16.322 - Stochastic Estimation and Control
Final Project

Massachusetts Institute of Technology
Fall 2012

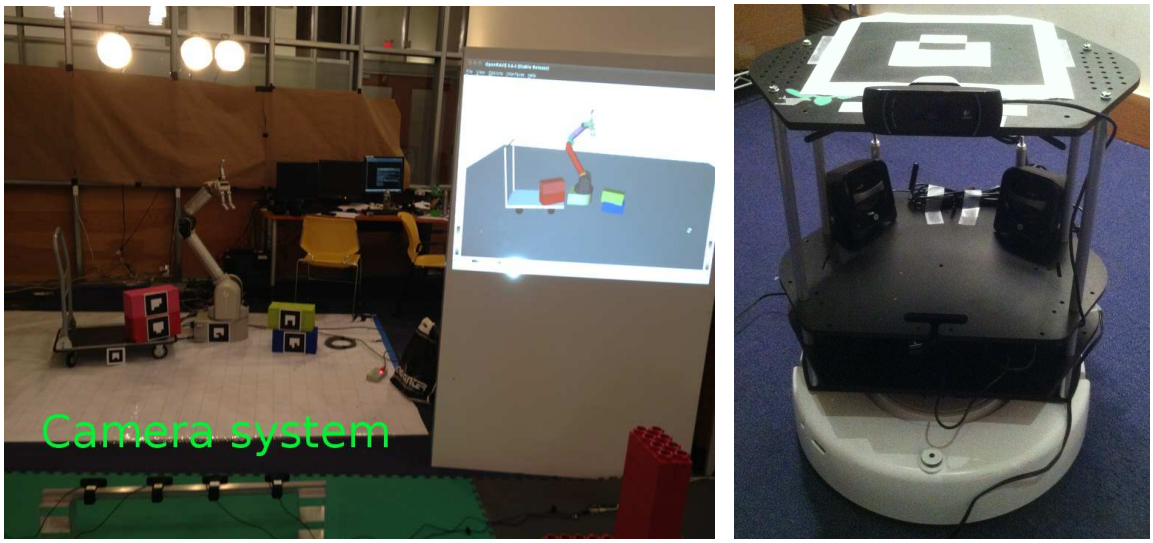
Pedro Santana (psantana@mit.edu), MIT ID 926275295

December 11^{th} 2012

Object tracking using unreliable sensors

1 Introduction

The inspiration for this project came from the two robotic test beds shown in Figure 1 that are currently being developed in my research group. The robotic manufacturing test bed in Figure 1(a) requires humans and robots to work in close proximity in a dynamically-changing environment. Therefore, the camera system shown at the bottom has to be robust to occlusions and occasional failures of one or more of its components in order to keep track of the system's state at all times. The modified iRobot Create platform shown in Figure 1(b) is part of a disaster relief test bed where autonomous aerial scouts are responsible for guiding a humanitarian aid vehicle through dangerous terrain in order to provide help to the victims. In both situations, we would like to leverage visual sensors for object recognition and tracking in order to limit the amount of structure that we need to add to our environment in order to be able to estimate the state of multiple agents. Not only that, we would like our localization system to rely on several inexpensive and commercially available sensors in order to increase redundancy, reduce costs, and facilitate the replacement of parts.



(a) Manufacturing test bed.

(b) iRobot Create

Figure 1: Test beds that use the proposed visual localization system.

This project brings two important contributions to the target tracking capabilities of these two test beds. First, it uses the Hidden Markov Model (MHH) formalism in order to model the sometimes unreliable behavior of the cameras used as sensors. Second, it compares the implementations of two state-of-the-art algorithms for pose estimation using the combined outputs of several different cameras. The distinguishing feature in this pose estimation problem is the nonlinear unit norm constraint that has to be enforced for the quaternions used to represent orientations, which add a new dimension of complexity to the problem.

