## First run of calibration data

BUGS:

1. The car may go the wrong way (clockwise) on a small amount of runs. So remove trajectories that have the **car’s y value < -0.5**. I tried to enforce the car going the right way (counterclockwise) by setting the initial state to have a small upward velocity (0.2), but that didn’t work for all trials
2. Obstacle spawning didn’t work like I wanted. The actual mean y-position of spawning is around 0.65

Obstacle position/size:

obstacle = struct('xlim', [-0.75, -0.25], 'ylim', [1.75, 2.25]);

Obstacle spawn settings:

obstacle\_spawn\_ylim = 1; % spawn obstacle once state passes this line

obstacle\_spawn\_var = 0.05;

Obstacle spawn criteria (system\_state(2) is y position):

if(system\_state(2) > obj.obstacle\_spawn\_ylim + …

mvnrnd(0, obj.obstacle\_spawn\_var, 1))

obj.obstacle.active = true;

end

Variable descriptions:

% time array

t\_arr = zeros(1, length(x\_hist(1,:)));

% state history of the actual car

x\_hist = zeros(length(car.x), length(0:1/f\_anc:T\_sim) + 1);

% state history of the MPPI nominal car at each time

x\_mppi\_hist = zeros(length(car.x), length(x\_hist(1,:)));

% the chosen MPPI rollout trajectory (cost-weighted average trajectory)

x\_mppi\_traj\_hist = cell(1, length(x\_hist(1,:)));

% control history of the actual car (MPPI + ancillary + control noise)

u\_tot\_hist = zeros(length(car.u), length(x\_hist(1,:)));

% control history of the MPPI controller

u\_mppi\_hist = zeros(length(car.u), length(x\_hist(1,:)));

% control history of the ancillary controller

u\_anc\_hist = zeros(length(car.u), length(x\_hist(1,:)));

% true if the actual car’s position is outside the track

outside\_track = false(1, length(x\_hist(1,:)));

% true if the obstacle is active

obs\_isactive = false(1, length(x\_hist(1,:)));

% true if the actual car’s position within the obstacle

obs\_hit = false(1, length(x\_hist(1,:)));