CMPEN 454 - Project 3

By: Rui Zheng, Jacob Minnich, Chenning Zhang, Nelson Huang



Summary

The project seeks to test our understanding with various motion detection techniques by allowing us to write computer algorithms that uses those techniques. Our project is divided into four sub tasks.

* Simple Background Subtraction
* Simple Frame Differencing
* Adaptive Background Subtraction
* Persistent Frame Differencing

The goal of this project is to implement the four motion detection algorithms in one program, and generate a four panel frame that shows the results of each of motion detection techniques on each frame of video.

For the simple background subtraction implementation, we want to remove the static background of the frame. To achieve this, we first created a for-loop that iterates through the entire frame of a directory. Then, we set a variable inside the for-loop that represents the first frame in the directory. We do this by creating an if-statement, such that the variable will be created if the algorithm is at the first iteration. As the loop continues to iterate, we are creating the frame with the background subtraction technique applied by computing these calculations:

Here, represents the first frame in the directory and represents each of the changing frame in the directory. After computing the differences between the two frames, we then create a binary image from where The variable is the resultant frame we want to produce.

For the simple frame differencing implementation, we want to produce a frame where we are not only removing the static background, but also produce a frame that shows the only obvious presence we see in the frame is the object that is moving. The algorithm we used is the exact same procedure we used for the simple background subtraction, except that we did not use in our calculations. Instead, we decided to create a variable called , which represents the frame in the iteration, where is the current iteration in the loop. To illustrate this a bit, these are the computations we computed:

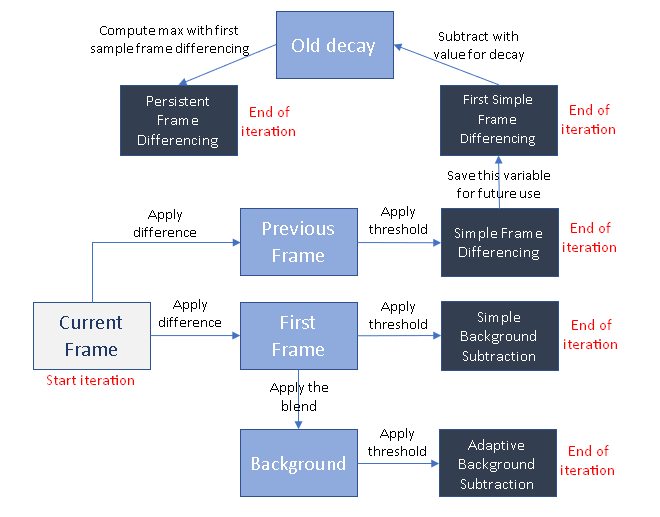
For the adaptive background subtraction, we want to produce a frame where our current frame is “blended” into the background, such that fast and small moving objects are well segmented that leave behind short “trails”of pixels. When an object stops, there is a “ghost” left behind by object that would begin to gradually fade into the background. The implementation is somewhat similar to that of the Simple Background Subtraction and the Simple Frame Differencing. The only difference is that we have to compute the calculation that “blends” our current frame into the background. To do this, these are the computations we computed:

Here, is a constant that we set to be equal to 0.25. The variable is used for the such that when , will yield a simple background subtraction frame, and when it will yield a frame differencing image.

For the Persistent Frame Differencing, we want to produce a frame that is responsive to changes in illumination and camera motion, and static objects fade away but would leave a trail of pixels if they are moving. To implement this, we computed these calculations:

In order to implement the persistent frame differencing, we first need to compute the simple frame differencing and save its frame during the first iteration to a variable.With that, we compute by subtracting from the first simple frame differencing image we saved. Here, we set the value of to be 25 in order to gives moving objects a clear and long trail. The final result is , which we computed by finding the maximum value of and . Finding the maximum value of these two frames is basically equivalent to finding which frame is brighter.