2 System Overview

Our experimental setup consists of the two test beds shown in Figure 1 equipped with the webcams and fiducial markers (referred to as "fiducials" throughout the text) shown in Figure 2. All code for this project was written in C++ and Java and runs on top of Willow Garage's Robot Operating System (ROS). As our sensors, we use the ROS *ar_pose* wrapper for the ARToolKit library (`http://www.hitl.washington.edu/artoolkit/`), which outputs a 3D position vector and a 4D quaternion for each fiducial currently visible to the camera.

(a) Logitech C910 webcam.

(b) Fiducial marker used by ARToolKit.

Figure 2: Hardware for the localization system consisting of one or more webcams used to extract the pose of multiple black-and-white fiducial markers.

The next sections will describe the internal details of the "HMM filter", "BDM" (Bayesian decision maker), and "SCKF" (State-constrained Kalman Filter) components shown in the block diagram of Figure 3 that describes our visual localization system.

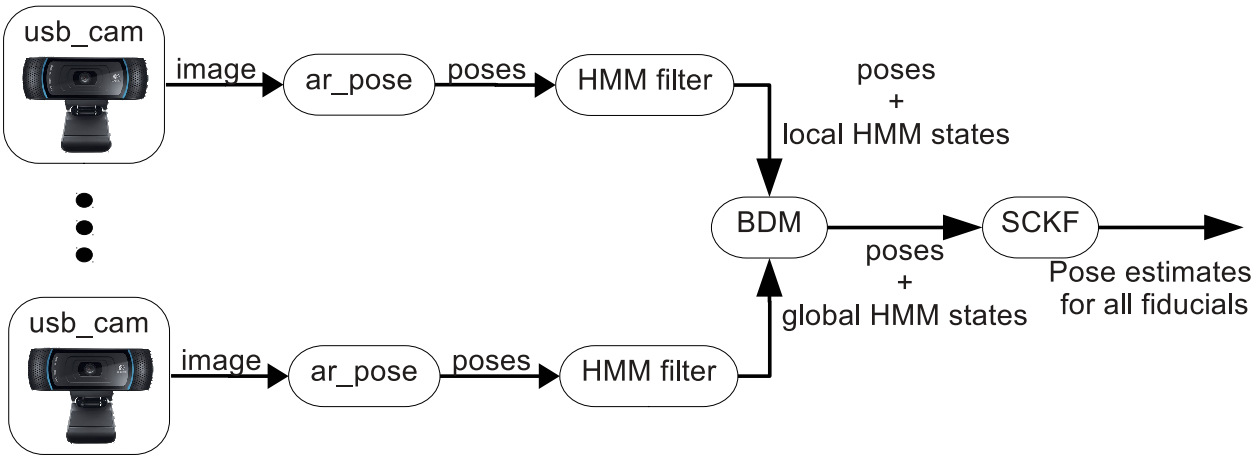
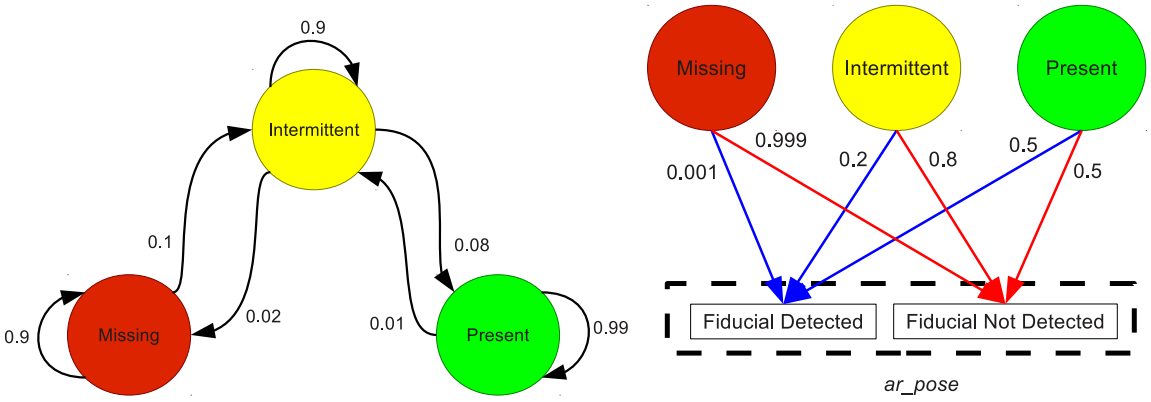


Figure 3: Block diagram for the localization system.

