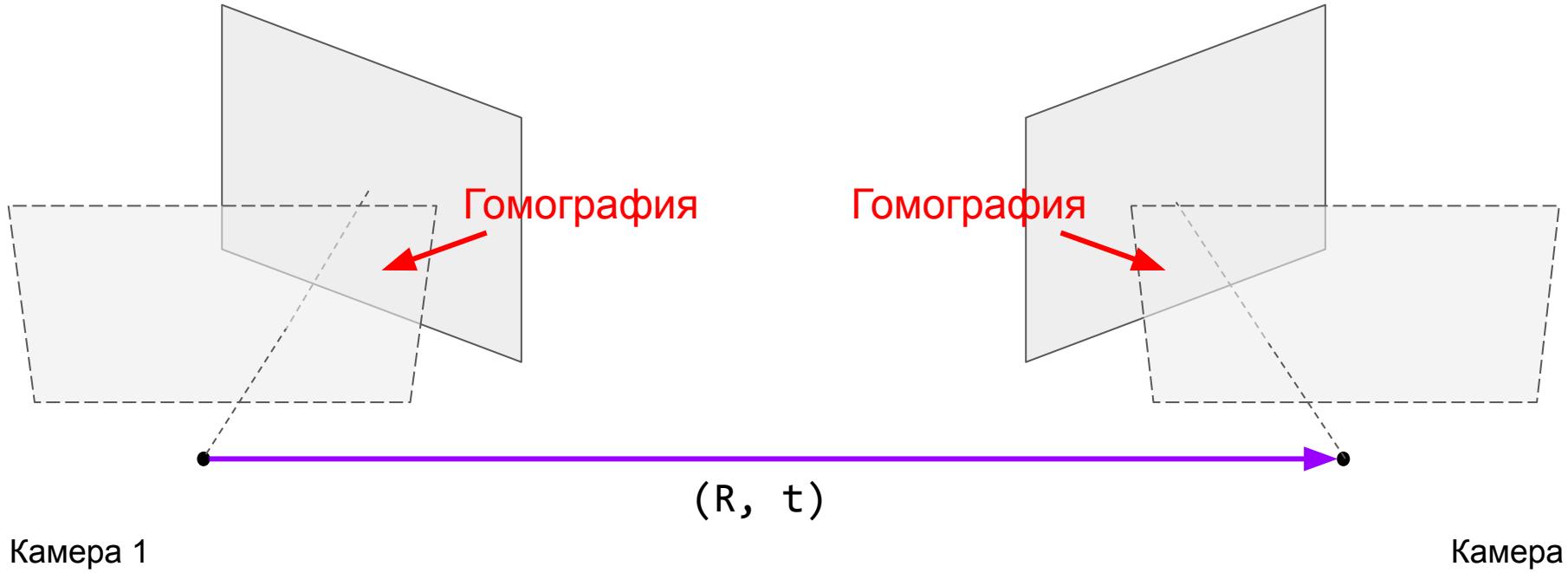
Камера 1

$(R, t)$

Камера 2

# Стерео: вычисление гомографий

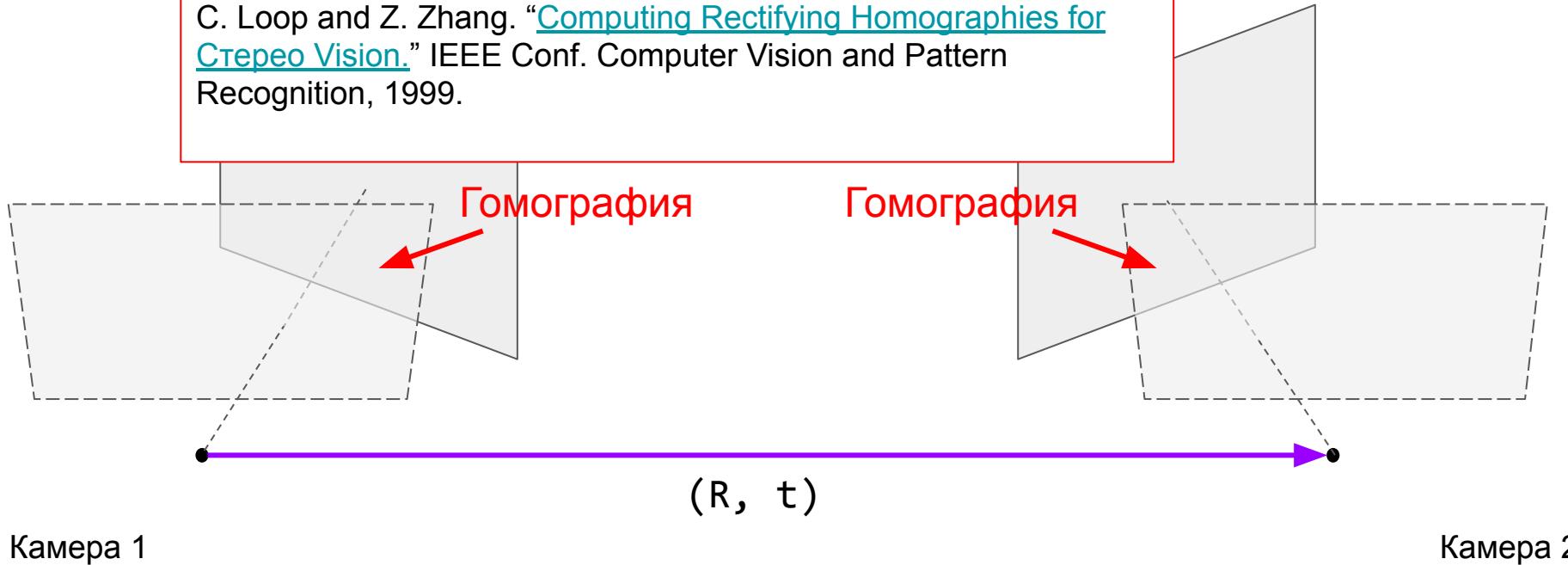
---



# Стерео: вычисление гомографий

## Stereo image rectification

C. Loop and Z. Zhang. “[Computing Rectifying Homographies for Stereo Vision.](#)” IEEE Conf. Computer Vision and Pattern Recognition, 1999.



