Lab 04. Image Matching and Image Stitching

Introduction to Computer Vision, Lab 04.

Today

- Image Matching
  - Harris Response Map
  - SIFT and Ratio-test
- Image Stitching
  - Transformation
  - RANSAC

Harris response map

- Compute the covariance matrix at each point

$$H = \sum_{(u,v)} w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

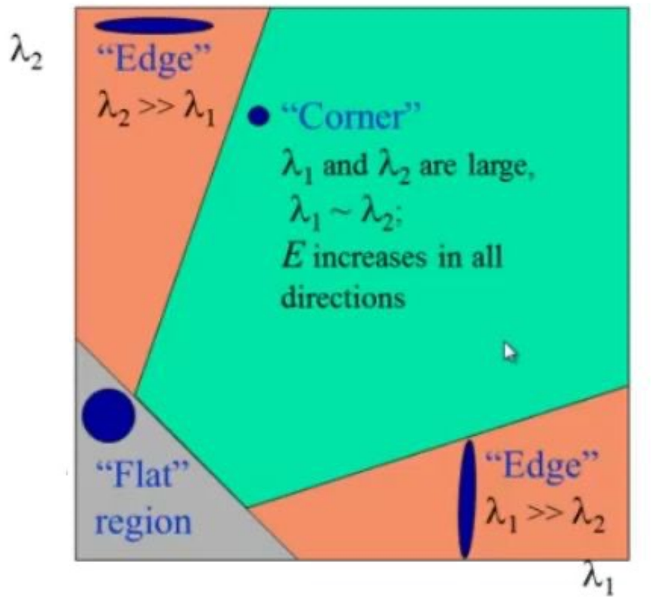
  - where $I_x = \frac{\partial f}{\partial x}, I_y = \frac{\partial f}{\partial y}$

Harris response map

- Theoretically, we can compute eigenvalues

$$H = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \lambda_{\pm} = \frac{1}{2}((a+d) \pm \sqrt{4bc + (a-d)^2})$$

and then classify points using eigenvalues of H, like:



Harris response map

- However, computing eigenvalues are expensive, so we use the following alternative

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} = \frac{determinant(H)}{trace(H)}$$

  - where

$$det(\begin{bmatrix} a & b \\ c & d \end{bmatrix}) = ad - bc \quad trace(\begin{bmatrix} a & b \\ c & d \end{bmatrix}) = a + d$$

Harris response map

- Threshold f and visualize
  - we skip non-maximum suppression operation here. You only need to visualize the thresholded harris response map.

