

Exercise May 19: Geometric Transform of Image

Align multiple images. Then, Generate a “stitched” image from the aligned multiple images.

1. Take a **series of interesting overlapping photos**. **(more than 3 images, >3)**

2. Detect feature points from the images and match the feature points.

You can use a library function, such as AKAZE, SIFT, ORB, supergru, etc.

3. Assign a unique index i to each track.

Then you can have $\{x_i, y_i\}$ for N feature points for the series of input images.

4. Estimate a transform for each image **using the linear solution**.

Try every transform: translation, similarity, and affine as the motion model.

- 4'. For perspective transform (homography), use OpenCV function, or other ways.

example: `cv2.getPerspectiveTransform`

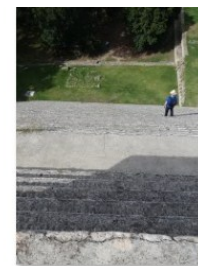
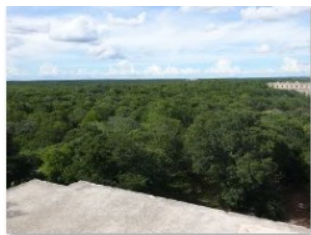
5. Compute the size of the resulting composite canvas.

6. Warp each image into its final position on the canvas.

`cv2.warpPerspective`

7. Average all of the images.

1. Take a series of interesting overlapping photos



2. Detect feature points from the images and match the feature points.

You can use a library function, such as AKAZE, SIFT, ORB, etc.

AKAZE https://docs.opencv.org/3.4/db/d70/tutorial_akaze_matching.html

ORB https://docs.opencv.org/4.x/d1/d89/tutorial_py_orb.html

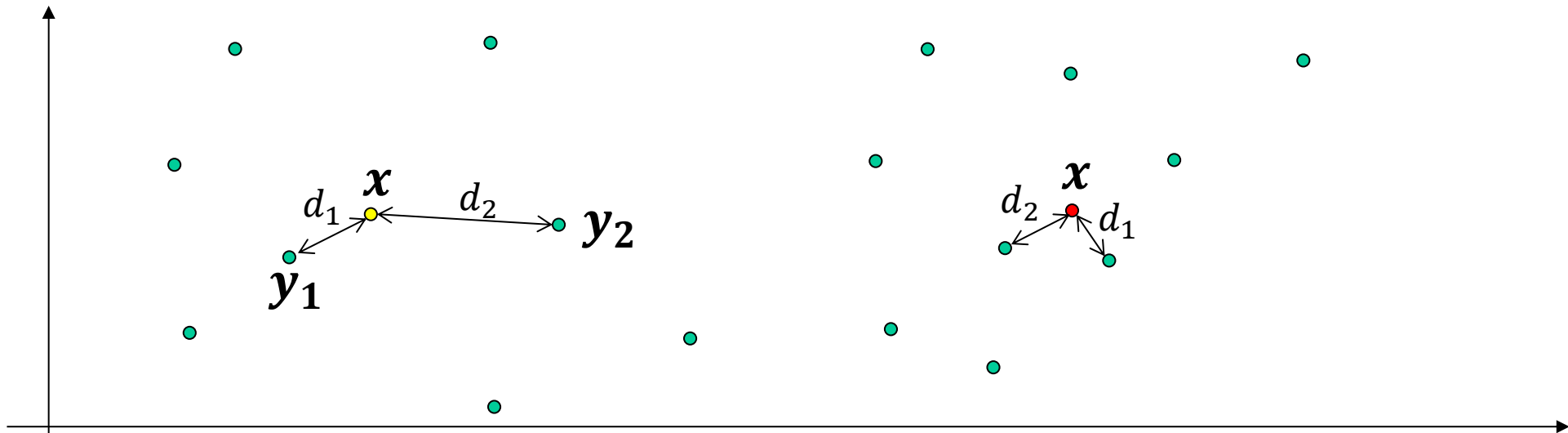
SUPERPOINT <https://github.com/rpautrat/SuperPoint>
<https://github.com/magicleap/SuperPointPretrainedNetwork>



