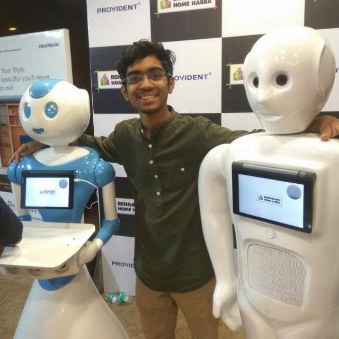
Submission

Preparation for Summer 2019





Sai Siddartha Maram

Former Intern Invento Robotics, India

Thapar Institute of Engineering and Technology

**Abstract**

Mapping of transportation infrastructure and its health is a critical area when we discuss the bringing up smart-cities. Monocular images are easy images which can be collected though pinhole cameras. Photometry deals with extracting geometrical information from these images. The report discusses methods, validation, applications and insights generated form the dataset.

**Keywords**: Birds Eye View, Aspect ratio, GPS, UTM N16, Camera Matrix, Distortion

**Understanding the Dataset**

Understanding the dataset is a critical element before beginning programming, the dataset provided has been obtained from data collected from GeorgiaTech Sensing Vehicle (GTSV). The GTSV consists of a camera which faces straight ahead and collects images after a particular interval. The sign annotations of each expressway, contains the coordinates through which you can extract the center of the sign located in the image. Converting the json file into a csv file which allows easier parsing of it from the programming perspective. We have the sign annotations as follow in csv format for various expressways.

Similarly, we have, csv files which indicate the real-world coordinates of cameras while taking the image. Other important file is the ground truth of the sign in the real world. This will never be used as input in the algorithm/application but will only be used for validation and evaluating our algorithm.

**Understanding the images and camera**

Images taken from cameras have certain extend of distortion associated it. This distortion can be identified and eliminated using the intrinsic parameters of the camera. The config.yaml, contains all the particular details and intrinsic parameters specific to the camera. Here we have two kinds of cameras which are in use, the camera associated with GTSV which takes images of the dimensions 2248 x 2048 pixels and then the AllGather application from the smartphone camera which takes images of the dimension 1140 x 1080 pixels. Through camera calibration we can develop two very critical matrices which will be used across multiple programming sections. The camera matrix and distortion matrix of the different cameras are as follows:

Camera Matrix of 2248 \* 2048 :

|  |  |  |
| --- | --- | --- |
| 2468.6668434782608 | 0 | 1228.876620888020 |
| 0 | 2468.6668434782608 | 1012.976060035710 |
| 0 | 0 | 1 |

Distortion Matrix of 2248 \* 2048

|  |  |  |  |
| --- | --- | --- | --- |
| 0.00125859 | 0 | -0.00010658 | 0 |

In the same manner we have matrices for the smart phone application also.

**Please do look into submission 1.0, where the approach in terms of calculation has been clearly elaborated**

**Task 1.0**

In this particular section we will be trying to develop a user interface where, two images are bought in front of the user, the user is allowed to select two points, one on both of the images. The application keeps track of these points and then based on the algorithm provides the corresponding UTM N16 coordinates, these UTM N16 coordinates can be easily converted into GPS coordinates which can be mapped as required.

Details and instruction to set up and run the program have been clearly explained in the video submission. The application takes in two input arguments which are images on a particular expressway. The rest is left to the program to track user cursor coordinates and on double click store image coordinates. Once two points are selected one from each image, the user can quit the application and note down the coordinates. The user is free to use the coordinates as required.

