CMSE

# Road Line Identification

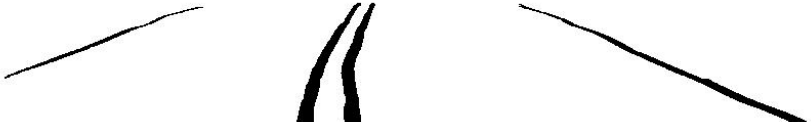
James Kervin

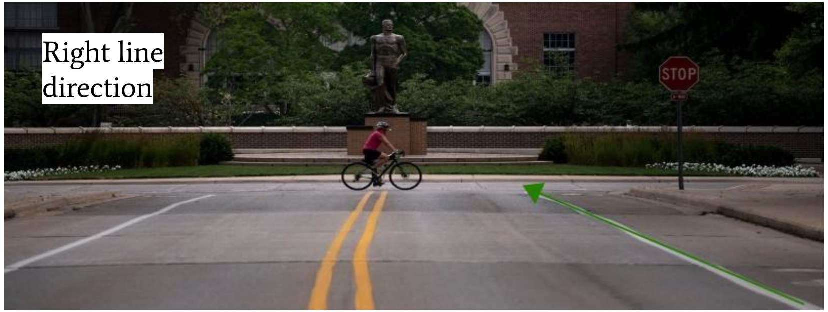
Fall 22 CMSE 202 Honors Project

# Goals:

1. Create a method for identifying road lines
2. Create a method for finding the right line vector
3. Analyze the effectiveness of each method

Key





# Key Assumptions and Limitations

**Assumptions:**

- Each image is 800 x 300
- Each image contains a road in it

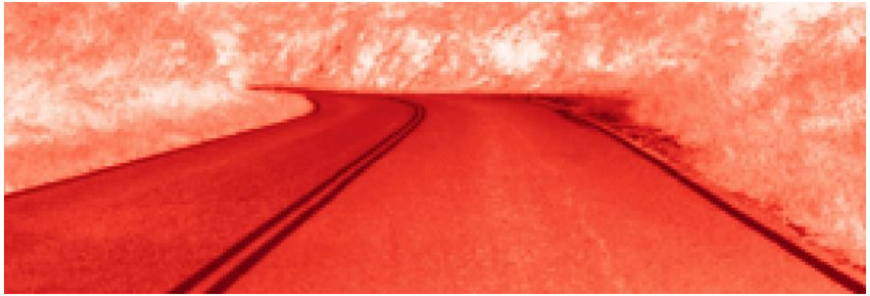
**Limitations:**

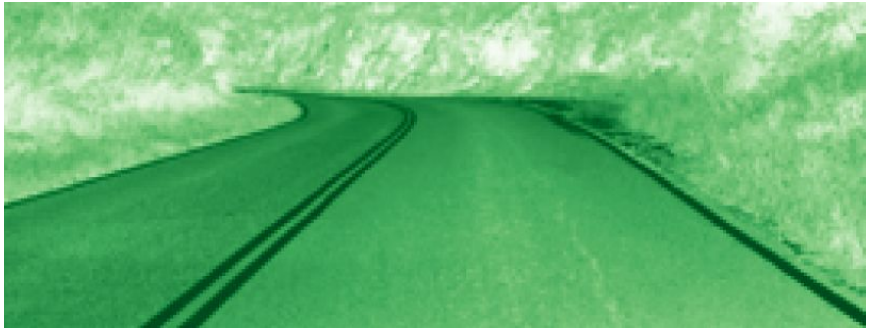
- Not all images have the same angle to the road
- Images come from different cameras

# Training the SVM

(Support Vector Machine)

- Find images and create keys
- Break the image into its components
- Train using a set amount of data
- Test with new images