Outline



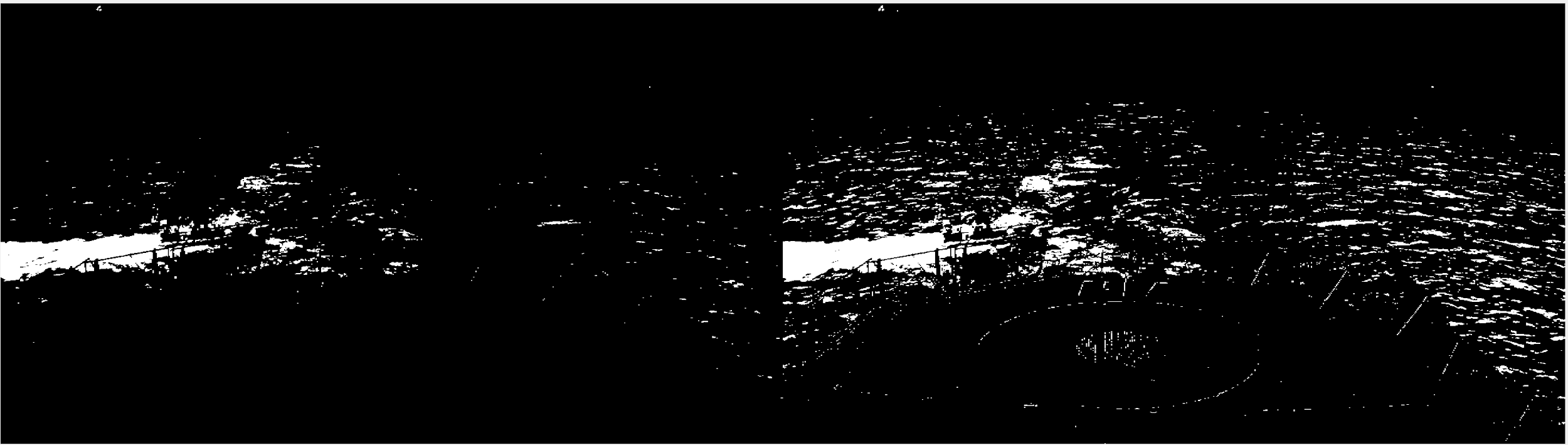
The structure of our MatLab code is fairly basic. We named our main function ‘project3.m’ and that function accepts a parameter named ‘folder’. In order to run our program, the user must type in the name of the main function (project3) followed by the name of a folder in the directory that contains a bunch of images (i.e. trees) into the command line on MatLab. So, typing ‘project3 trees’ onto the MatLab command line would produce a four panel frame showing the results of each motion detection techniques. In addition, our directory contains 4 more MatLab function files containing algorithms for each of the motion detection techniques.

In terms of how our program runs, we basically set a for-loop that iterates through all of the images for a specified folder. Inside the loop, we are calling the 4 MatLab function files, as well as creating additional variables to produce the final product.

Experimental Observation

**Pixel difference threshold lambda**

We use lambda = 0.1 for simple and adaptive background subtraction, and lambda = 0.2 for simple and persistent frame differencing. We test it mainly by setting it to different values and see how the output looks.

