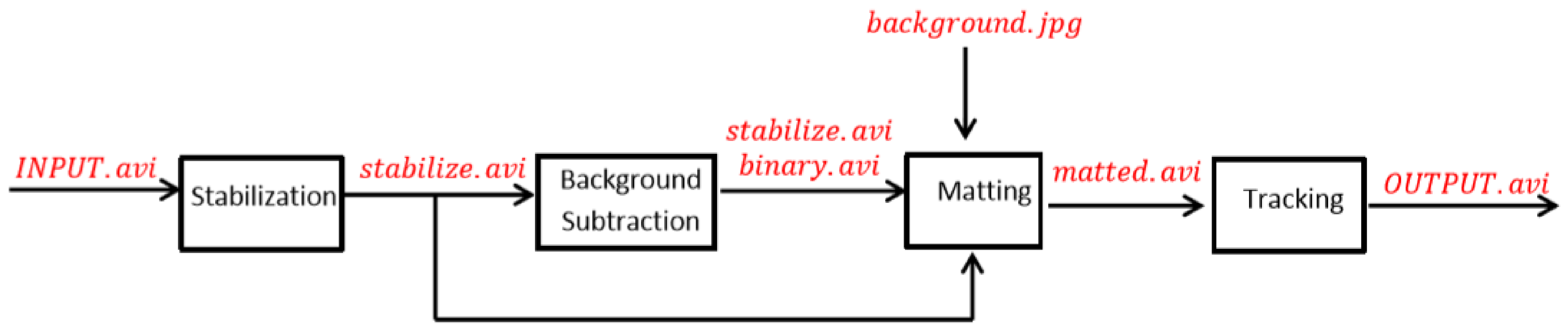
Video Processing Course
Final Project

Tips and solution proposal

Final Project Flow Diagram



Final Project - Reminder

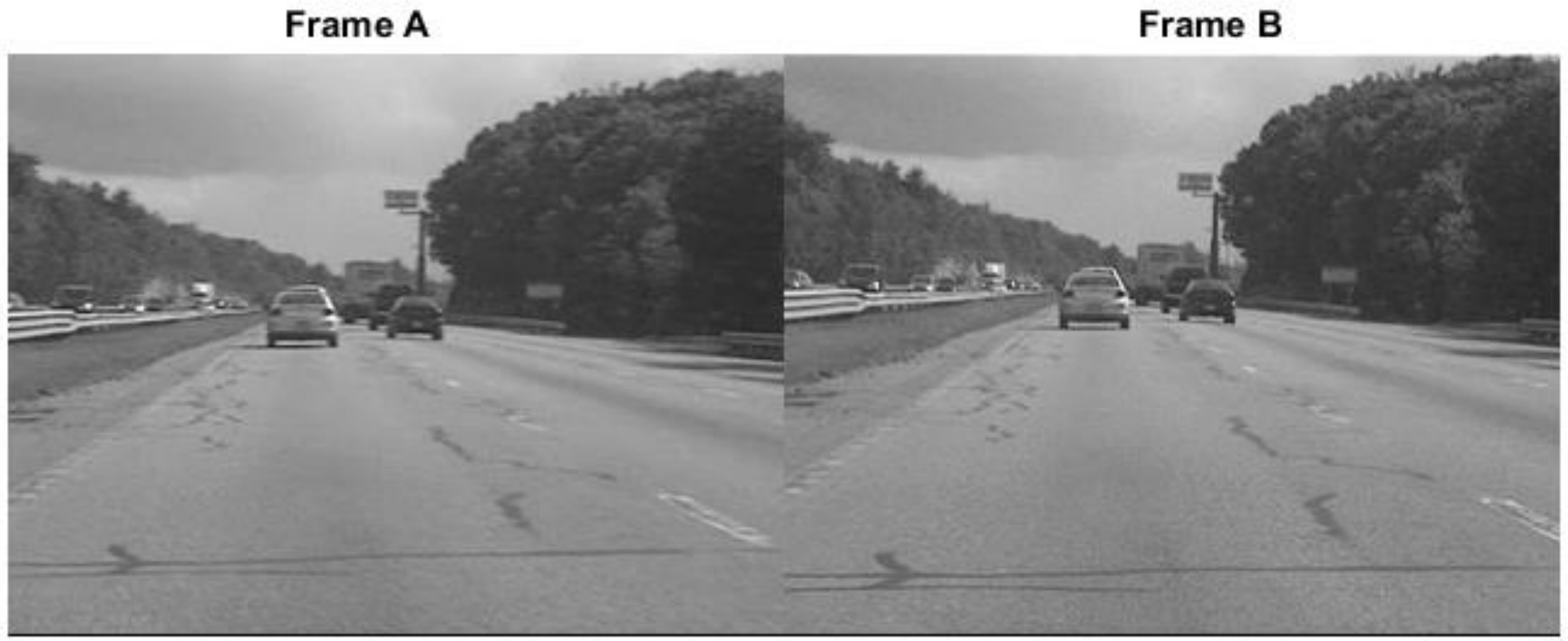
- You may use your creativity in solving this project as long as you don’t use someone else’s code. You may use standard python functions etc. as was allowed in the homework assignments.
- You may also use additional python libraries (but ask for my approval first). If you choose to use a function that is too specific (i.e. a function that includes the required steps to stabilize a given frame), add an explanation of its main functionality as well as its source code to your documentation (if your explanation won’t be detailed enough you will lose points).
- If you implement a solution not studied in class you *have* to add a short explanation about the operation just so I know you understood what you did.