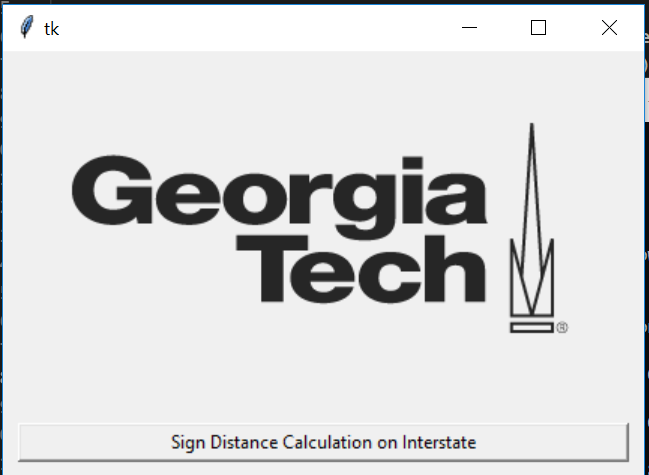
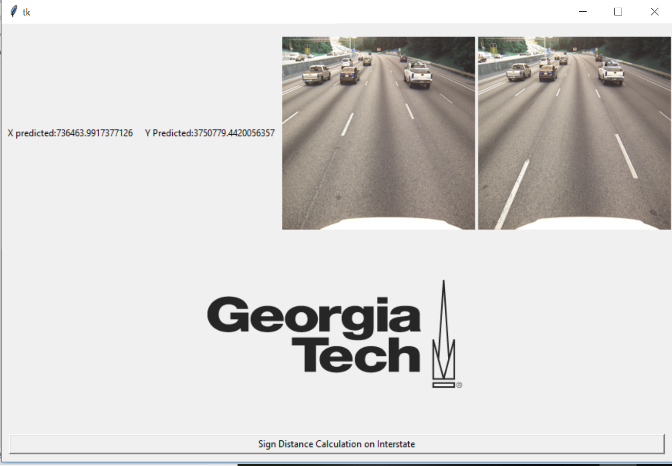


Note the user marked sign, with the mouse cursor, the user marked sign is indicated by a blue marker and its points coordinates are stored within the application. Once the user has finished selecting his points, the algorithm outputs the predicted location   
Predicted: (736463.9917377126, 3750779.4410056357)

Ground Truth: (736465.737487371, 3750769.21175686)

The corresponding GPS coordinates can be found and marked on mapping applications. To the left marked in a blue pin is the predicted and the pin marked in red indicates the actual location of the sign.

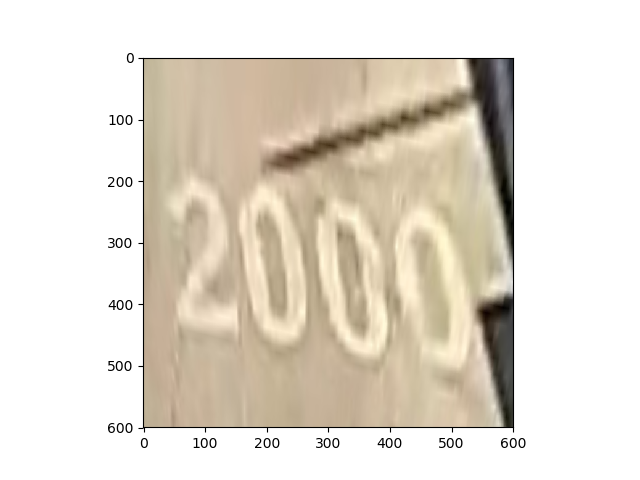
To remove the burden of selection of signs from the user, we can achieve the location of the center of the sign using the given annotations. For the selection of images we can use the developed GUI. This GUI prompts the user to select two images, which have the sign.



|  |  |
| --- | --- |
|  |  |

The algorithm then performs the concerned calculations and displays the result on the screen shown above.

**Dimensions of Pavement markings and Potholes: (Choose the smartphone images since they have pavement signs)**

Consider the following image, here we see the pavement marking 2000 written across the lane. Whenever we look to obtain the area of an object on planar surfaces on an image, our intuition says pick the four corners and then calculate the dimensions. But on repeated testing what has been found is this is only valid if and only if the camera is placed exactly vertical about the object. So we have to figure out a way to develop a birds eye perspective to generate the true dimensions of the pavement marking or pothole.

Here in this case 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

After exploiting the camera matrix and the perspective transform matrix which we used to get the birds eye view, we have pixels per meter along the y direction as 58pixels.

Now marking the four coordinates of the pavement sign we get a rectangle which is 164pixels along the y\_axis and 418pixels along the x\_axis.

