Robotics
Exercise 10

Marc Toussaint
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

January 10, 2014

1 Kalman filtering

I collected new data from the racer's IMU, with higher frame rate. To collect this data I fixed the wheels (motors don't turn, the motors' encoder is constantly zero) and moved the racer by hand back and forth from lying on the ground to approximately balancing.

Please find the data files `01-imu.dat`, `02-imu.dat`, `01-times.dat`, `02-times.dat` on the course page. `01` and `02` refer to two different trials—start with `02`. The `imu` files contain the 4D IMU signal (the 4th entry is constantly zero: the motor encoders). The `times` files contain the real time in seconds that correspond to these readings (you will need these to determine the time interval τ between two steps).

Also, find on the webpage the two files `racer.h` and `racer.cpp`, which implement an updated model of the racer.

Implement a Kalman filter to estimate the state trajectory $q(t)$ from this data. For this,

- Initialize the state of the `Racer` model with `R.q(1)=MT_PI/2.;` (lying down)
- Assume the following simplified dynamic model:

$$A = \mathbf{I}_4 + \tau \begin{pmatrix} 0 & \mathbf{I}_2 \\ 0 & 0 \end{pmatrix}, \quad a = 0\,, \quad Q = \text{diag}(10^{-6}, 10^{-6}, 1, 1) \tag{1}$$

  This dynamics $\dot{x} = Ax + a$, with $x = (q, \dot{q})$ simply says that the velocities are "copied" with high precision to the next time slice, but the accelerations in the next time slice are $\mathcal{N}(0, 1)$ distributed (very uncertain). Clearly this is a rough approximation – but fully sufficient for the current scenario.

- In the Kalman filter loop step as follows:
  - Retrieve the observation model (C, c, W) for the current `Racer` state using `Racer::getObservation`. Also retrieve y_{pred} here: the predicted sensor readings.
  - Use the true sensor readings (from the data file) and the dynamics and observation model for a Kalman step. Compute τ from the data files.
  - Set the state of the `Racer` model to the Kalman estimate using `R.q = ...` and `R.q_dot = ...`, and display the state using `R.gl.update()`
  - Output the Kalman's mean estimate x, the predicted y_{pred}, and the true sensor readings y_{true} in one line of a file
- Plot all curves of the output file. In particular, compare the predicted sensor outputs y_{pred} with the true ones y_{true}. Do they match?

2 Identification of the sensor model

Now that we have an estimated underlying state trajectory $q(t), \dot{q}(t)$, we can learn an even better sensor model. Usually this means to learn a mapping from the dynamic state to the sensor readings:

$$x(t) \mapsto y(t)$$

However, we exploit that we already have a sensor model implemented, with hand-tuned parameters, and want to learn a model that improves upon this (or corrects this). Therefore we learn a mapping

$$(x, y_{\text{pred}}) \mapsto y_{\text{true}}$$

where y_{pred} is the output of the implemented sensor model.

We take the output data file of the previous exercise as the basis to learn this mapping.

Use multivariate linear regression, to compute such a linear map. See http://ipvs.informatik.uni-stuttgart.de/mlr/marc/teaching/13-MachineLearning/02-regression.pdf if you need details. Use

$$\hat{\beta}^{\text{ridge}} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda I)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

where $\boldsymbol{y}$ is a matrix, containing the multivariate output in each row; $\boldsymbol{X}$ is a matrix containing the multivariate input (and an appended 1) in each row, and $\lambda \approx 10^{-4}$ is some small number.

a) What is the mean squared error of $(y_{\text{pred}} - y_{\text{true}})^2$ (not using the learned mapping)

b) What is the mean squared error of $(f(x, y_{\text{pred}}) - y_{\text{true}})^2$ using the learned linear map f

c) (Bonus) Can you weave in this learned mapping into the Kalman filter of the first exercise?