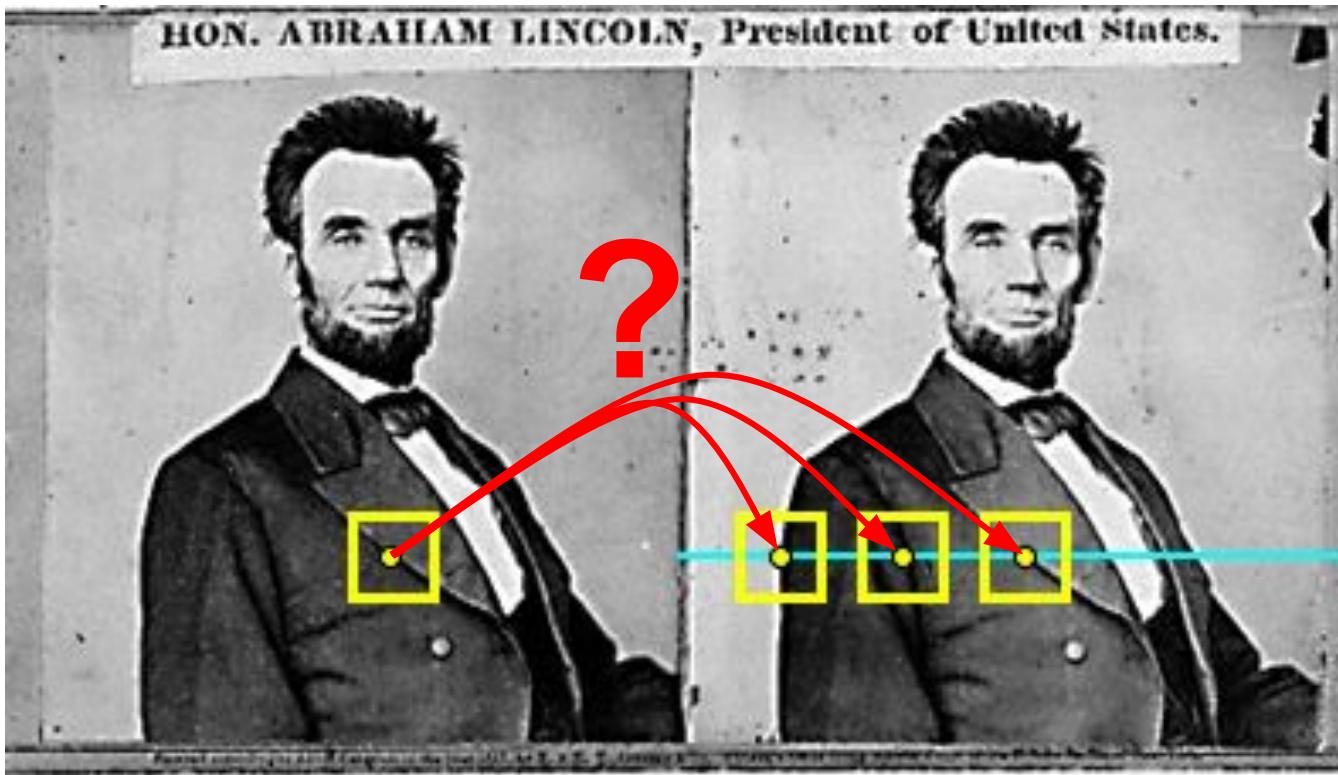
Камера 1

$(R, t)$

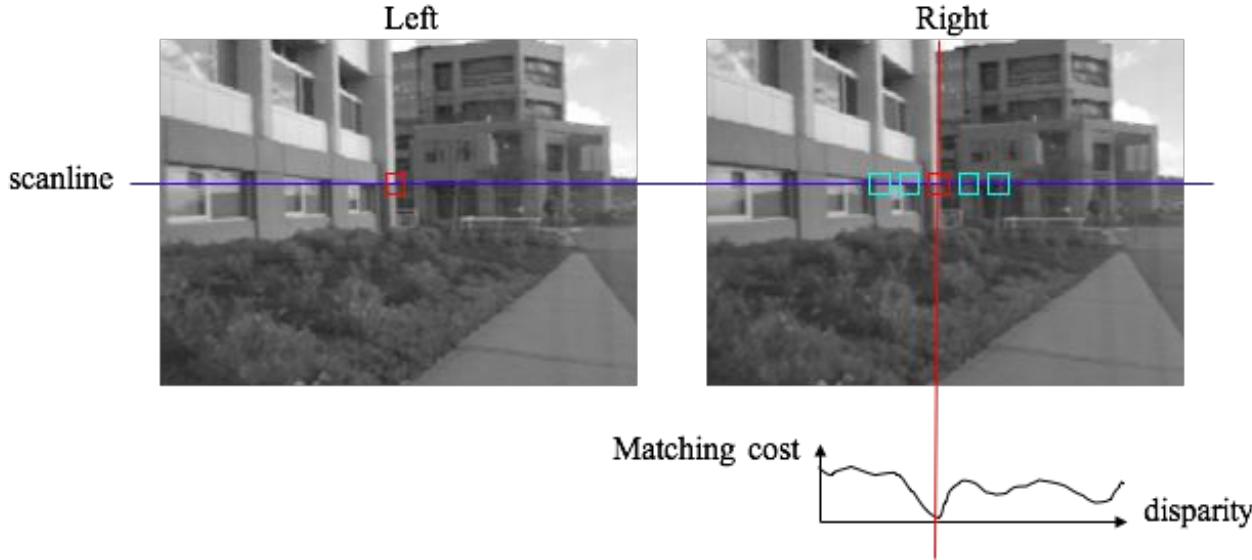
Камера 2

# Матчинг признаков

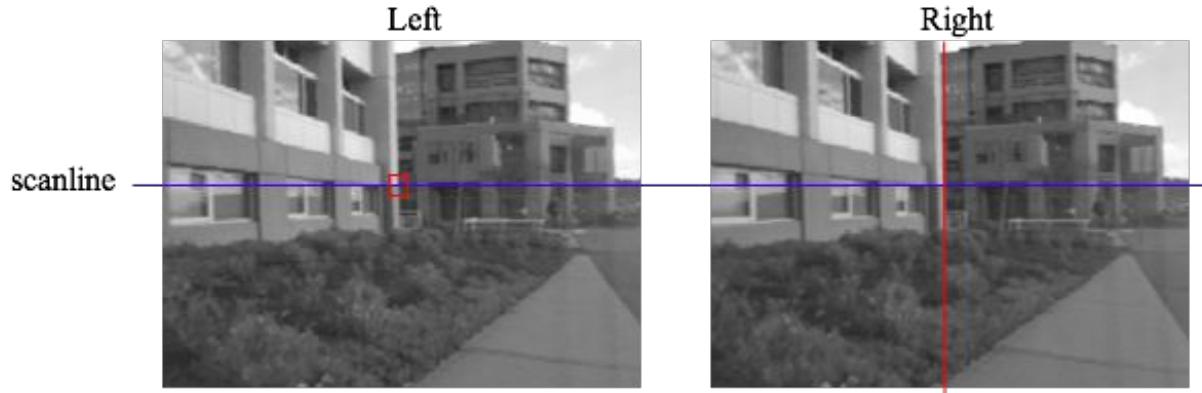
---



# Матчинг признаков

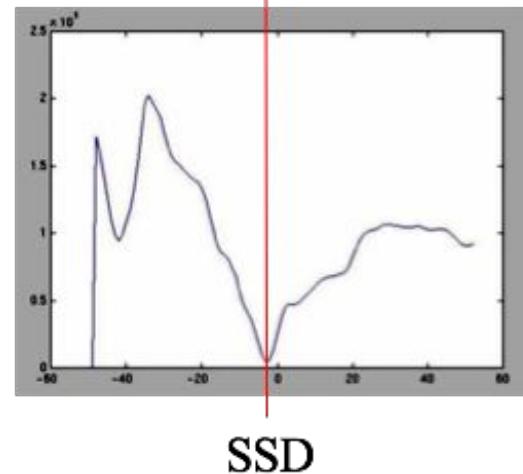


