Lab 1: Smoothing

Convolution, Separable Filters, Sliding Window

ECE 4310

Brian Guzman

The purpose of this lab was to implement three versions of a 7x7 mean filter. The first version was to be a 2D convolution algorithm, the second version was to use separable filters, and finally separable filters with a sliding window. The output from all three filters should be the same. To verify that the output images were exactly the same, all outputs were to be compared using "diff". Apart from being compared for likeliness, the filters were to be timed multiple times with the average taken for comparison. The screenshots below show the excerpts from the code for each filter as well as the corresponding outputs.

2D Convolution:

```
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_in[((r+r2)*COLS)+(c+c2)];
            }
        }
        image_out[(r*COLS)+c] = sum/49;
    }
}
```

Separable Filters:

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

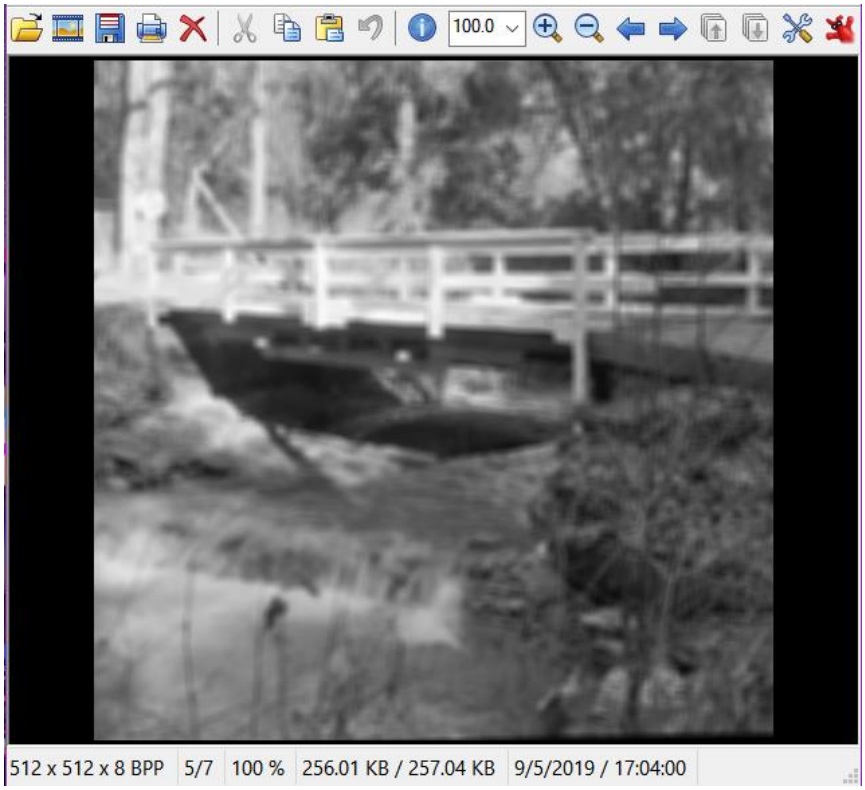
Sliding Window:

```
for(r = 0; r < ROWS; r++)
{
    sum = 0;
    for(c = 0; c < COLS; c++)
    {
        if(c < 7)
        {
            sum += image_in[(r * COLS)+c];
            if(c == 6)
            {
                temp_image[(r*COLS)+3] = sum;
            }
        }
        else
        {
            sum = image_in[(r*COLS)+c] + (sum - image_in[(r*COLS)+(c-7)]);
            temp_image[(r*COLS)+(c-3)] = sum;
        }
    }
}
for(c = 0; c < COLS; c++)
{
    sum = 0;
    r2 = 3;
    for(r = 0; r < ROWS; r++)
    {
        if(r < 7)
        {
            sum += temp_image[(r*COLS) + c];
            if(r == 6)
            {
                image_out[(r2*COLS)+c] = sum/49;
                r2++;
            }
        }
        else
        {
            sum = temp_image[(r*COLS)+c] + (sum - temp_image[((r-7)*COLS) + c]);
            image_out[(r2*COLS)+c] = sum/49;
            r2++;
        }
    }
}
```

Outputs:

Original Image:

2D Convolution:



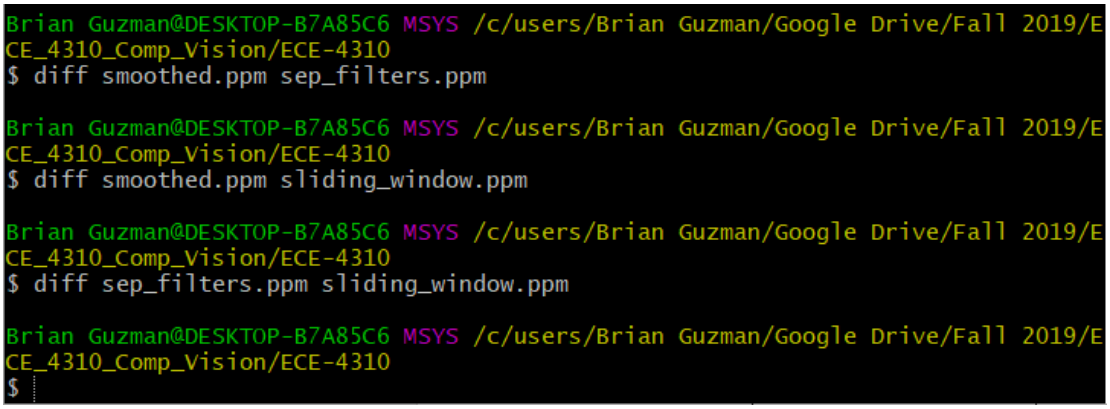
Separable Filters:

Sliding Window:

Timing Results:

2D Convolution	Separable Filters	Sliding Window
32455400	8314800	3185300
32049800	8027800	2972500
31054700	8639900	2836700
31153000	8270100	2764700
31021500	8306700	2720200
30786700	8406500	2777600
30086800	8555600	2960500
29549600	8895800	2720800
29127300	8370900	2720300
30629900	8454400	2720300
Average		
30791470	8424250	2837890

Diff results:



Conclusion:

From the above data, it's observed the three different filters produced the same output. The table of times above shows that the sliding window filter was much faster than the 2D convolution and separable filters.