CS696, Applied Computer Vision

Homework Assignment 6

Instructor: Xiaobai Liu

**How to submit**

* Submit your source codes and writeup through SDSU Blackboard, before 11:59pm, April 23, 2017
* Three submissions are allowed.

1. **Overview**

The goal of this mini-project is to track an object of interest (e.g. a mug) in video using a feature-based matching approach. The recommended pipeline is a simplification of the popular KLT method. We consider moving cameras in this setting whereas the developed techniques can be used for videos by static cameras. You will work on two video sequences whose first images are shown in Figure 1.

Figure.1 left: sequence 1(seq1); right: sequence 2 (seq2).

For the main assignment, you should complete the starter codes to generate demo videos and calculate numeric results. In the writeup, you are expected to analyze your results.

You can optionally implement advanced recommendations (see the later part of this instruction) or analyze this pipeline on your own video sequences.

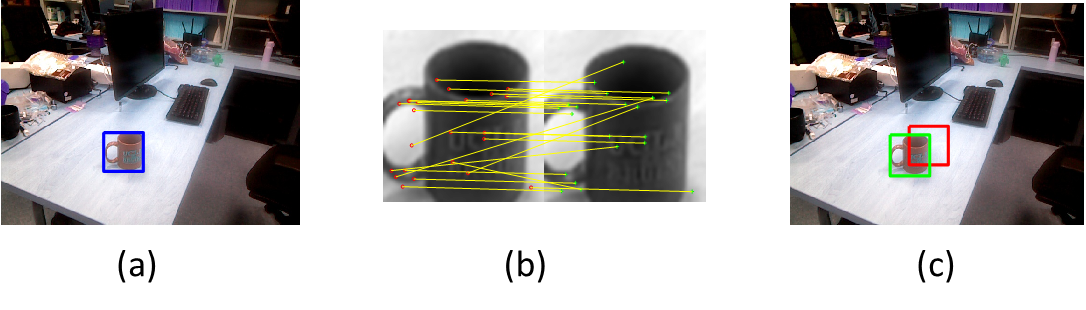


Figure 2 exemplar results over Seq1 at time t. (a): the image at the previous time, i.e. t-1, overlaid with the detected box (blue); (b): matched points between the two images cropped from the images t and t-1, according the detected box at t-1; (c) the image t overlaid with the groundtruth box (green) and predicted box (red).

1. **Details**

For this project, we will need to predict the object box in video (or image sequence). The initial box at the first image is provided. For every image t, we use the following tracking pipeline to predict the object box

* According to the object box at time t-1 (either detected or initialized), crop object images from the images t-1 and t, respectively
* Detect interests points and compute correspondences between the two cropped images;
* Estimate translation displacement between two images from these correspondences.

Figure 2 illustrates an exemplar result of the suggested tracking method at a time. There are numerous papers in the computer vision literature addressing object tracking. You are encouraged to experiment with more sophisticated algorithms!

In the starter codes, i.e., proj6.m, we breakdown the above pipeline into 6 steps. We will go through each of them in the following. You are only required to implement two of them.

The above tracking method will process the sequence of images one by one. To track the object of interest at time t,

**Step-1: Crop images of objects**

According to the bounding-box at time t-1 (known), crop object images from the images t and t-1. Each box is described as a rectangle [minx, miny, width, height], where (minx, miny) is the top-left corner of the box.

In this step, we assume the object motion is small over consecutive frames and there are considerable overlapping between the two boxes at the images t-1 and t. Therefore, we expect the box at time-1 can still cover majority of the object at time t.

[IMPORTANT] To crop images of objects, you might use either the box in the first image (method 1) or the box at the previous time t-1 (method 2). Both are provided in the starter codes. You are required to report the performance of these two methods, and discuss their Cons&Pros.

**Step 2: Detect interest points**

This step is used to detect Interest points (Harris Corners) for the two cropped images. You are encouraged to use your own codes.

**Step 3: Extract feature descriptors for every interest point and find the correspondences between two cropped images.**

Again, you are encouraged to evaluate the performance of different descriptors, e.g., template features, histogram, or template+histogram.

**Step 4: Estimate translation displacement (your own codes)**

Given a collection of correspondences, you are requested to implement the least square method to solve the optimal translation displacement.

Remember that you need to convert the problem to be the standard form: Ax=b. Then you can solve x in a closed form: .

You are encouraged to implement the least square method in the RANSAC framework.

You are strongly encouraged to consider more freedom in deformation, e.g. scale, shear, rotation, or affine. See Lecture\_10\_ransac\_II.

**Step 5: calculate average tracking error (your own codes)**

For every image in the sequence, calculate the Euclidean distance between the estimated object box (top-left corner) and the ground-truth object box. Calculate average errors over the number of images.

**Step 6:** **Generating video demos**

In proj6.m, Simply set bOutputVideo=true, then the starter codes will generate a video (\*\_results.avi) according to your tracking results. Put this demo video in your homepage if you want.

1. **Using the starter code and videos (proj6.m)**

*[IMPORTANT] You need to install computer vision toolbox in MATLAB*

The top-level proj6.m script provided in the starter code includes the steps of image cropping (step 1), detecting interest points (step 2), matching (step 3) and generating video demos (step 6). You are required to implement step 4 and step 5.

For step 4, the program in proj6.m uses a few lines of place-holder codes to simply pick up one of the matched pair and use it to estimate translation vector.

For step 5, simply add your own codes to calculate the differences (Euclidean distances) between the estimated box (rectCur) and groundtruth box (rectGD). This is only one line of code.

If you run proj6.m without any modification, you might be able to run proj6.m through for the first few frames. It will eventually fail, though. This is reasonable since, without proper fitting algorithm, the errors are accumulated over time and the trajectories will drift.

We provide two video sequences each with about 50 images. You are required to report results on both sequences. You are also strongly encouraged to run your algorithms on your own video sequence, e.g., shooting videos using your cellphone. If you choose to do so, you might like to use main\_getAnno.m to create the groundtruth boxes for your videos, although it is not required. For a short video sequence of 50 images, it might cost 10 minutes to get the complete annotations.

1. **Writeup**

In the report you will describe your algorithm and any decisions you made to write your algorithm a particular way. Then you will show and discuss the results of your algorithm.

In the case of this project, show how well your method can tracking the object. Include both qualitative results (visualized figures), and quantitative results (average tracking errors).

A good writeup will assess how important various design decisions were. E.g. by using RANSAC, I went from 10 pixels error to 5 pixel error.

You should clearly demonstrate how your additions changed the behavior on particular images.