William Benton

Lab 1 Report

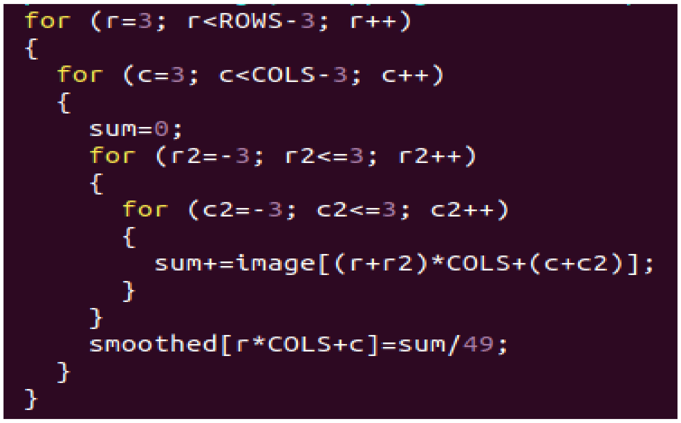
ECE 4310 – Computer Vision

For this lab, we implemented 3 filters for smoothing an image. The filters were all 7x7 but were implemented three different methods. The three methods were a standard 7x7 filter with convolution, a separable 7x7 filter, and a separable 7x7 filter implemented with a sliding window. All three methods were timed to compare the difference in computing time. The program was run 10 times and the results were saved and the results for each method were averaged to try to get an accurate estimation of how long each would take for the same image. The results are shown in the table below. All times are in nanoseconds.

Standard 7x7	Separable Filter	Sliding Window
35455761	21653400	4200306
24873426	18472696	6531653
25301705	31049322	4343478
23193572	39289849	4062543
28090392	22166731	5599234
38541614	13705009	7712747
24061359	20275162	6070453
24661431	17184809	3955671
22918436	20678081	5994772
22307284	16675229	12374962
AVG: 26940498	AVG: 22115028	AVG: 6084581

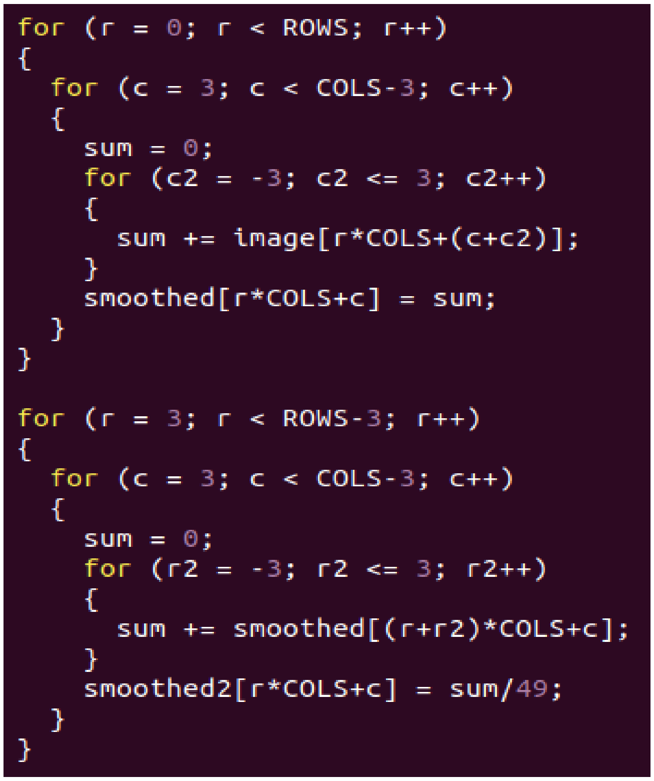
These results were what was expected for the most part. The only outlier was the final run of the sliding window which was over double the average. However, the averages were still in line with the expectations. The standard 7x7 filter was the slowest with the separable filter decreasing run time and the combination of the separable and sliding window being 4.42 times faster on average then the standard filter.

This was implemented in C. The image was read in and the image was processed and a temp image was used for processing and the final result was written to a *.ppm file. The baseline code was provided and modified to perform a 7x7 filter and then new code was added to perform the other filters. The portion of the code that performs the filtering is shown below.



```
for (r=3; r<ROWS-3; r++)
{
  for (c=3; c<COLS-3; c++)
  {
    sum=0;
    for (r2=-3; r2<=3; r2++)
    {
      for (c2=-3; c2<=3; c2++)
      {
        sum+=image[(r+r2)*COLS+(c+c2)];
      }
    }
    smoothed[r*COLS+c]=sum/49;
  }
}
```