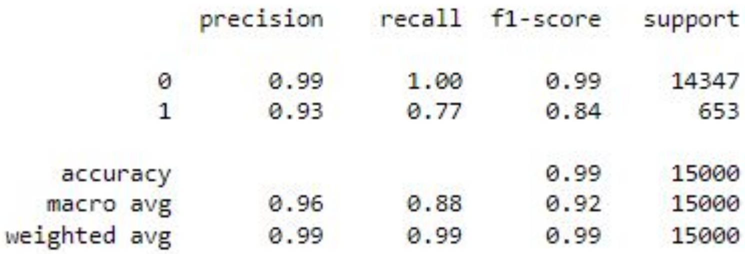


# One Picture Training Set

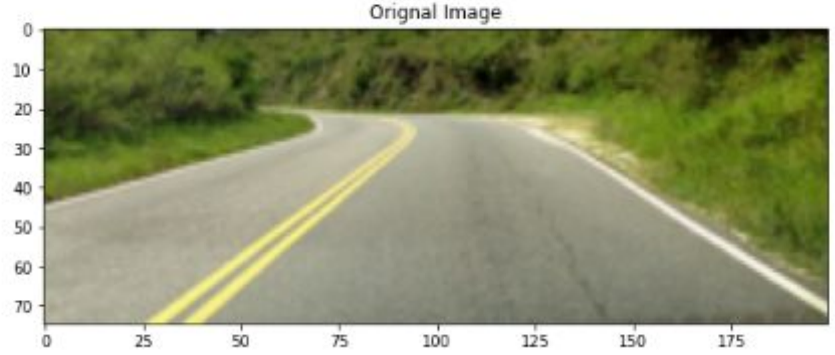
**SVM Parameters:**

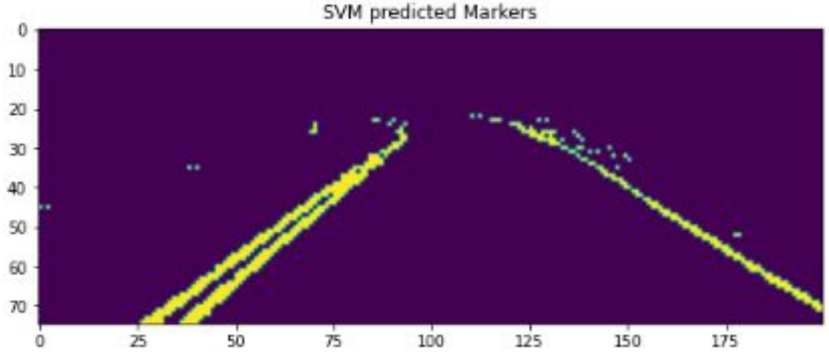
C: 100
Γ : 0.005

**Results:**

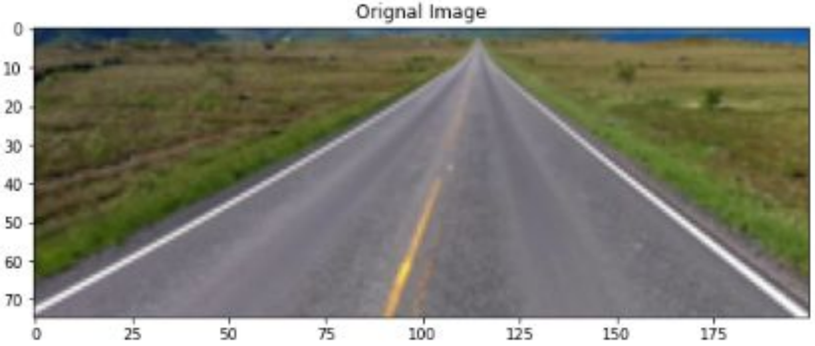


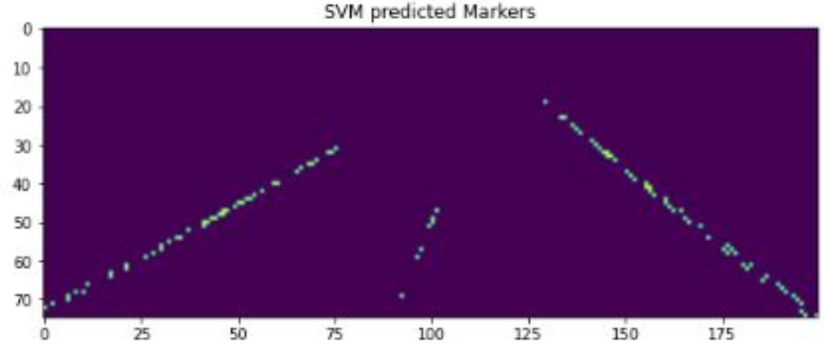
```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     14347
           1       0.93      0.77      0.84       653

    accuracy                           0.99     15000
   macro avg       0.96      0.88      0.92     15000
weighted avg       0.99      0.99      0.99     15000
```