2.45 meters

171 pixels 🡪 1meter

418 pixels 🡪 2.45m along the x-axis

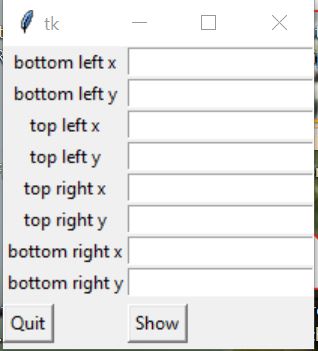
Similarly,

2.9 meters

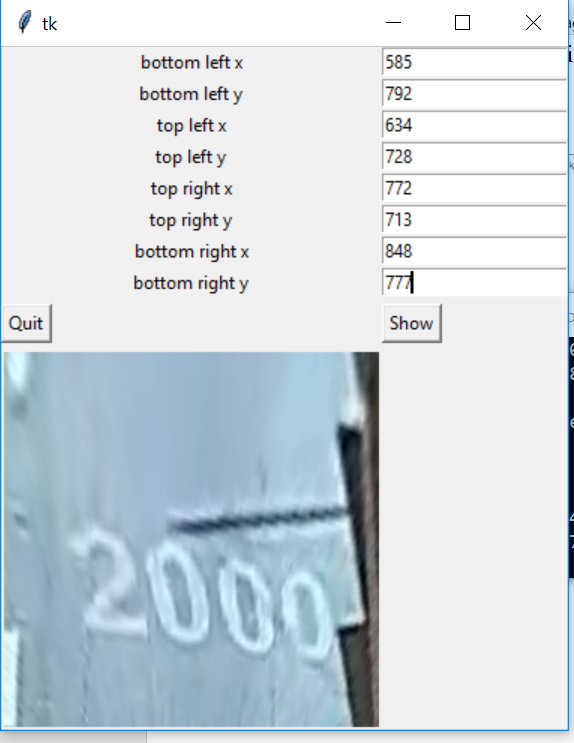
58 pixels 🡪 1 meter

164pixels 🡪 2.9 m

These are screenshots from google satellite view which gives an the width and height of the pavement sign, we see that the sign dimensions we have measured is similar.







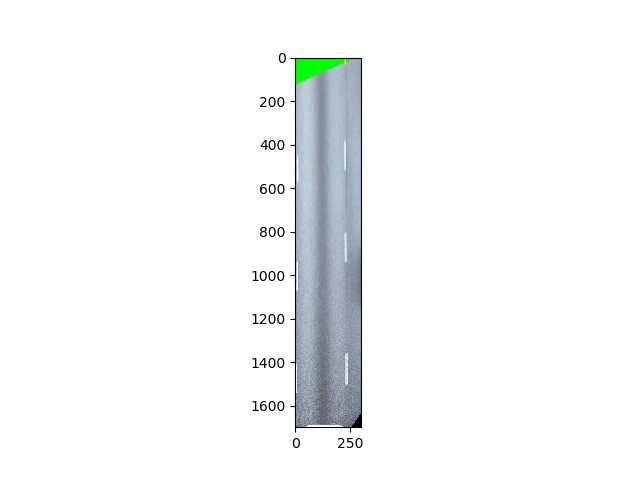
**Error Analysis and Possible different methods for distance calculations on interstate expressways**

Apart from the method mentioned in the appendix, one possible way of obtaining the distance I looked to explore was the Birdeye view perspective. There is no specific application as such developed for this but the algorithm has been developed.

