Trey McAtee

12/2/20

CS 350 Assignment 3

**pimEdge.c**

**Problem Definition/Specification**

We need to write an image analysis program. In the making of this program, our understanding of some key concepts from operating systems such as processes and threads will be enhanced. For this program, we are assuming that there are multiple servers available to perform the task that the user will need to be completed. The user will request the server to perform their task and they will provide an image for the task.

**Problem Analysis**

As for inputs, we need the user to give an ***m x m*** image **A** that we are assuming to be a square image. As for outputs, we will output the image and return it to the client.

**Algorithm Design**

I will follow the guidelines and formulae in the assignment 3 handout to design the algorithm and use what I can from my existing (although incomplete) program from Assignment 2. I will attempt to utilize pipes as well, according to the recommendations and benefits named from Professor Adjeroh in class.

**Test Plan**

For testing, I will give the provided image to the program and compare it to what I believe it should look like as in the example of the assignment handout.

**Testing**

As for testing, it went well in some regards, and not well in others. At this point, I can successfully put in the amount of processes that I want my program to use, but I failed in performing correct calculations, and so I took out some of the code that performed the incorrect calculations and commented out a small portion that I thought worked correctly until recently. I can put any of the 3 images provided into my program and they will correctly output the 3 basic masked images (vertical, horizontal, both). I will display these results in this section via snipping tool screenshots, with example command lines that I used. I will also output “binary” images, but they are incorrect. So, I will just show that they do appear in the folder as they are not working correctly.