There is no one definite way to solve this project and you will be graded based on the quality and performance of your algorithm and presentation.

In the following slides I will present a solution proposal for the different steps in the project.
It is only a suggestion, you may solve it as you please!

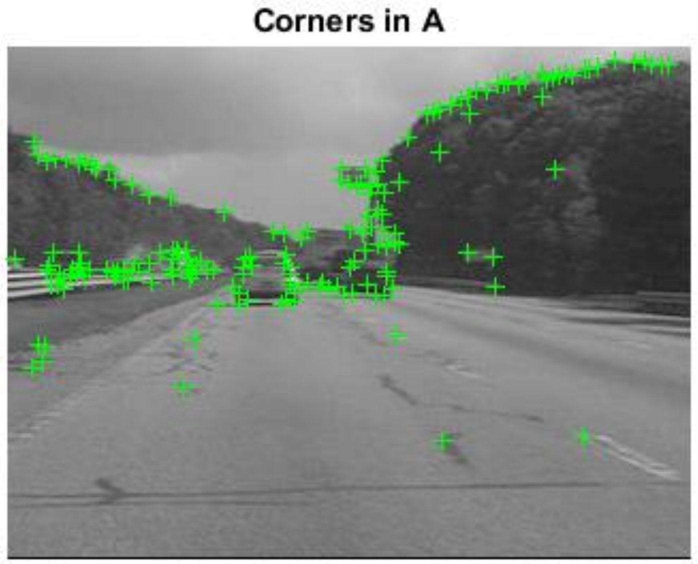
Solution Proposal - Video stabilization

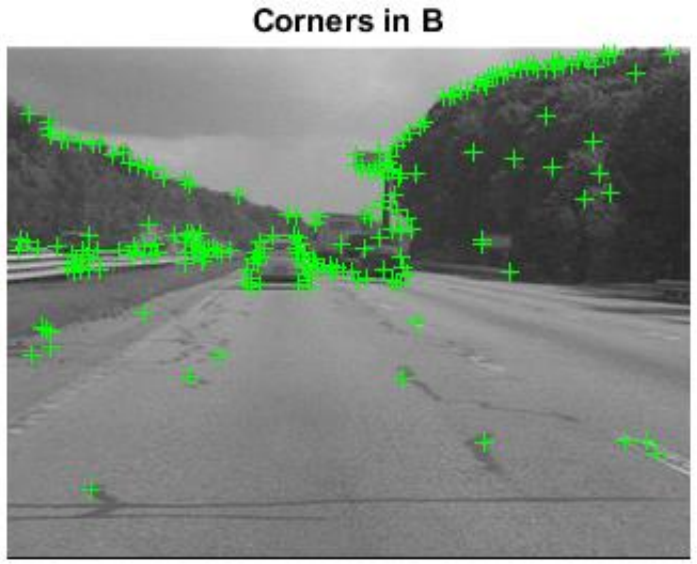
Example on two frames:
We have two frames of a shaky video. We want to warp frame B to the coordinates of frame A.



