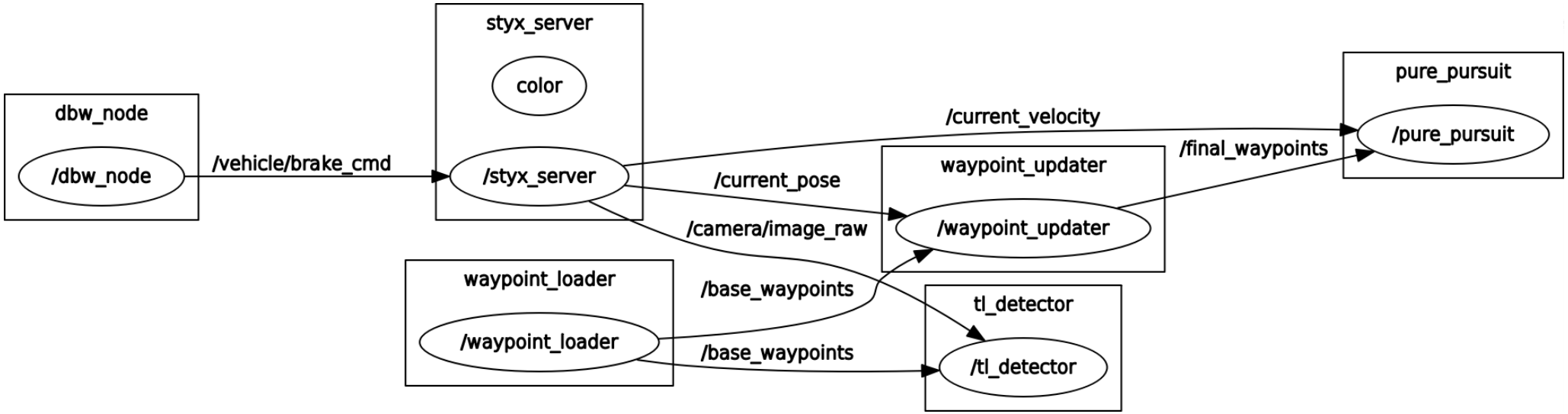
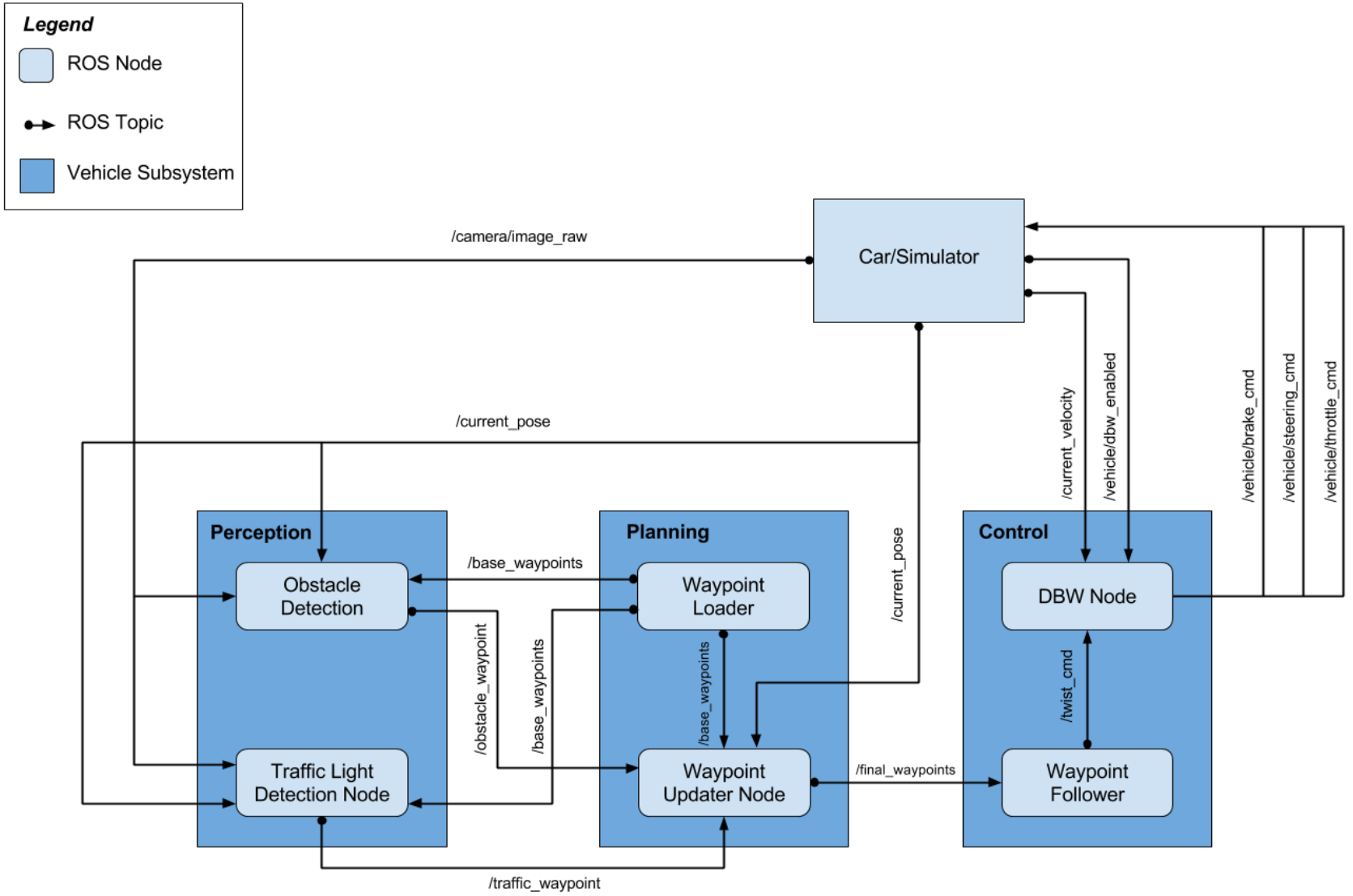
RQT Graph


