Bye Bye Google Maps. Hello Locus Maps.

At Locus, we plan the on-ground movement for logistics companies. Unlike some of the new age companies like Uber, Ola, Swiggy who need very ‘real time’ movement; a large part of logistics companies (e-commerce, FMCG etc) still relies on planning the movement ahead of time for efficient delivery.

Locus currently powers more than 25% of e-commerce in India. Most of the Locus clients who use Locus ‘Dispatcher’ to plan their delivery boys’ movement, also have ‘Locus On the Road’ (LOTR) app installed on their apps. Using the app, the delivery boy can mark his task complete & collect signatures from the recipient.

The LOTR app also sends the delivery boys’ coordinates to Locus server on regular intervals. Locus provides a live view to warehouse managers to see where their delivery boys are what path they have taken, how much delayed they are etc. The coordinates sent by LOTR app alongside the timestamp helps us to show the live information to the warehouse manager.

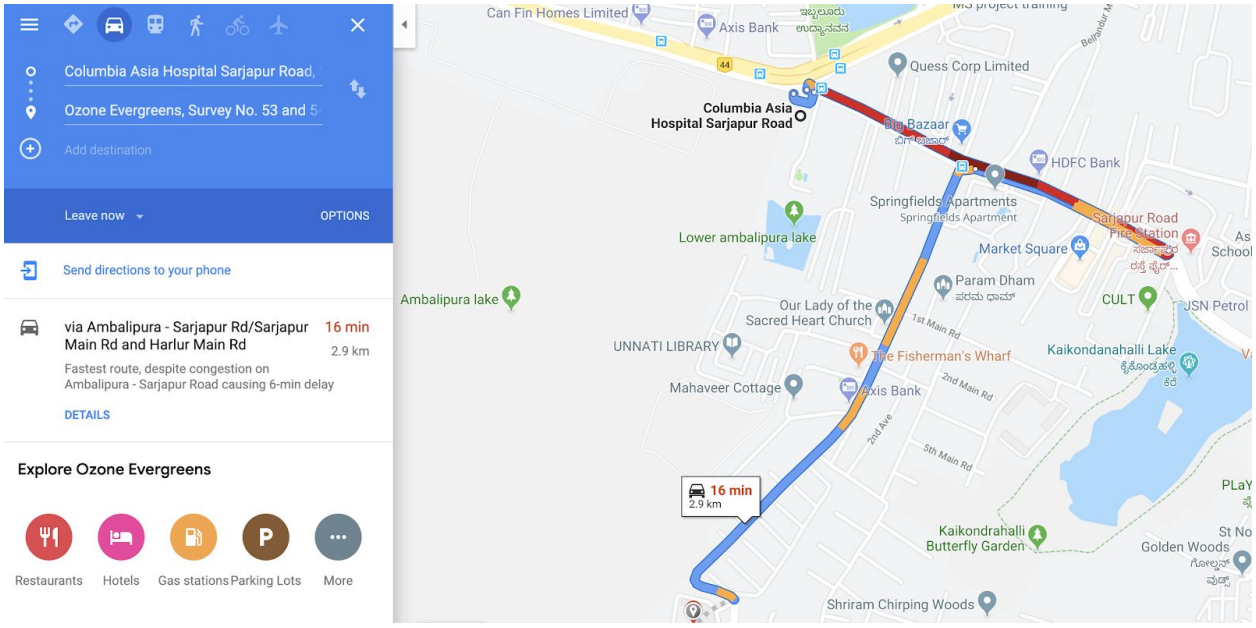
Your mission, should you choose to accept, is to use the LOTR GPS data (we call this *trail data*) to build an equivalent of Google Maps. Let us break this problem down to more ‘implementable’ parts.