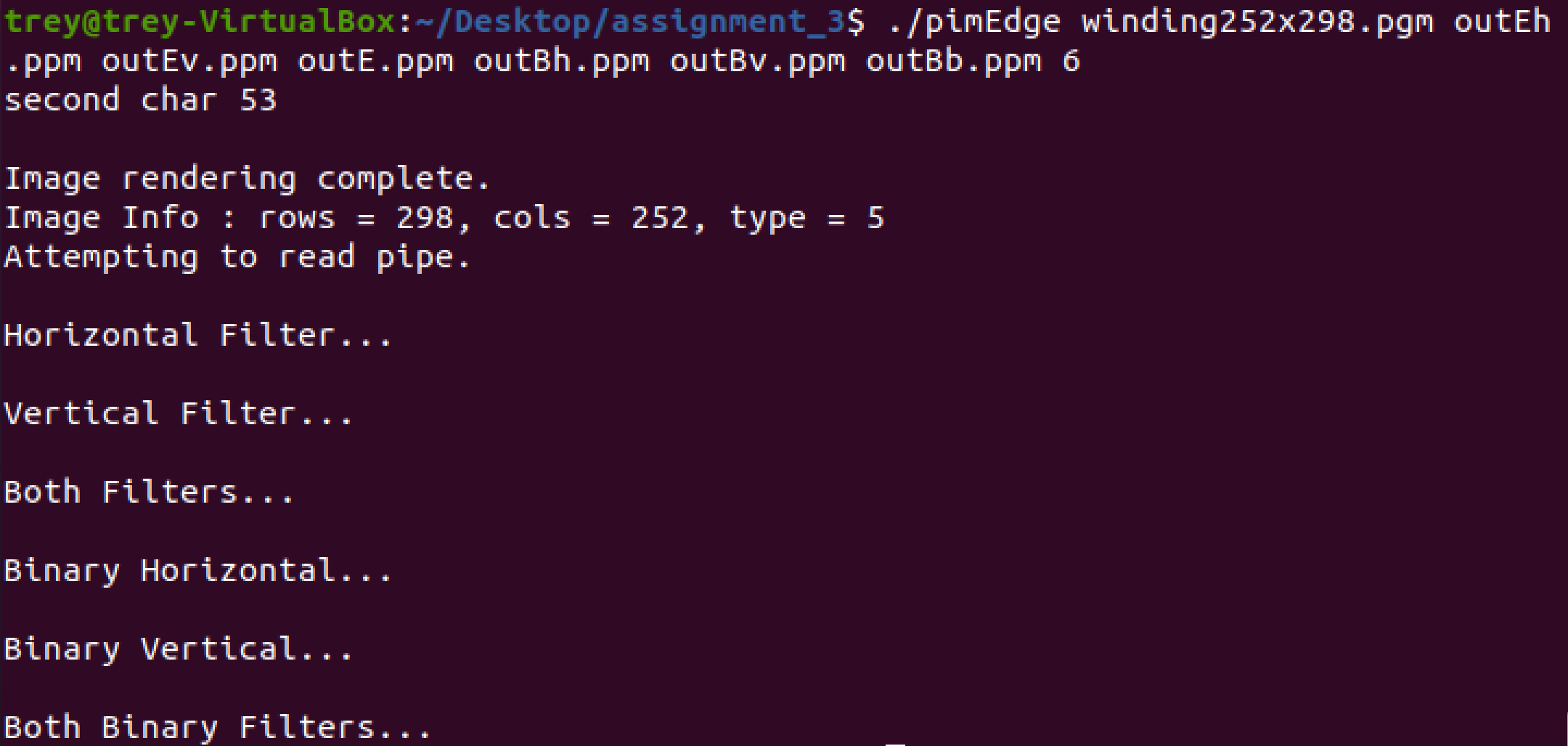


Figure 1, example command line input with messages for each output write function working correctly