dbw_node
/dbw_node
/vehicle/brake_cmd
styx_server
color
/styx_server
/current_velocity
/current_pose
/camera/image_raw
waypoint_updater
/waypoint_updater
/final_waypoints
pure_pursuit
/pure_pursuit
waypoint_loader
/waypoint_loader
/base_waypoints
/base_waypoints
tl_detector
/tl_detector

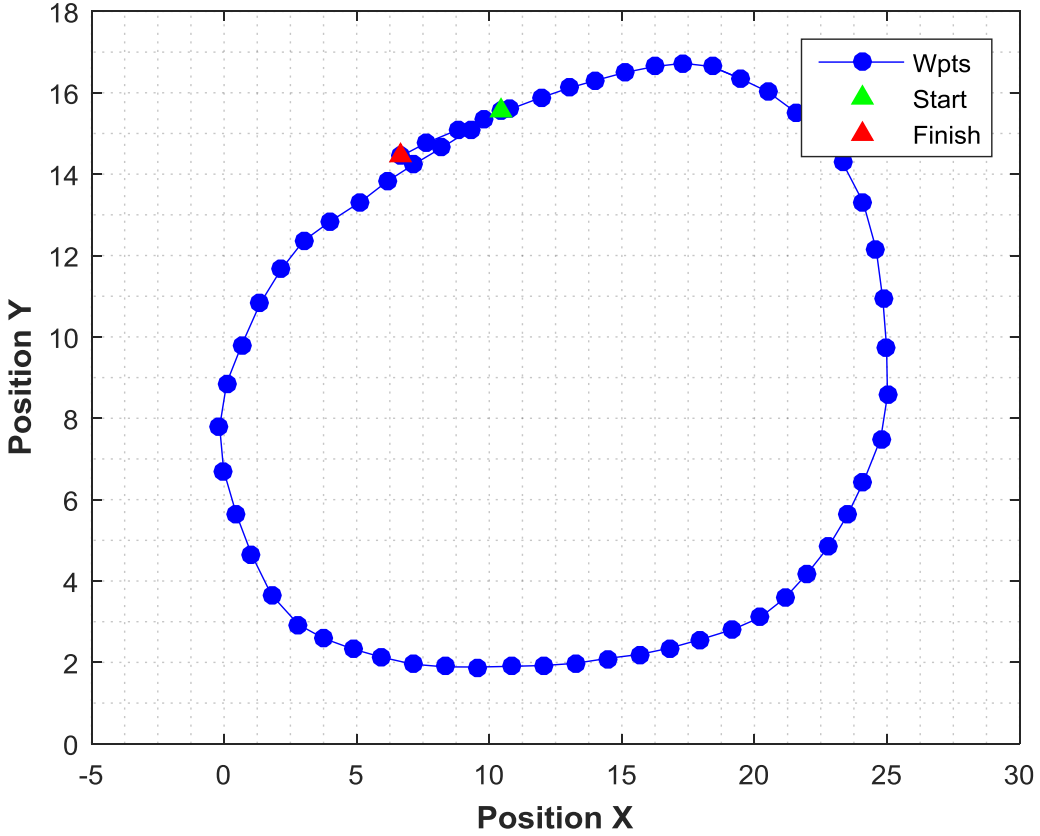
ROS Nodes Layout

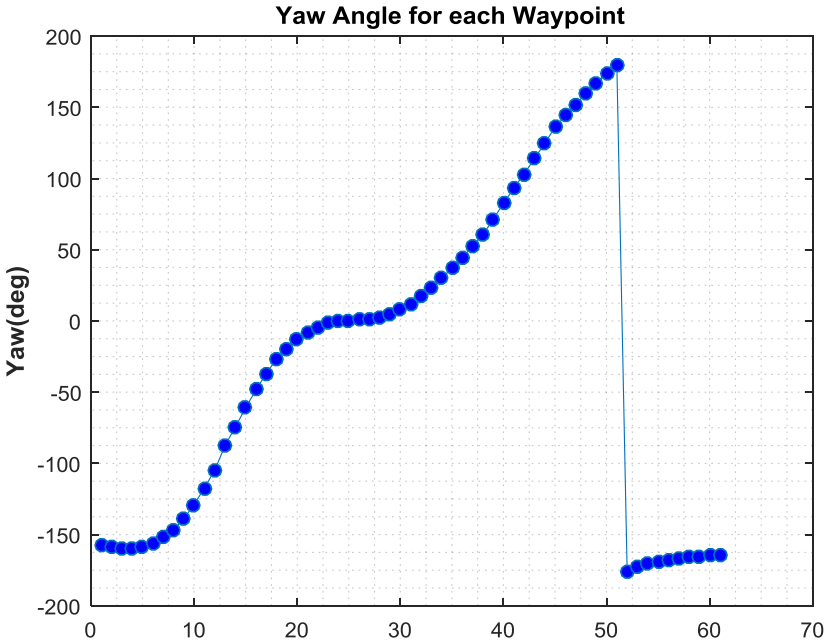


