CSE 152 Introduction to Computer Vision
Homework 2

Instructions:

- Total points: 100
- **Due: 11:59 pm, Monday, Nov 12, 2018**
- For the implementation part, we've provided you with a number of functions and scripts in the hopes of alleviating some tedious or error-prone sections of the implementation. **Please implement your part under the comment line "Write your code here", and stick to the headers, variable names, and file conventions provided.**
- For the theory problems, please justify your solutions by necessary derivations or explanations.
- To speed up the running time, try to optimize your implementation using vector-based operation and avoid to loop for every pixel.
- Please pack your codes and write-up into a single file named **YourPID_hw2.zip** and submit it to descartes.ucsd.edu. **And also submit your write-up (pdf) to Gradescope.**
- A complete submission consists of the following files:
  - testCarSequence.m
  - iterative_KLT.m
  - p1_3.m
  - carPosition.mat
  - A write-up (.pdf format)

  Make sure that we can run your code without any modification, otherwise you are at risk of losing points.
- **Start early!** Please familiar yourself with MATLAB and allow enough time for debugging.

1 KLT Tracker

In this section you will track a specific object in an image sequence (`carSequence.mat`) by template tracking.

The first groundbreaking work on template tracking was the KLT tracker. It basically assumes that the template undergoes constant motion in a small region. The KLT Tracker works on two frames at a time, and does not assume any statistical motion model throughout the sequence. The algorithm estimates the deformations between two image frames under the assumption that the intensity of the objects has not changed significantly between the two frames.

In this homework, we assume the motion is translation (the matrix W we learned from class is a translation matrix here). Starting with a rectangle R_t on frame I_t, the KLT Tracker aims to move it by an offset (u, v) to obtain another rectangle R_{t+1} on frame I_{t+1}, so that the pixel squared difference in the two rectangles is minimized:

$$\min_{u,v} J(u,v) = \sum_{(x,y)\in R_t} (I_{t+1}(x+u, y+v) - I_t(x,y))^2 \tag{1}$$

1.1 Preliminary [14 points]

Starting with an initial guess of (u, v)(usually (0,0)), we can compute the optimal (u^*, v^*) iteratively. In each iteration, the objective function is locally linearized by first-order Taylor expansion and optimized by solving a linear system that has the form $A\Delta p = b$, where $\Delta p = (u, v)^T$ is the template offset.

- What is $A^T A$?
- What conditions must $A^T A$ meet so that the template offset can be calculated reliably?

1.2 Iterative KLT Tracker [50 points]

Complete the script `iterative_KLT.m` , which computes the optimal local motion from frame I_t to frame I_{t+1} that minimizes Eq. 1. Please read the script file for details. **Note that moving the template by u and v will lead to fractional coordinates of the pixels. To deal with fractional movement of the template, you will need to interpolate the image using the MATLAB function `interp2`**. You will also need to iterate the estimation until the change in (u, v) is below a threshold. You are allowed to use MATLAB function `gradient`.

1.3 Testing [25 points]

Complete the script `testCarSequence.m` that loads the video sequence (`carSequence.mat`) and run the KLT Tracker to track the speeding car. The rectangle in the first frame is $[x1, y1, x2, y2] = [328, 213, 419, 265]$. In other words, the rectangle starts from (328, 213) (row 213 and column 328 in the image) and ends at (419, 265).

Save your results as file `carPosition.mat` which has top-left and bottom-right corners of your tracked object for us to evaluate your tracking result. The `carPosition.mat` file should contain a variable called `box` which is a $n \times 4$ matrix; n is the number of frames and each row in the matrix contains 4 numbers: $[x1, y1, x2, y2]$, where indicates the coordinates of top-left and bottom-right corners to the object you tracked.

If you complete this script correctly, it will play a video with a rectangle tracking the car.

In your write-up, print the tracking result (image + rectangle) at frame 1, frame 20, frame 50 and frame 100. You can use `p1_3.m` to produce the figures. And your visualization result should be like figure 1.

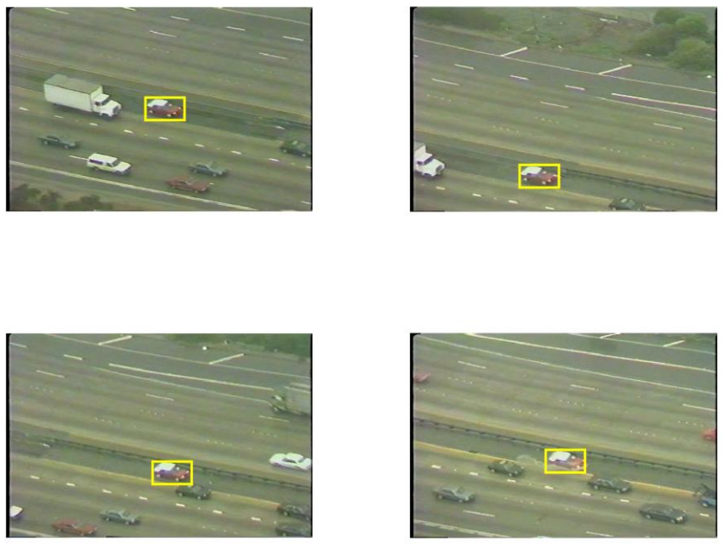

Figure 1: Iterative KLT tracking. The bounding box of the car in the first frame is given, and you need to implement the KLT tracker to locate the car's position in the rest of the frames like this.

1.4 Failure analysis [10 points]

In your write-up, give a short analysis on the possible reasons the tracker could fail; give a possible solution to fix some of these problems.

2 Questionnaire [1 point]

Approximately how many hours do you spend on this homework? Please answer it in your write-up.