Tips and Tricks

• Review the previous lessons! Andrei, Dominik and co. have given you everything you need. In fact, you've built most of an Extended Kalman Filter already! Take a look at the programming assignments and apply the techniques you used to this project.

• The R matrix values and Q noise values are provided for you. There is no need to tune these parameters for this project. In the unscented Kalman Filter lectures, we'll discuss how to determine these parameters.

• For lidar measurements, the error equation is `y = z - H * x'`. For radar measurements, the functions that map the x vector [px, py, vx, vy] to polar coordinates are non-linear. Instead of using H to calculate `y = z - H * x'`, for radar measurements you'll have to use the equations that map from cartesian to polar coordinates: `y = z – h(x')`.

• In C++, `atan2()` returns values between -pi and pi. When calculating phi in `y = z - h(x)` for radar measurements, the resulting angle phi in the y vector should be adjusted so that it is between -pi and pi. The Kalman filter is expecting small angle values between the range -pi and pi.

• Before and while calculating the Jacobian matrix Hj, make sure your code avoids dividing by zero. For example, both the x and y values might be zero or `px*px + py*py` might be close to zero. What should be done in those cases?

• Test! We're giving you the ability to visualize your estimations and calculate RMSE. As you make changes, keep testing your algorithm!

• For the fun of it, try removing radar or lidar data from the filter. Observe how your estimations change when running against a single sensor type! Do the results make sense given what you know about the nature of radar and lidar data?