# Матчинг признаков

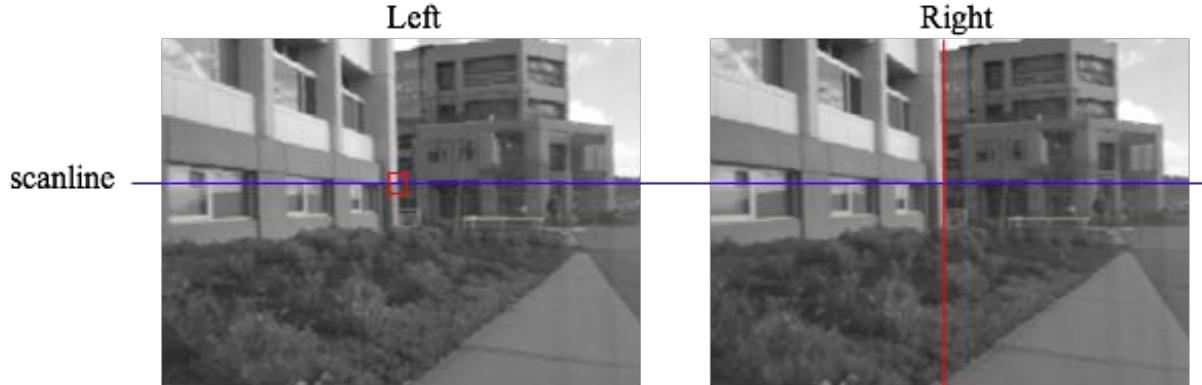


Сумма квадратов разностей (SSD)

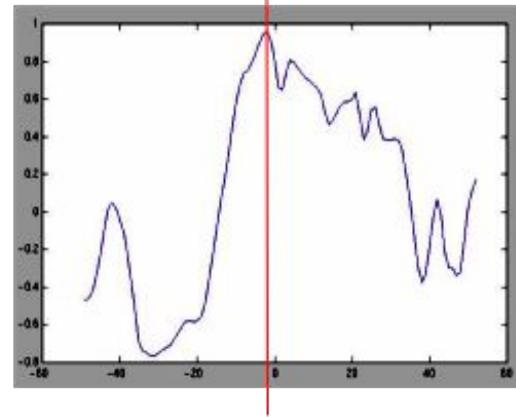
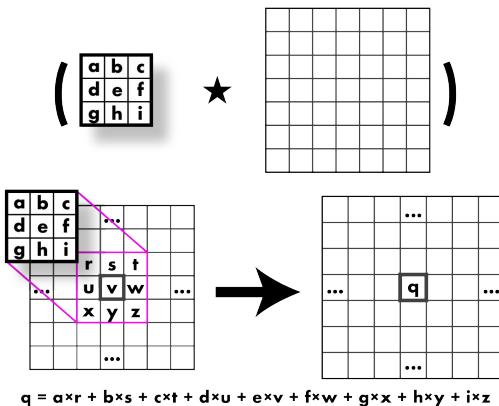
$$||\text{Патч}_L - \text{Патч}_R||^2$$