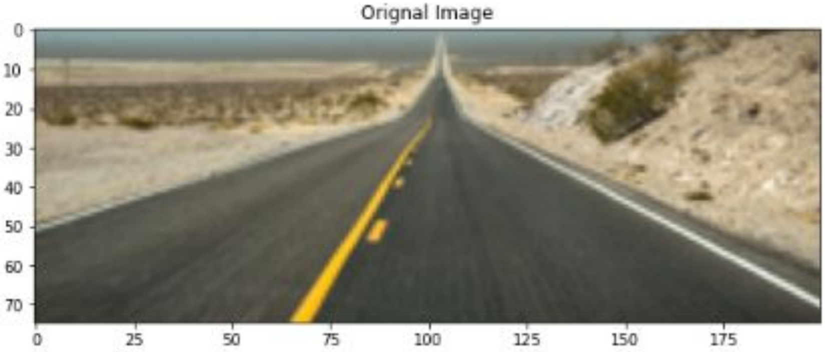


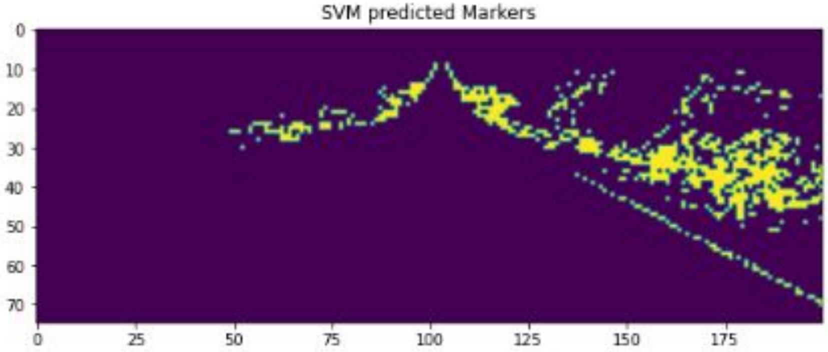


# One Picture Training Set Results

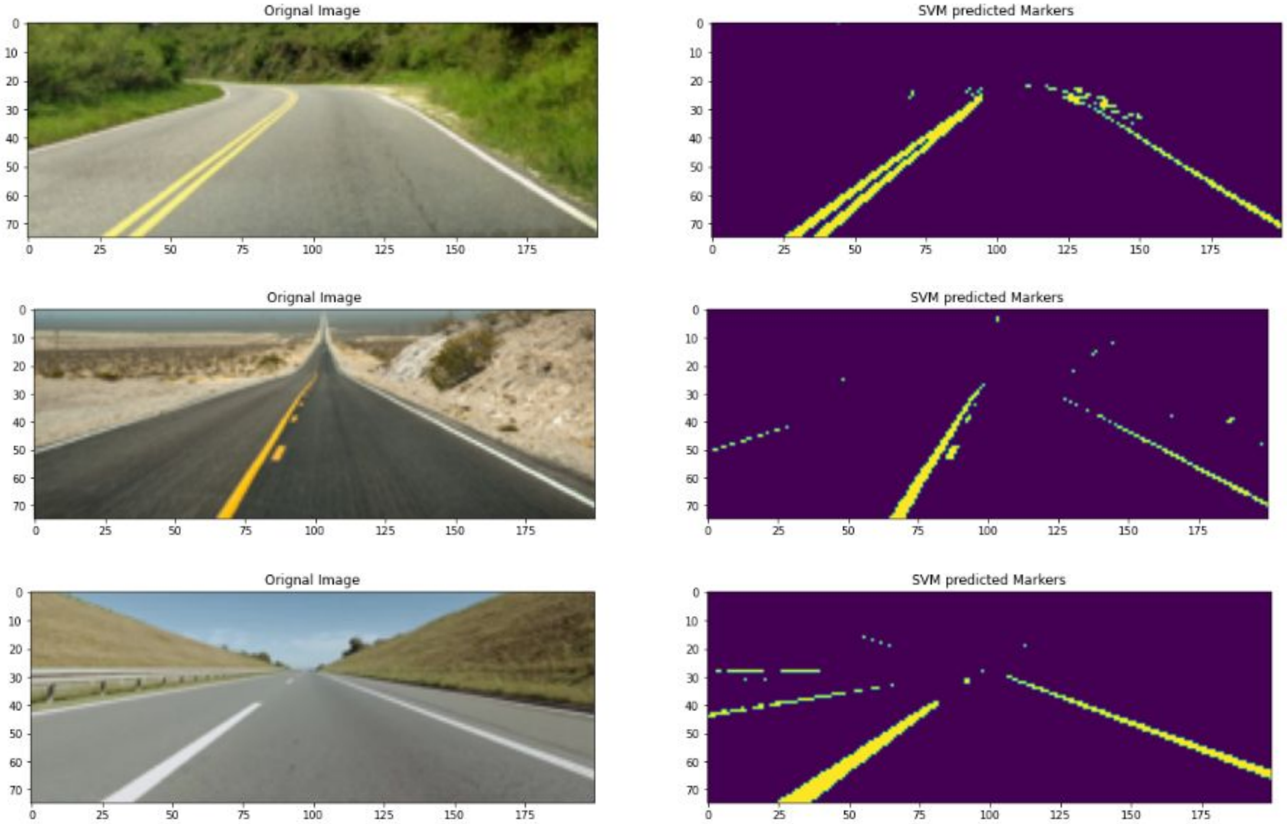






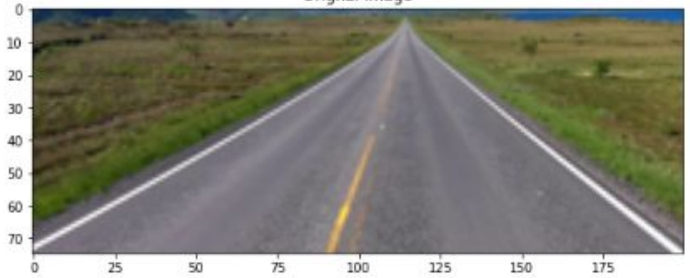


# Multi-picture Training Set

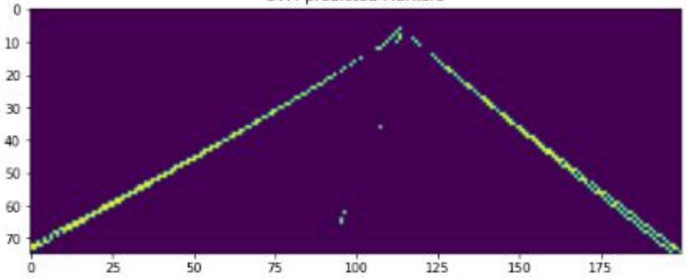