3 Modeling unreliable sensors using HMMs

The pose measurements outputted by the *ar_pose* nodes in Figure 3 depend not only on the fact that a fiducial is present in the camera's field of view or not, but also on an unobservable hidden state related to the lighting conditions. In order to model this behavior, we represent the detection of each fiducial by means of the HMM shown in Figure 4. For each camera, we define the HMM state of a fiducial as being "Missing" (0), "Intermittent" (1), or "Present" (2). As their names suggest, "Missing" and "Present" describe, respectively, the behavior of *ar_pose* when the fiducial is missing from the scene or is definitely present in it. "Intermittent" is a somewhat transient state in which a fiducial is repeatedly detected by *ar_pose*, but not as much as we would expect if the fiducial were truly present in the scene. Whenever the fiducial is in the "Intermittent" state, the output measurements from *ar_pose* tend to be less precise. Using the transition and observation models from Figures 4(a) and 4(b), respectively, the next section addresses the fact of how to determine the most likely discrete HMM state of a fiducial given a history of detections.



(a) Hidden Markov chain.

(b) Stochastic output model.

Figure 4: Hidden Markov Model (HMM) describing the camera system's behavior. The probabilities shown are the currently implemented ones.

3.1 HMM filtering

This section presents the HMM filtering equations that I derived and implemented for the "HMM Filter" block shown in Figure 3. Even though I acknowledge that HMM filtering is a well-known problem in the literature, it is also simple enough so that I could derive its equations in a convenient way for my particular implementation. I start by presenting the results for the filtering of a single fiducial tracked by a camera, followed by its extension to multiple fiducials in the case when we assume that their Markov Chains are mutually independent. In the next section, I present the decision algorithm that takes in the filtered HMM states from all the cameras and outputs the estimate of the HMM state of each fiducial on the global scene.

For the i-th fiducial being tracked within the field of view of one camera, define $m_{i,k} \in \{0, 1, 2\}$ as the state of its corresponding HMM at time step k. The numerical values $m_{i,k}$=$\{0, 1, 2\}$ correspond, respectively, to the fiducial being "Missing", "Intermittent", or "Present". In order to obtain the most likely HMM state $m^*_{i,k}$, we first have to determine the posterior probability $\Pr(m_{i,k}|y_{1:k})$,

where $y_{1:k}$ is the full measurement history for that particular fiducial. This is done as follows:

$$\Pr(m_{i,k}|y_{1:k}) = \frac{\Pr(y_k|m_{i,k}, y_{1:k-1}) \Pr(m_{i,k}|y_{1:k-1})}{\Pr(y_k|y_{1:k-1})}, \tag{1}$$

$$= \eta \Pr(y_k|m_{i,k}) \sum_{q=0}^{2} \Pr(m_{i,k}|m_{i,k-1} = q) \Pr(m_{i,k-1} = q|y_{1:k-1}), \tag{2}$$

where $\Pr(y_k|m_{i,k})$ is the measurement likelihood corresponding to Figure 4(b); $\Pr(m_{i,k}|m_{i,k-1} = q)$, $q \in \{0, 1, 2\}$, are the transition probabilities for the Markov Chain shown in Figure 4(a); $\Pr(m_{i,k-1} = q|y_{1:k-1})$ is the probability of having $m_{i,k-1}$=q at time step $k-1$, which is a known quantity from the previous filtering step; and η is a constant given by

$$\eta = \left(\sum_{s=0}^{2} \Pr(y_k|m_{i,k} = s) \sum_{q=0}^{2} \Pr(m_{i,k} = s|m_{i,k-1} = q) \Pr(m_{i,k-1} = q|y_{1:k-1}) \right)^{-1}. \tag{3}$$

