Georgia Institute of Technology

School of Electrical and Computer Engineering

Homography Matrix Estimation

Author:
Xueren Ge

Supervisor:
Nope

Robot

March 12, 2021

1 Graph Transformation

1.1 Rotation

If you rotate a picture around origin $(0, 0)$ counterclockwise by θ agnle, we can have,

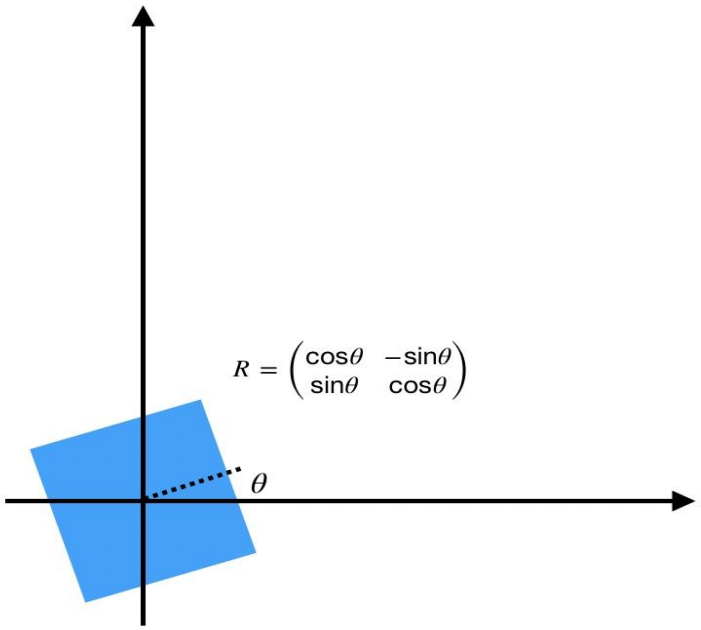

Figure 1: rotation

$$x' = x \cdot \cos\theta - y \cdot \sin\theta \tag{1}$$
$$y' = x \cdot \sin\theta + y \cdot \cos\theta \tag{2}$$

written in the form of matrix multiplication,

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{3}$$

1.2 Translation

Translation can be described as,

$$x' = x + t_x \tag{4}$$
$$y' = y + t_y \tag{5}$$

In matrix form, we have

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{6}$$

Here comes the problem, that is we can't write it as matrix multiplication form. To solve this problem, let's introduce homogeneous coordinate system, where $(x, y) \iff (x, y, 1)$, and then write it as matrix multiplication,

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{I}_{2\times 2} & \boldsymbol{T}_{2\times 1} \\ \boldsymbol{0}^T & \boldsymbol{1} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{7}$$

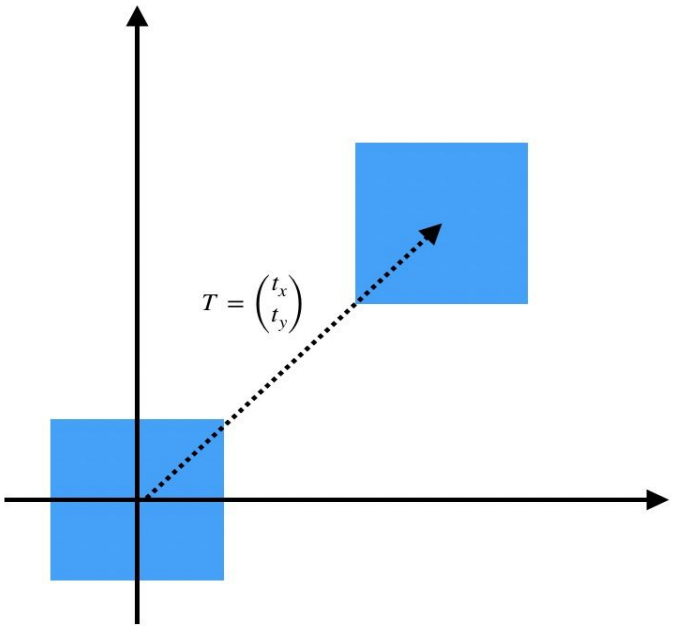

Figure 2: translation

and then, we can combine rotation with translation, which is **Rigid Body Transformation**,

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R}_{2\times 2} & \boldsymbol{T}_{2\times 1} \\ \boldsymbol{0}^T & \boldsymbol{1} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{8}$$

Note that here rotation matrix $\boldsymbol{R}_{2\times 2}$ is orthogonal matrix.