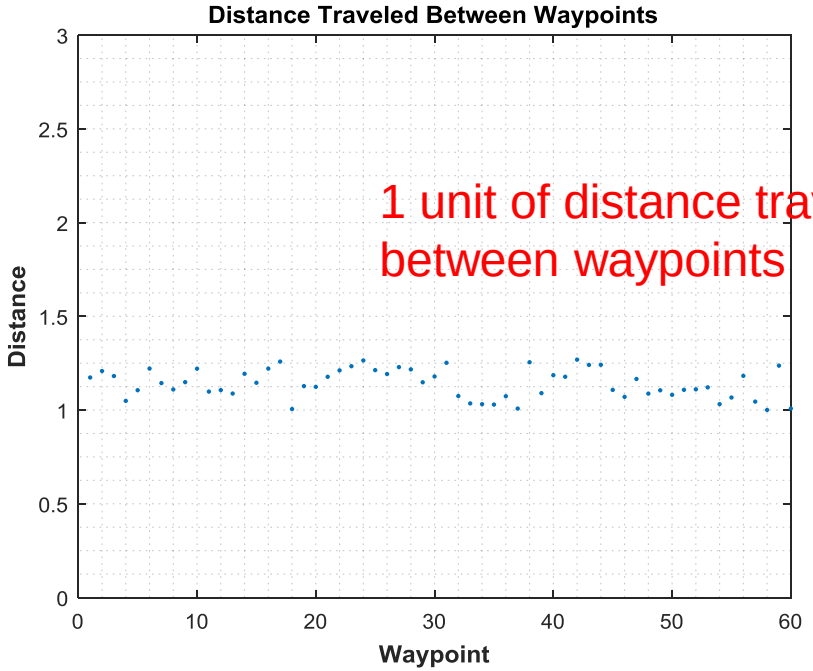
Waypoint Files Summary

Filename	Column headers	Notes
churchlot_with_cars .csv	Pos x,y,z, yaw(rad)	Yaw is in rad (+/- Pi)
sim_waypoints.csv	Pos x,y,z, yaw(?)	Yaw is constant – bad? Same path as wpt_yaw and wpt_yaw_const
wp_yaw.txt	Pos x,y,z, yaw(deg)	Identical to wp_yaw_const.txt
wp_yaw_const.txt	Pos x,y,z, yaw(deg)	Identical to wp_yaw.txt Yaw is in deg(0-360)

