ECE 549 Homework 2

Fangwen Wu (fwu11)
February 27, 2017

1 A featrue tracker

1.1 Keypoint Selection

Algorithm 1 Get keypoints using harris detector

Input: Input image, threshold for selecting keypoints τ.
Output: x and y coordinates of the keypoints
Steps:

1.Smooth the image with gaussian filter before taking the derivative.
2.Computet the x and y gradient value of the smoothed image I_x,I_y.
3.Calculate the second moment matrix

$$\begin{bmatrix} I_xI_x & I_xI_y \\ I_xI_y & I_yI_y \end{bmatrix}$$

4. Further gaussian filtering of the image.

$$\begin{bmatrix} g(I_xI_x) & g(I_xI_y) \\ g(I_xI_y) & g(I_yI_y) \end{bmatrix}$$

5. Construct the cornerness function

$$har = \det \begin{vmatrix} g(I_xI_x) & g(I_xI_y) \\ g(I_xI_y) & g(I_yI_y) \end{vmatrix} - \alpha[g(I_xI_x) + g(I_yI_y)]^2$$

6. Thresholding using harris criteria for selecting the key points

$har > \tau$

7. Non-maxima suppression of 5 by 5 window centered at each key point.

Here α is chosen to be 0.05 and threshold τ is 4×10^{-8}. The Gaussian filter that had been used to smooth the image is chosen to be the size of 7 by 7 with $\sigma = 2$.

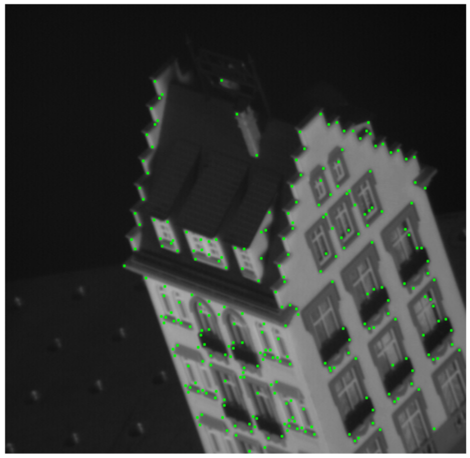

Figure 1: Keypoints detected by the harris detector

1.2 Tracking

Algorithm 2 Kanade-Lucas-Tomasi tracking procedure

Input: x and y coordinates of the keypoints.
Output: path of the key points over the 50 frames
for frame t and t+1 from 1 to 50
  for each key point x_i, y_i from 1 to N
    Consider a interpolated 15 by 15 window centered at each keypoint
    if the window has points outside the image
      stop tracking window
    else
      calculate the second moment matrix of the window
      While the refinement is not converged
        1.Calculate the temporal gradient I_t by taking the difference between the windows of the adjacent frames.
        2.Solve displacement u,v by
        $$\begin{bmatrix} \sum_W I_x I_x & \sum_W I_x I_y \\ \sum_W I_x I_y & \sum_W I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum_W I_x I_t \\ \sum_W I_y I_t \end{bmatrix}$$
        3. Updated predicted $(x'_{i+1}, y'_{i+1}) = (x'_i, y'_i) + (u, v)$
      end
    end
  end
end

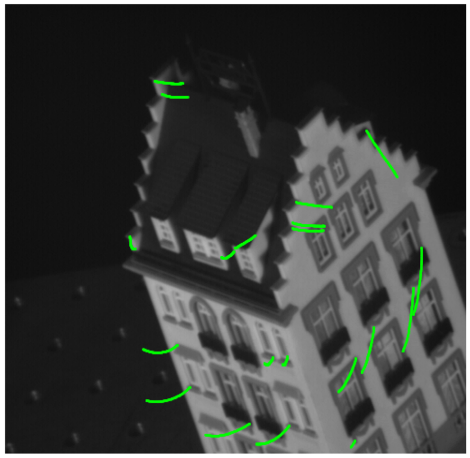

Figure 2: 20 Key points path over 50 frames

Figure 3: Points moved out of the frame during the tracking

2 Shape alignment

Summary of the algorithm with pseudocode

Algorithm 3 ICP Algorithm with affine transformation

Input: The original image and the image that needs to be aligned.
Output: Alignment result.
Steps:
1.Initialization: Calculate the mean, standard deviation of the boundary pixels in image 1 (Set1) and image 2 (Set2)
2.Align two sets of pixels by doing
$x_i' = \frac{x_i - \mu_{x_1}}{\sigma_{x_1}} \sigma_{x_2} + \mu_{x_2}$
Same for y coordinates.
While the refinement is not converged > threshold (0.0000001)
3. Assign each point in the Set1 to its nearest neighbor in Set2
4. Set up the affine transformation model and calculate the parameters
5. Solve the affine transformation matrix by least squares method.
6. Transform the points in Set1 using the affine transformation matrix
7. Calculate the difference between the transformed points and the points that were transformed.
end

This algorithm uses Iterative Closest Points Algorithm to iteratively estimate the transform between two sets of points in order to achieve the shape alignment task. The transformation model used is the affine transform which combines the translation, scaling, rotation and shearing that give 6 degree of freedom so it is a suitable transformation model in this question. The affine matrix is solved by least squares method.