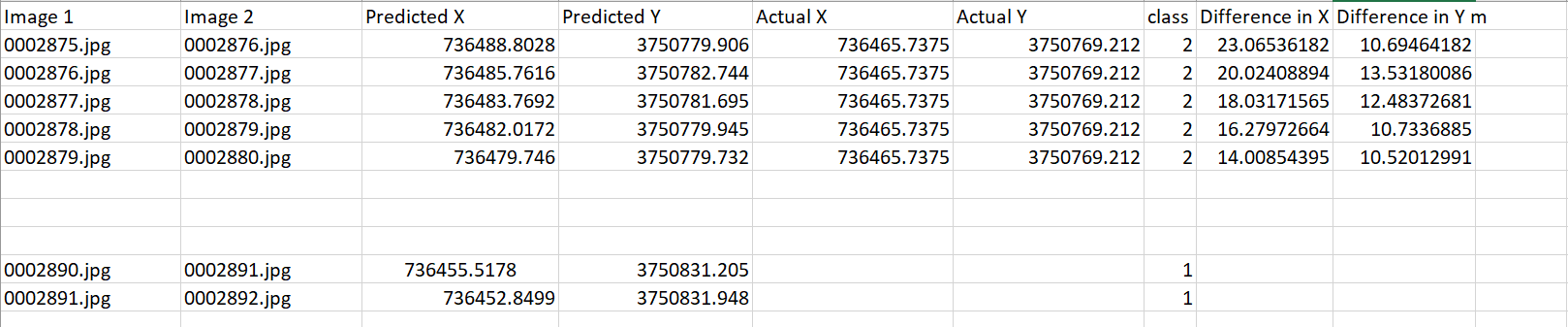
Consider the image



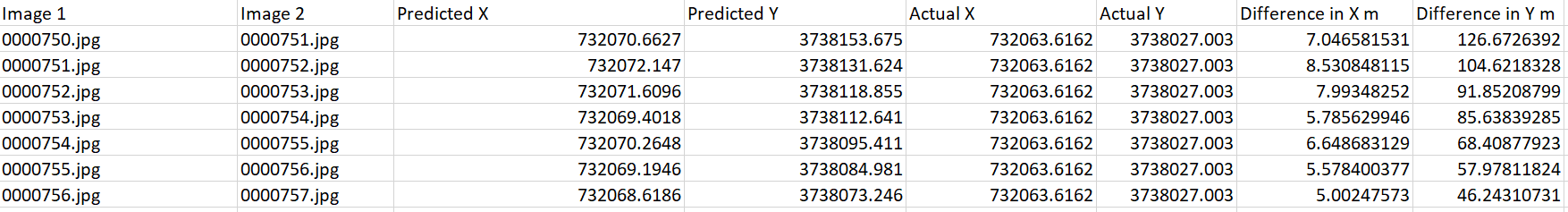
Consider the image where we need to find how ahead the green sign on the left is. Now project this bottom of sign using our draw on image application onto the road. To get the true perspective of the distance we will need the top view. We generate this through perspective transform matrix.

Now we have generated the birds view of the road where we can get the true dimension of the road. The Birds view is 300pixels wide and contains one lane of the road which is 12 feet of 3.5meters. This indicates, we have 83pixels per meter. Using the perspective transform matrix we can calculate the pixels per meter along the y axis which is 55 pixels per meter along y-axis. Now the we have 1700 pixels from the car to the green mark, which leads to the conclusion 1700/55 meters which is nearly 31meters. Which is nearly 15 meters off the actual distance. This error raises because of two major issues.

