HW4 – submission 26.1.19 23:55

Guide lines

1. Include all your personal details including name, id, and e-mail address.
2. You should submit all function and script files written in MATLAB or Python. Your code should be well documented and clear. The code should run from **any** computer and include all path definitions (You should take care of this in the code).
3. Please divide the code by questions.
4. Final report – should include explanations on the implementation and the execution, answers to the questions, results, conclusions and visual results. Do elaborate on all parts of the algorithms/solution. **Please submit a PDF file and not a DOC file.**
5. Please post question regarding this HW on the facebook group: https://www.facebook.com/groups/294298727746314/
6. The grades are highly depended upon the analysis depth of the report.
7. HW can be submitted in pairs.
8. Eventually submit one compressed file including the code + images PDF.

Good luck!

Question 1 – Optical flow:

In this assignment, you are given a short video "seq.gif". Your goal is to estimate the optical flow between every two neighboring frames, using the Lukas-Kanade optical flow algorithm. Remember that for Lukas-Kanade, for each flow vector that you estimate, you will be choosing a region over which to analyze the two frames.

For this assignment, use regions of size NxN pixels which are **not overlapping**, where N=16.

(Since the input frames are 512 by 512, you should have an array of 32 by 32 optical flow vectors at the end of your procedure.)

The Lukas-Kanade algorithm works by trying to find an optical flow vector with the components (u,v) for each region that minimizes the error of the following series of equations:

$$\begin{bmatrix} I_{x1} & & I_{y1} \\ I_{x2} & & I_{y2} \\ & \text{M} & \\ I_{xN^2} & & I_{yN^2} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -\begin{bmatrix} I_{t1} \\ I_{t2} \\ \text{M} \\ I_{tN^2} \end{bmatrix}$$

Recall that I_{x1} is an estimation of the image derivative in the horizontal direction at the first pixel in the image region (hence the superscript "1"). For the purposes of this assignment you can just take that to be the difference between that first pixel and the one immediately to its right. (NOTE: You will have to do something special for the image regions that are on the rightmost edge of the image, since those regions will not have a pixel immediately to the right of the rightmost pixel. Do whatever you like to deal with this. It won't have a major impact on the results.) The last index N^2 is the size of the region (16x16 here).

This equation can be rewritten in matrix form as

$$A\vec{u} = b$$

Where

$$A = \begin{bmatrix} I_{x1} & & I_{y1} \\ I_{x2} & & I_{y2} \\ & \text{M} & \\ I_{x255} & & I_{y255} \end{bmatrix}, \quad \vec{\mathbf{u}} = \begin{bmatrix} u \\ v \end{bmatrix}, \quad b = -\begin{bmatrix} I_{t1} \\ I_{t2} \\ \text{M} \\ I_{t255} \end{bmatrix}$$

And the solution can be found using least square:

$$\vec{u} = \left(A^T A\right)^{-1} A^T b$$

For all the neighboring pairs of frames:

1. For each region in the image:
   a. Compute the matrix $A^T A$ and the vector b
   b. Compute the eigen-values of $A^T A$
   c. If the smallest eigan-value is larger than a threshold t=1, continue to (d), otherwise move to the next region.
   d. Estimate (u,v) for the current image region by solving $\vec{u} = \left(A^T A\right)^{-1} A^T b$.
2. From the estimations of (u,v) at each region, display the flow over the full frame using `quiver()`. This function plots a set of two-dimensional vectors (the flow field in our case) as arrows. **Please plot the flow on top of the frame**, using `hold all`**.**
3. Repeat for t=0.1
4. Repeat for N = 8, t=1,0.1
5. Describe in general effect of changing N and t.
   (There is no need to present the flow of all the frames. Select only a few examples for each case)

   Note: it is important to convert the images into double (use `im2double`**).**