Эпиполярные линии

- Линия в плоскости изображения одной из двух камер, проходящая через проекцию трехмерной точки и проекцию центра другой камеры, называется эпиполярной

$$l' = Fq$$
$$l = F^T q'$$

- Проекция центра одной камеры на другую называется эпиполем

Свойства

- Если F – фундаментальная матрица для пары камер (P,P’), то F^T – фундаментальная матрица для пары камеры (P’,P)
- Эпиполярные линии
- Эпиполь камеры P является нулем F
- F имеет 7 степеней свободы
- Зависит только от системы координат на изображении, не зависит от выбора трехмерной системы координат

Эпиполярные линии соответствуют друг другу

- $\forall q \neq e$ $\forall q'$такого, что $q' \neq e'$ и q' лежит на эпиполярной линии $\mathrm{F}q$ эпиполярная линия на первом изображении, соответствующая q', будет проходить через q, то есть $F^T q' = e \times q$

Соотношение между эпиполярными линиями

$$l' = F\left[e\right]_{\times} l$$

Связь фундаментальной матрицы с положением камер

$$F = [e']_\times P' P^+ \qquad P = K[I|0] \qquad P' = K'[R|T]$$

$$F = K'^{-T}[T]_\times RK^{-1}$$

Каноническая форма фундаментальной матрицы

$$F = [e']_{\times} P' P^{+} \qquad P = [\mathrm{I}|0] \qquad P' = [R|T]$$

$$F = [T]_{\times} R$$

Фундаментальная матрица для параллельного переноса вдоль оси X

$$F = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$$

Робототехника

Boston Dynamics, Spot Launch, https://www.youtube.com/watch?v=wlkCQXHEgjA

Модельные задачи

1 камера повернута относительно второй матрицей R и сдвинута на вектор T. Найти фундаментальную матрицу при единичных матрицах внутренних параметров

$$Q_2 = RQ_1 + T$$

$$F = [T]_{\times} R$$

Матрица поворота в python

```
import numpy as np
from scipy.spatial.transform import Rotation as R

anglez = np.pi/4
r = R.from_rotvec([0,0,anglez]).as_matrix()
```

Использование np.cross для получения $[T]_\times$

```
>>> r
array([[ 0.70710678, -0.70710678,  0.        ],
       [ 0.70710678,  0.70710678,  0.        ],
       [ 0.        ,  0.        ,  1.        ]])

>>> t
[10, 0, 0]

>>> tx
array([[  0.,   0.,   0.],
       [  0.,   0., -10.],
       [  0.,  10.,   0.]])
```

Использование np.cross для получения $[T]_\times$

```
>>> np.dot(tx,r)
array([[  0.        ,   0.        ,   0.        ],
       [  0.        ,   0.        , -10.        ],
       [  7.07106781,   7.07106781,   0.        ]])

>>> np.cross(t,r)
array([[  0.        ,   0.        ,  -7.07106781],
       [  0.        ,   0.        ,   7.07106781],
       [  0.        , -10.        ,   0.        ]])

>>> tx = np.cross(t,np.eye(3)).transpose()
>>> tx
array([[  0.,   0.,   0.],
       [  0.,   0., -10.],
       [  0.,  10.,   0.]])
```

Using np.cross axisa,axisb,axisc options

```
>>> r
array([[ 0.70710678, -0.70710678,  0.        ],
       [ 0.70710678,  0.70710678,  0.        ],
       [ 0.        ,  0.        ,  1.        ]])
>>> t = [10,0,0]

>>> np.cross(t, r, axisa=0, axisb=0)
array([[  0.        ,   0.        ,   7.07106781],
       [  0.        ,  -0.        ,   7.07106781],
       [  0.        , -10.        ,   0.        ]])


>>> np.cross(t, r, axisa=0, axisb=0, axisc=0)
array([[  0.        ,   0.        ,   0.        ],
       [  0.        ,  -0.        , -10.        ],
       [  7.07106781,   7.07106781,   0.        ]])
```