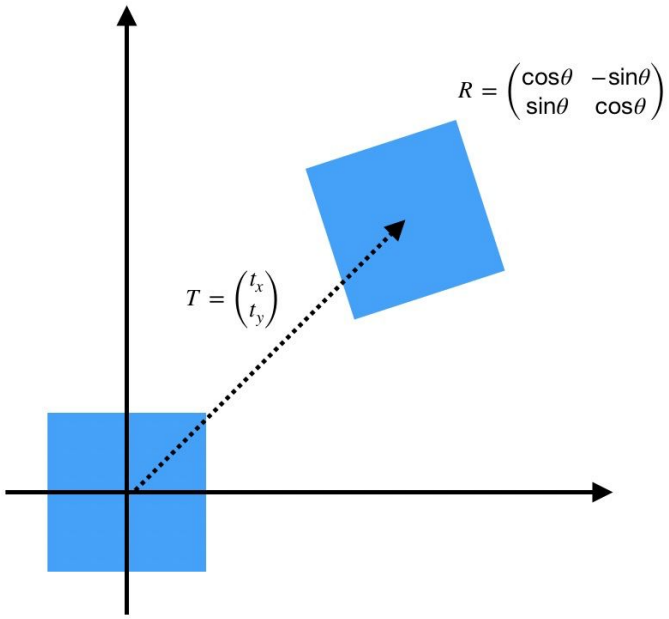

Figure 3: rigid body transformation

1.3 Affine Transformation

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{A}_{2\times 2} & \boldsymbol{T}_{2\times 1} \\ \boldsymbol{0}^T & \boldsymbol{1} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{9}$$

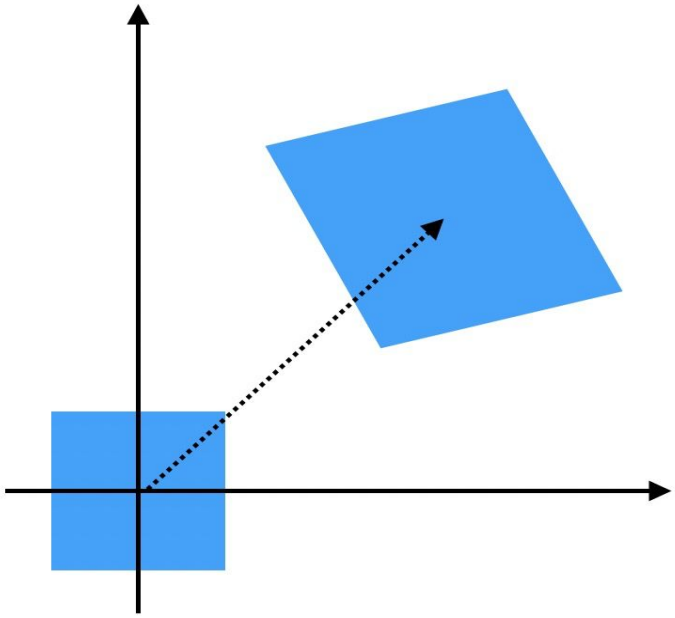

Figure 4: affine transformation

where $\boldsymbol{A}_{2\times 2} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ can be any 2×2 matrix, which is different from Rigid Body Transformation.

Compare with Rigid Body Transformation, Affine Transformation will not only change the location but also shape of object. However, transformed object will keep "straightness".

1.4 Projection Transformation (Homograph Transformation)

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{A}_{2\times 2} & \boldsymbol{T}_{2\times 1} \\ \boldsymbol{V}^T & \boldsymbol{s} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \boldsymbol{H}_{3\times 3} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{10}$$

where $\boldsymbol{A}_{2\times 2}$ is parameters of homograph transformation, $\boldsymbol{T}_{2\times 1}$ is parameters of translation transformation. $\boldsymbol{V}^T = [v_1, v_2]$ represents a kind of "edge intersection after transformation" relationship and $\boldsymbol{s}$ represents scaling factor related to $\boldsymbol{V}$ because you will have $v_1 x + v_2 y + s = 0$. Usually, we will make $\boldsymbol{s} = 1$ so that,

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ v_1 & v_2 & s \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ v_1 x + v_2 y + s \end{pmatrix} \iff \begin{pmatrix} \frac{x}{v_1 x + v_2 y + s} \\ \frac{y}{v_1 x + v_2 y + s} \end{pmatrix} \tag{11}$$

Here, Homograph Transformation will totally change location and shape of object.