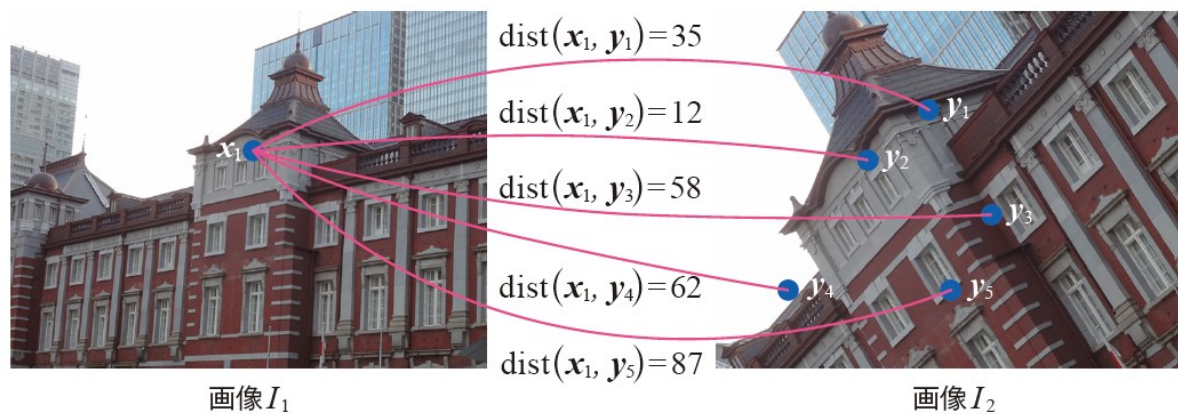
Matching strategy and error rates

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=0}^{N-1} (x_i - y_i)^2}$$

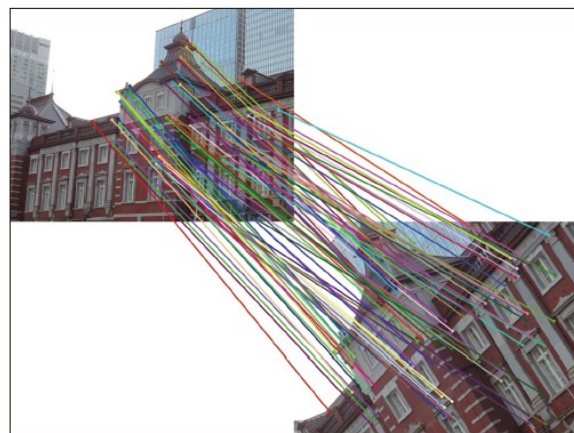
the feature descriptor space (N dim)



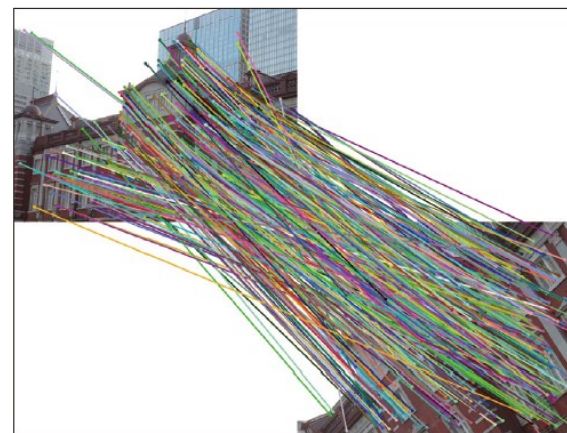
Nearest Neighbor Distances Ratio $\text{NNDR} = \frac{d_1}{d_2} < k$