# Multi-picture Training Set Results 1

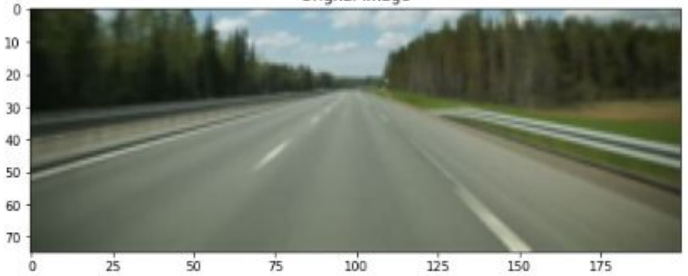
Orignal Image



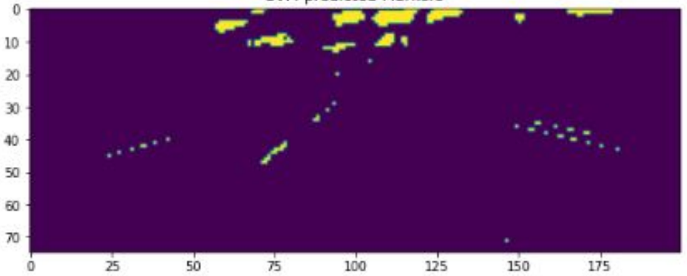
SVM predicted Markers



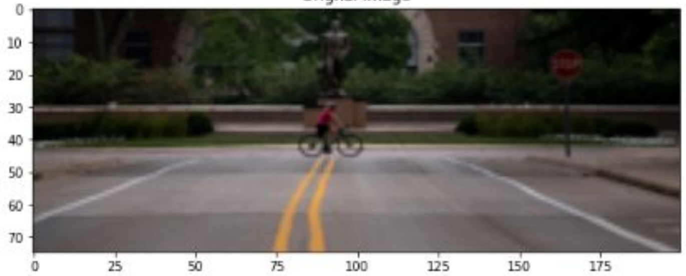
Orignal Image



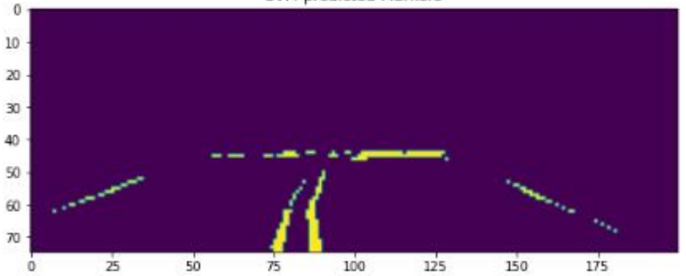
SVM predicted Markers



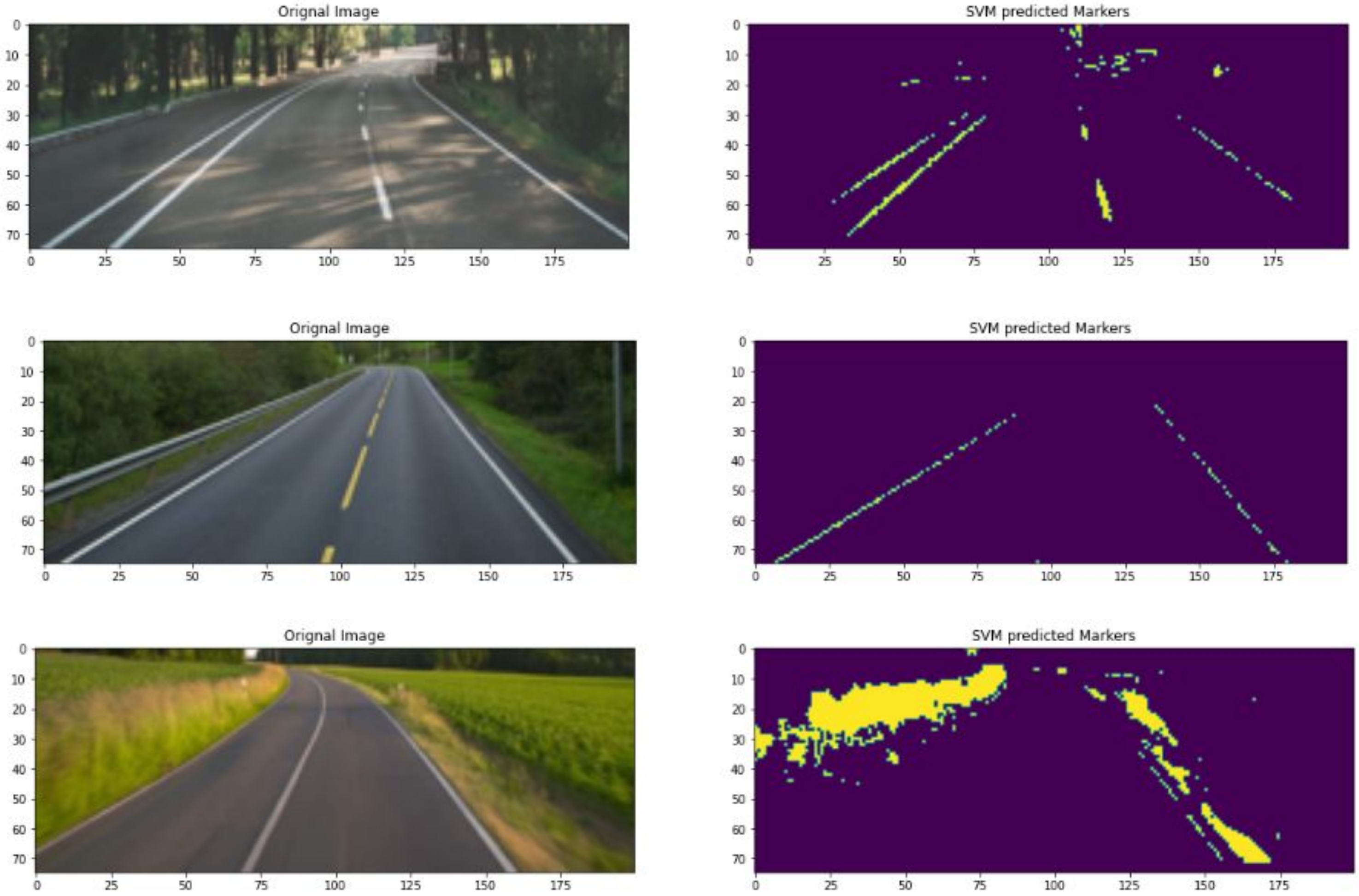
Orignal Image