# Матчинг признаков



Кросс корреляция



Norm. corr

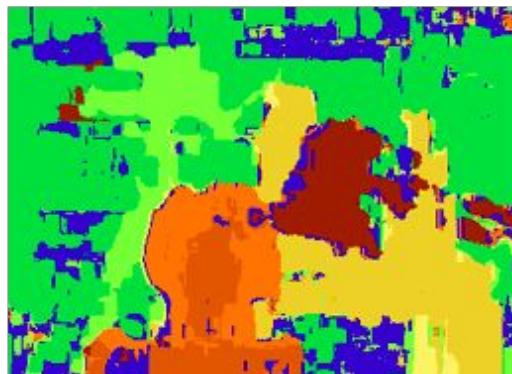
# Результаты

---

Data



Window-based matching



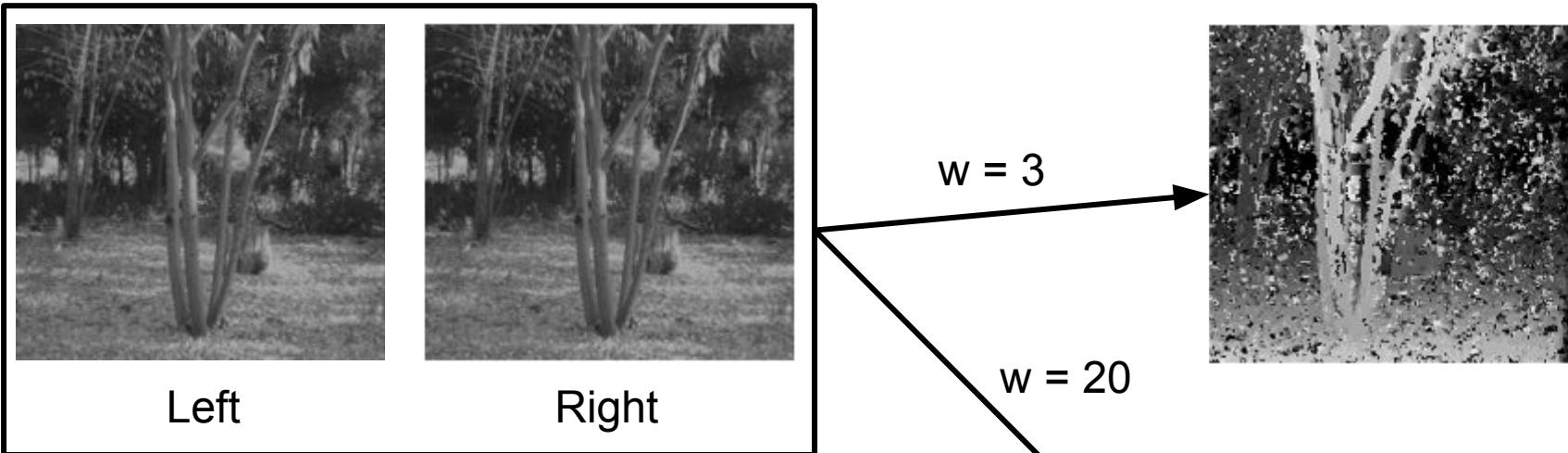
Ground truth



# Результаты: Мерседес



# Влияние размера окна

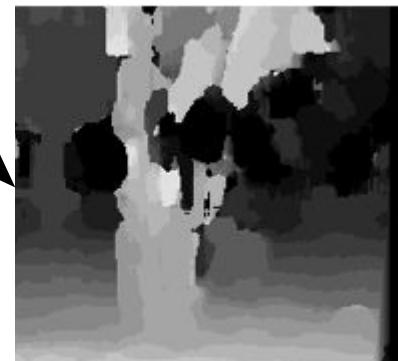