■ 図 11.21 —— 対応点の探索



[a] $k=0.25$ の場合の対応づけ結果

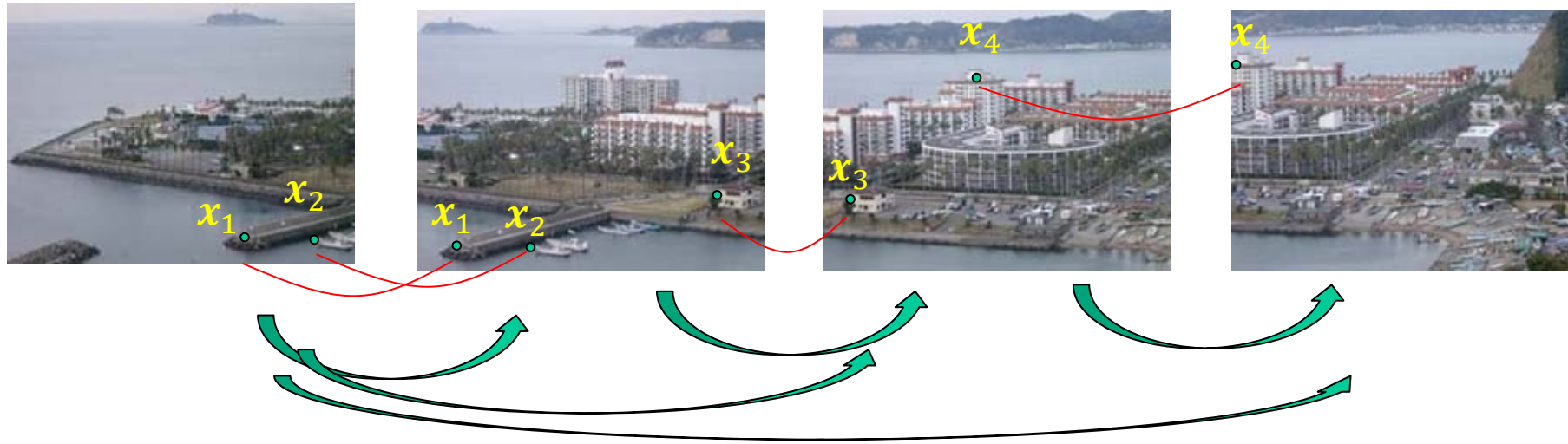


[b] $k=0.8$ の場合の対応づけ結果

■ 図 11.22 —— 2画像間の対応点とマッチング

3. Assign a unique index i to each track.

Then you can have $\{x_i, y_i\}$ for N feature points for the series of input images.

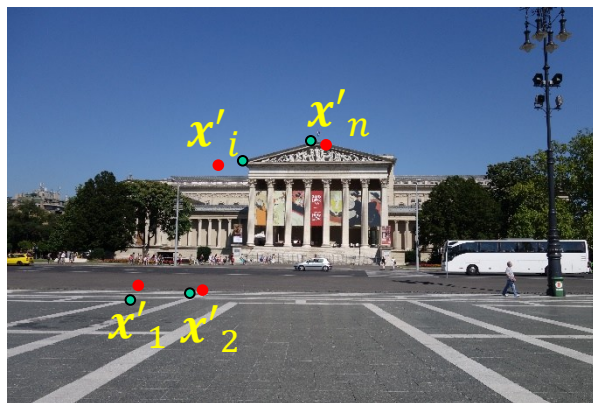
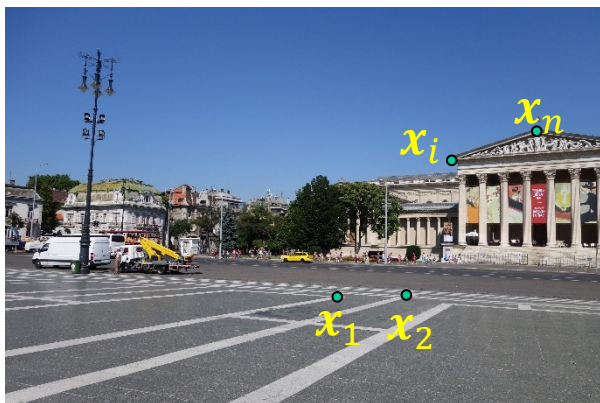