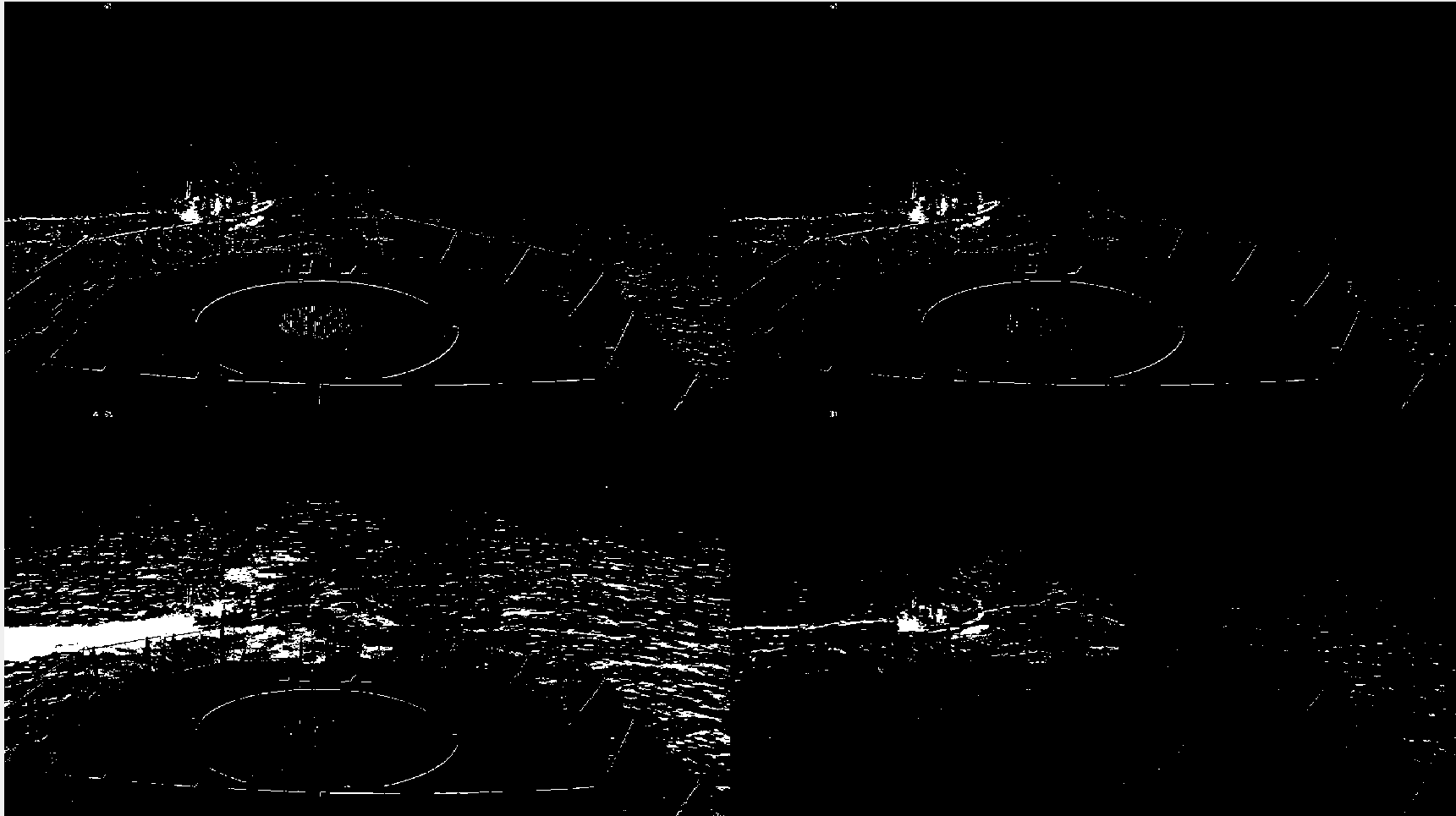


Simple background subtraction with lambda = 0.2 (left) and lambda = 0.1 (right)

We found that when lambda is higher ( > 0.3) the output frame would be very clear but the outline of the object would be vague as well. In contrast, when lambda is low ( < 0.1) the object would be clear but there would also be a lot more noise in the background. Ultimately, we chose threshold values to maximize performance across all input files.

**Adaptive background weight alpha**

We use alpha = 0.5 for the experiment. We also test it by using different values to see how different the outputs are. Ultimately, we decided that setting alpha = 0.5 would make the graph distinct enough from both frame differencing and background subtraction.