## Маленькое окно:

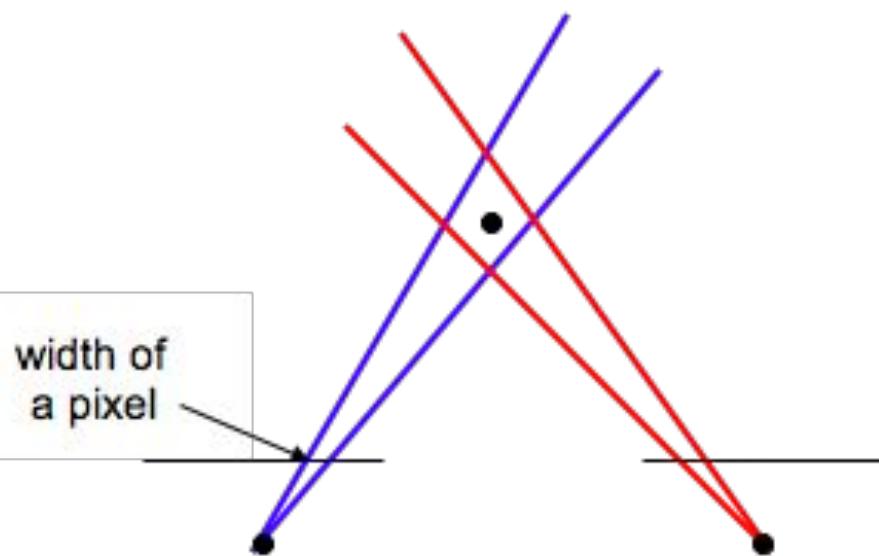
- + Хорошая точность много деталей
- Чувствительно к шуму

## Большое окно:

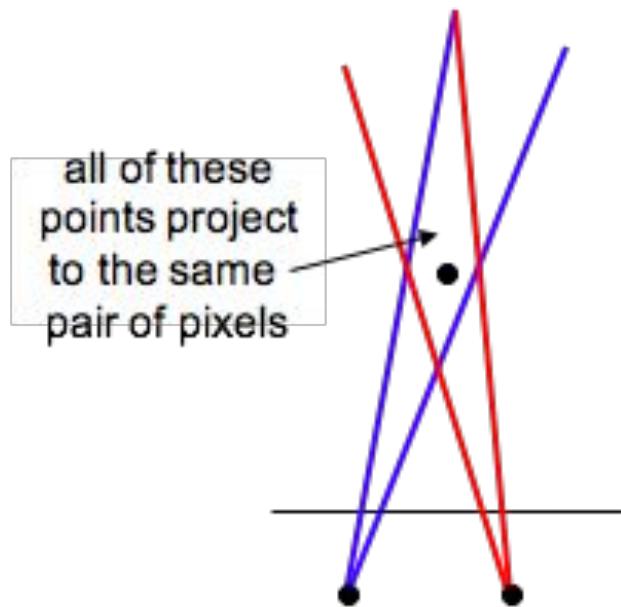
- + Устойчиво к шуму
- Маленькая точность, меньше деталей



