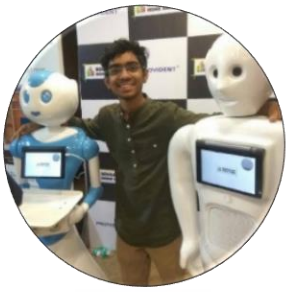
Summer 2019



Sai Siddartha Maram
Incoming Intern

Lane Marking Sizes
Lane markings are symbols shown on the lane, and the task we are looking to achieve is to get the dimensions of the lane marking. Starting of look at figure we see the pavement marking “2000”. The task is to approximate the near dimensions of this marking.

Figure 1: Image from the dataset

Understanding the Vanishing point and perspective transform:
We want to next get a perspective transform of the road so it appears the camera is placed over the road and the optical axis of the camera is perpendicular to the road. Why do we need to that, through intuition let us take an example of a small paining below.

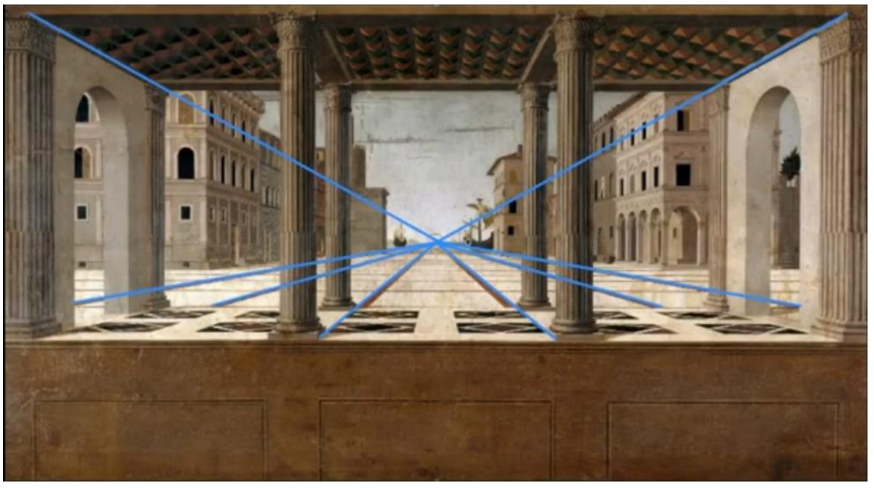
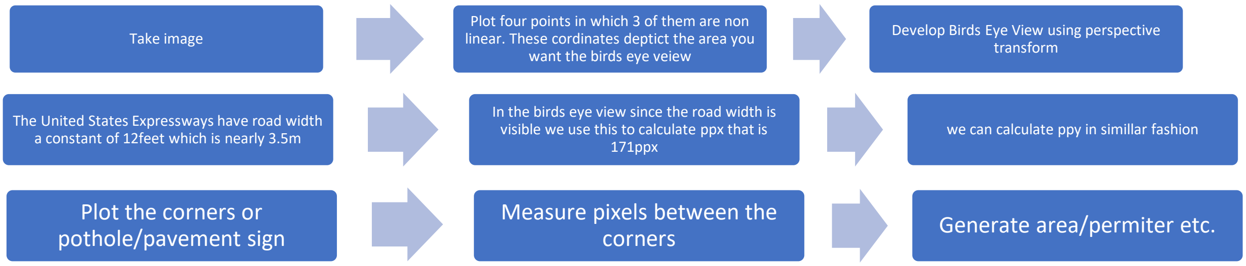
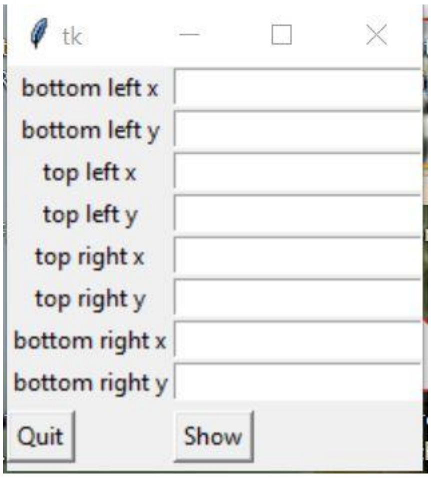


Figure 2: Architectural Veduta

The image in figure 2 is a painting named "Architectural Veduta" by Italian painter Francesco di Giorgio Martini. All the parallel lines meet at a single point called the vanishing point and the next important thing is, look at the square shaped tiles below on the floor, they appear like trapezoids, from this angle of vision we assume them to trapezoids which is wrong and are actually squares. Now for a bird flying above near the roof, when it looks down it appear clearly to be a square. [Inspired from Udacity Lane detection Self-Driving car nanodegree]. Now I our case at image 1 the lane marking also fits in the similar case study and for the same we need to develop a bird’s eye perspective of the same lane marking to get the true shape and dimensions.