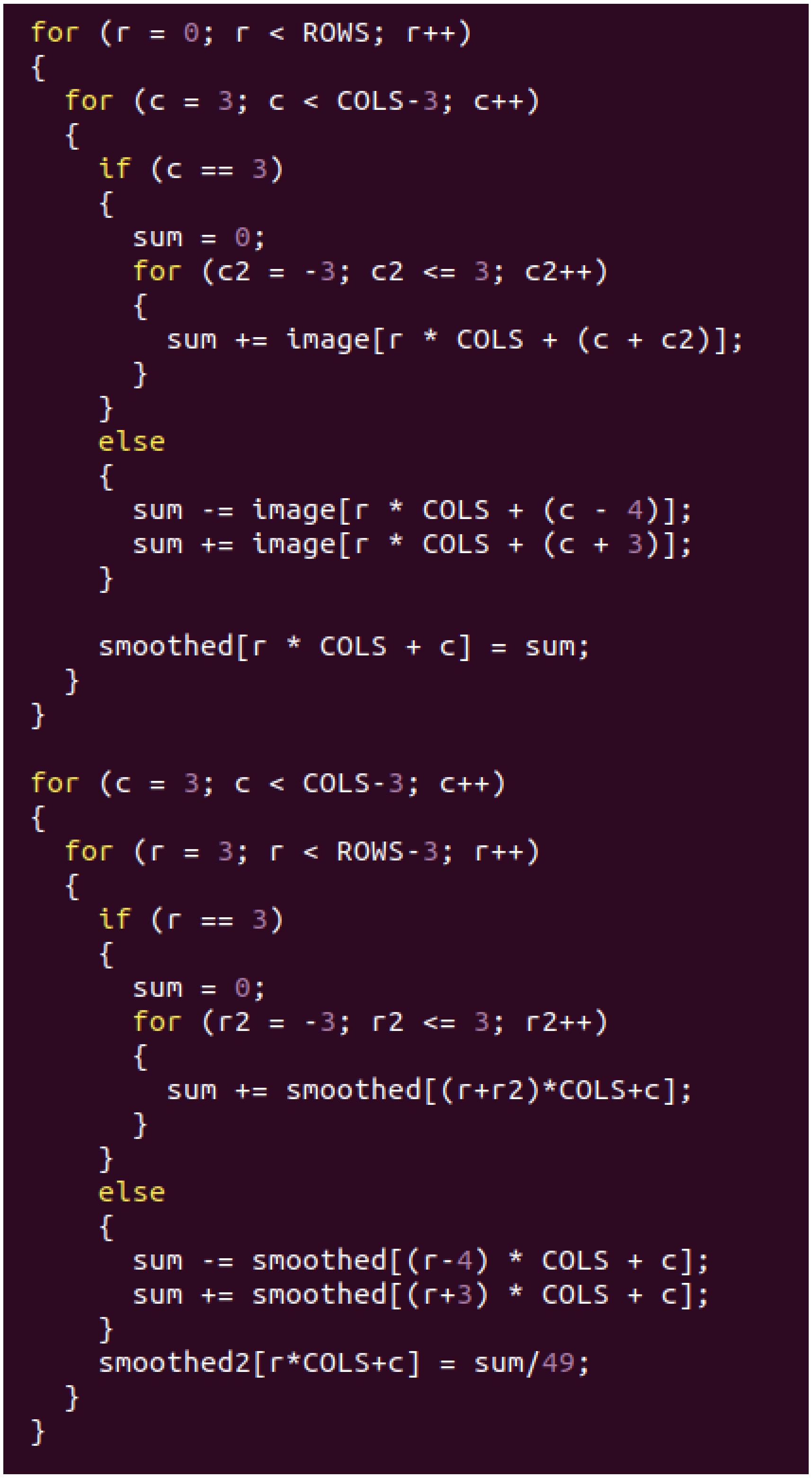
Figure 1: Standard 7x7 Filter Implementation



```
for (r = 0; r < ROWS; r++)
{
  for (c = 3; c < COLS-3; c++)
  {
    sum = 0;
    for (c2 = -3; c2 <= 3; c2++)
    {
      sum += image[r*COLS+(c+c2)];
    }
    smoothed[r*COLS+c] = sum;
  }
}

for (r = 3; r < ROWS-3; r++)
{
  for (c = 3; c < COLS-3; c++)
  {
    sum = 0;
    for (r2 = -3; r2 <= 3; r2++)
    {
      sum += smoothed[(r+r2)*COLS+c];
    }
    smoothed2[r*COLS+c] = sum/49;
  }
}
```