Solution Proposal - Video stabilization

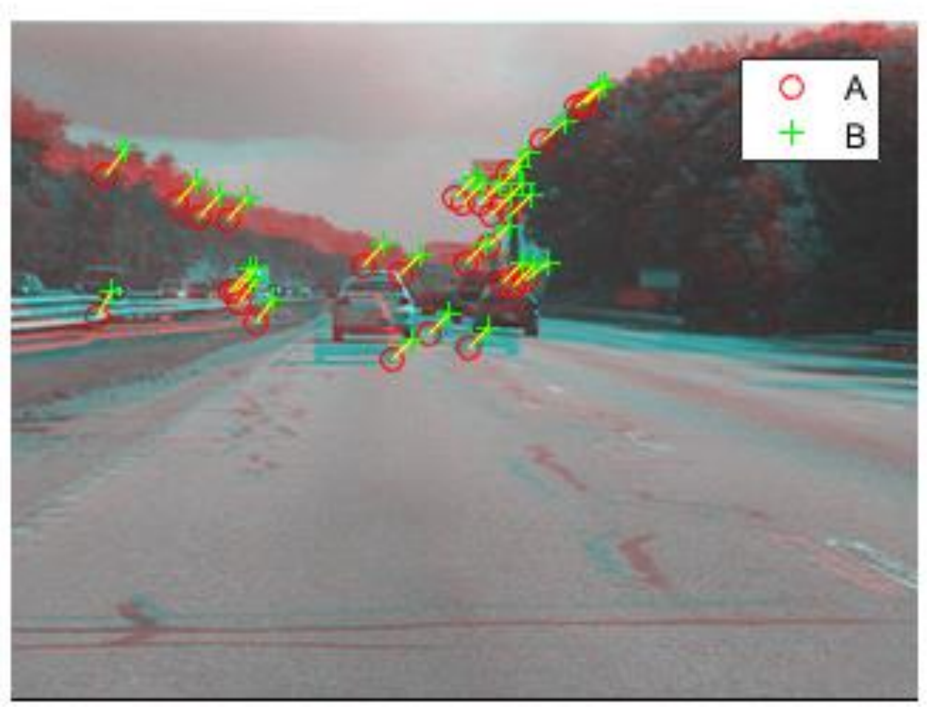
1. Find interest points in each frame





Solution Proposal - Video stabilization

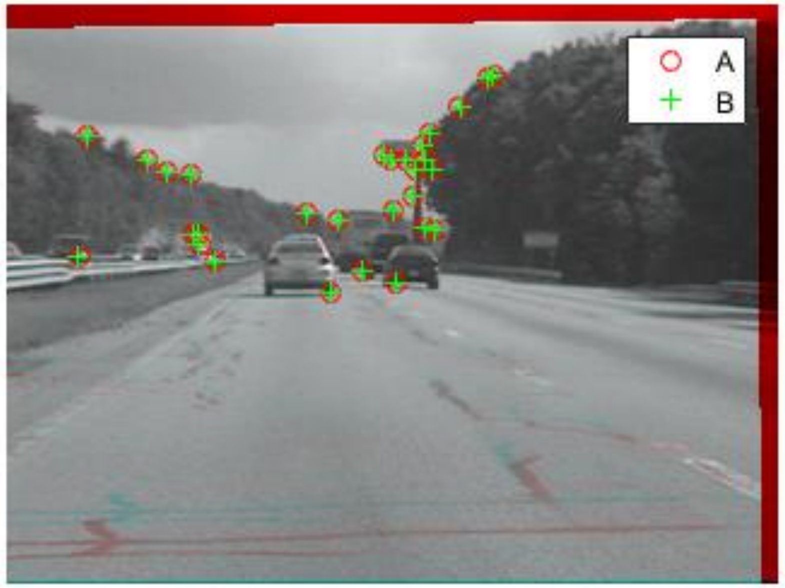
1. Find interest points in each frame
2. Extract descriptors for all points
3. Find pairs of matching points between frames



Solution Proposal - Video stabilization

1. Find interest points in each frame
2. Extract descriptors for all points
3. Find pairs of matching points between frames
4. Use random pairs of matching points to estimate various transformations (RANSAC)
5. Choose best transformation and warp the distorted frame

Example: https://www.learnopencv.com/video-stabilization-using-point-feature-matching-in-opencv/

