פרויקט מסכם לקורס

נא לקרא בעיון את כל הנהלים להגשת הפתרונות:

1. **נא להגיש את הפתרון בהתאם להוראות ההגשה המפורטות בהמשך (בתיבת ההגשה ב-Moodle) עד לתאריך ה- 30/06/2020. הארכות יינתנו לפי כללי הפקולטה (מקרים חריגים בלבד).**

2. הפרויקט כתוב באנגלית למניעת אי-הבנות בדרישות.
אתם יכולים לענות בעברית או באנגלית כרצונכם.

3. אם קיימות שאלות בנוגע לפרויקט **נא לרשום בפורום הקורס**.

4. התרגיל יוגש בזוגות בלבד בדומה להגשה של תרגילי הבית.

5. ניקוד יופחת על חוסר סדר, פתרון לא מתועד, קוד שלא רץ וכיו"ב בהתאם לשיקול דעת הבודק.

בהצלחה! ☺

Prerequisites:

1. Use the supplied shaky video and the background image (I may change the background image when testing your code). This image will be the new background for your video (see step 3 – 'matting'). **The image can be of any size (you need to resize the background image inside the code to match the input video frame size), no need to keep the same aspect ratio.**

The following outlines the general tasks you will have to implement. You may use your creativity in solving them **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 (according to the requirements file).

You may also use additional python libraries, but ask for my approval first (write a message **on the forum** and explain what do you need and if you haven't found it in the approved packages). 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 an explanation about the operation just so I know you understood what you did. No need to explain algorithms that we studied in this course (you can mention them without explaining), but if you did some changes to them, explain them.

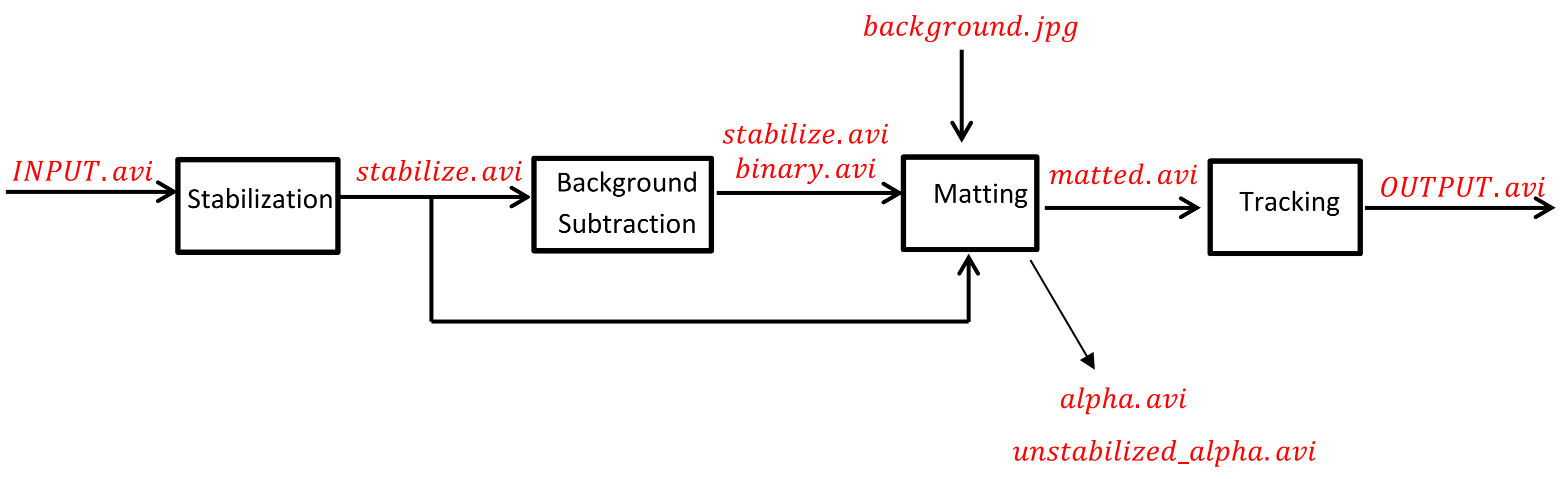
There is no one definite way to solve this project and you will be graded based on the quality and performance of your algorithm and presentation.