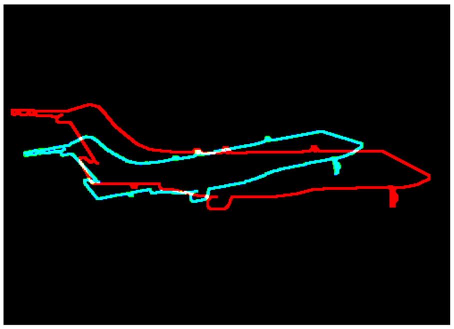

Figure 4: align object2 to object2t

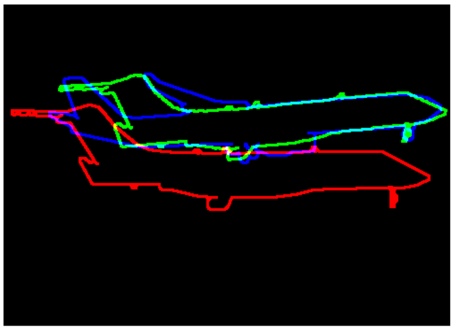

Figure 5: align object2 to object1

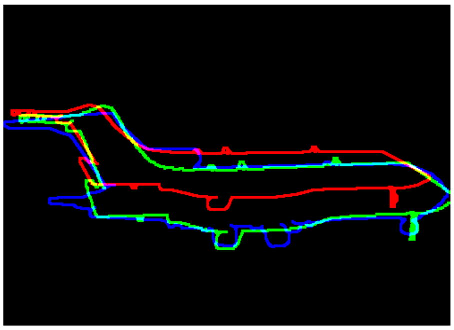

Figure 6: align object2 to object3

Image	Error	Runtime
object2t	0.26	0.58s
object1	5.39	1.60s
object3	4.52	1.61s

3 Object instance recogition

3.1 Keypoint matching

Algorithm 4 key point matching

Input: A key point descriptor g from one image, a set of keypoint descriptors $f_1 \dots f_n$ from a second image.
Output: The keypoint matches g.
Steps:
for $f_i = f_1 : f_n$
Calculate the euclidean distance between g and each individual descriptor f_i $||g - f_i||$
end
Find two nearest neighbors f_m and f_n of g with distances d_m and d_n
f_m matches g if $\frac{d_m}{d_n} < 0.7$ where 0.7 is the optimized threshold.

3.2 Object alignment

Known: (x_1, y_1, w_1, h_1), $(u_1, v_1, s_1, \theta_1)$, $(u_2, v_2, s_2, \theta_2)$.
We wish to transform (x_1, y_1) to (x_2, y_2) and find w_2, h_2, o_2.
In order to determine (x_2, y_2) and we already know keypoints pairs (u_1, v_1) and (u_2, v_2) they are matched. We start with (x_1, y_1) and setup the transformation between them by first subtracting bounding box center coordinates $\begin{bmatrix} u_1 - x_1 \\ v_1 - y_1 \end{bmatrix}$, then we multiply rotation matrix R $\begin{bmatrix} cos(o_2) & -sin(o_2) \\ sin(o_2) & cos(o_2) \end{bmatrix}$ and scaling matrix $\begin{bmatrix} \frac{s_2}{s_1} & 0 \\ 0 & \frac{s_2}{s_1} \end{bmatrix}$
Now we need to do subtraction from (u, v) to center and we put all together we get the center of the bounding box in image 2

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} u_2 \\ v_2 \end{bmatrix} + \begin{bmatrix} \frac{s_2}{s_1} & 0 \\ 0 & \frac{s_2}{s_1} \end{bmatrix} \begin{bmatrix} cos(o_2) & -sin(o_2) \\ sin(o_2) & cos(o_2) \end{bmatrix} \begin{bmatrix} x_1 - u_1 \\ y_1 - v_1 \end{bmatrix}$$

Notice that the rotation angle $o_2 = \theta_2 - \theta_1$.
$w_2 = s_2 \frac{w_1}{s_1}$, $h_2 = s_2 \frac{h_1}{s_1}$ by scaling.