Here we initially define a region where the lane marking is present and then, try move the camera over the lane marking. The approach we will be trying to use is as follows:





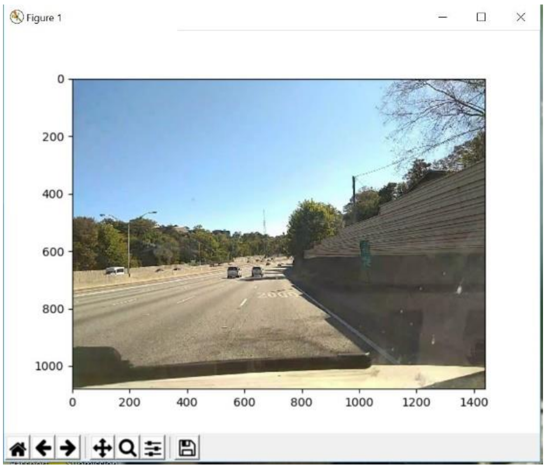


Figure 3: Application to enter image coordinates of the lane marking on the image.

Talking in terms of code, what we looking is for a coordinate (u,v) in the original image, what is the coordinate in the transform? Let it be (u_w , v_w). Now the relation between them would be as such

$$(u_w,v_w,1) = sH(u,v,1)$$

Now, what is H? H is the perspective transform matrix, we generate this matrix using the cv2.perspectiveTransform() function. The source points and destination size are taken in as parameters for computing the H as shown below in the code snippet and the result beside it.

```
source_points=np.float32([ [bl_x,bl_y],[tl_x,tl_y],[tr_x,tr_y],[br_x,br_y]  ])
destination_points = np.float32([ [0, 600], [0, 0], [600, 0], [600, 600] ])
image=cv2.imread(path_a)
matrix = cv2.getPerspectiveTransform(source_points, destination_points)
result = cv2.warpPerspective(image, matrix, (600,600))
result_image_y,result_image_x,_=result.shape
```

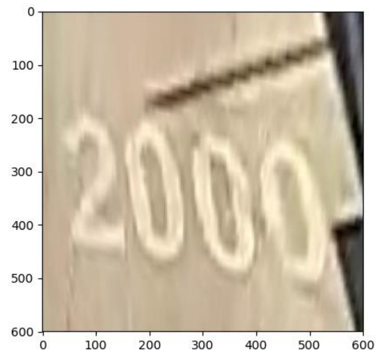


Figure 4: Showing the Code Snippet and results

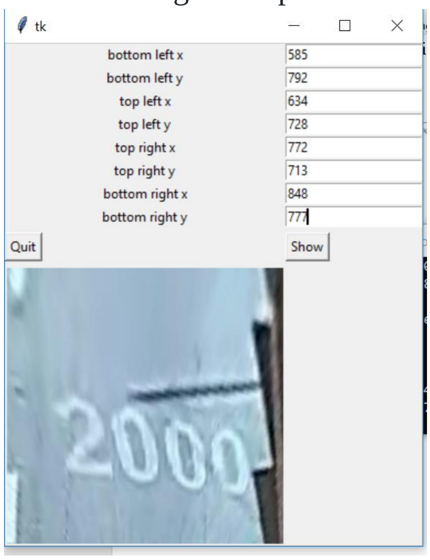
Estimating pixel to distance ratios

Now we have the true form of it, we want to calculate how many pixels represent one meter along the x-axis and how many pixels represent one meter along the y-axis. In the united states, the lane length is 12 feet, which corresponds to nearly 3.5 meters, and our perspective view is 600 pixels wide and 600 pixels in height, using this we have the following,
We clearly see that the birds-eye view we have is a 600 x 600 image, now the lane width which is 12ft nearly 3.5meters corresponds to 600pixels width. From here we have
3.5 meters → 600pixels
1 meter → 171 pixels
This is how we find the resolution in the x axis, but along the y-axis this is not the case, so we need to figure out a way to calculate along the y-axis.

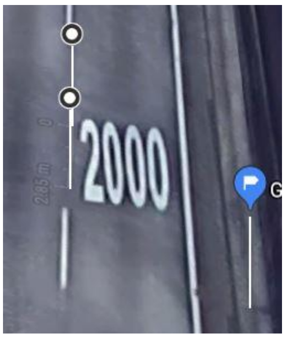
Lets say that position and of the road coordinate frame in camera coordinate frame (in bird eye view) is given with rotation matrix **R**=[r1 r2] and translation vector t. The same point (u_w,v_w) on the bird view corresponds to a 3D world point. The general concept of rotation matrixes is they are orthogonal in nature,

$[u\ v\ 1]^T = sM[X\ Y\ Z\ 1]^T$
$[u\ v\ 1]^T = sMR[X\ Y\ Z\ 1]^T$
$[u_w\ v_w\ 1]^T = sM[r1\ r2\ t][X\ Y\ 1]^T$
Applying perspective transform we have $[u_w,v_w,1]=sMH[r1\ r2\ t][X\ Y\ Z\ 1]^T$
With all the other variables known we can go for calculating r2. (For more detailed explanation I would highly recommend going through https://github.com/ajsmilutin/CarND-Advanced-Lane-Lines/blob/master/README.md) the above link walks through the math with a more visual perspective which is easier to grab.

Final Steps
We know have the image in its true dimension and also the corresponding ppm, we can check how many pixels wide and how many pixels in height the image is and get the final dimensions after multiplying with the corresponding factor. Please ding it compared with the satellite image on Google world.




```
height 3.4461484487768606m
length 2.2923976608187133m
```



Possible Extensions:

In the below image please find the thick green line, whose objective is to mark on the road how far the pole is, post that we again try using the birds eye perspective transform and try getting the distance as mentioned in the previous method as explained above.