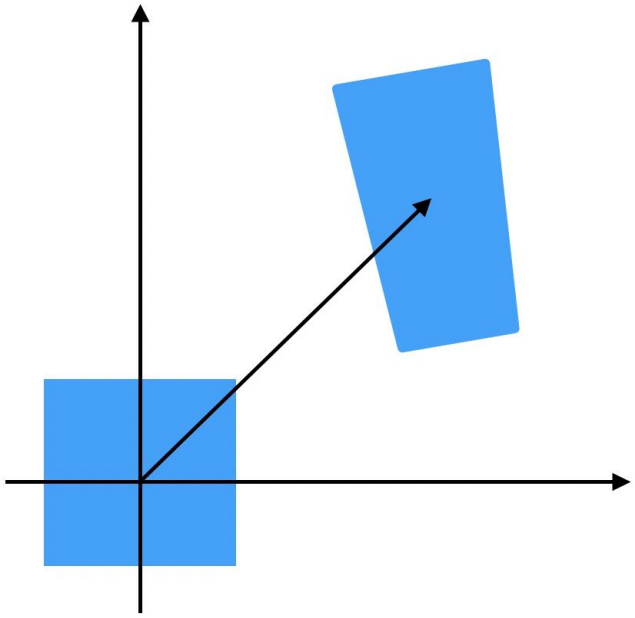

Figure 5: homograph transformation

2 Plane coordinate system & homogeneous coordinate system

Homogeneous coordinate plane $(x, y, w) \in \mathbb{P}^3$ is different from 3D coordinate system $(x, y, z) \in \mathbb{R}^3$, it only has 2 degrees of freedom.

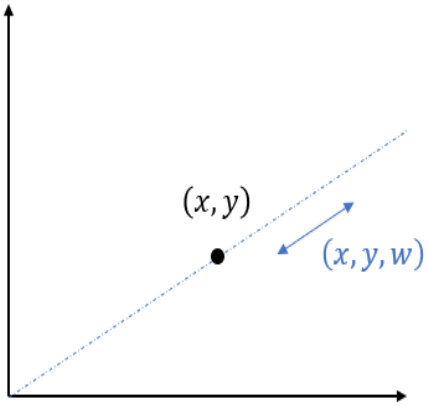

Figure 6: rotation

$$\begin{pmatrix} \frac{x}{w} \\ \frac{y}{w} \end{pmatrix} \iff \begin{pmatrix} x \\ y \\ w \end{pmatrix} \tag{12}$$

where $w > 0$ is scaling factor of x and y. When $w = 1$,

$$\begin{pmatrix} x \\ y \end{pmatrix} \iff \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{13}$$

and when $w = 0$, it's infinity.

$$\begin{pmatrix} \infty \\ \infty \end{pmatrix} \iff \begin{pmatrix} x \\ y \\ 0 \end{pmatrix} \tag{14}$$

you can clearly see that (x, y, w) slides on the ray from origin to (x, y).

3 Homograph Transformation

Wiki definition: In projective geometry, a homography is an isomorphism of projective spaces, induced by an isomorphism of the vector spaces from which the projective spaces derive. mathmatically speaking,