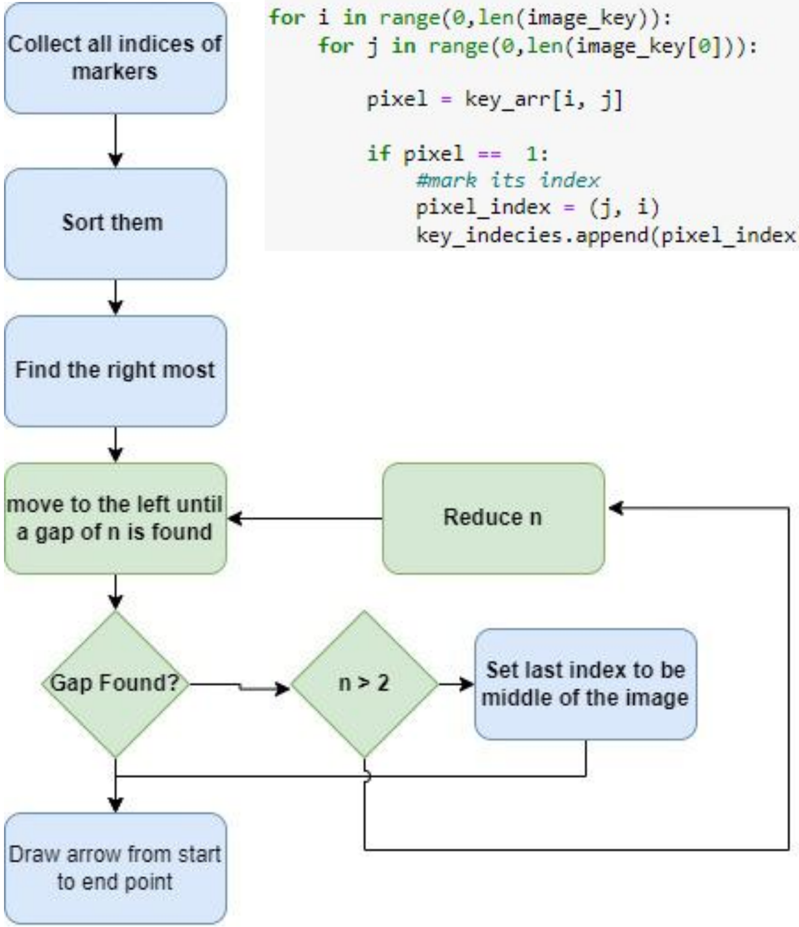
SVM predicted Markers



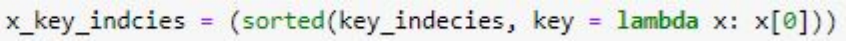
# Multi-picture Training Set Results 2



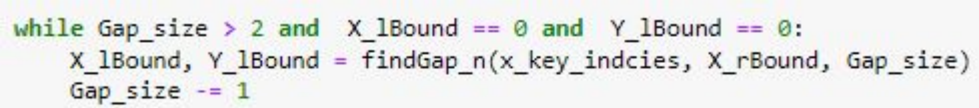
# Right Line Vector Algorithm


Collect all indices of markers
Sort them
Find the right most
move to the left until a gap of n is found
Reduce n