churchlot_with_cars.csv



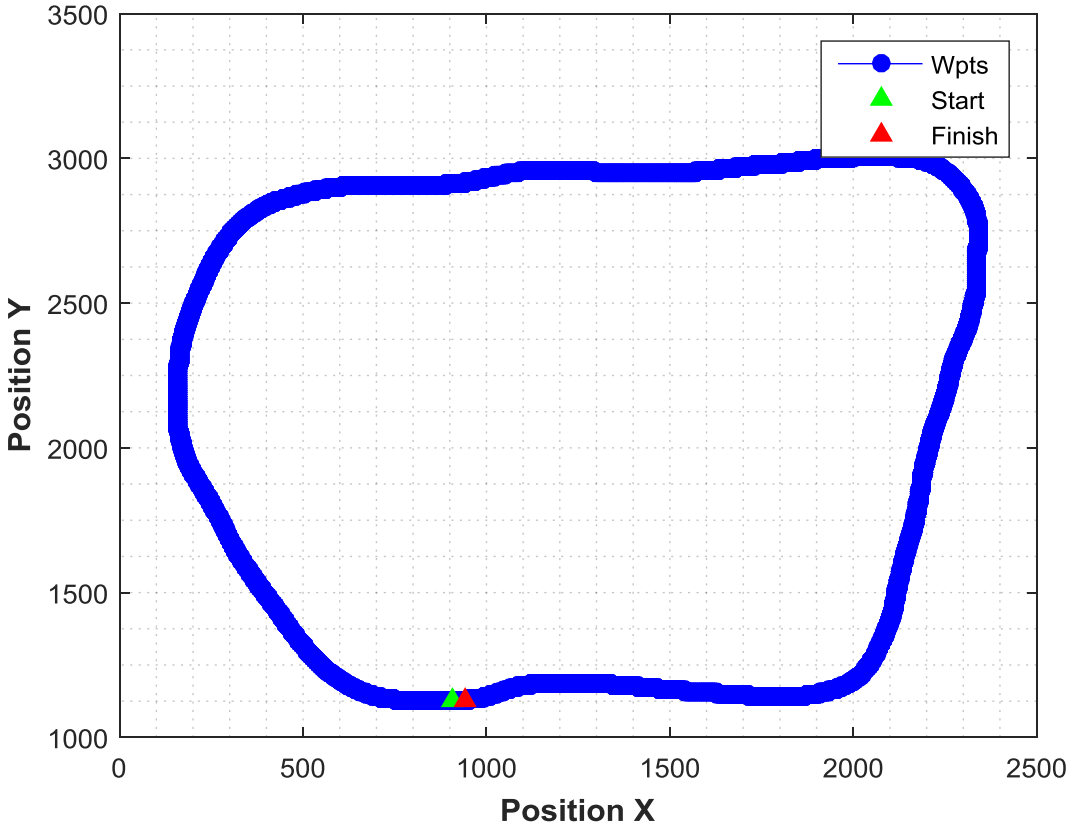


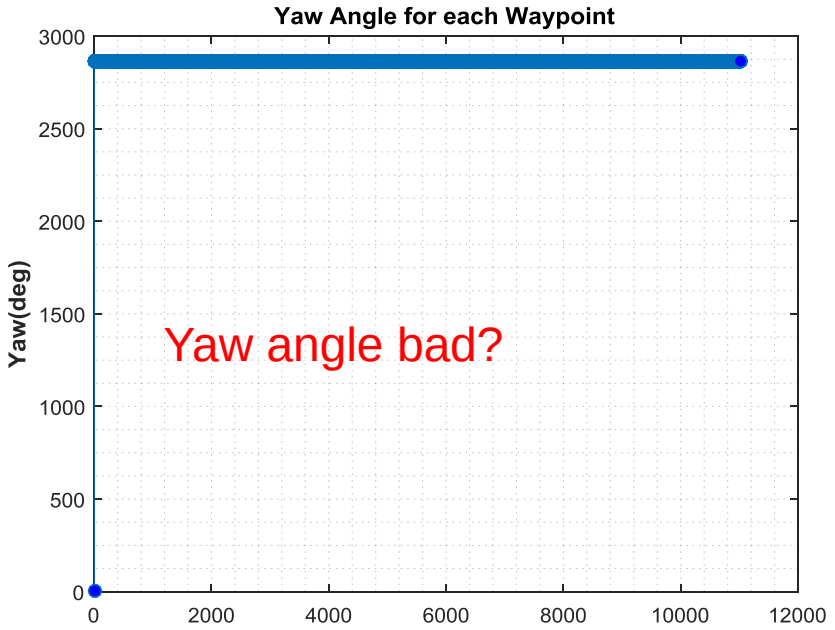


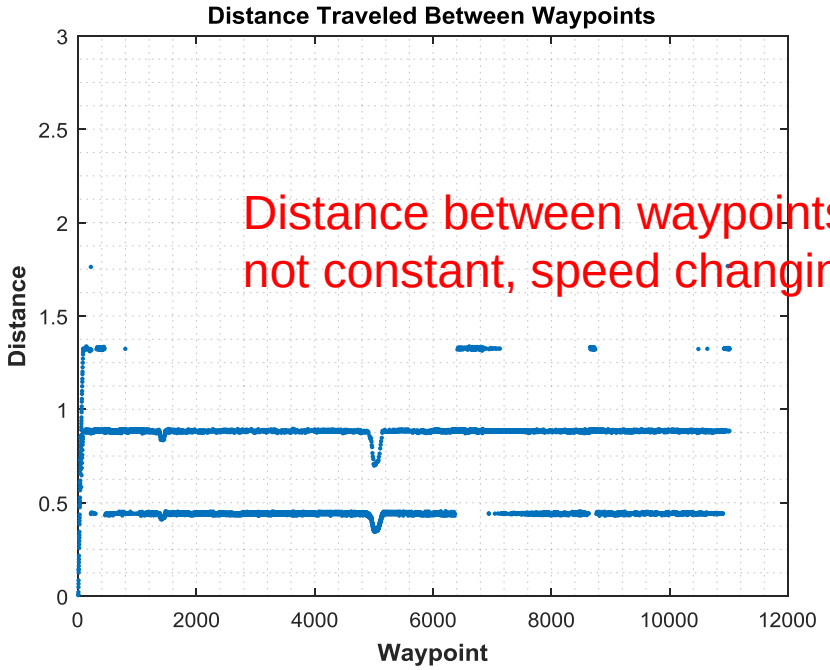
Total Waypoints: 61
Distance Traveled Over Entire Waypoint Path: 68.4871
Column Headers: Pos x,Pos y, Pos z, yaw(rad)
Yaw goes from +/- pi (rad)