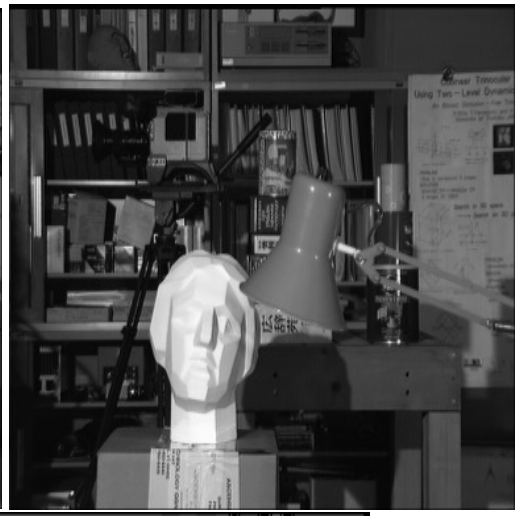
Определение трехмерных координат

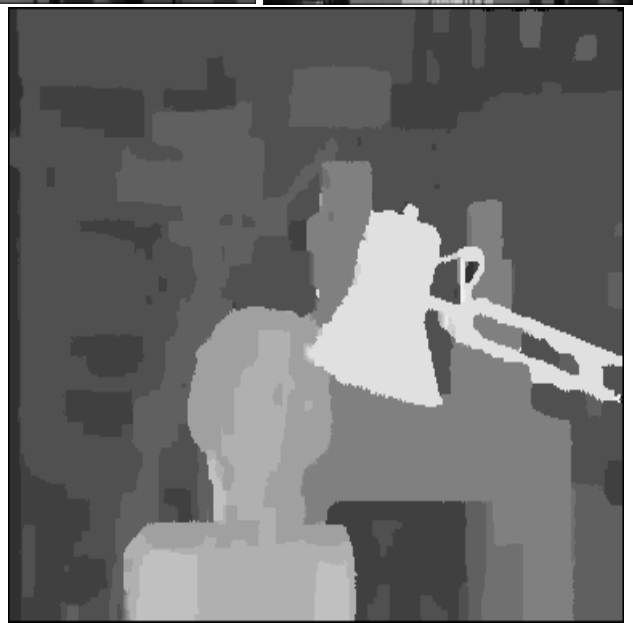
$$P = K[I|0] \qquad P' = K\left[I \mid T\right]$$

$$Z = \frac{f_x T}{u' - u} = \frac{f_x T}{D}$$

Изображение, в котором пиксель принимает значение D, называют картой смещений (disparity map)

Карта смещений по двум изображениям





Stereo correspondence block matching

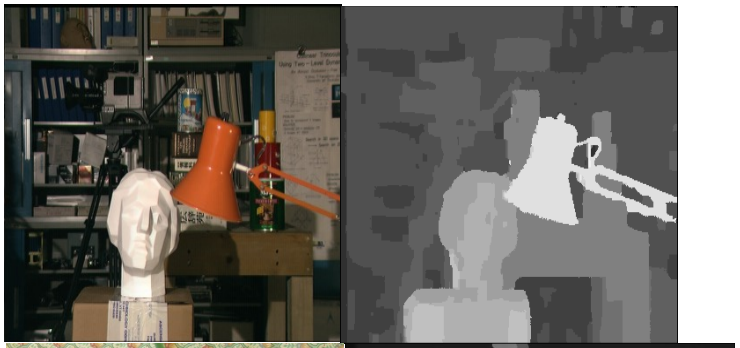
For each block in left image:

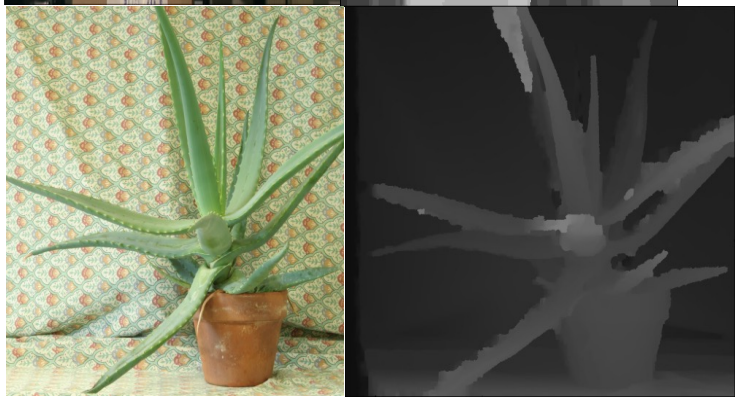
- Search for the corresponding block in the right image such that SSD or SAD between pixel intensities is minimum

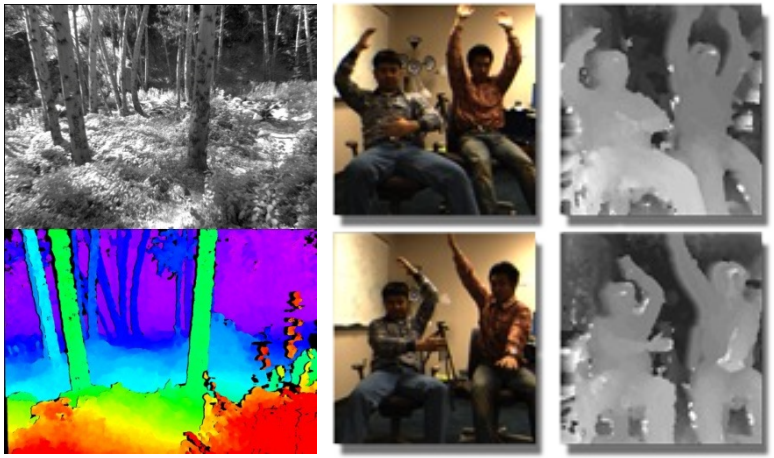
Stereo correspondence

- Без априорной информации: *Heiko Hirschmüller, "Accurate and efficient stereo processing by semi-global matching and mutual information", CVPR 2005.*
- С глубоким обучением: *Matteo Poggi et al, Guided Stereo Matching, CVPR 2019.*
- Проблемы:
  - Сложности с однородными по цвету областями
  - Временная синхронизация (global shutter vs rolling shutter, hardware synced shutter)

Stereo Matching







$$Z = f_x \frac{T}{D}$$

iPhone X True Depth camera