Gap Found?
n > 2
Set last index to be middle of the image
Draw arrow from start to end point

```
for i in range(0,len(image_key)):
    for j in range(0,len(image_key[0])):

        pixel = key_arr[i, j]

        if pixel ==  1:
            #mark its index
            pixel_index = (j, i)
            key_indecies.append(pixel_index)
```

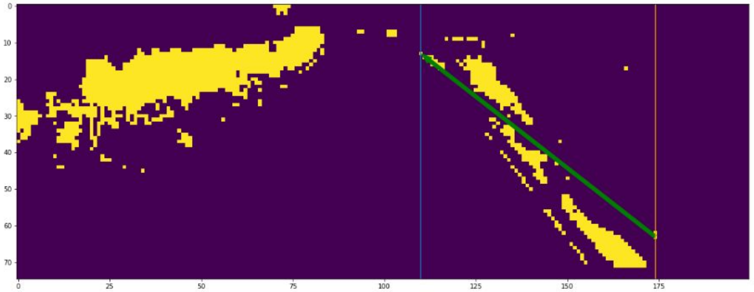


```
x_key_indcies = (sorted(key_indecies, key = lambda x: x[0]))
```

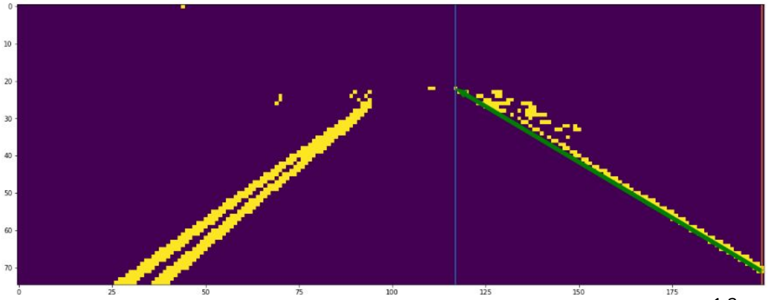


```
while Gap_size > 2 and  X_lBound == 0 and  Y_lBound == 0:
    X_lBound, Y_lBound = findGap_n(x_key_indcies, X_rBound, Gap_size)
    Gap_size -= 1
```