Figure 2, output file outE.ppm

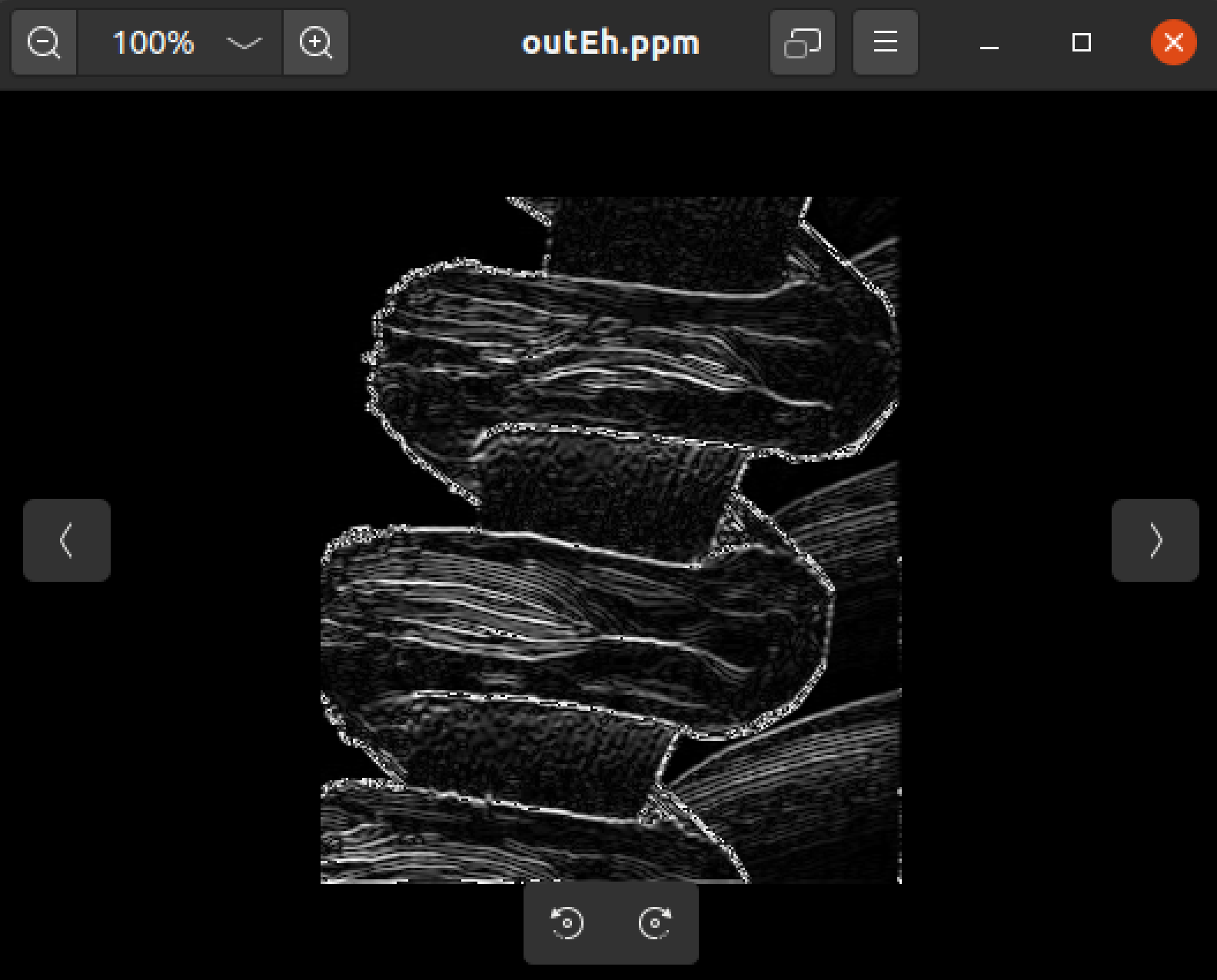


Figure 3, output file outEh.ppm

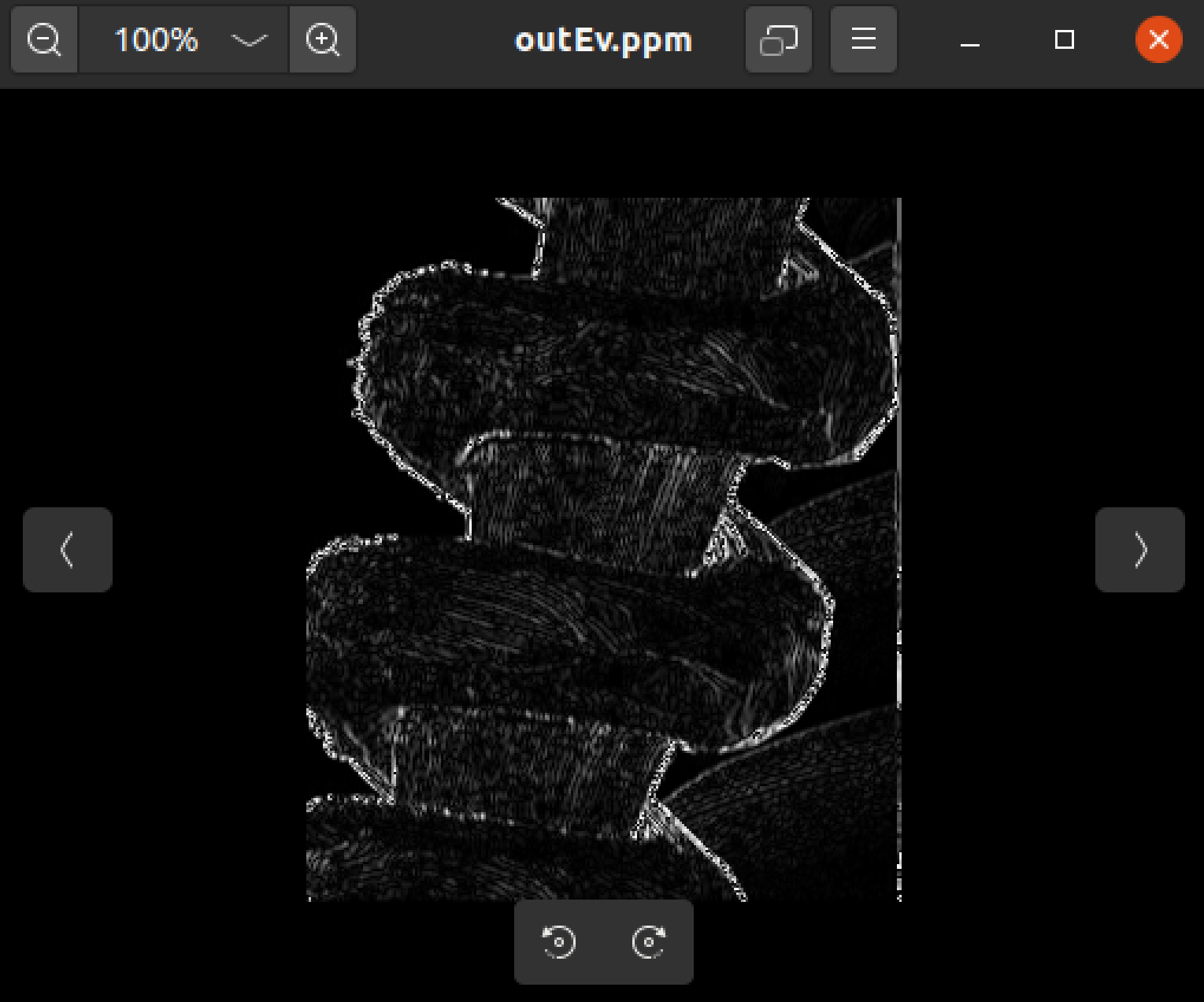