Harris response map

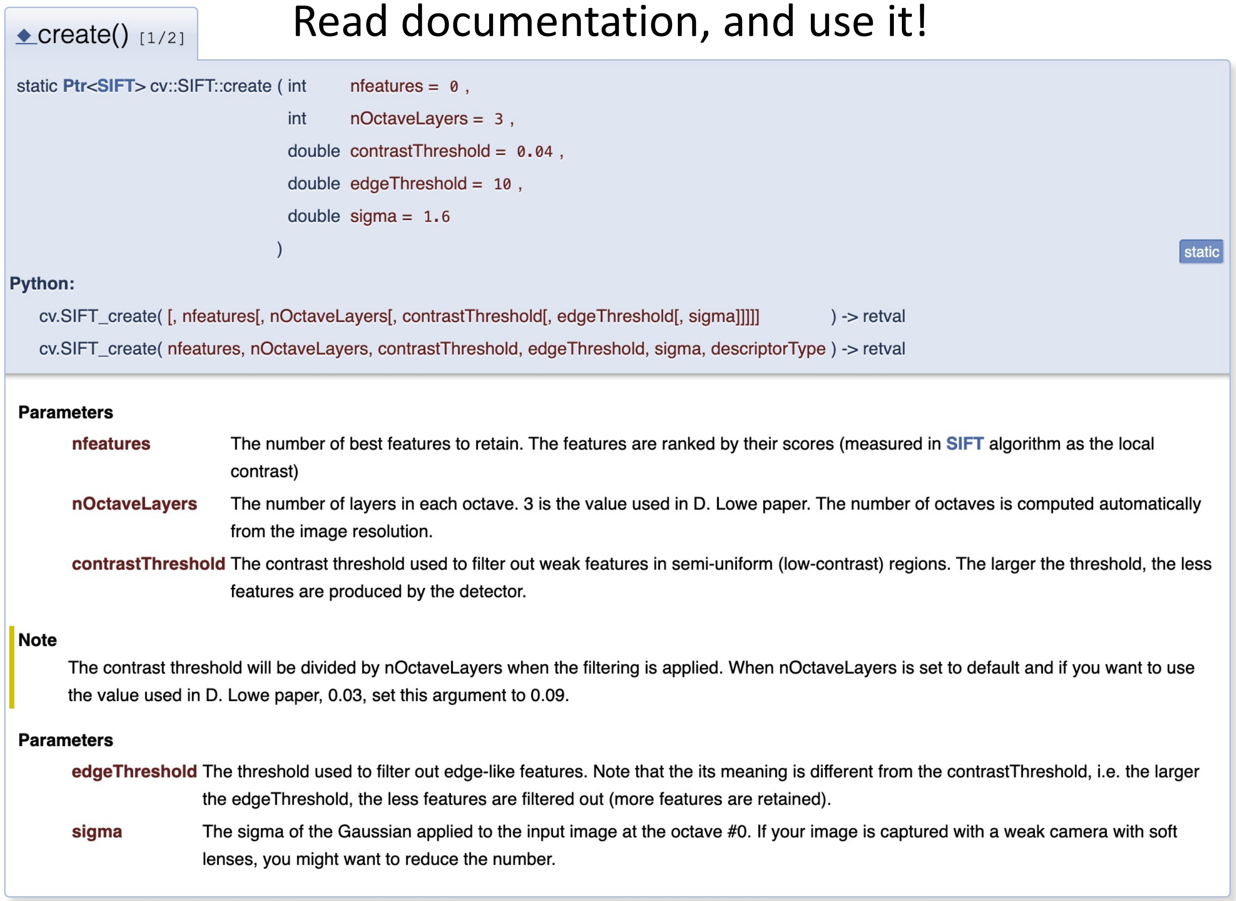
- https://zhuanlan.zhihu.com/p/36382429

Today

- Image Matching
    - Harris Response Map
    - SIFT and Ratio-test
- Image Stitching
    - Transformation
    - RANSAC

SIFT and Ratio-test

Read documentation, and use it!



◆ create() [1/2]

```
static Ptr<SIFT> cv::SIFT::create ( int     nfeatures = 0 ,
                                    int     nOctaveLayers = 3 ,
                                    double  contrastThreshold = 0.04 ,
                                    double  edgeThreshold = 10 ,
                                    double  sigma = 1.6
                                  )
```

static

Python:

```
cv.SIFT_create( [, nfeatures[, nOctaveLayers[, contrastThreshold[, edgeThreshold[, sigma]]]]] ) -> retval
cv.SIFT_create( nfeatures, nOctaveLayers, contrastThreshold, edgeThreshold, sigma, descriptorType ) -> retval
```

Parameters

nfeatures	The number of best features to retain. The features are ranked by their scores (measured in SIFT algorithm as the local contrast)
nOctaveLayers	The number of layers in each octave. 3 is the value used in D. Lowe paper. The number of octaves is computed automatically from the image resolution.
contrastThreshold	The contrast threshold used to filter out weak features in semi-uniform (low-contrast) regions. The larger the threshold, the less features are produced by the detector.

Note

The contrast threshold will be divided by nOctaveLayers when the filtering is applied. When nOctaveLayers is set to default and if you want to use the value used in D. Lowe paper, 0.03, set this argument to 0.09.

Parameters

edgeThreshold	The threshold used to filter out edge-like features. Note that the its meaning is different from the contrastThreshold, i.e. the larger the edgeThreshold, the less features are filtered out (more features are retained).
sigma	The sigma of the Gaussian applied to the input image at the octave #0. If your image is captured with a weak camera with soft lenses, you might want to reduce the number.