1 unit of distance traveled between waypoints

sim_waypoints.csv

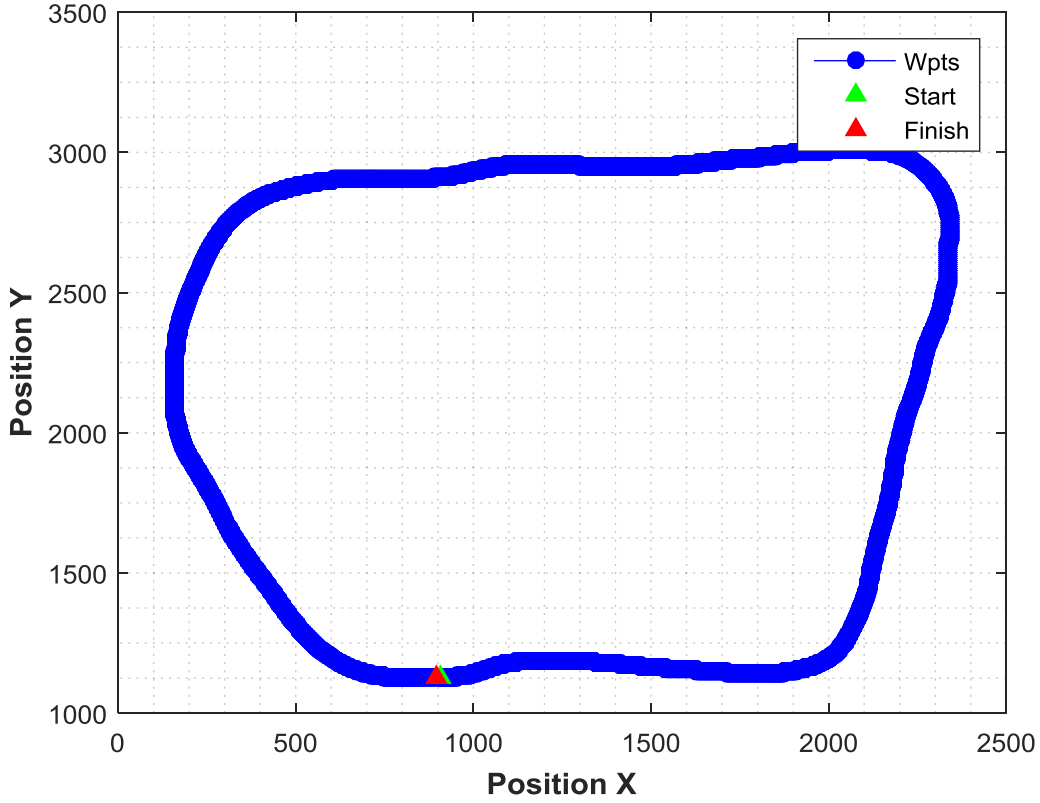


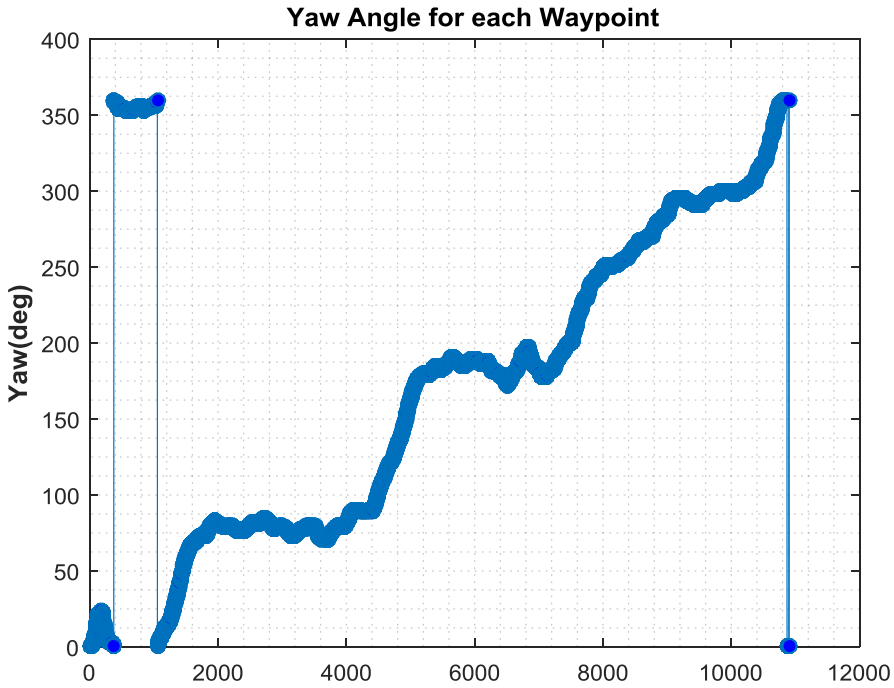


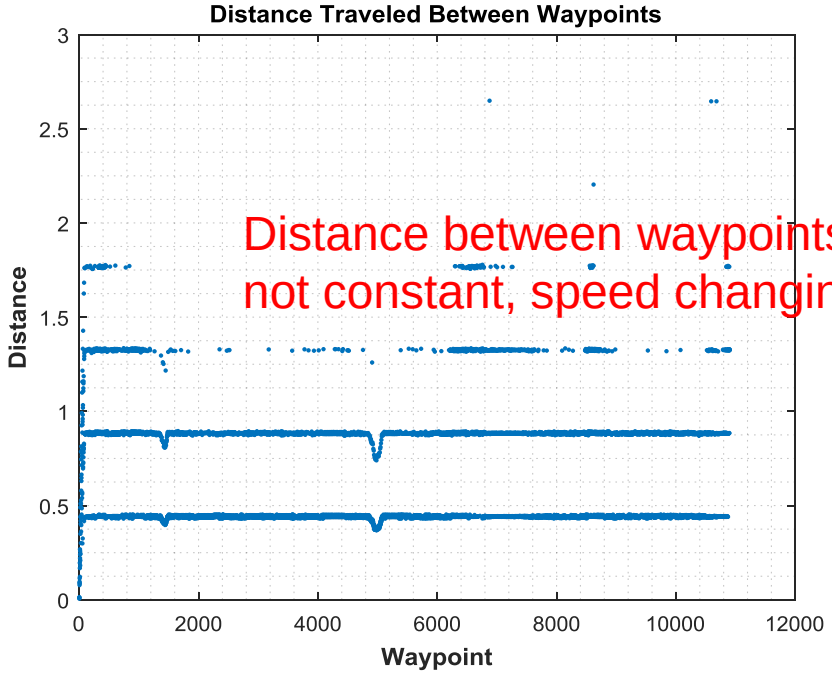


Total Waypoints: 11011
Distance Traveled Over Entire Waypoint Path: 7014.1148
Column Headers: Pos x,Pos y, Pos z, yaw(rad)
Yaw is constant, doesn't make sense