Therefore, we obtain the most likely HMM state $m^*_{i,k}$ for the i-th fiducial by selecting the maximum *a posteriori* (MAP) value of (2), i.e.,

$$m^*_{i,k} = \underset{m_{i,k}}{\operatorname{argmax}} \, \eta \Pr(y_k|m_{i,k}) \sum_{q=0}^{2} \Pr(m_{i,k}|m_{i,k-1} = q) \Pr(m_{i,k-1} = q|y_{1:k-1}), \tag{4}$$

where η is computed according to (3). If we assume that the probability distribution of $m_{i,k}$ is independent from $m_{j,k}$, $\forall i, j \in \{1, 2, \ldots, N\}$, the joint probability of the HMM state of all fiducials becomes

$$\Pr(m_{1,k}, m_{2,k}, \ldots, m_{N,k}|y_{1:k}) = \prod_{i=1}^{N} \Pr(m_{i,k}|y_{1:k}). \tag{5}$$

Hence, we conclude from (5) that the joint MAP estimate for all fiducial being tracked by a camera is just the concatenation of the MAP estimate of each individual fiducial.

3.2 Bayesian decision maker (BDM)

Given the MAP HMM state estimates of the fiducials within each camera's field of view, the Bayesian Decision Maker (BDM) block in Figure 3 performs two tasks. First, it determines the discrete HMM state of a fiducial within the global scene given the local MAP state estimates coming from each one of the cameras. Second, it assigns higher uncertainty to the pose measurements of fiducials in the "Intermittent" state, since this is usually an indication that *ar_pose* might be generating spurious detections. For the first task, the BDM uses the following rule:

- If the fiducial is "Present" within the field of view of at least one camera, it is "Present" on the global scene.
- Else, if the fiducial is in the "Intermittent" state for at least one of the cameras, it is "Intermittent" on the global scene.
- Otherwise, the fiducial is deemed "Missing".

4 State-constrained Kalman Filtering (SCKF)

4.1 System dynamics

The continuous state vector[1] for each fiducial being tracked is given by $x_k = \left[p_k^T \; b_k^T \; q_k^T\right]^T$, where $p_k = [p_{x,k} \; p_{y,k} \; p_{z,k}]^T$ is a 3D position vector; $b_k = [b_{x,k} \; b_{y,k} \; b_{z,k}]^T$ is a bias vector; and $q_k = [q_{0,k} \; q_{1,k} \; q_{2,k} \; q_{3,k}]^T$ is a unit quaternion representing the object's 3D orientation. For this system, we define the discrete-time dynamical model

$$
\begin{aligned}
& x_{k+1} = I_{10\times 10} x_k + w_k, && E\{w_k\} = 0, E\{w_k w_k^T\} = \begin{bmatrix} W_p & 0 & 0 \\ 0 & W_b & 0 \\ 0 & 0 & W_q \end{bmatrix}, \\
& y_{k+1} = \begin{bmatrix} I_{3\times 3} & I_{3\times 3} & 0_{3\times 4} \\ 0_{4\times 3} & 0_{4\times 3} & I_{4\times 4} \end{bmatrix} x_{k+1} + r_k, && E\{r_k\} = 0, E\{r_k r_k^T\} = \begin{bmatrix} R_p & 0 \\ 0 & R_q \end{bmatrix}, \\
& \text{s.t.} \|q_k\| = 1, && \qquad (6)
\end{aligned}
$$

which is a linear model with a nonlinear equality state constraint $\|q_k\| = 1$. Before we proceed, it is worthwhile to mention the reasons behind the choice of such a simple model for the object tracker.