# Влияние $\Delta x$ (baseline)



**Large Baseline**

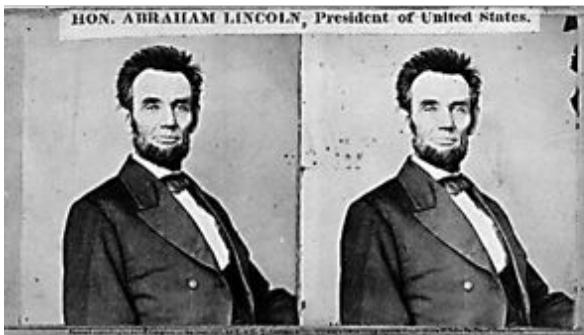


**Small Baseline**

# Когда алгоритм не работает?

# Когда алгоритм не работает?

---



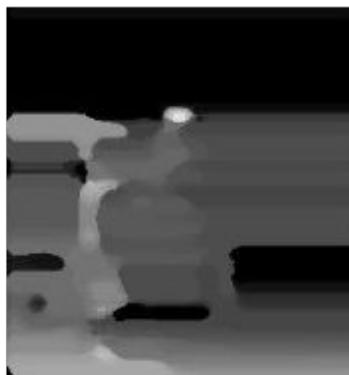
Textureless surfaces



Occlusions, repetition



Non-Lambertian surfaces, specularities



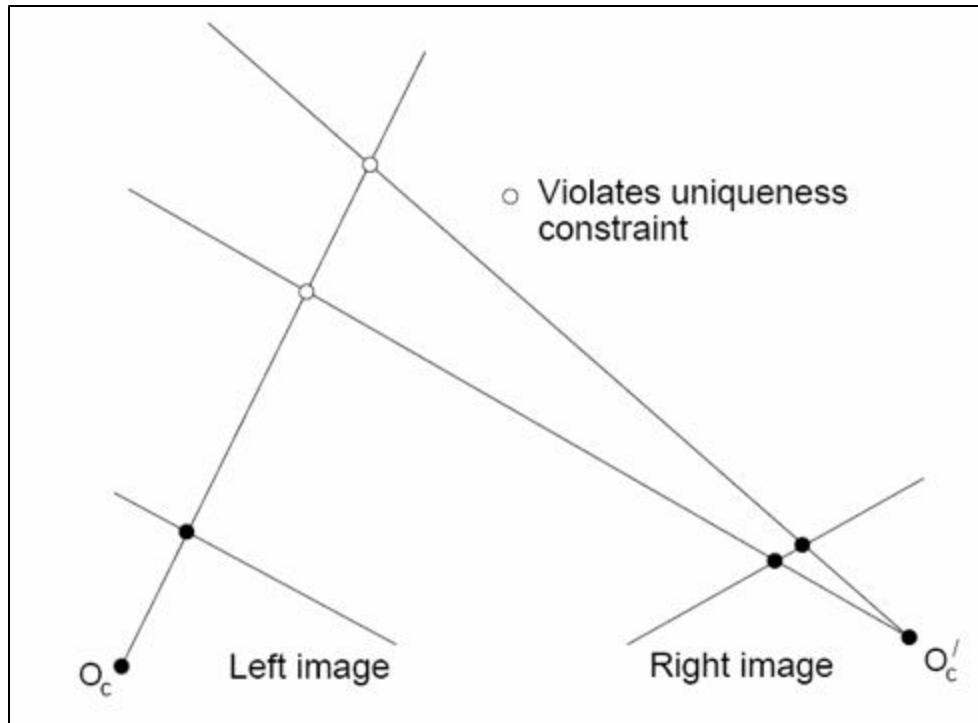
---

# Как можно улучшить матчинг?

# Стерео ограничения

# Стерео ограничения

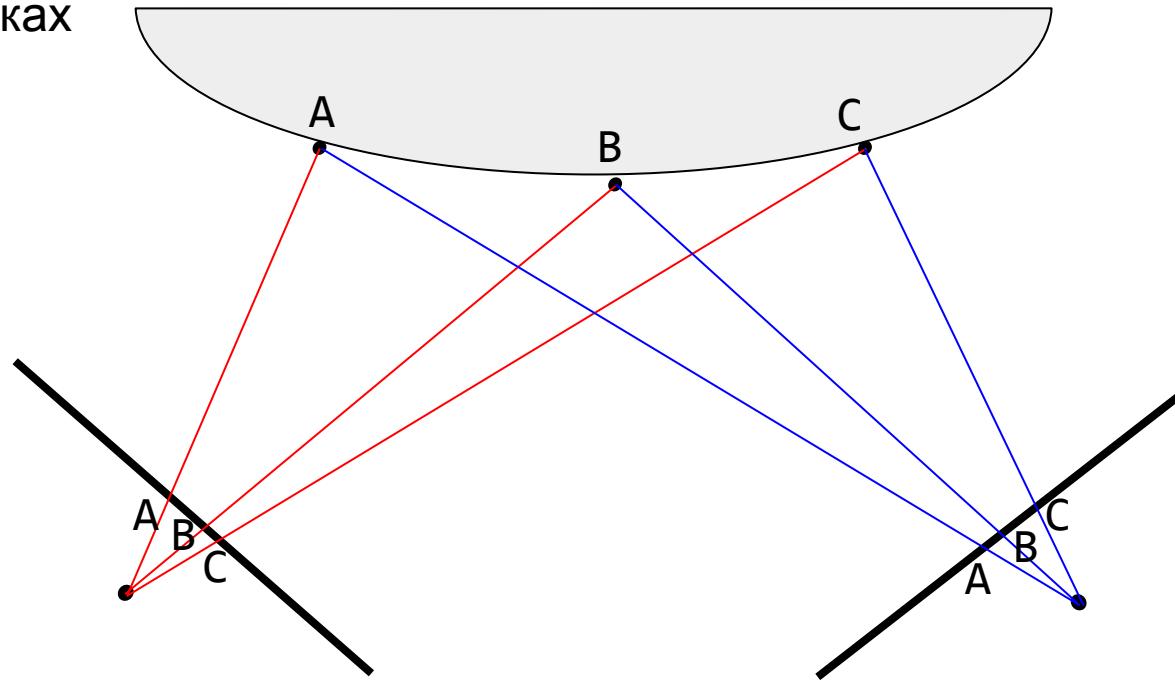
- Уникальность: Для каждой точки одного изображения, может быть только одна точка-матч из другого изображения



