CMPEN/EE 454, Project 3, Spring 2019

Due Friday, April 19 by 11:59pm EST, submitted in Canvas

1 Motivation

The goal of this project is to implement the four simple motion detection algorithms described in Lecture 24, run them on short videos, and compare the results. As described (and pseudocoded) in Lecture 24, the four algorithms are:

- Simple Background Subtraction (Slides 4-8)
- Simple Frame Differencing (Slides 9-14)
- Adaptive Background Subtraction (Slides 15-17)
- Persistent Frame Differencing (Slides 18-20)

You are to implement all four in one program and generate, as output, a four-panel frame showing the results of each algorithm, on each video frame (see Figure 1), and generate a video of the results. Note: you don't have to label them with text but should be the same order as shown in Figure 1.

While this project is a set of simple temporal processing tasks, you will discover that practical applications have hurdles to overcome. Specifically, there are 8 different videos to process that each require 4 images to be generated to make each frame. That is approximately 12,000 output images needed to make the quad layout video frames shown in Figure 1. Processing all frames of all videos is required for this project.

Your group will be provided:

- A zip file that contains eight (8) different video sequences. Each sequence is stored as a short video that you will need to extract into a series of JPG images number consecutively. Some video content courtesy of https://motchallenge.net/data/PETS2017/.

The specific project outcomes include:

- Experience in efficient and effective Matlab programming
- Understanding temporal frame differencing algorithms
- Using videos visualizing the temporal performance of implemented algorithms
- Implementing efficient code to ensure timely operation and testing of algorithms

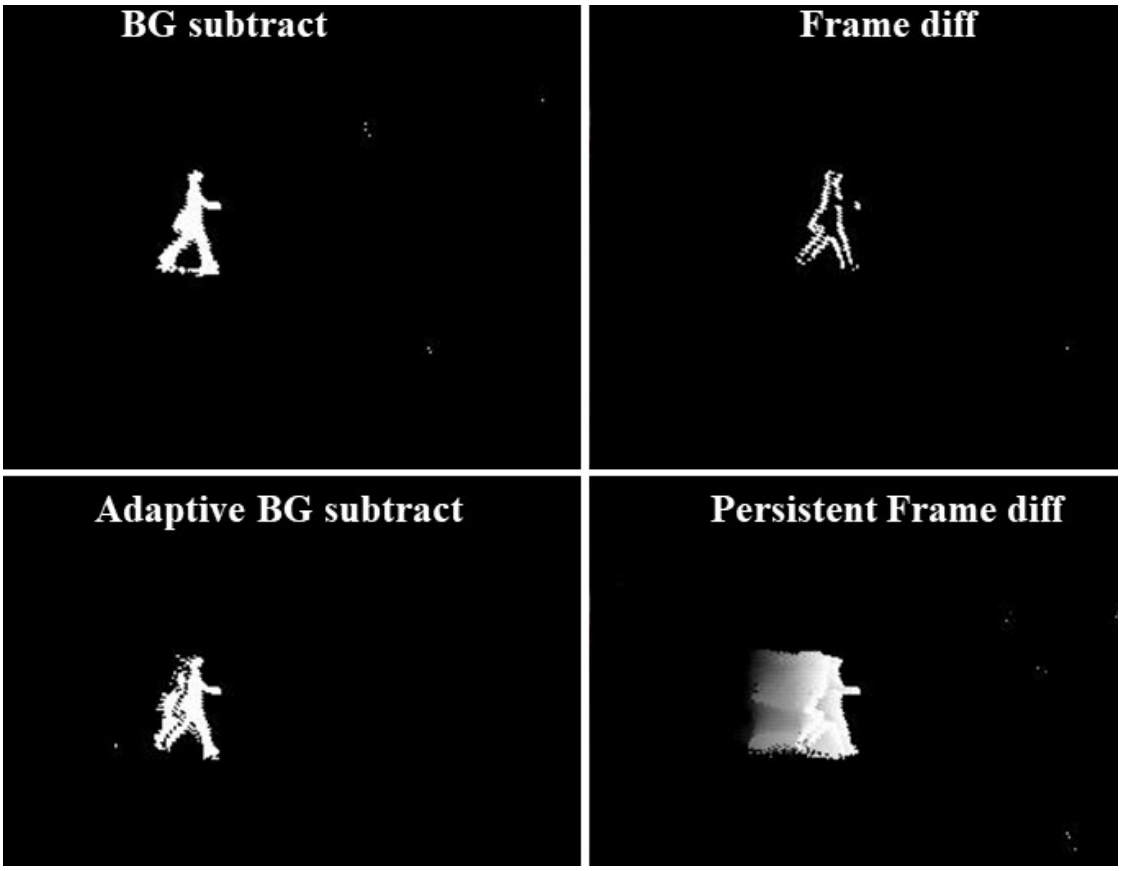

Figure 1: Example output frame showing results of the four (4) motion detection algorithms run on the same input sequence. Simple Background Subtraction (top left), Simple Frame Differencing (top right), Adaptive Background Subtraction (bottom left), Persistent Frame Differencing (bottom right)