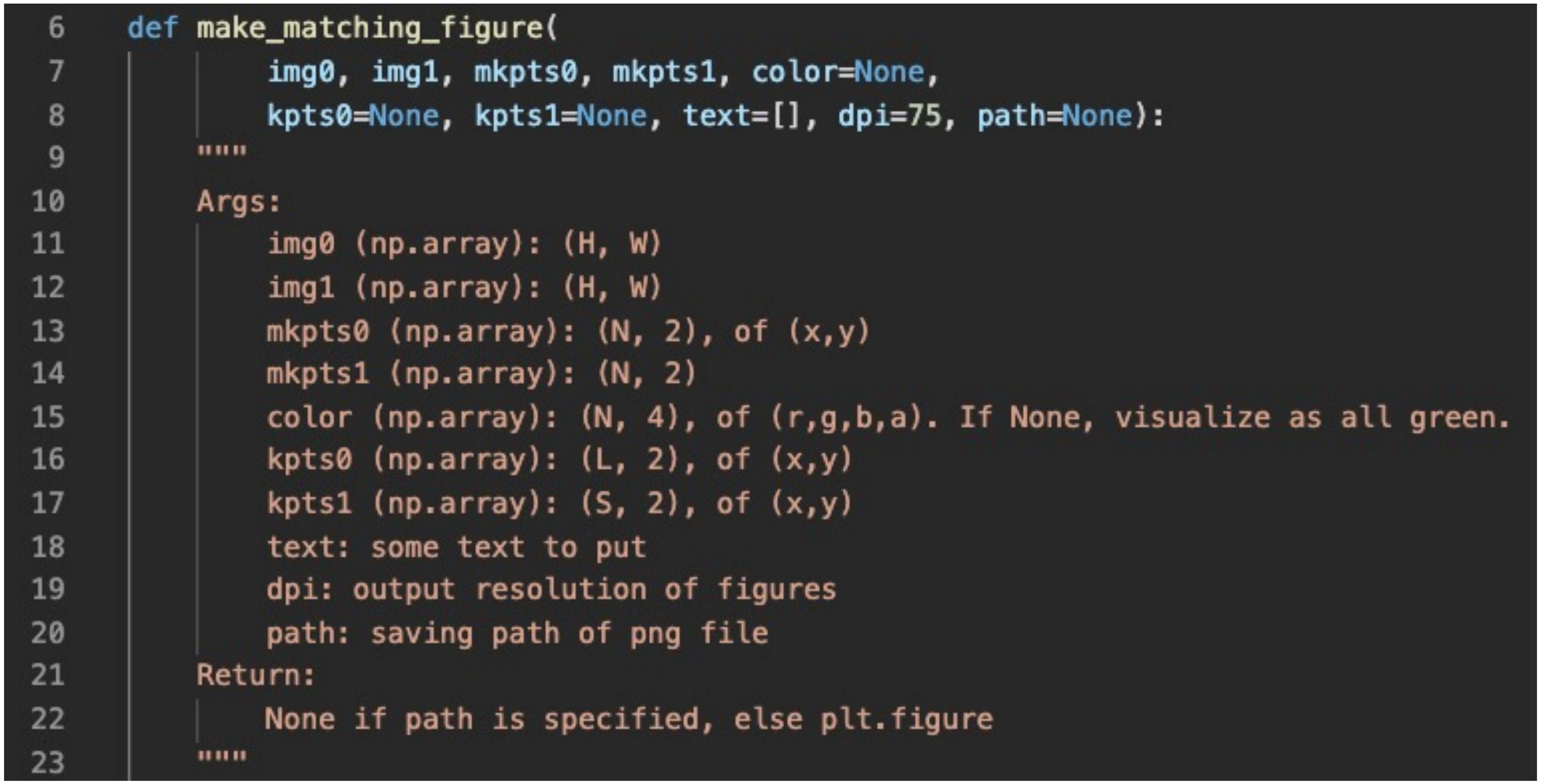
SIFT and Ratio-test

For SIFT descriptors, people usually match them with ratio-test.

(1) Please list the main advantage of ratio-test in matching SIFT descriptors.

(2) Do you think mutual-nearest-neighbor method can also work?