wpt_yaw_const.txt

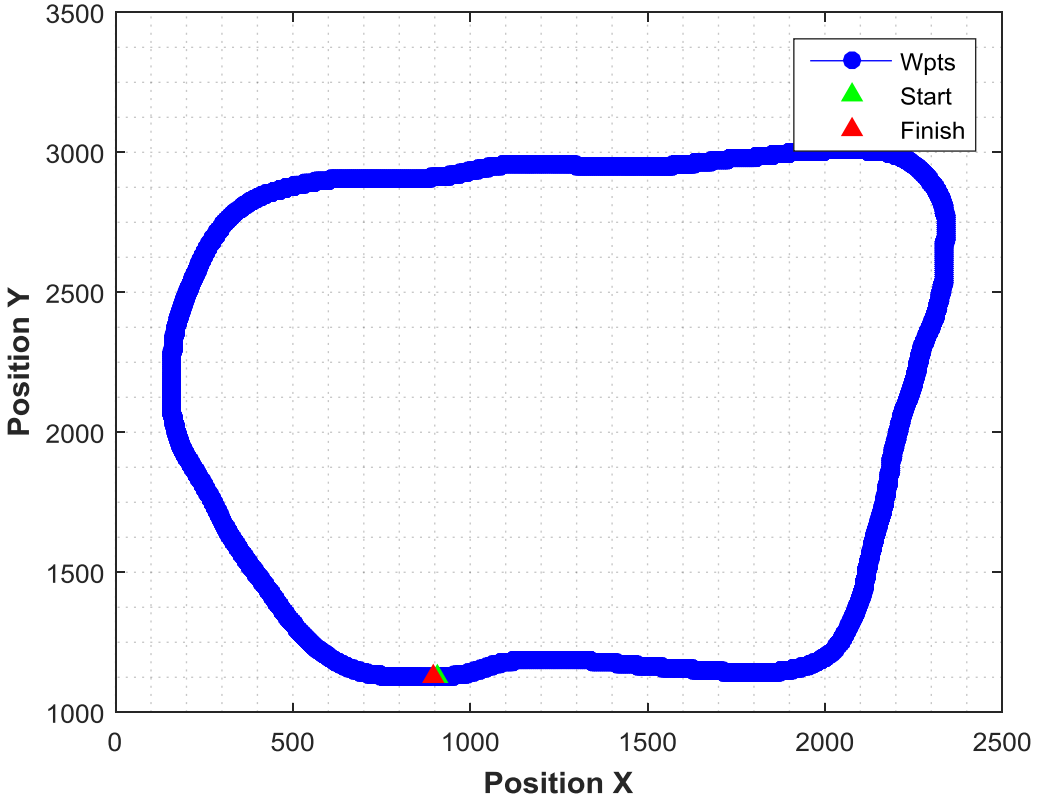


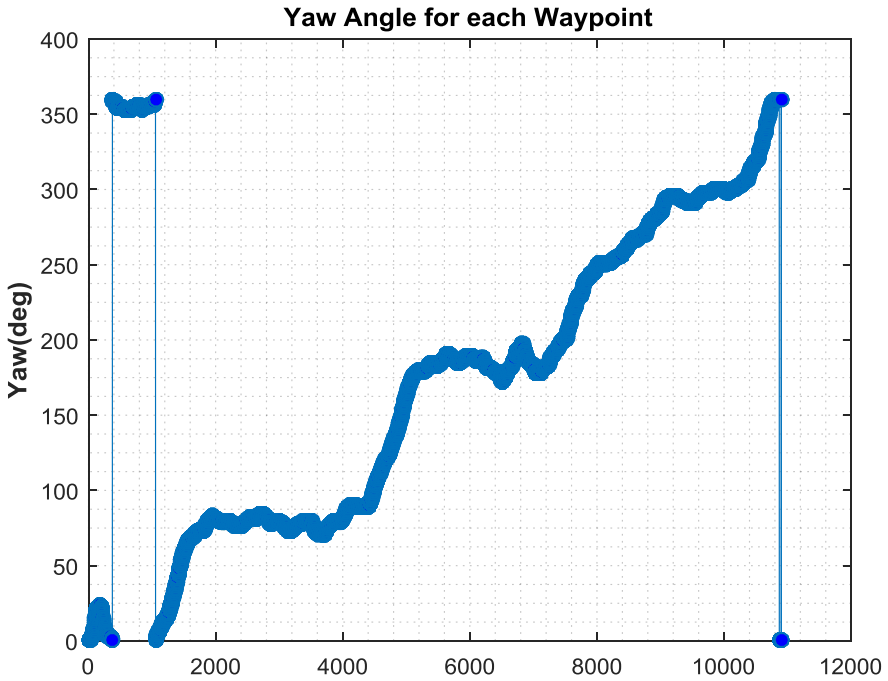


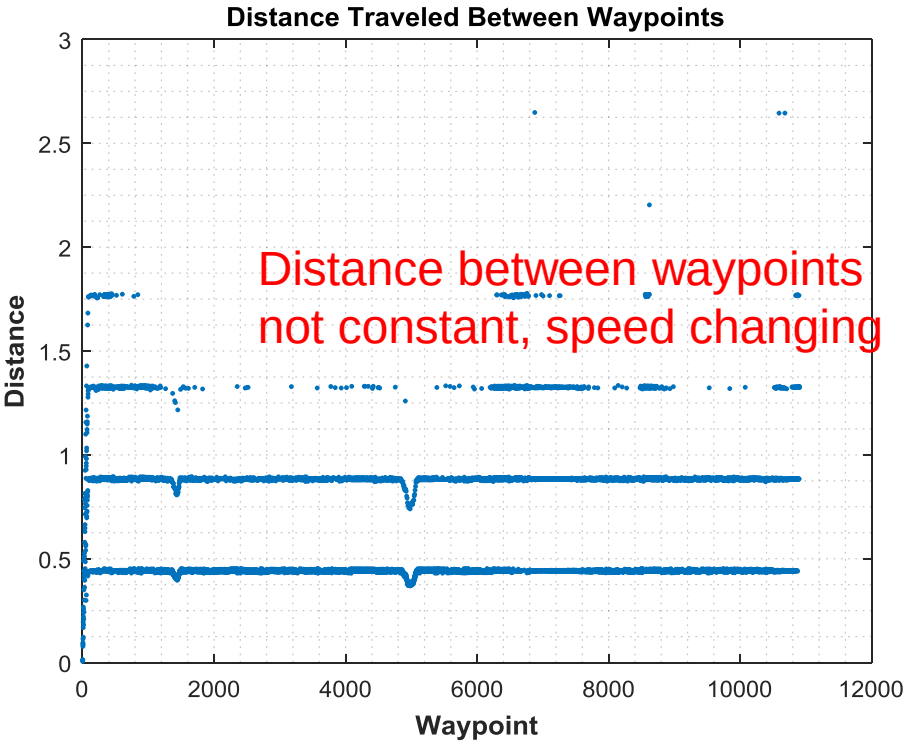


Total Waypoints: 10902
Distance Traveled Over Entire Waypoint Path: 6968.739
Column Headers: Pos x,Pos y, Pos z, yaw(deg)
Yaw goes from 0 to 360 deg
Same path as sim_waypoints.csv, but yaw is correct in this file

wpt_yaw.txt







Total Waypoints: 10902
Distance Traveled Over Entire Waypoint Path: 6968.739
Column Headers: Pos x,Pos y, Pos z, yaw(deg)
Yaw goes from 0 to 360 deg
Same path as sim_waypoints.csv, but yaw is correct in this file

waypoint_loader.py

Random Notes:
0.27778 is km/hr to m/s conversion
3.6 is m/s to km/hr conversion
Output waypoints are in /world frame
Target speed set in launch file (in km/hr)

Function: Loads one of the csv files containing waypoints of the track and publishes them to /base_waypoints

waypoint_updater.py

Random Notes:
Hardcoded number of waypoints to publish (5)
Publishing more that 5-10 waypoints ahead causes simulator to lag

Implementation Details:
We find the closest n waypoints ahead of us and assign a linear speed to them

Similar to the path planning project, loop over all the base waypoints and find the closest one to our current position. This waypoint needs to be ahead of us so compute the bearing from current position to the waypoint. If bearing is positive then the waypoint is ahead.

waypoint_follower

Random Notes:

AKA *pure_pursuit* node
Not clear if this requires changes at this time
Produces estimated angular rate based on curvature of road, need to combine this with CTE control for steering control

twist_controller: launch files

3 launch files:
dbw.launch – runs dbw_node.py and sets params below
dbw_sim.launch – runs dbw_node.py and sets similar params
dbw_test.launch – loads rosbag dbw_test.rosbag.bag
Assigns topics from rosbag:
/vehicle/throttle_cmd = /actual/throttle_cmd
/vehicle/steering_cmd = /actual/steering_cmd
/vehicle/brake_cmd = /actual/brake_cmd