Figure 4, output file outEv.ppm

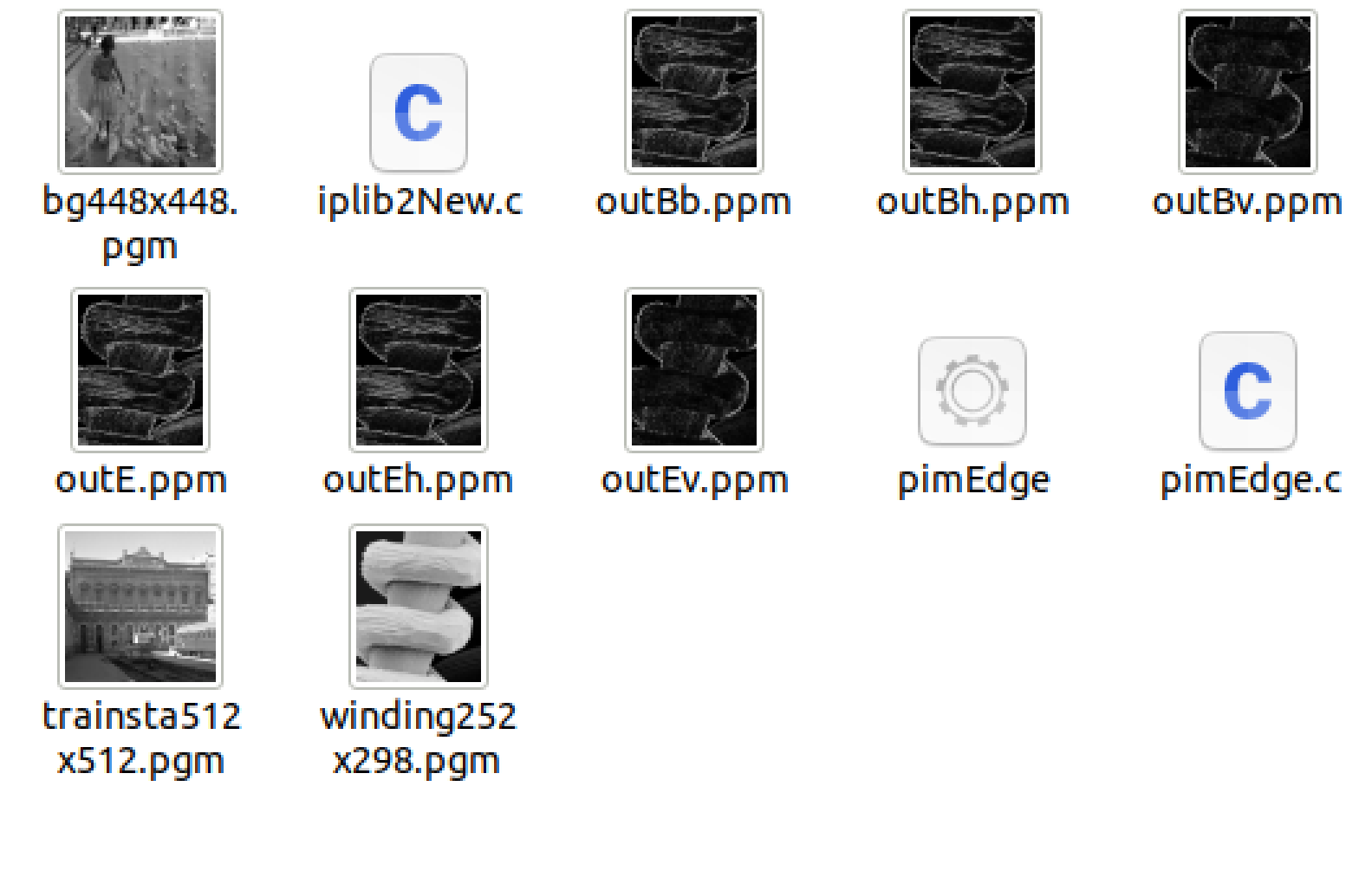
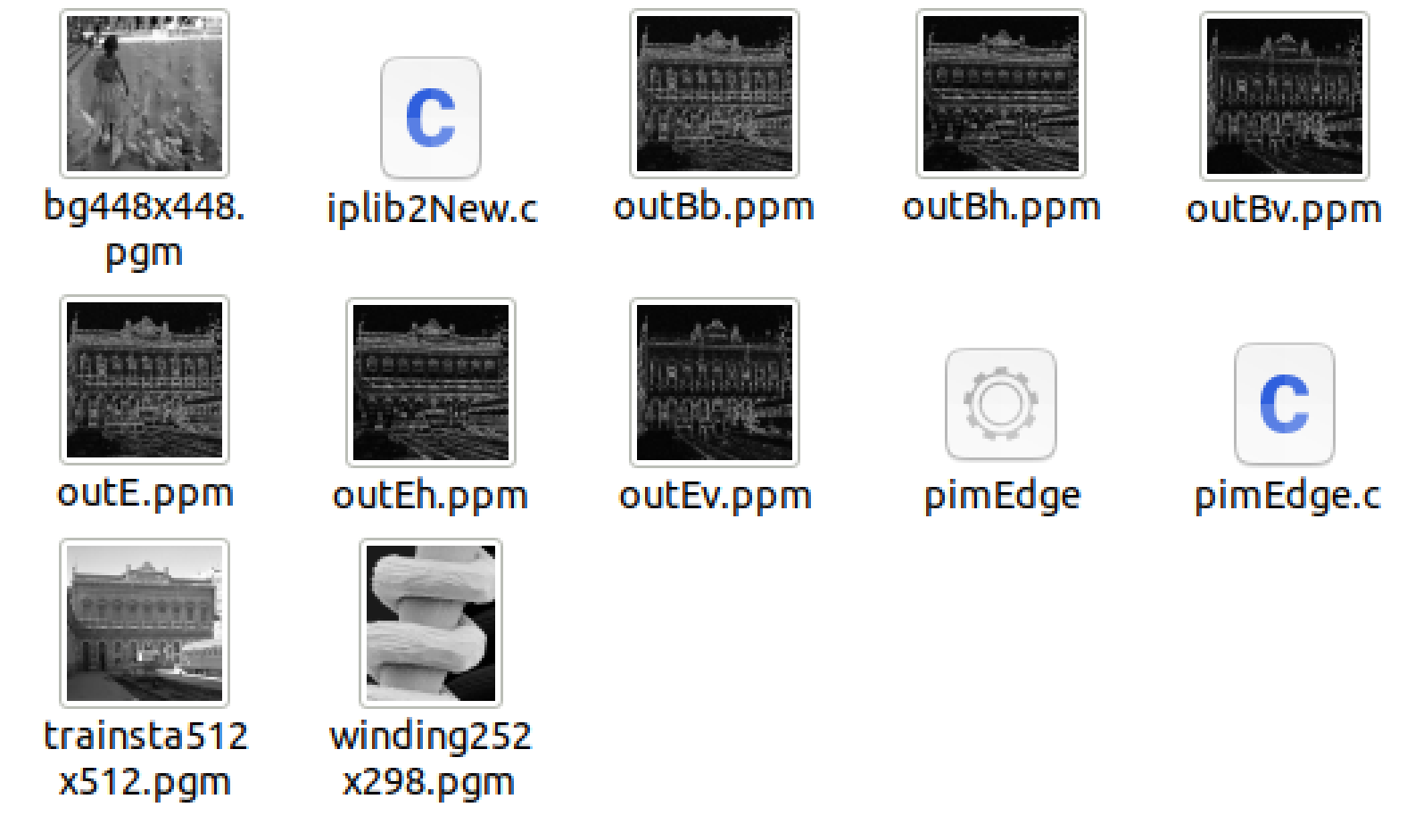


Figure 5, example of folder showing all images (including incorrect binary ones)

Figure 6, example of folder briefly showing that other provided images can be written