SIFT and Ratio-test

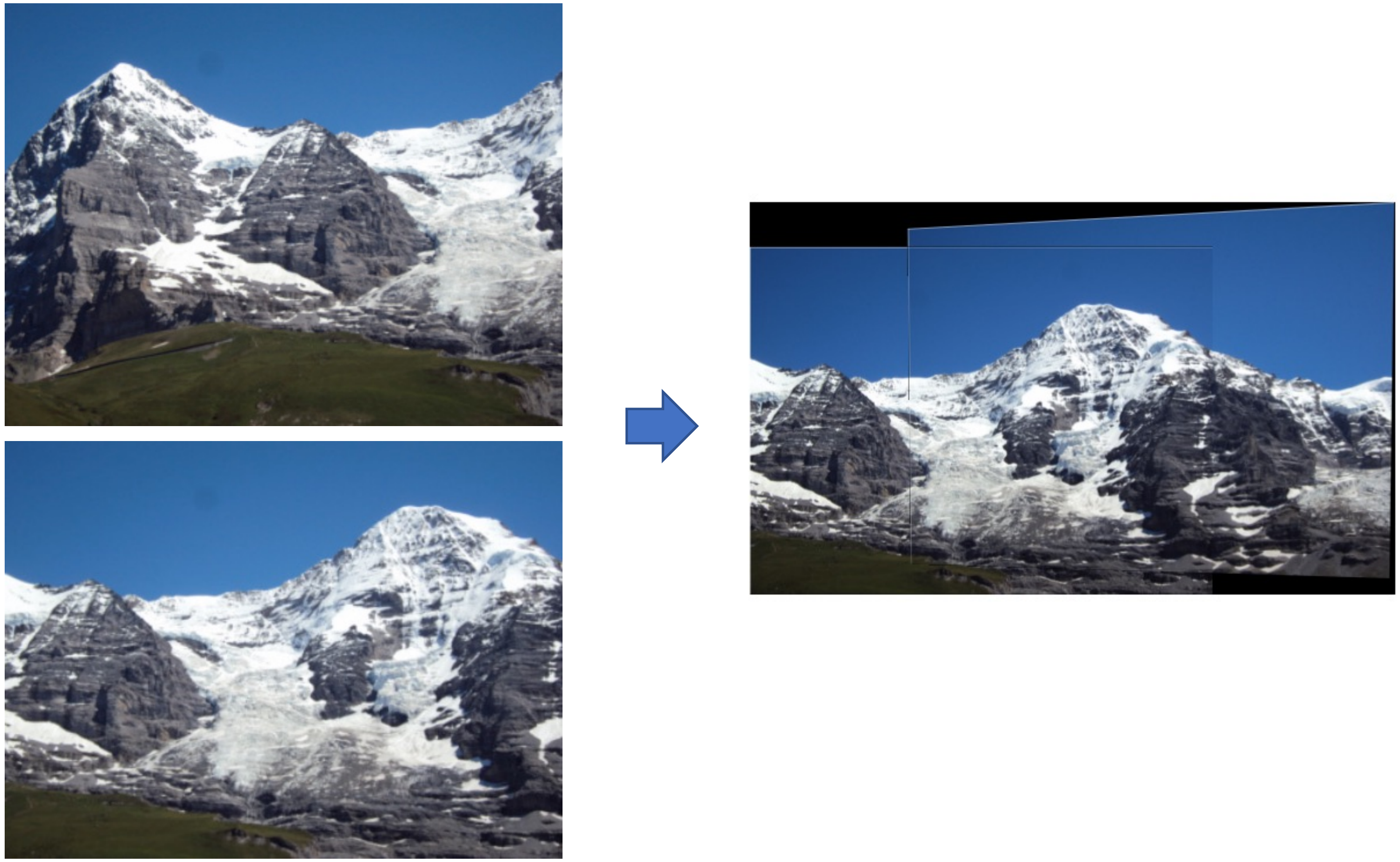


```python
def make_matching_figure(
        img0, img1, mkpts0, mkpts1, color=None,
        kpts0=None, kpts1=None, text=[], dpi=75, path=None):
    """
    Args:
        img0 (np.array): (H, W)
        img1 (np.array): (H, W)
        mkpts0 (np.array): (N, 2), of (x,y)
        mkpts1 (np.array): (N, 2)
        color (np.array): (N, 4), of (r,g,b,a). If None, visualize as all green.
        kpts0 (np.array): (L, 2), of (x,y)
        kpts1 (np.array): (S, 2), of (x,y)
        text: some text to put
        dpi: output resolution of figures
        path: saving path of png file
    Return:
        None if path is specified, else plt.figure
    """
```

Today

- Image Matching
    - Harris Response Map
    - SIFT and Ratio-test
- Image Stitching
    - Transformation
    - RANSAC

Transformation



Transformation

Here, the transformation *H* is defined as

$$\begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

(1) What type is this transformation?

(2) Please write down the converted equation in the form of Ah=0. To solve this equation, what's the minimal number of matches that we need?

Transformation

- Randomly select K matches to compute the transformation

Today

- Image Matching
  - Harris Response Map
  - SIFT and Ratio-test
- Image Stitching
  - Transformation
  - RANSAC

RANSAC

- To use naive ransac algorithm, we need N sample-points(样本点), to solve the model, we need K sample-points as a minimal requirement. Then perform:

1. Randomly sample K sample-points.
2. Fit the model with K sample-points. Denoted as h’.
3. Compute error of other sample points according to h’. Count the inliers within some threshold.
4. Repeat M times, the final h is the h’ with most inliers.

RANSAC

- To get the panorama

  pano = cv2.warpPerspective(img1_rgb, dsize, img0_rgb, cv2.INTER_LINEAR)

- Hot to solve the artifacts in the overlapping region? Name 2 possible methods.

- End-of-the-slides