The first version of (6) had a velocity vector $v_k = [v_{x,k} \; v_{y,k} \; v_{z,k}]^T$ as part of the state vector x_k, whose purpose was to improve the time propagation step of the filter in the absence of visual measurements. This initial formulation, however, failed to yield good tracking performance in the manufacturing test bed shown in Figure 1(a) unless a very small covariance W_v were assigned to the process noise affecting v_k. Due to the natural jitter of the pose measurements generated by *ar_pose*, nonzero velocities were erroneously computed by the state estimator even when the objects were static on the scene. This in turn caused the objects to constantly oscillate, hindering our manipulator's ability to plan a trajectory in order to correctly grasp the object. As mentioned before, the workaround found for this problem was to force W_v to be very small, causing velocities to slowly drift from zero due to the "low-pass" effect induced by a small W_v. This, however, caused the velocity estimates to remain around 0 almost all the time, defeating their purpose of aiding in the estimates' time propagation. Hence, we decided to remove v_k from the state vector in order to reduce the filter's complexity and adopted the static time propagation model shown in (6). This assumption clearly makes this localization system not suitable for the tracking of agile systems, for which it is important to incorporate inertial measurements into the filtering loop and an accurate dynamical model in order to improve the time propagation performance between measurement update steps.

The presence of the bias vector b_k as part of the state vector was also motivated by the particular features of the measurements generated by *ar_pose*. Using a grid and a tape measure in order to position objects at well-defined coordinates, I realized that the position measurements coming from the cameras were always off by a few centimeters. This bias seemed to be different depending on the position of the fiducial with respect to the camera, but was approximately constant for the same relative orientation. The bias vector b_k was then introduced as part of the state vector in order to compensate for that effect. Nevertheless, future improvements of this localization system should consider a different bias vector for each camera tracking the fiducial.

[1] As opposed to the discrete HMM state.

Despite the simplicity of (6), the nonlinear constraint $\|q_k\|=1$ renders the filtering problem harder to tackle. The next section briefly describes the two state-of-the-art algorithms that I have implemented for this project.

4.2 Dealing with state constraints

Equality-constrained state estimation extends our usual filtering setup by adding a constraint

$$g(x_k) = 0, \tag{7}$$

where the function $g(.)$ can have arbitrary form. For the particular case in which $g(.)$ is a linear function, the authors in [1] develop three different estimate projection methods that yield the optimal solution. On the other hand, when $g(.)$ is a nonlinear function of the state such as in $\|q_k\|=1$, estimate optimality cannot be assured. For the latter case, the same authors propose in [2, 3] several different methods to deal with nonlinear state constraints that yield good performance. In this project, I decided to implement and compare two of these approaches, namely the Measurement-Augmented Extended Kalman Filter (MAEKF) and the Measurement-Augmented Unscented Kalman Filter (MAUKF). The measurement-augmented term comes from the fact that we extend the model in (6) to include a virtual measurement of the form

$$1 = q_{0,k}^2 + q_{1,k}^2 + q_{2,k}^2 + q_{3,k}^2 + v_{n,k}, E\{v_{n,k}\} = 0, E\{v_{n,k}^2\} \approx 0. \tag{8}$$

The idea of augmenting your model with a pseudo-norm measurement (8) in order to enforce the constraint (7) is not new and can also be found in [4], where the authors also perform a thorough analysis of potential stability problems caused by the "brute force" approach of normalizing q_k using the inverse of its quadratic norm at each time step. Ideally, the covariance of the normalizing measurement (8) should be zero, but the authors in [2–4] point out that this might lead to filter instability issues. Hence, a small nonzero number should be chosen for $E\{v_{n,k}^2\}$[2].

Dealing with (8) within an UKF is straightforward, but the EKF requires the computation of the Jacobian matrix of (8). For this simple measurement equation, the Jacobian is given by

$$H(\hat{x}_k) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 2\hat{q}_{0,k} & 2\hat{q}_{1,k} & 2\hat{q}_{2,k} & 2\hat{q}_{3,k} \end{bmatrix}. \tag{9}$$

5 Experimental Results

This section presents the performance of the MAEKF and the MAUKF when tracking the Create platform shown in Figure 1(b) while it is visiting a series of waypoints in closed-loop with a pose controller. For the manufacturing test bed, my original intention was to show videos of the manipulator using the filtered pose estimates in order to localize blocks and move them around. However, due to a recent hardware damage, this could not be performed. In order to assess the performance of the object tracking system when operating in the manufacturing test bed, please refer to the links below.

- Raw *ar_pose* data on a completely static scene: `http://youtu.be/plLPCT7gYJA`.