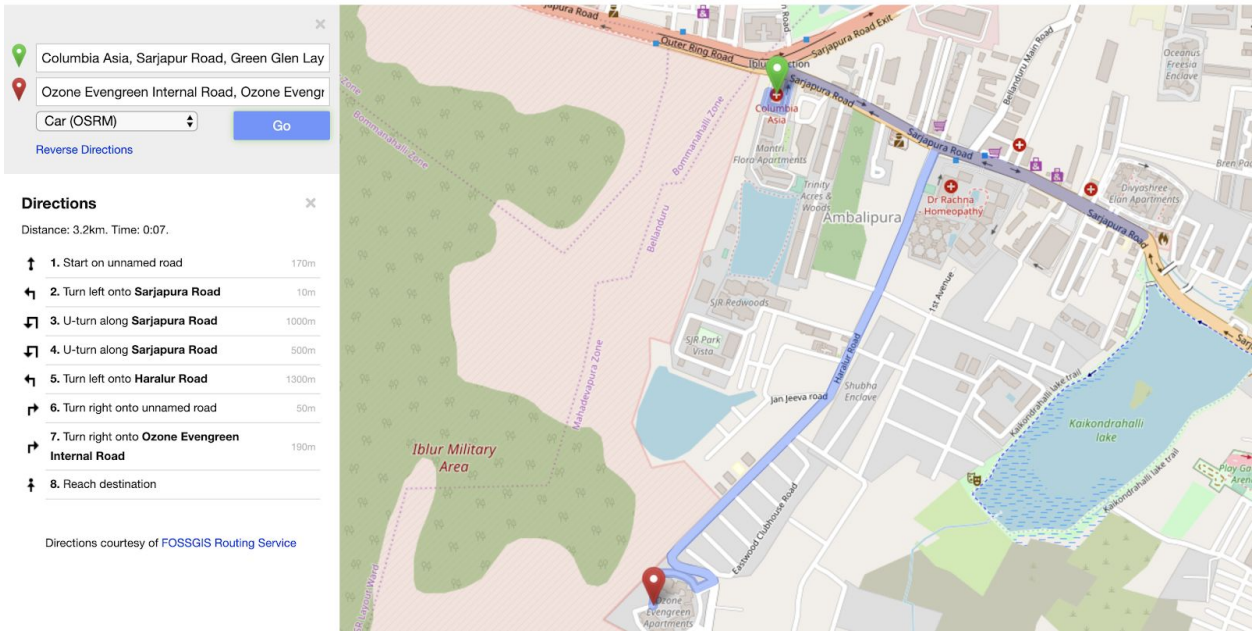
So what exactly Google Maps does for you ? When you give start & end location and time of the day (you would usually be giving it as that instance when you are looking up the route), it tells you the most ‘appropriate’ path. The word appropriate is mostly path optimized on time taken to cover the distance from the start point to the end point. Afterall, we are in Bangalore.

Inputs : We will provide you the Locus GPS data around **Sarjapur road area** for entire **July**. You can download the data here. The data will be in the format - rider-id, latitude, longitude, timestamp . The latitude longitude across riders when plotted on a map looks like below.

Problem : You need to predict the time taken to travel between two locations around Sarjapur road area. You can get the path between the points from Open Source Routing Machine (OSRM) hosted at Locus. You can then use Locus trail data to **predict** the time taken for the path.
OSRM URL - http://osrm-1644136849.us-east-1.elb.amazonaws.com . Example output can be seen here.

Why OSRM ? Usually, OSRM gives exactly the same path as that of Google maps for two closely located points. OSRM uses Open Street Map (OSM).





Validation : Predict time taken navigating between two locations from your engine. You can then compare your prediction with Google map's time prediction. Above are two examples - one from Google Maps & the same path between the two points suggested from Open Street Maps. You can see that while the path suggested is the same, the time suggested by Open Street Map is 'wrong'. Your mission is to fix this time :).

Caveats :

1. GPS data is not 'clean'. There could be 'time jumps'. Plus not all latitude, longitude will lie on the road. You would need to clean the data first before you can start modeling your prediction. Simplest way to clean is to ignore these erroneous GPS data.
2. A rider could be stationary for a long time at some locations like taking a break. You need to come up with an algorithm to remove such outliers while you do the prediction.

Backend Design :

While Data Science folks can work on coming up with an efficient predictive model, the backend folks should worry on structuring the model in a way that calculation of time between point A & B can be done 'faster'. We are NOT talking about caching & distributed systems here. We would mostly look at how you are able to structure your data for a quick/efficient lookup.

Frontend/App Visualization :

We would like to see how you can show us 'traffic snapshot' on a map averaged over day / week / month. Ideally, the road parts that have slower average speed of vehicles at a given point of time should be shown in darker shade of red. The parts that have fast moving traffic should be shown in shades of green. **The problem statement is to visualize how the traffic situation changes over time**. The evaluation criteria is not entirely engineering centric, but also product centric. How have you thought about to come up with great visualization.

Resources :

1. Uber tried solving similar problem which is documented at https://d3i4yxtzktqr9n.cloudfront.net/web-movement/static/pdfs/Uber-Movement-Speeds-Calculation-Methodology-56b3b1999e.pdf . You may read till 'Map Matching' & ignore the parts after that.

2. Think about how you need to predict & store the data. Right storage will help you with both the problems - predicting time between point A & B and showing traffic snapshot wrt to the average speed at a certain time of the day.