Figure 2: Separable 7x7 Filter



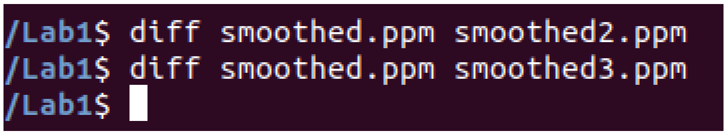
```
for (r = 0; r < ROWS; r++)
{
  for (c = 3; c < COLS-3; c++)
  {
    if (c == 3)
    {
      sum = 0;
      for (c2 = -3; c2 <= 3; c2++)
      {
        sum += image[r * COLS + (c + c2)];
      }
    }
    else
    {
      sum -= image[r * COLS + (c - 4)];
      sum += image[r * COLS + (c + 3)];
    }

    smoothed[r * COLS + c] = sum;
  }
}

for (c = 3; c < COLS-3; c++)
{
  for (r = 3; r < ROWS-3; r++)
  {
    if (r == 3)
    {
      sum = 0;
      for (r2 = -3; r2 <= 3; r2++)
      {
        sum += smoothed[(r+r2)*COLS+c];
      }
    }
    else
    {
      sum -= smoothed[(r-4) * COLS + c];
      sum += smoothed[(r+3) * COLS + c];
    }
    smoothed2[r*COLS+c] = sum/49;
  }
}
```

Figure 3: Sliding window 7x7 Filter

Below is a result from running the diff command on the three images that were output.



```
/Lab1$ diff smoothed.ppm smoothed2.ppm
/Lab1$ diff smoothed.ppm smoothed3.ppm
/Lab1$ 
```

Figure 4: Diff Results

If the images differ, a line would appear that states “The binary files differ.” so this shows that the images do not differ. Below is the result of the filtering. Only one image has been added because the command line above shows that the three images do not differ.

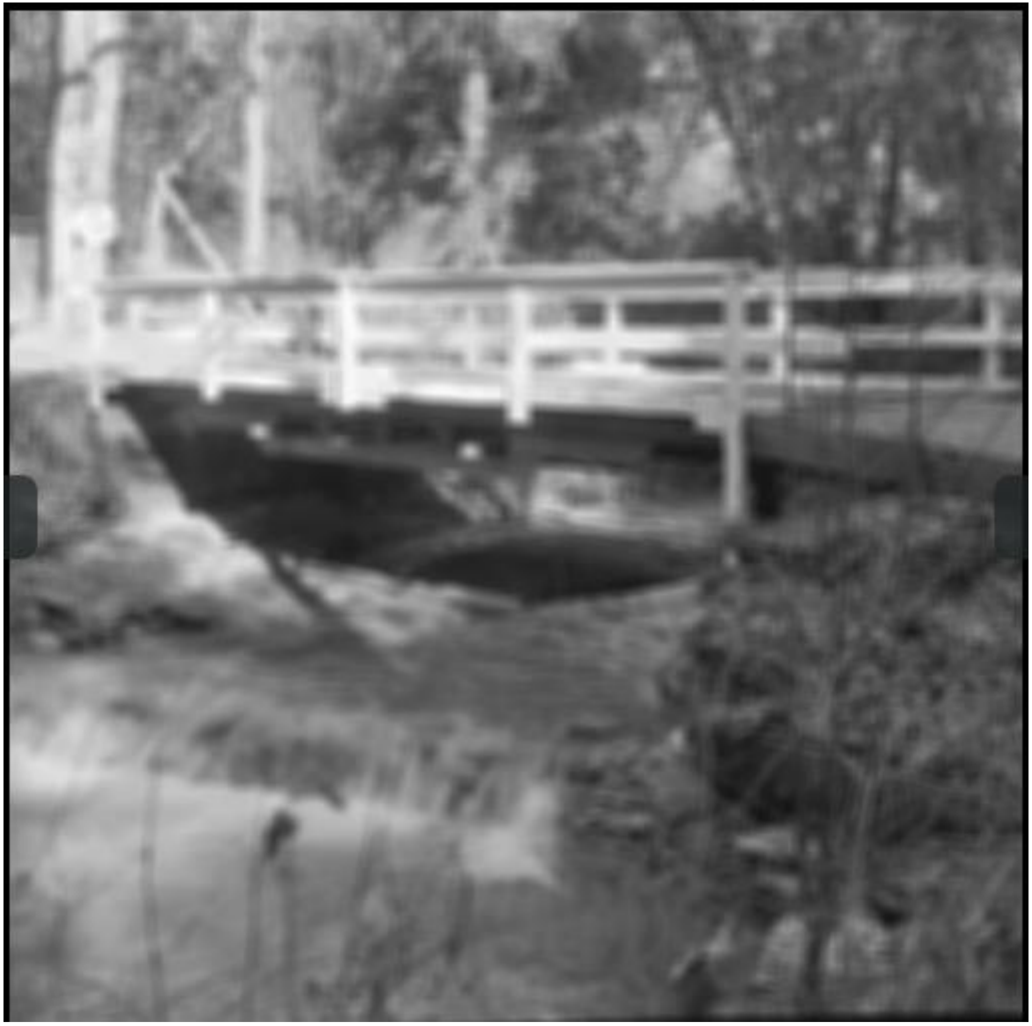


Figure 5: Smoothed Image