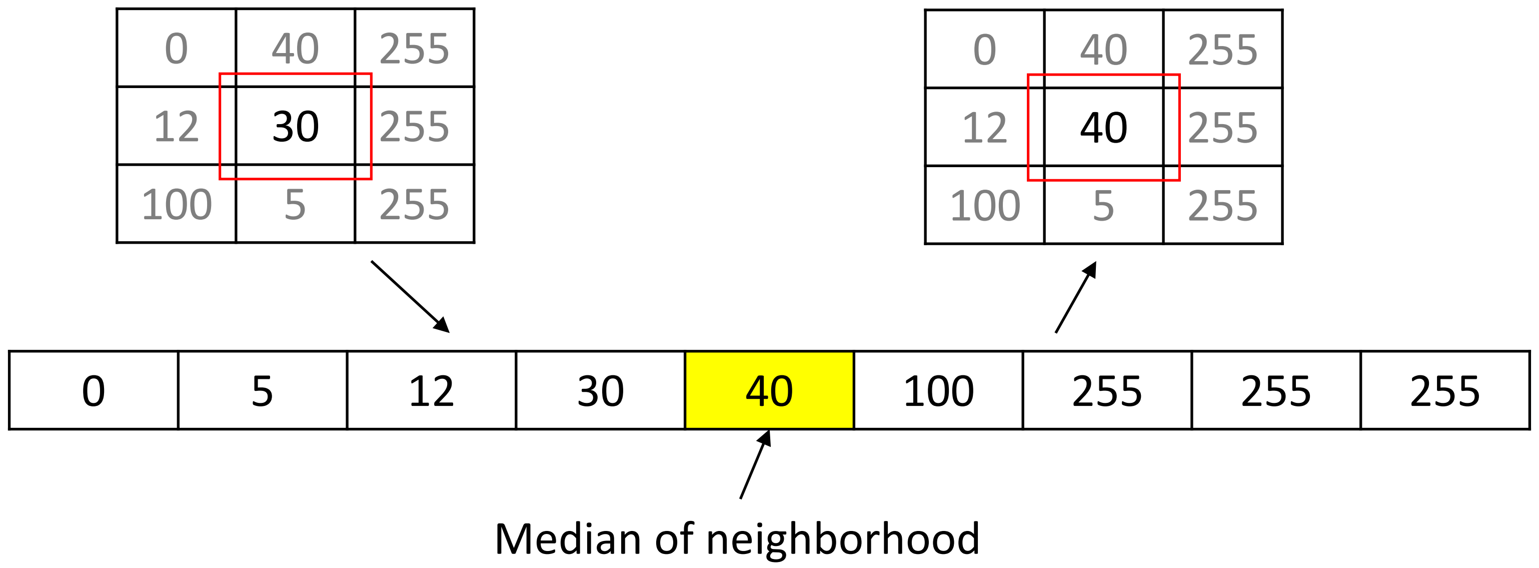


Useful vision functions:
- cv2.estimateRigidTransform
- cv2.goodFeaturesToTrack
- cv2.calcOpticalFlowPyrLK

Solution Proposal – Background Subtraction

Median Filter - Quick Reminder

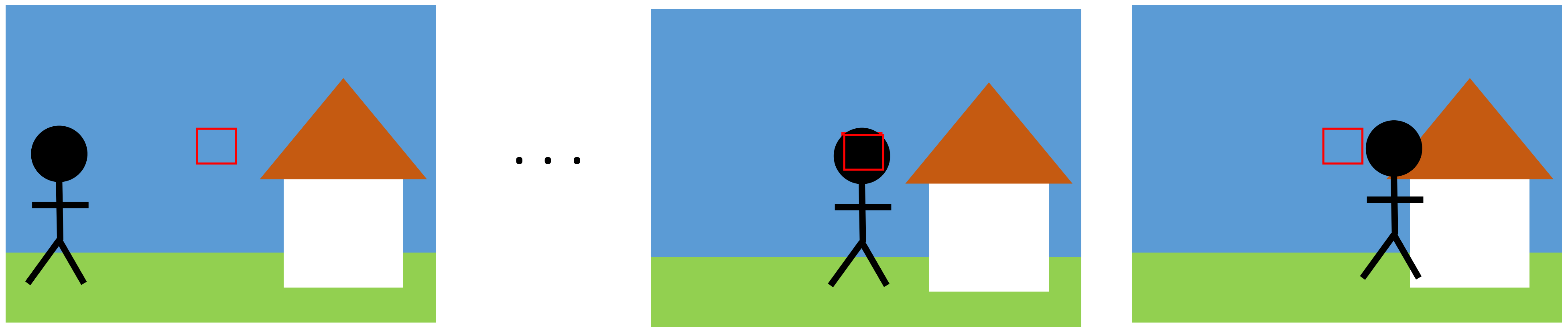
The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the median of neighboring entries.