# Стерео ограничения

---

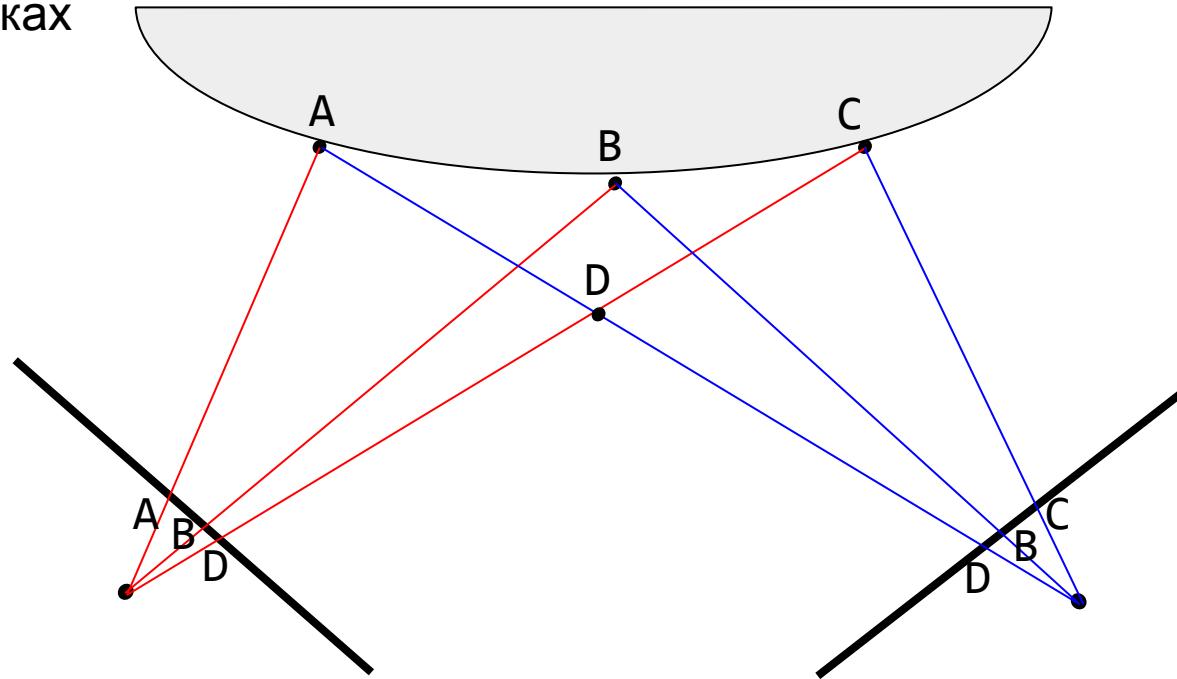
- Уникальность: Для каждой точки одного изображения, может быть только одна точка-матч из другого изображения
- Упорядочивание: Точки должны быть в одном и том же порядке на обоих картинках