2 Detailed Description

Refer to lecture 24 slides for detailed descriptions of each of the 4 motion detection algorithms enumerated in Section 1. The following steps will be essential to the successful completion of the project:

1) For a given video sequence, read in each image in a loop and convert it to greyscale using whatever method you are familiar with (for example by taking the green channel).
2) After reading each image, compute a grayscale output frame using each of the four motion detection algorithms and concatenate them together into a single four-panel (quad) frame. Concatenation is easy to do in Matlab, since images are arrays. For example, if the four motion detection results image arrays are A, B, C and D (you will want to make them more descriptive names), the output quad image can be generated as outimage = [A B; C D];. The persistent frame differencing image will have a different greyscale range (0-255) than the other three methods (0-1), when generating the output

quad image, you want to convert it to the range 0-1 by dividing by 255, so all output brightness ranges are compatible. Alternatively, you could scale up the intensity range of the other three by multiplying by 255).

3) Generate a numbered filename for your output image (e.g. if you are currently processing frame 0036 of the input sequence your output image will be numbered 0036) and save the image, either as a JPG or PNG. If you save as a PNG, your results will have better resolution in the final movie, since it does not compress the image while writing it to a file. Continue the loop until all input images have been processed.
4) Generate a video of the results. How you do this is totally up to you. One program that can take numbered images and combine them into a movie file is "ffmpeg", which is open-source software with versions that work on Windows, Linux or Mac. You may alternatively use Windows moviemaker, Mac iMovie, Matlab videoWriter, or whatever video creation tool you would like. Make your movies in some obviously playable format, like mp4. If we can't playback what you produce, we'll be unable to view it either and then we do not know that you completed the project.

Pro Tip: Although you are running all four algorithms at the same time, it is a good idea to keep their data structures independent from each other. For example, in the lecture pseudocode, variable B or B(t) is used to denote the current background frame. However, this image will be different for each algorithm, so you really should be keeping four current background frames, one computed by each algorithm.

3 Evaluation

As part of your report, and to convince us that your code is working correctly, we would like you to show the following quantitative and qualitative evaluations of your work.

Quantitative

Quantifying the accuracy of the algorithm's implementation without ground-truth results is not possible, but no ground truth results are provided. Therefore, evaluation will be primarily qualitative, by watching the video results.

Qualitative

The report should provide visualizations of each step of the data processing starting with the formatting for input to each of the 4 algorithms, the mathematical equations explaining each of the four motion detection algorithms, and the combination/formatting of each frame/video that is processed. An output video for each input video should be generated and submitted with specific frames provided as example in the report. A relative qualitative comparison should be completed to explain the impact of the control variables of each motion detection algorithm implemented. Provide empirical evidence of testing to find the "optimal" parameter configuration.

Efficiency

Use the Matlab profile() function to analyze and quantify the efficiency of your implementation. This tool allows for a full breakdown of the processor time used by each function call in a Matlab script. The efficiency of your implementation will also help you complete the project assignment more quickly. More information on the Matlab profile function is here: https://www.mathworks.com/help/matlab/ref/profile.html

4 What Libraries Can I Use?

The intent is that you will implement the various computational modules described using general processing functions (https://www.mathworks.com/help/matlab/functionlist.html) in Matlab. The notable exceptions are that you MAY use the image processing toolbox (https://www.mathworks.com/help/images/index.html) and the videoReader/videoWriter functions. Specifically, you MAY NOT use anything from the computer vision toolbox, or any third-party libraries/packages. Don't plagiarize any code from anywhere or anyone else.

5 Project Reporting and Evaluation

Half of your grade will be based on submitting a fully operational program with 8 results videos, and the other half will be based on a written report discussing your program, design decisions, and experimental observations. The following components will be submitted:

1) Submit a written report in which you discuss at least the following:
   a) Summary: Summarize *in your own words* what you think the project was about. What were the tasks you performed; what did you expect to achieve?
   b) Procedure Outline: Present an outline of the procedural approach used in your implementation along with a flowchart showing the control flow and subroutine structure of your Matlab code. Explain any design decisions you had to make. For example, did you keep the four (4) motion detection algorithms completely independent from each other or did you try to use the results for some of them when computing others? Basically, explain at each step why you chose to do whatever you did. Be sure to document any deviations you made from the above project descriptions (and why), or any additional functionality you added to increase robustness, generality, or efficiency of the approach.
   c) Experimental Observations: What do you observe about the behavior of your program when you run it? Does it seem to work the way you think it should? Run the different portions of your code and show pictures of intermediate and final results that convince us that the program does what you think it does.
   d) Quantitative Results: All methods have user-settable thresholds and parameters. One is the pixel difference threshold for determining what level of intensity change is needed to declare there has been a significant change at a pixel, not just sensor noise. What value did you decided to set it at, and how did you decide? What do you observe when

you set this higher or lower? An important parameter for adaptive background subtraction is the "alpha" parameter for merging new images into the background model. Try different values for this and explain what you observed as you change it higher and lower. Can you verify that setting it to 0 and 1 yields results very similar to two of the other algorithms? How did you set linear decay parameter in the persistent frame differencing algorithm and what happens if you make it larger or smaller?