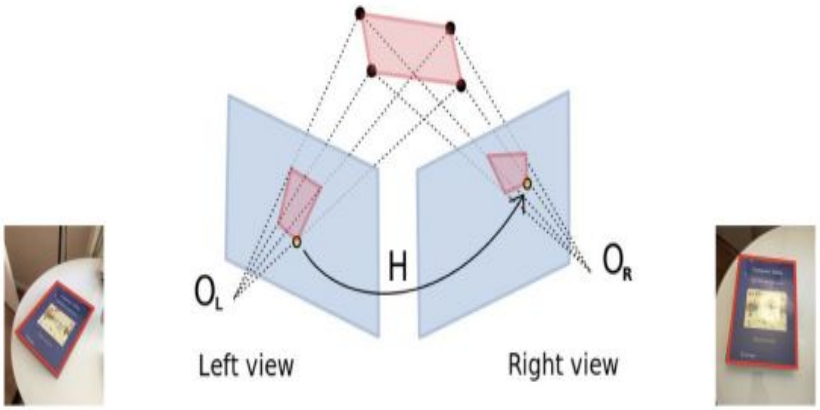

Figure 7: homograph

$$\begin{pmatrix} x_l \\ y_l \\ 1 \end{pmatrix} = \boldsymbol{H}_{3\times 3} \times \begin{pmatrix} x_r \\ y_r \\ 1 \end{pmatrix} \tag{15}$$

where (x_l, y_l) is the point in one picture and (x_r, y_r) is the point in another picture. For every matching point $(x_i, y_i) \to (x_i', y_i')$ we can have,

$$\begin{pmatrix} x_i' \\ y_i' \\ 1 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h23 \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11}x_i + h_{12}y_i + h_{13} \\ h_{21}x_i + h_{22}y_i + h_{23} \\ h_{31}x_i + h_{32}y_i + h_{33} \end{pmatrix} \tag{16}$$

If we translates it to homogeneous coordinate system, (16) can also be described as,

$$x_i' = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \tag{17}$$

$$y_i' = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \tag{18}$$

To see things more clearly, let's write it as form of $\boldsymbol{AX} = 0$

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & x_i' \\ 0 & 0 & 0 & x_i & y_i & 1 & -y_i'x_i & -y_i'y_i & -y_i' \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0 \tag{19}$$

That's to say, every pair of matching points will have 2 groups of equations.

3.1 8 degree of freedom

Here you will notice something interesting, that is homography matrix H has no difference with aH where $a \neq 0$, because,

$$x_i' = \frac{ah_{11}x_i + ah_{12}y_i + ah_{13}}{ah_{31}x_i + ah_{32}y_i + ah_{33}} = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \tag{20}$$

$$y_i' = \frac{ah_{21}x_i + ah_{22}y_i + ah_{23}}{ah_{31}x_i + ah_{32}y_i + ah_{33}} = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \tag{21}$$

And if we make $a = \frac{1}{h_{33}}$, we can have,

$$\boldsymbol{H}' = a\boldsymbol{H} = \begin{bmatrix} \frac{h_{11}}{h_{33}} & \frac{h_{12}}{h_{33}} & \frac{h_{13}}{h_{33}} \\ \frac{h_{21}}{h_{33}} & \frac{h_{22}}{h_{33}} & \frac{h_{23}}{h_{33}} \\ \frac{h_{31}}{h_{33}} & \frac{h_{32}}{h_{33}} & 1 \end{bmatrix} \tag{22}$$

here you will only get 8 equations and 8 parameters, hence if you have 4 pairs of matching points which is nonlinear, you can have unique solutions.

4 Python implementation

```
'''
chest robot project
Author: Xueren Ge
Time: Mar,12,2021
'''
import numpy as np
import json
import os
import cv2
import sys


def read_coordinate(filename):
    data = json.loads(filename)
    points = data['coordinates']
    return points

def H_transforamtion(filename, points):
```

```
    img = cv2.imread(filename)
    src_points = points
    dst_points = np.array([[70, 70], [950, 70], [950, 1060], [70, 1060]])
    H, _ = cv2.findHomography(src_points, dst_points)
    w = 1020
    h = 1130
    Perspective_img = cv2.warpPerspective(img, H, (w, h))
    return Perspective_img


if __name__ == '__main__':
    path = sys.argv[1]
    savepath = sys.argv[2]
    if not os.path.exists(savepath):
        os.makedirs(savepath)
    points = np.array([[201, 318], [913, 313], [1006, 1137], [119, 1134]])
    for root, dirs, files in os.walk(path):
        for file in files:
            img_path = os.path.join(root, file)
            Perspective_img = H_transforamtion(img_path, points)
            savepath = os.path.join(savepath, os.path.basename(root)+'.jpg')
            cv2.imwrite(savepath, Perspective_img)
```