4. Estimate a transform for each image using the linear solution.

Transform	Matrix M
translation	$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix}$
Euclidean	$\begin{bmatrix} c_\theta & -s_\theta & t_x \\ s_\theta & c_\theta & t_y \end{bmatrix}$
similarity	$\begin{bmatrix} 1+a & -b & t_x \\ b & 1+a & t_y \end{bmatrix}$
affine	$\begin{bmatrix} 1+a_{00} & a_{01} & t_x \\ a_{10} & 1+a_{11} & t_y \end{bmatrix}$
Homography projective	$\begin{bmatrix} 1+h_{00} & h_{01} & h_{02} \\ h_{10} & 1+h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix}$

2D alignment using least squares (LS)

Given a set of matched feature points $\{(x_i, x'_i)\}$



$$x' = f(x; p)$$

$$x' = f(x; \mathbf{p})$$

Estimate \mathbf{p} by minimizing the sum of squared residuals: E_{LS}

$$E_{LS} = \sum_i \|\mathbf{r}_i\|^2 = \sum_i \|\underbrace{f(x_i; \mathbf{p})}_{\text{Predicted location by transformation model } \mathbf{p}} - \underbrace{x'_i}_{\text{measurement}}\|^2$$

Matched Feature Points $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_1^N$

If we can assume a **linear relationship** between the amount of motion $\Delta \mathbf{x} = \mathbf{x}' - \mathbf{x}$ and the unknown parameters \mathbf{p}

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}; \mathbf{p}) = \left(\frac{\partial \mathbf{f}(\mathbf{x}; \mathbf{p})}{\partial \mathbf{p}} \right) \mathbf{p} + \mathbf{x} = \underset{\text{Jacobian}}{\mathbf{J}(\mathbf{x})} \mathbf{p} + \mathbf{x}$$

$$\mathbf{x}' - \mathbf{x} = \Delta \mathbf{x} = \mathbf{J}(\mathbf{x}) \mathbf{p} \quad \sum_i \|\mathbf{J}(\mathbf{x}_i) \mathbf{p} - \Delta \mathbf{x}_i\|^2 \rightarrow 0$$

$$E_{LLS} = \sum_i \|\mathbf{J}(\mathbf{x}_i) \mathbf{p} - \Delta \mathbf{x}_i\|^2$$