Parameters and their units:

param name="vehicle_mass"	value="1736.35"	(kilograms)
param name="fuel_capacity"	value="13.5"	(gallons)
param name="brake_deadband"	value="0.1"	(?)
param name="decel_limit"	value="-5.0"	(meters/sec/sec)
param name="accel_limit"	value="1.0"	(meters/sec/sec)
param name="wheel_radius"	value="0.2413"	(meters)
param name="wheel_base"	value="2.8498"	(meters)
param name="steer_ratio"	value="14.8"	(unitless)
param name="max_lat_accel"	value="3.0"	(meters/sec/sec)
param name="max_steer_angle"	value="8.0"	(rad)

twist_controller: dbw_node.py

This node calls the controller object in twistcontroller.py that has the controllers for throttle, steering and brakes, it publishes the final commands to the car/simulator.

Car model parameters are provided for use in the control design.

We read in the final waypoints from the waypoint updater to get the desired linear velocity and subtract current velocity to feed the throttle PID controller.

Waypoint follower (PurePursuit) provides the desired angular velocity that should be used in the yaw predictive controller. Still need to figure out how this works.

We compute CTE in this file similar to the MPC project last Term to feed the PID controller on steering.

Still need to figure out how to use brakes.

PID loops and low pass filters need to be tuned.

Sign Conventions and Units

Filename	Sign Convention	Units	Range	Notes
yaw	Nose Left	rad	0 to 360	Assuming right handed coordinate system positive Z axis must be pointing up
Steering	Nose Left	rad	-8.2 to 8.2	8.2 rad is 470 deg, divided by gear ratio of 14.8 gives steering angles of +/-31 deg
Brake	-	Nm	0-3250	
Throttle	-	%	0-1	

twist_controller: low_pass.py

Simple Low Pass Filter:

$$output = \frac{1}{\frac{tau}{ts} + 1} * current + \frac{\frac{tau}{ts}}{\frac{tau}{ts} + 1} * previous$$

$$= \frac{1}{\frac{tau}{ts} + 1} * current + \frac{\frac{tau}{ts}}{\frac{tau}{ts} + 1} * previous$$

tau =0, ts = 1 will shut off this filter

twist_controller: pid.py