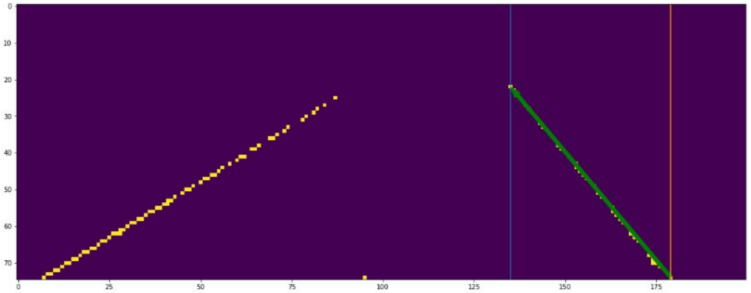
# Right Line Vector Results



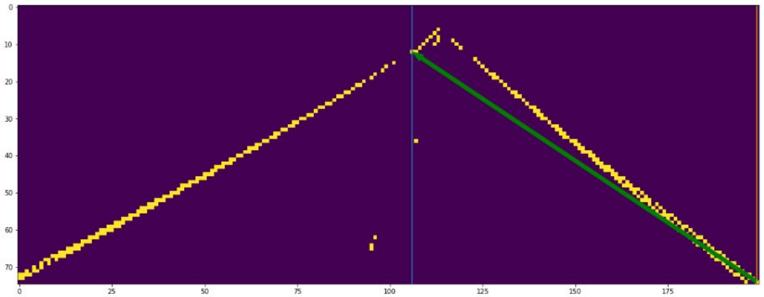
1,1



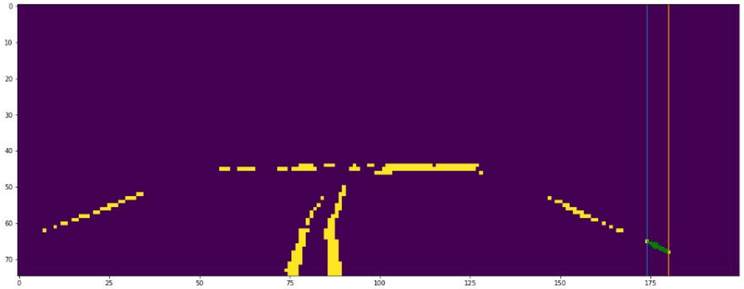
1,2



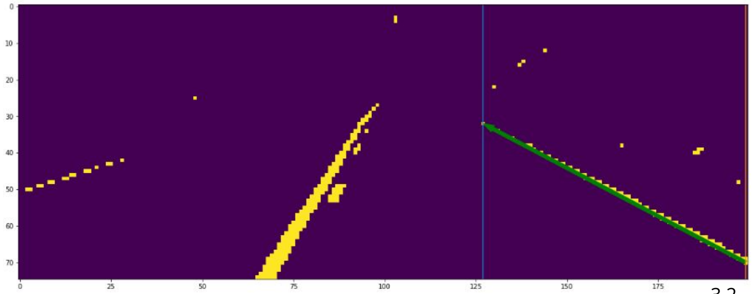
2,1



2,2



3,1



3,2

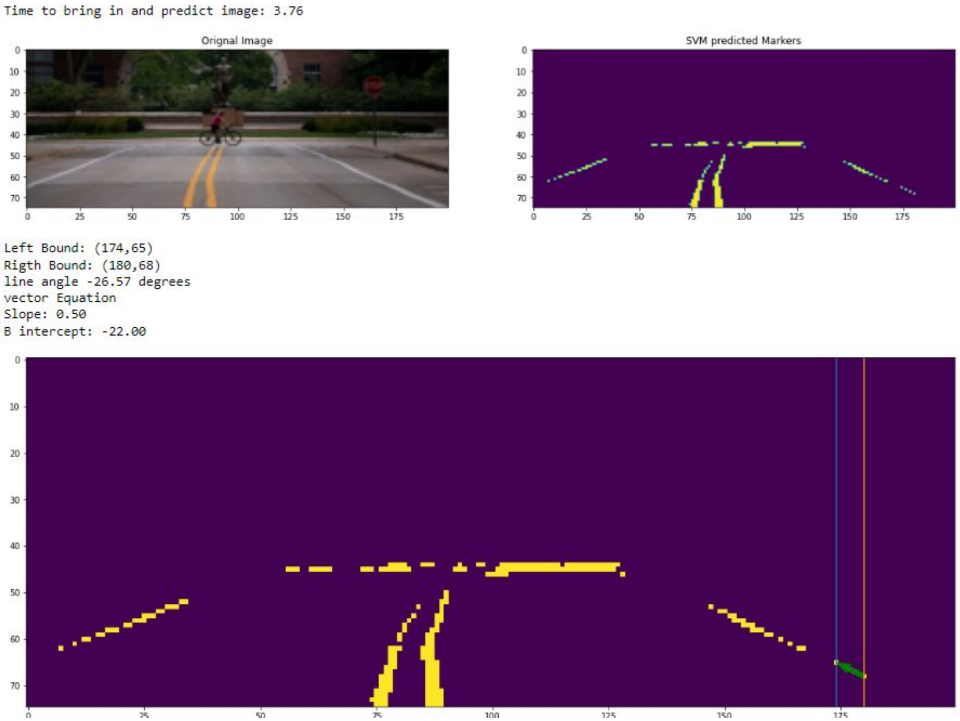
# Conclusion

## The RGB SVM method:

- can determine the pixels of the road
- very sensitive to noise

## The gap algorithm:

- successful on clean pictures
- also very sensitive to noise



```
Time to bring in and predict image: 3.76
```

```
Left Bound: (174,65)
Rigth Bound: (180,68)
line angle -26.57 degrees
vector Equation
Slope: 0.50
B intercept: -22.00
```