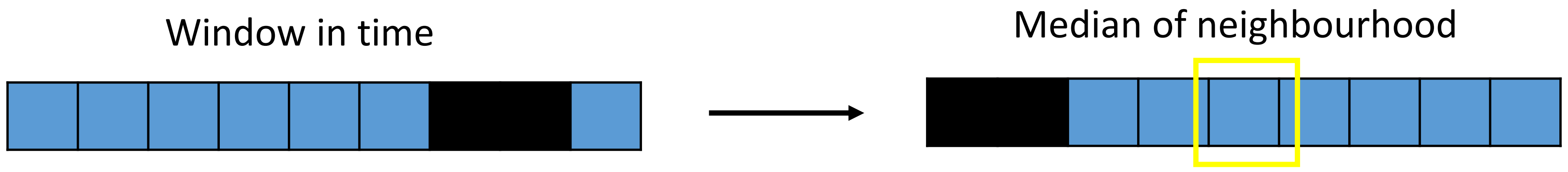
Solution Proposal – Background Subtraction

An idea – in order to perform background subtraction we can use a median filter, where the pixels neighbourhood is a window in time.

Say we are interested in the background value of the pixel that is marked in red.



Frame 1 Frame 8 Frame 9

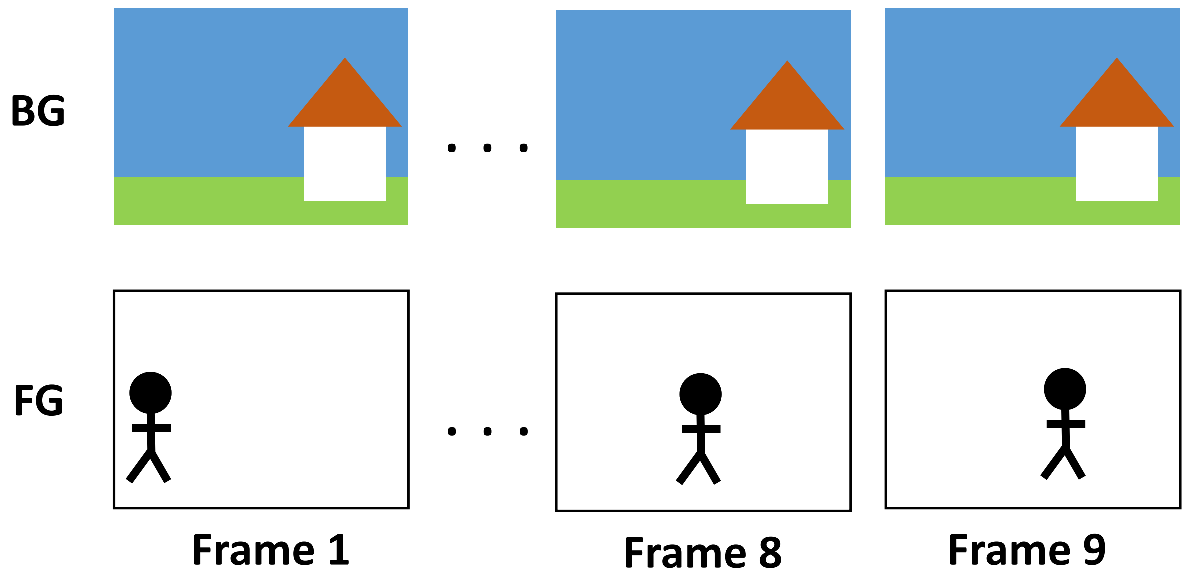


(say the person walked through the marked pixel during frames 7,8)

Solution Proposal – Background Subtraction

If we use a large enough window in time, the walking person presence in the pixels will be negligible.
Meaning – we will be left with the background (assuming the video is stabilized).

After we have the background, we can use it in order to find the walking figure (foreground).



Solution Proposal – Matting & Tracking

Matting

Use the method you learned in class.

- Check out the file "Implementation of Video Colorization and Matting" for instructions (in moodle)
- Use kde function: https://jakevdp.github.io/blog/2013/12/01/kernel-density-estimation/

Tracking

- You can use the particle filter that you implemented in HW3
- Check out CAMShift algorithm: https://docs.opencv.org/3.1.0/db/df8/tutorial_py_meanshift.html

General advice

- Don't try your algorithm on 500 video frames before you see it works on a simple synthetic data set. (take a still image and transform and crop it in several ways: imrotate, imresize, a bit of translation etc. so you create a few made up 'frames' in a completely static sequence)

- Search for ideas in the internet, in python tutorials, check the articles that were taught in class.

- Try to keep things simple!

Good luck and have fun! ☺