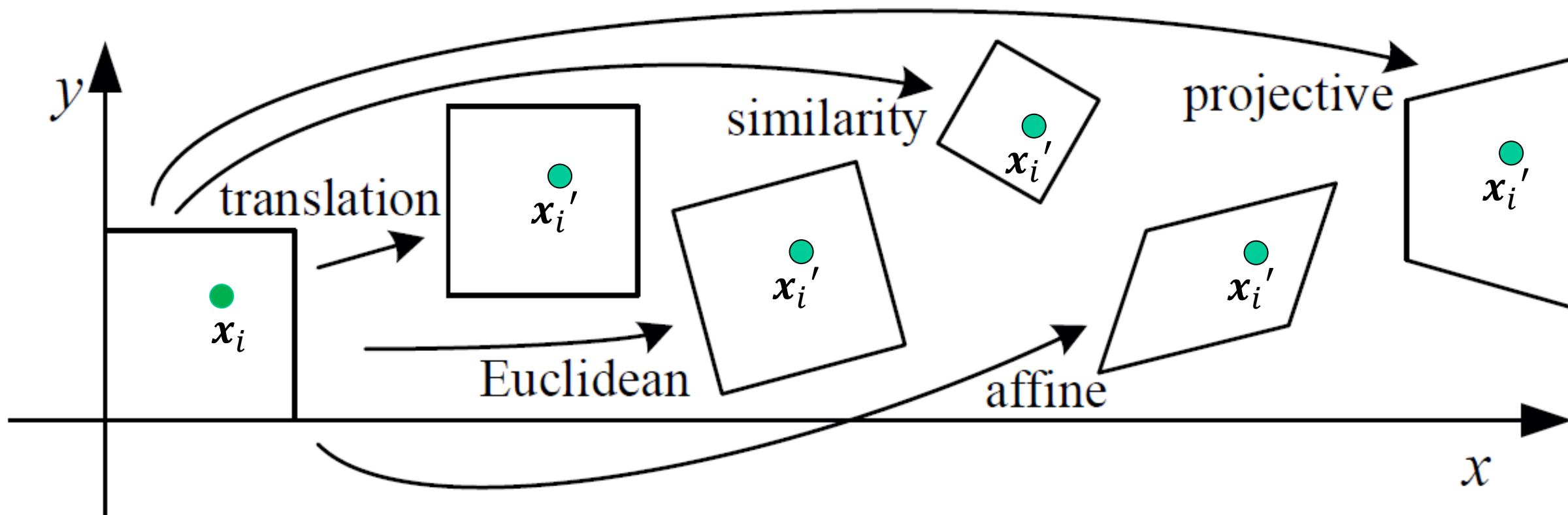
LLS: linear least squares

$$\begin{aligned} &= \mathbf{p}^T \left[\sum_i \mathbf{J}^T(\mathbf{x}_i) \mathbf{J}(\mathbf{x}_i) \right] \mathbf{p} - 2 \mathbf{p}^T \left[\sum_i \mathbf{J}^T(\mathbf{x}_i) \Delta \mathbf{x}_i \right] + \sum_i \|\Delta \mathbf{x}_i\|^2 \\ &= \mathbf{p}^T \mathbf{A} \mathbf{p} - 2 \mathbf{p}^T \mathbf{b} + c \end{aligned}$$

Minimum E_{LLS} can be found by solving symmetric positive definite (SPD) system :

$$\mathbf{A} \mathbf{p} = \mathbf{b} \quad \mathbf{A} = \sum_i \mathbf{J}^T(\mathbf{x}_i) \mathbf{J}(\mathbf{x}_i) \quad \mathbf{b} = \sum_i \mathbf{J}^T(\mathbf{x}_i) \Delta \mathbf{x}_i$$

Matched Feature Points $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_1^N$



Jacobians of the 2D coordinate transformations

Transform	Matrix	Parameters p	Jacobian J
translation	$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix}$	(t_x, t_y)	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Euclidean	$\begin{bmatrix} c_\theta & -s_\theta & t_x \\ s_\theta & c_\theta & t_y \end{bmatrix}$	(t_x, t_y, θ)	$\begin{bmatrix} 1 & 0 & -s_\theta x - c_\theta y \\ 0 & 1 & c_\theta x - s_\theta y \end{bmatrix}$
similarity	$\begin{bmatrix} 1+a & -b & t_x \\ b & 1+a & t_y \end{bmatrix}$	(t_x, t_y, a, b)	$\begin{bmatrix} 1 & 0 & x & -y \\ 0 & 1 & y & x \end{bmatrix}$
affine	$\begin{bmatrix} 1+a_{00} & a_{01} & t_x \\ a_{10} & 1+a_{11} & t_y \end{bmatrix}$	$(t_x, t_y, a_{00}, a_{01}, a_{10}, a_{11})$	$\begin{bmatrix} 1 & 0 & x & y & 0 & 0 \\ 0 & 1 & 0 & 0 & x & y \end{bmatrix}$
projective	$\begin{bmatrix} 1+h_{00} & h_{01} & h_{02} \\ h_{10} & 1+h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix}$	$(h_{00}, h_{01}, \dots, h_{21})$	(see Section 8.1.3)

re-parameterized the motions so that they are identity I for $p = \mathbf{0}$.

Jacobians of the 2D coordinate transformations

Translation

$$f(x; \mathbf{p}) = f\left(\begin{bmatrix} x \\ y \end{bmatrix}; \begin{bmatrix} t_x \\ t_y \end{bmatrix}\right) = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad J = \frac{\partial \mathbf{f}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial x'}{\partial \mathbf{p}} \\ \frac{\partial y'}{\partial \mathbf{p}} \end{bmatrix} = \begin{bmatrix} \frac{\partial x'}{\partial t_x} & \frac{\partial x'}{\partial t_y} \\ \frac{\partial y'}{\partial t_x} & \frac{\partial y'}{\partial t_y} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Similarity

