Traffic Intelligence

GUI for semi – automated road user tracking

Akhil Vakayil

Indian Institute of Technology Madras

Objective

The objective of this project is to build a GUI (Graphical User Interface) to manually correct the output from Traffic Intelligence video tracker (https://bitbucket.org/Nicolas/trafficintelligence)

Error: Distorted Trajectory

The tracker often results in distorted trajectories, as shown in the image below: L



This error can be corrected by either shifting the centroid of features or deleting the erroneous features

A user friendly method to shift the centroid would be to trace the right trajectory and record the coordinates as a new feature



Now deleting all the features of the object in the frame range of the trace and adding our new feature will result in the correct centroid and the trajectory is as expected

Implementation

The first step is to identify the object when the user clicks on it. The function *findObject* serves this purpose. It takes the coordinates of the click and frame number as input and then checks for the objects present in that frame and then pin-points the object based on its pixel range.

Now as the user traces the right trajectory the coordinates, object id and frame number is recorded. The coordinates are then converted to the real world coordinates using the homography matrix.

Function *sqlTrack* deletes records from table *positions* in the frame range of the trace in order to delete all features within that frame range. Before deleting the least trajectory id in that frame range is recorded and this is the trajectory id given to the new feature.

In case few frames are missing while tracing (value set to 5, can be edited), *sqlTrack* fills in data for those frames with the average of previous known frame and the next.

How to use

After running the code, pressing key *t* takes you to the track mode, indicated by a yellow border highlight. If you wish to do the tracing frame by frame change the *cv2.waitKey* value to *cv2.waitKey(0)*, 25 to 30 for normal speed and higher values to slow down the video.

Click on the object you wish to edit and trace the trajectory, just click on the object if the edit is to be made at a single position. The trace takes effect once you close the program and will be visible in the next run.

Error: Single object tracked as 2 or more objects

The image below clearly depicts the issue:



The solution to this issue is to merge the objects as one. A user friendly way to do this would be to select the objects to be merged as one while the video is running.

The features of the objects can be linked to one single merged object. There will be cases where an object is not tracked for a stretch, in such cases we can add a feature with constant coordinates for that stretch. Later with the track feature the user can add the required trajectory.

The image below shows the trajectory after merging:



Implementation

While in merge mode, the objects the user clicks on gets added to the *mergeList* and when the user comes out of the merge mode the function *sqlMerge* is called.

*sqlMerge* deletes all the objects present in in *mergeList* from table objects, except the first one. It then links all features from *objects\_features* for the deleted object to the first object in *mergeList.*

If there is discontinuity in the trajectory, the function looks for the feature with max *trajectory\_id* before the discontinuity and then extends it with its position remaining the same throughout the vacant stretch.

How to use

Pressing *m* takes you to the merge mode, indicated by a green border highlight. Once in merge mode select the objects you wish to merge by clicking on them and press *m* again to come out of the merge mode and the merge to take effect.

If any discontinuity exists go to track mode and trace the trajectory for the vacant stretch.

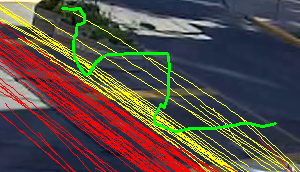
Error: 2 or more objects tracked as one

The image below depicts the issue:



We need to group the features into separate objects. We can ask the user to select the features which he wants to separate out as a new object. A user friendly way to do this would be to show all the features and let user select the desired features by dragging the cursor over them.

The image below shows the feature selection process:



The image below shows the trajectory after the split:



Implementation

When the user drags over the video the coordinates get added to the list *cArray* (this happens inside the function *coordinates*).

Every time splitting happens a new object is being created, *newObjID* increments itself accordingly.

Whenever in split mode, function *findTrajectory* is invoked which looks for any feature being selected by the user depending upon the coordinates in *cArray* and then adds the feature id to list *splitSelect*.

Function *sqlSplit* finds the object being split based on the feature id present in the list splitSelect. It then creates a new object in table *objects* with same *road\_user\_type* and *n\_objects* as the original object.

sqlSplit then updates the *object\_id*, in table *objects\_features*, of all features present in *splitSelect* to the new object id just created in the table *objects*.

How to use

Pressing *s* takes you to split mode, indicated by a blue border highlight. Once in split mode you will be able to see all the features, select those features you want to separate out as a new object by dragging the mouse over them or individually selecting them by clicking on them.

Press *s* again will bring you out of split mode and invoke *sqlSplit* to initiate the split. You will be able to see the new object in your next run.

Future work

* Removing unwanted objects



As seen in the image above, shrub movement is being detected as an object

Solution: Delete the object and its features

* Deleting unwanted features

The user should be able to select the trajectory and delete them

* Updating *velocities* table according to the edits made by the user
* Undetected vehicles

Sometimes some vehicle go undetected.

Solution: The user should be able to add features

OR

Manually initializing the tracker with the option of dynamically adjusting some parameters to detect enough features on the object