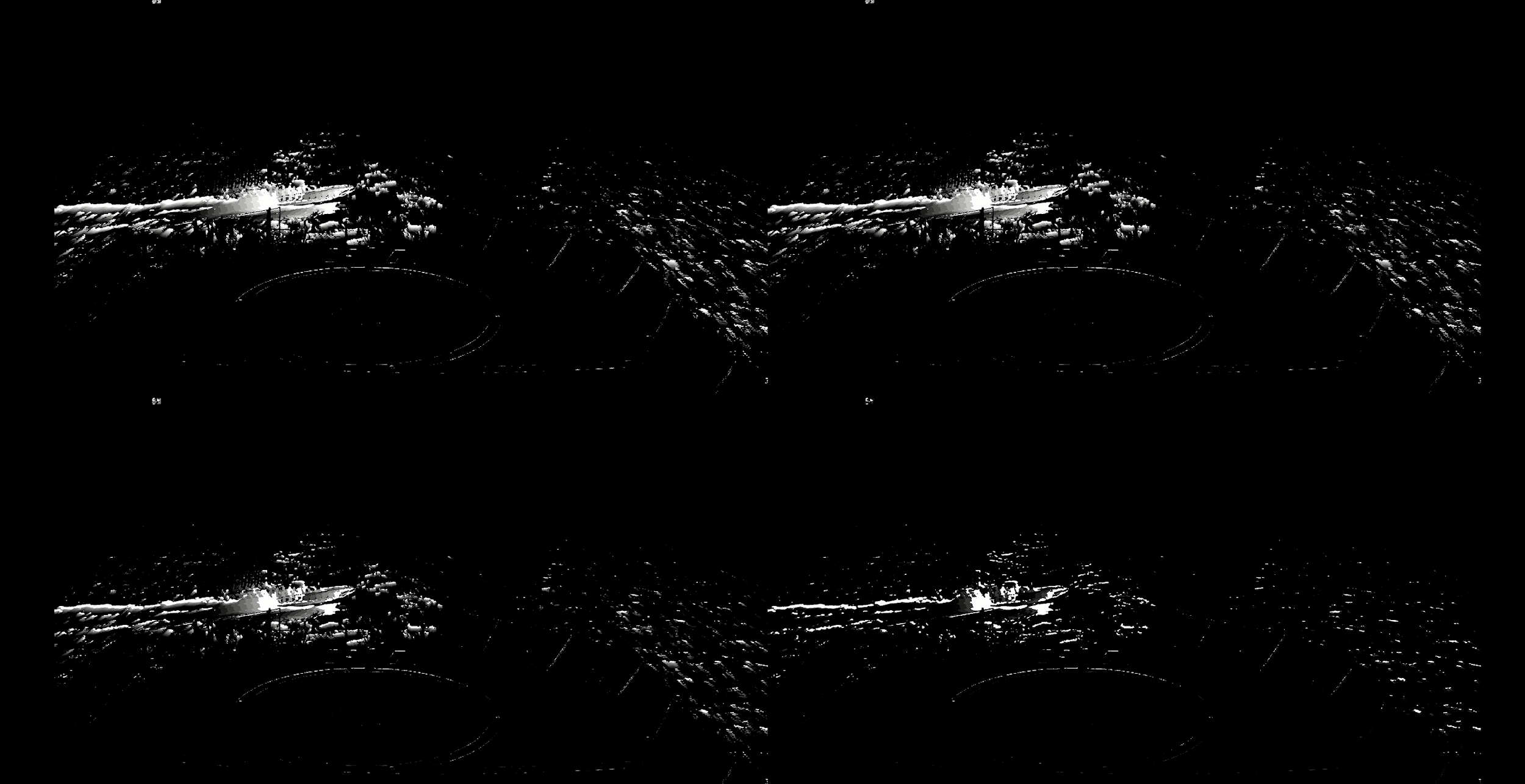
Top left: ABS w/ alpha = 0.25 Top right: ABS w/ alpha = 0.75

Bottom left: Simple background subtraction Bottom right: Frame differencing

We verify the result of alpha = 0 and alpha = 1 by subtracting the output with output of background subtraction and frame differencing.

**Decay Parameter**

We set the decay parameter to 25 (25/255) for persistent frame differencing, which gives moving objects a clear and long trail.



Top left: decay = 25 Top right: decay = 50 Bottom left: decay = 100 Bottom right: deay = 200

**Others**

We set the frame difference to 1 for frame differencing. We tested three-frame differencing with frame difference from 1 to 20, but the result was not good, so we set it back to simple frame differencing with fame difference = 1.

Experimental Comparison

**ArenaA**

Simple background differencing doesn’t work well due to relatively large movement due to wind. Frame differencing and adaptive background subtraction work well. Persistent frame differencing doesn’t work quite as well in the last part, since trails of different persons overlap too much.

**ArenaN**

SBD doesn’t work well because in the first frame, which we use as background, the moving object is already in there, causing an error in detection. ABS, FD, and PFD work equally well.

**AShipDeck**

ABS, FD, and PFD work equally well again. Depending on the need, SBD might be the best or the worst as it also detects the ship’s effect on the waves and the distance from the horizon.

Contributions

Rui Zheng:

* Coding portion for Adaptive Frame Subtraction
* Coding portion for Persistent Frame Differencing
* Experimental Comparisons (Report)

Nelson Huang:

* Coding portion for Simple Frame Differencing
* Coding portion for Persistent Frame Differencing
* Outline (Report)

Chenning Zhang:

* Coding portion for Simple Background Subtraction
* Experimental Observations (Report)

Jacob Minnich:

* Coding portion for Persistent Frame Differencing
* Summary (Report)