$$f(x; \mathbf{p}) = f\left(\begin{bmatrix} x \\ y \end{bmatrix}; \begin{bmatrix} t_x \\ t_y \\ a \\ b \end{bmatrix}\right) = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 + a & -b & t_x \\ b & 1 + a & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad J = \frac{\partial \mathbf{f}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial x'}{\partial \mathbf{p}} \\ \frac{\partial y'}{\partial \mathbf{p}} \end{bmatrix} = \begin{bmatrix} \frac{\partial x'}{\partial t_x} & \frac{\partial x'}{\partial t_y} & \frac{\partial x'}{\partial a} & \frac{\partial x'}{\partial b} \\ \frac{\partial y'}{\partial t_x} & \frac{\partial y'}{\partial t_y} & \frac{\partial y'}{\partial a} & \frac{\partial y'}{\partial b} \end{bmatrix} = \begin{bmatrix} 1 & 0 & x & -y \\ 0 & 1 & y & x \end{bmatrix}$$

Affine

$$f(x; \mathbf{p}) = f\left(\begin{bmatrix} x \\ y \end{bmatrix}; \begin{bmatrix} t_x \\ t_y \\ a_{00} \\ a_{01} \\ a_{10} \\ a_{11} \end{bmatrix}\right) = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 + a_{00} & a_{01} & t_x \\ a_{10} & 1 + a_{11} & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad J = \frac{\partial \mathbf{f}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial x'}{\partial \mathbf{p}} \\ \frac{\partial y'}{\partial \mathbf{p}} \end{bmatrix} = \begin{bmatrix} \frac{\partial x'}{\partial t_x} & \frac{\partial x'}{\partial t_y} & \frac{\partial x'}{\partial a_{00}} & \frac{\partial x'}{\partial a_{01}} & \frac{\partial x'}{\partial a_{10}} & \frac{\partial x'}{\partial a_{11}} \\ \frac{\partial y'}{\partial t_x} & \frac{\partial y'}{\partial t_y} & \frac{\partial y'}{\partial a_{00}} & \frac{\partial y'}{\partial a_{01}} & \frac{\partial y'}{\partial a_{10}} & \frac{\partial y'}{\partial a_{11}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & x & y & 0 & 0 \\ 0 & 1 & 0 & 0 & x & y \end{bmatrix}$$

Motion Alignment in Translation Case

Matched Feature Points $\{(\mathbf{x}_i, \mathbf{x}'_i)\}_1^N = \left\{ \left(\begin{bmatrix} x_i \\ y_i \end{bmatrix}, \begin{bmatrix} x'_i \\ y'_i \end{bmatrix} \right) \right\}_1^N$ $\{\Delta \mathbf{x}_i\}_1^N = \left\{ \begin{bmatrix} x'_i - x_i \\ y'_i - y_i \end{bmatrix} \right\}_1^N$

$$\mathbf{A} = \sum_i \mathbf{J}^T(\mathbf{x}_i) \mathbf{J}(\mathbf{x}_i)$$

$$\mathbf{A} = \sum_{i=1}^N \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} N & 0 \\ 0 & N \end{bmatrix}$$

$$\mathbf{b} = \sum_i \mathbf{J}^T(\mathbf{x}_i) \Delta \mathbf{x}_i$$

$$\mathbf{b} = \sum_{i=1}^N \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x'_i - x_i \\ y'_i - y_i \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^N (x'_i - x_i) \\ \sum_{i=1}^N (y'_i - y_i) \end{bmatrix}$$

$$\mathbf{A} \mathbf{p} = \mathbf{b}$$

$$\begin{bmatrix} N & 0 \\ 0 & N \end{bmatrix} \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^N (x'_i - x_i) \\ \sum_{i=1}^N (y'_i - y_i) \end{bmatrix}$$