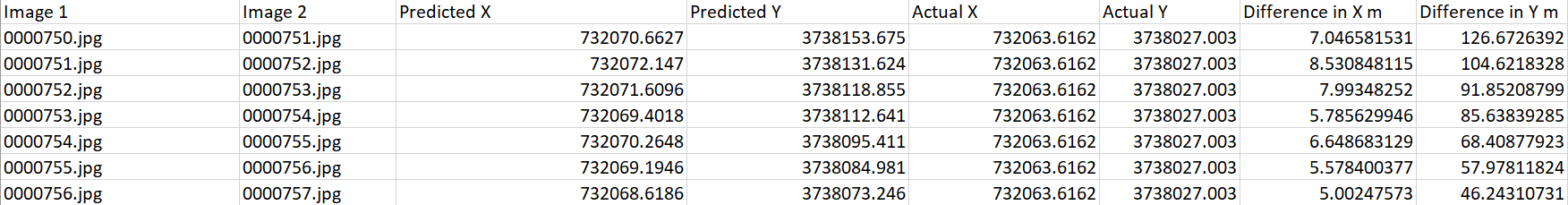
GTSV i75 Error Log

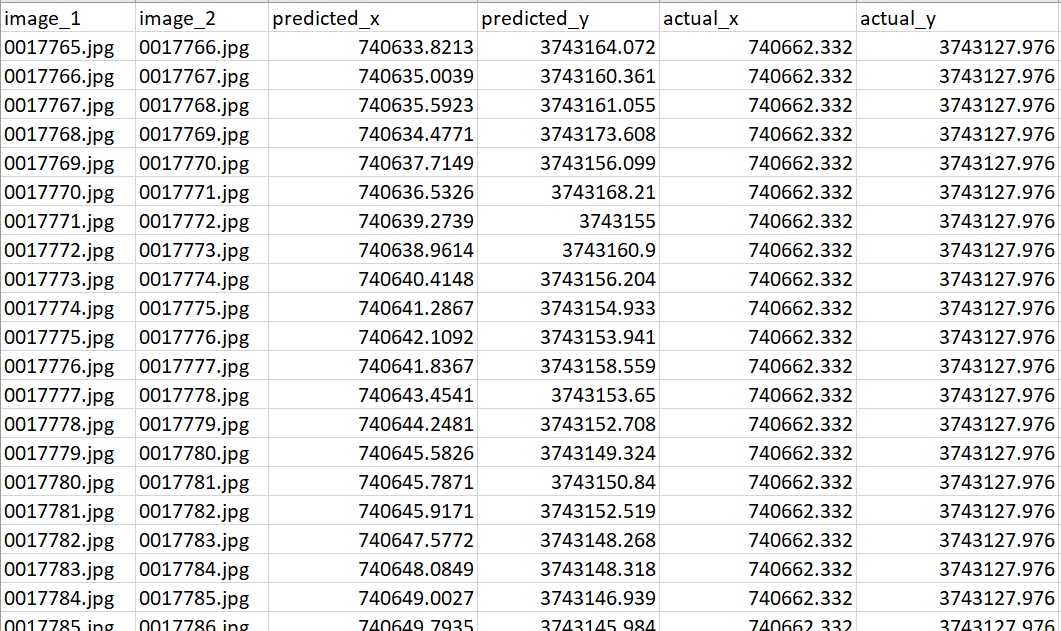


GTSV i285 Error Log

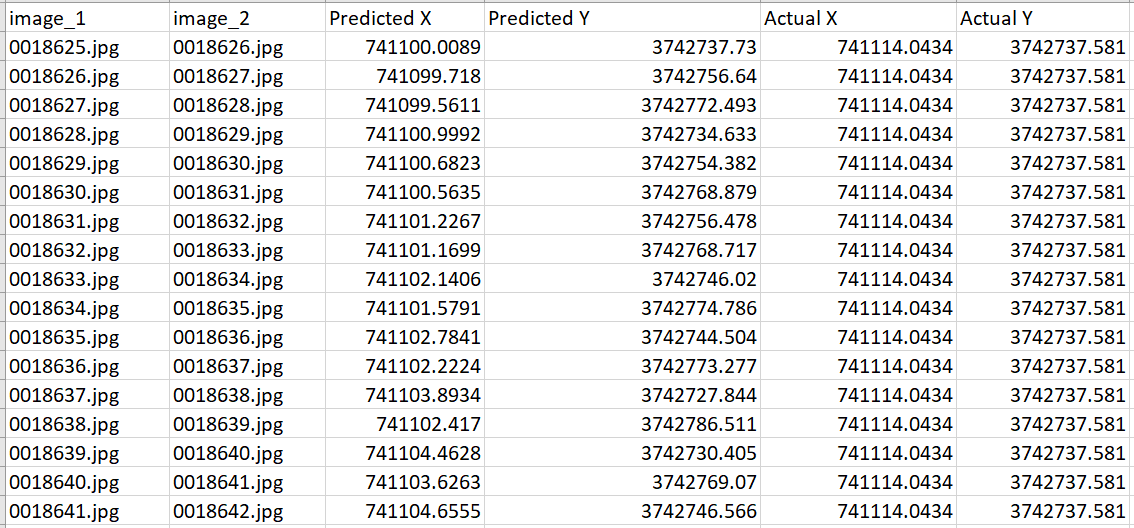


GTSV StateWay



SmartPhone 0

SmartPhone 1



Things to work on:

Application development is a full time task, and currently I could not convert what I envisioned in terms of work flow, UI/UX into