e) Qualitative results: At least one quad frame showing the results from each of the 8 videos, should be included in the report in addition to the 8 results videos you will produce and include with the report. Additionally, the report should include intermediate images of parameter variations as discussed in the quantitative portion of the report. Compare the results of all four algorithms on the 8 videos (which should be easy to do since they are conveniently displayed as four panels in your output video). Do you notice any significant differences between them? Are there some situations where one or more of them produce terrible results? Identify some events in the videos where the algorithms produce very different quality results and explain why the different algorithms are either succeeding or failing on these events (you should include images in your report to illustrate these specific conditions). Overall, which method would you prefer to use, if you were only able to choose one, and justify your answer there is no right or wrong answer to this, as long as you have reasonable and sufficient justification.

f) Algorithm Efficiency: Provide an image showing the Matlab profiler output. If you decide to make performance improvements, show the profiler results both before and after to convince us that the efficiency has been improved, and in what functions. The code implementation should focus on efficiency implementation and use of Matlab to optimize speed and reduce code complexity.

g) Document what each team member did to contribute to the project. It is OK if you divide up the labor into different tasks (it is expected), and it is OK if not everyone contributes precisely equal amounts of time/effort on each project. However, if two people did all the work because the third team member could not be contacted until the day before the project was due, this is where you get to tell us about it, so the grading can reflect the inequity.

2) Turn in a running version of your main program and all subroutines in a single zip/tar archive file (e.g. all code/data/results in a single directory, then make a zip file of that directory).

a) Include a demo routine that can be invoked with no arguments and that loads the input data from the local directory and displays intermediate **and** final results showing the different portions of your program are working as intended.

b) We might be running the code ourselves on other input datasets, so include a routine that takes as input a path to directory of image frames and generates a video of the results of your 4 algorithm implementations on these video frames.

c) Include enough comments in your functions so that we have a clear understanding of what each section of code does. The more thought and effort you put in to demonstrating / illustrating in your written report that your code works correctly, the

less likely we feel the need to poke around in your code, but in case we do, make your code and comments readable.

6 Grading Criteria

Project 3 grade = (100 − Deductions + Additions) × (1− Late Penalty)

Deductions - Implementation

- Implementation of all four (4) motion detection algorithms [30pts]: visualizations for specific frame indexes of the quad video showing proper implementation of each algorithm. Each algorithm should be clearly defined and implemented to generate each of the four (4) images needed to generate the final result video. Results videos must be included for all eight (8) provided video sequences.
- Program structure and readability [10pts]: The code should have enough comments to explain the functions and procedures while being properly divided into small modules (function calls).
- Code should be runnable [10pts]: Each function created should operate properly when called as part of your implementation and should not cause failures in any demonstration code. Functions may be spot checked to verify proper coding style and commenting. Main function routine should process a directory of sequentially numbers video frames and generate a directory of result frames with matched frame numbers.

Deductions - Report

- Implementation Description [10pts]: Outline of your code along with proper diagrams. Unless the operation is predefined in Matlab, explanation of what operation is being applied is required. This is a step by step of WHAT was done to achieve the desired outcome. Provide *your own version* of pseudocode and diagrams to explain each algorithm and specifically how it was implemented.
- Experiment Observations and Explanations [15pts]: This is a step by step explanation of WHY each operation was used in the implementation. This should specifically answer the questions asked in early parts of this assignment.
- Experiment Visualizations [15pts]: This is a step by step visual representation of intermediate and final results including example frames from the eight (8) results videos.
- Evaluation and Discuss [10pts]: This is an explanation of the results, the overall successes/failures of the implementation, and what steps were taken to resolve technical hurdles that occurred during this project.

Additions

- Process your own video [5pts]: demonstrating the operation of your implementation on a video of your own. The results should be similar in format to the other 8 videos but of different content. Discuss what in your video images impacted each of the 4 change detection algorithms. Along with the result video, provide input image frames as part of the project submission so that your algorithm can be tested on your unique video.

Late Penalty

Deduction	**Late Time**	**Time Stamp Submitted**
0%	Not Late	By 11:59**pm** 04/19/18
10%	Up to 12 hours	By 11:59**am** 04/20/18
20%	Up to 24 hours	By 11:59**pm** 04/20/18
40%	Up to 36 hours	By 11:59**am** 04/21/18
80%	Up to 48 hours	By 11:59**pm** 04/21/18
100%	More than 48 hours	After 11:59**pm** 04/21/18