Main steps:

ALL STEPS REQUIRE COLOR FOOTAGE PROCESSING AS INPUTS AND OUTPUTS – NO GRAYSCALE (the binary.avi video may be excluded from this demand)! WITHIN THE FUNCTIONS YOU MAY WORK ON ANY CHANNELS YOU WISH.

1. Stabilize the video in 'INPUT'. Save the stabilized video as 'stabilized.avi'. Try not to damage the quality of the image (like blur) as it can cause you problems in the matting stage.
2. Extract the walking person from 'stabilized.avi' using background subtraction. This video will only contain the walking person. Save the extracted person as 'extracted.avi'. (save also a binary mask of the walking person as 'binary.avi').
3. Using image matting, place the walking person from 'extracted.avi' in 'background.jpg'. Save this video as 'matted.avi'. The new video file will have a walking person with a static background. You will also save a video named 'alpha.avi' which is the alpha (you will calculate) of each frame in the matted video (values in range of [0,1] where 0 means background and 1 means foreground and different values for pixels near the boundaries). You will also need to un-stabilize the alpha video (so it will be shaky exactly like the input video) and save it as 'unstabilized_alpha.avi' (**same size as input video**).
4. Automatically/Manually select the walking person in 'matted.avi' and track him throughout the video showing a rectangle around him in all frames.
   Save the video with the rectangle highlight as 'OUTPUT.avi'.

Flow diagram:



General guidelines:

You may write a simple GUI used for presenting your results; it should be self-contained, have basic functionality of loading the files, computing each step, showing the current video after each step (can open in another window), manually selecting the person for tracking (and also have a default option toggled at the beginning of that step – so I can see your selection) etc.

The GUI name must be runme.py – I will use this GUI to test your algorithm. You get to choose the GUI layout as long as it has all the above mentioned functionality. I suggest you start by building a simple GUI to present simple video files and then add to it. Don't try to start out with a GUI that does everything.

Important Note: Make sure you have a title on screen (as well as an explanation in the submitted document) so I know what tool I'm using and how ('click on the top left corner of the target and then double-click on the bottom right corner').

You are required to add an indication of current function processing in percentage so we know the computer isn't stuck (print the functions name, the current step and progress percentage). **Neglecting to have such an indicator = minus 5 points**. Don't keep debug prints you used to develop your algorithm.

As written above, you don't have to create a GUI (it's optional). If you don't – make a main function to call all other functions in sequence. **Call this function runme**.

It should allow me to run the functions in any order I want as long as the input files are available. For example – If I test the matting function only, the function will load the extracted.avi + stabilized.avi + binary.avi video files from Output folder and Background.jpg from Input folder. **The tracking step can take as input the matted.avi video file only and not the extracted.avi etc.**

It is recommended to output all parameters to a config file (file names, thresholds…) and use the config file across different functions, so it will be easy for you and me to make changes. Read about configuration files in Python.

About automatic vs. manual operations in the algorithm (matting/tracking):

You may choose how to implement things. For example – you may either choose automatic scribble points (matting process) or manual ones (the user selects them only in the first frame- use discretion in the number of pixels you use as supervision).

The tracking should have an automatic selection for your files with an option to manually override the selection (for my testing files).

But – as I will test your code, I need an *easy* way to recreate your results (in case you use manual selection), so you should put a default values (scribbles/bounding box). You might lose points if it doesn't work well.

Runtime: Add a final txt containing the runtime of each important block (stabilization, subtraction, matting, tracking) and the total runtime of the entire process. As some of you will include a manual process, exclude the manual process from the logging. You can make your life easier if you will work with a single log file (and each function can write into this log). You can use **logging** package, there are many examples online.

What you submit:

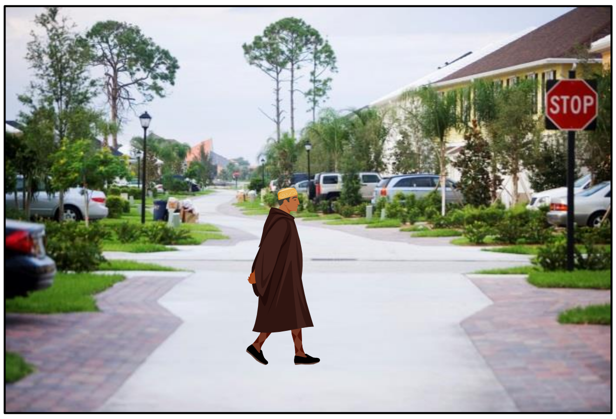
1. Powerpoint presentation or a word/pdf document containing a detailed explanation of your solution for each step. Do not underestimate the presentation; it is my way to understand your solution and will be graded separately as part of the final grade.
2. Complete Python files (containing your implementation) + input video file + new background image + output video files (7) + runtime log file.
3. Screen recording of running your code (from start to finish, until the output is created).
4. Place all the files in relative folders so the project runs perfectly using the GUI or by running the file runme.py as explained above.
   Do not use any direct path folders in your code ('cd c:\video_project\' etc. is unacceptable). Use relative paths.
5. The root folder - FinalProject_ID1_ID2.zip (which would be submitted via the Moodle submission box) will contain folders, each containing the files written in the brackets:
   - **Document\** (containing a Powerpoint presentation **or** a Word/PDF document)
   - **CODE\** (containing runme + all Python files)
   - **Input\** (containing INPUT.avi + background.jpg)
   - **Outputs\** (containing stabilized.avi + extracted.avi + binary.avi + alpha.avi + unstabilized_alpha.avi + matted.avi + OUTPUT.avi + RunTimeLog.txt)
   - **Temp\** (If needed to save temporal artifacts – you can choose if you want to submit this files).

Zip the Output folder and upload to Moodle. If the files total size would be above 50MB, please send the file as an attached Google Drive URL.

Note that running the main function (runme.py) should create all required outputs.

Illustrated example (without stabilization and alpha videos – hard to show in stills but you will provide a stabilized video as well). Of course, you won't be doing this on cartoons ☺ :

INPUT

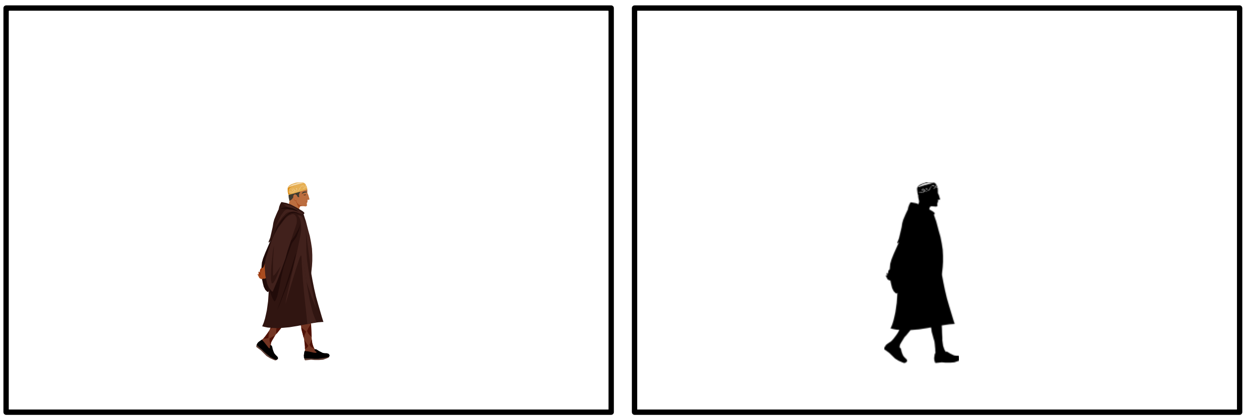


background

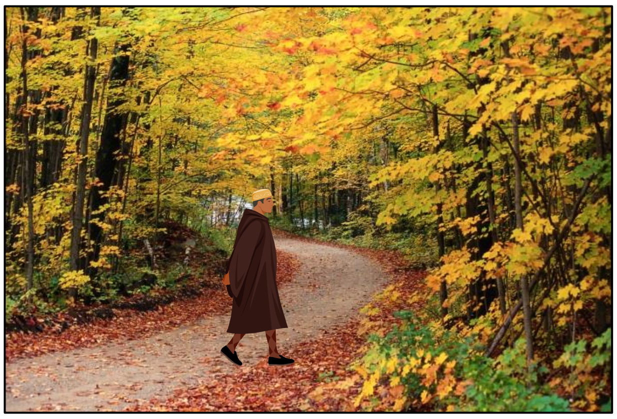
Extracted Person

+ Binary mask

(After stabilization)



matted



OUTPUT