# Стерео ограничения

---

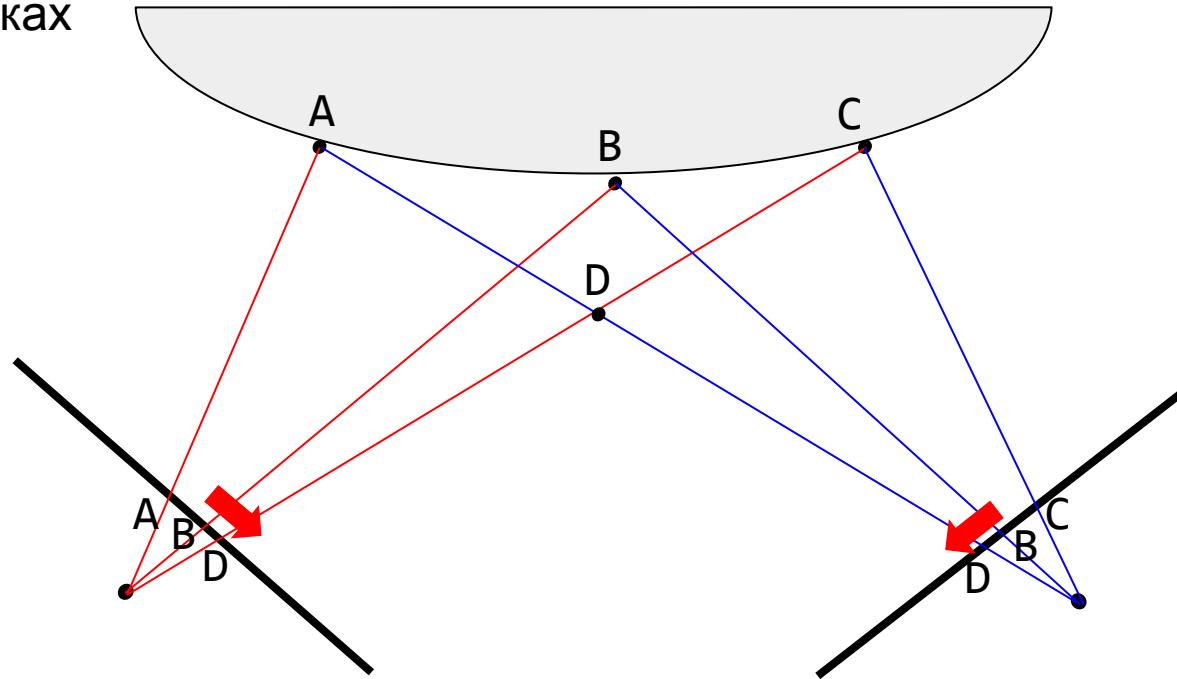
- Уникальность: Для каждой точки одного изображения, может быть только одна точка-матч из другого изображения
- Упорядочивание: Точки должны быть в одном и том же порядке на обоих картинках



# Стерео ограничения

---

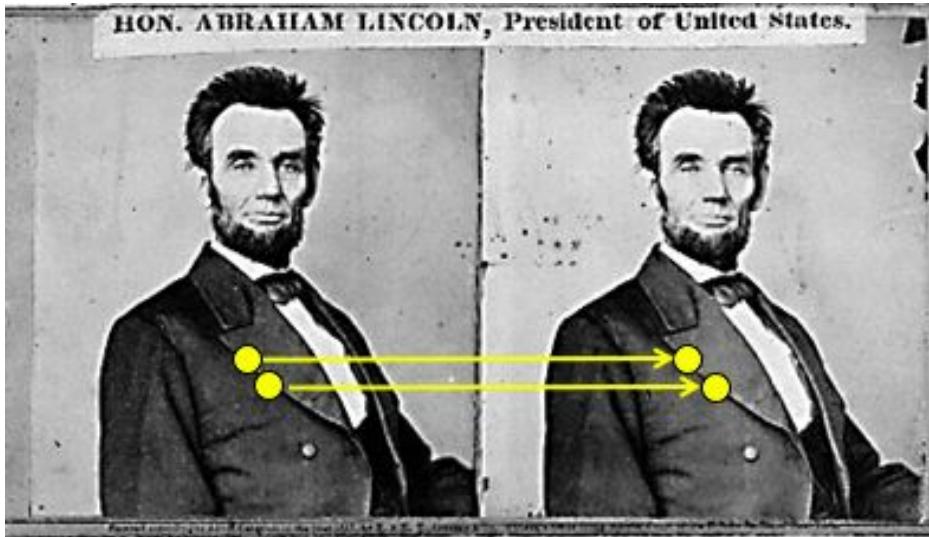
- Уникальность: Для каждой точки одного изображения, может быть только одна точка-матч из другого изображения
- Упорядочивание: Точки должны быть в одном и том же порядке на обоих картинках



# Стерео ограничения

---

- Уникальность: Для каждой точки одного изображения, может быть только одна точка-матч из другого изображения
- Упорядочивание: Точки должны быть в одном и том же порядке на обоих картинках
- Плавность: Близкие пиксели смещают недалеко друг от друга



# Разрез графа

---

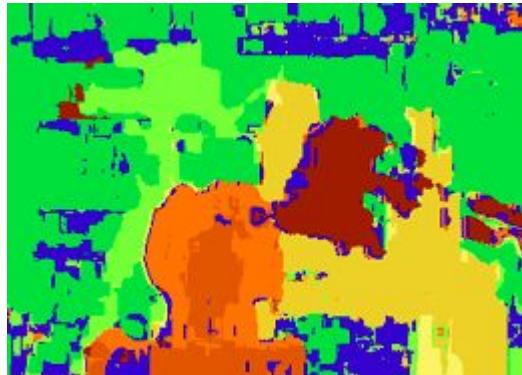
Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](#), PAMI 2001

# Разрез графа

Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](#), PAMI 2001



Before



Graph Cuts



Ground Truth





It's coding time!