Motion Alignment in similarity transform

$$\text{Matched Feature Points } \{(\mathbf{x}_i, \mathbf{x}'_i)\}_1^N = \left\{ \left(\begin{bmatrix} x_i \\ y_i \end{bmatrix}, \begin{bmatrix} x'_i \\ y'_i \end{bmatrix} \right) \right\}_1^N \quad \{\Delta \mathbf{x}_i\}_1^N = \left\{ \begin{bmatrix} \Delta x_i \\ \Delta y_i \end{bmatrix} \right\}_1^N = \left\{ \begin{bmatrix} x'_i - x_i \\ y'_i - y_i \end{bmatrix} \right\}_1^N$$

$$\mathbf{A} = \sum_{i=1}^N \begin{bmatrix} 1 & 0 & x_i & -y_i \\ 0 & 1 & y_i & x_i \\ x_i & y_i & x_i^2 + y_i^2 & 0 \\ -y_i & x_i & 0 & x_i^2 + y_i^2 \end{bmatrix} \quad \mathbf{b} = \sum_{i=1}^N \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ x_i & -y_i \\ y_i & x_i \end{bmatrix} \begin{bmatrix} \Delta x_i \\ \Delta y_i \end{bmatrix} = \sum_{i=1}^N \begin{bmatrix} \Delta x_i \\ \Delta y_i \\ x_i \Delta x_i - y_i \Delta y_i \\ y_i \Delta x_i + x_i \Delta y_i \end{bmatrix}$$

赤のところの符号が逆？

$$\mathbf{A} \mathbf{p} = \mathbf{b}$$

$$\begin{bmatrix} N & 0 & \sum_{i=1}^N x_i & -\sum_{i=1}^N y_i \\ 0 & N & \sum_{i=1}^N y_i & \sum_{i=1}^N x_i \\ \sum_{i=1}^N x_i & \sum_{i=1}^N y_i & \sum_{i=1}^N x_i^2 + y_i^2 & 0 \\ -\sum_{i=1}^N y_i & \sum_{i=1}^N x_i & 0 & \sum_{i=1}^N x_i^2 + y_i^2 \end{bmatrix} \begin{bmatrix} t_x \\ t_y \\ a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^N \Delta x_i \\ \sum_{i=1}^N \Delta y_i \\ \sum_{i=1}^N x_i \Delta x_i - y_i \Delta y_i \\ \sum_{i=1}^N y_i \Delta x_i + x_i \Delta y_i \end{bmatrix}$$

4'. Use OpenCV function for estimating homography. cv2.getPerspectiveTransform

https://www.devdoc.net/linux/OpenCV-3.2.0/da/d6e/tutorial_py_geometric_transformations.html

```
pts1 = np.float32([[176,187],[465,94],[98, 411],[572,356]])  
pts2 = np.float32([[100,200],[500,200],[100,700],[500, 700]])  
M = cv2.getPerspectiveTransform(pts1,pts2)
```

5. Compute the size of the resulting composite canvas.



6. Warp each image into its final position on the canvas.

`cv2.warpPerspective`

```
dst = cv2.warpPerspective(img, M, (600, 800))
```

Stitching and Panograph

This exercise is similar as Ex 8.2: Panography in the reference book p.549.

Panograph and **stitching** are both methods of combining multiple photographs to create a larger image, but with different objectives and results.

A **panograph** (<https://edelberto-cabrera.pixels.com/featured/panograph-of-a-beach-edelberto-cabrera.html>)

- an artistic technique where overlapping photos are arranged in a non-linear, mosaic-like fashion, emphasizing their individual perspectives and angles, resulting in a creative, fragmented appearance.
- celebrates the differences between individual photos

stitching (or photo stitching)

- aims to produce a seamless, cohesive image.
- aligns and blends photos carefully to ensure continuity in perspective, scale, and lighting, commonly used for panoramic images in landscape and architectural photography.
- seeks to make these differences invisible.

Report submission

- Deadline: June 2
- Language: English or Japanese
- Content: Goal, Method, Results, Discussion, Conclusion

Discussion

- Comparison of different geometric transforms
- Comparison with recent applications for stitching
 - <https://paperswithcode.com/task/image-stitching>
 - <https://mattabrown.github.io/autostitch.html>
 - <https://www.microsoft.com/en-us/research/project/image-composite-editor/>
 - <https://paperswithcode.com/task/image-stitching>