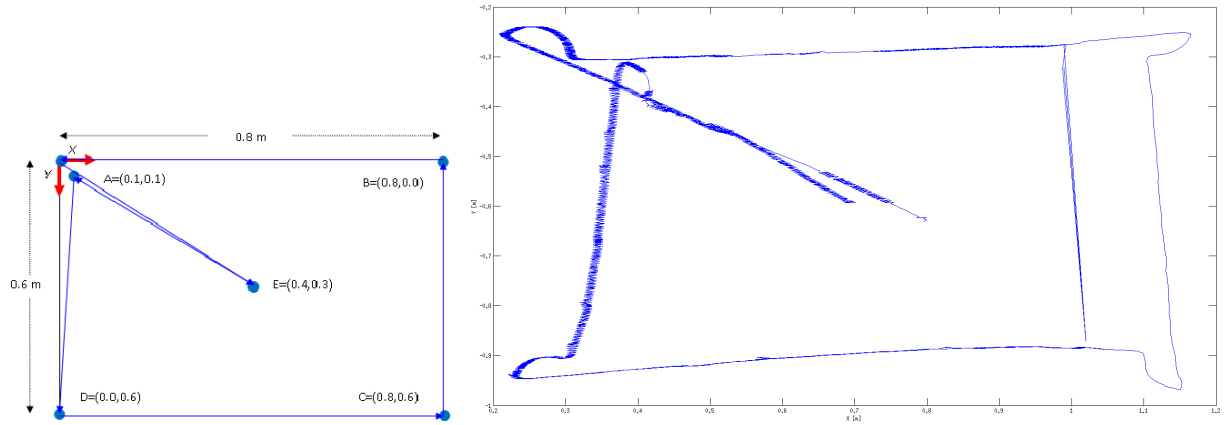
[2]This project uses $E\{v_{n,k}^2\} = 1e{-}13$.

$\bar{i}_{p_x}$	$\bar{i}_{p_y}$	$\bar{i}_{p_z}$	$\bar{i}_{q_0}$	$\bar{i}_{q_1}$	$\bar{i}_{q_2}$	$\bar{i}_{q_3}$
-7.3e-04	6.06e-0.5	1.71e-0.4	0.0102	0.0038	0.0030	-0.0026

Table 1: Average value for each component of the innovation process.

- Filtered block pose while being handled: `http://youtu.be/NnR_NyR7GOO`.

First of all, it should be noticed that the tracking performance for the MAEKF and the MAUKF was very similar, so just the results for the MAEKF will be shown here due to space limitations. Given the waypoints shown in Figure 5(a), a pose controller drove the robot around the map using the localization system's output. The reconstructed trajectory is shown in Figure 5(b), where we can clearly see that the robot visits the desired waypoints despite its deviations from the "nominal" trajectory consisting of purely straight lines. Please keep in mind that there is a $0.3m$ offset in both X and Y between Figures 5(a) and 5(b). Despite our best efforts to circumvent the problem of inconsistent pose measurements coming from *ar_pose*, it is evident that a spurious measurement caused the pose trajectory to present a spike on the right side of Figure 5(b), which could have been problematic to the pose controller had it lasted for an extended period of time.



(a) Robot waypoints.

(b) Filtered trajectory.

Figure 5: Waypoints and filtered trajectory for the Create platform operating in closed-loop.

Another perspective about the performance of our filters is given by the analysis of the innovation process, which is shown in Figures 6 and 7 and Table 1. The very small values of the first columns in Table 1 suggest that our position estimates are unbiased, which is a very desirable feature. For the remaining columns corresponding to the quaternion innovations, the values are still small, but not as much as for the positions estimates. The plots in Figure 7 do suggest that our filters are not estimating quaternions as well as positions, but the higher values for the innovations could also be a result of the aforementioned jitter in the pose measurements generated by *ar_pose*. An evidence of this possible cause are the clearly oscillatory patterns in Figure 7, followed by periods of almost zero innovation, i.e., the filters are doing a good job estimating the orientation. Despite these errors, the pose controller successfully managed to drive the robot to the specified positions on the map using the filtered pose estimates, as shown on the videos below.

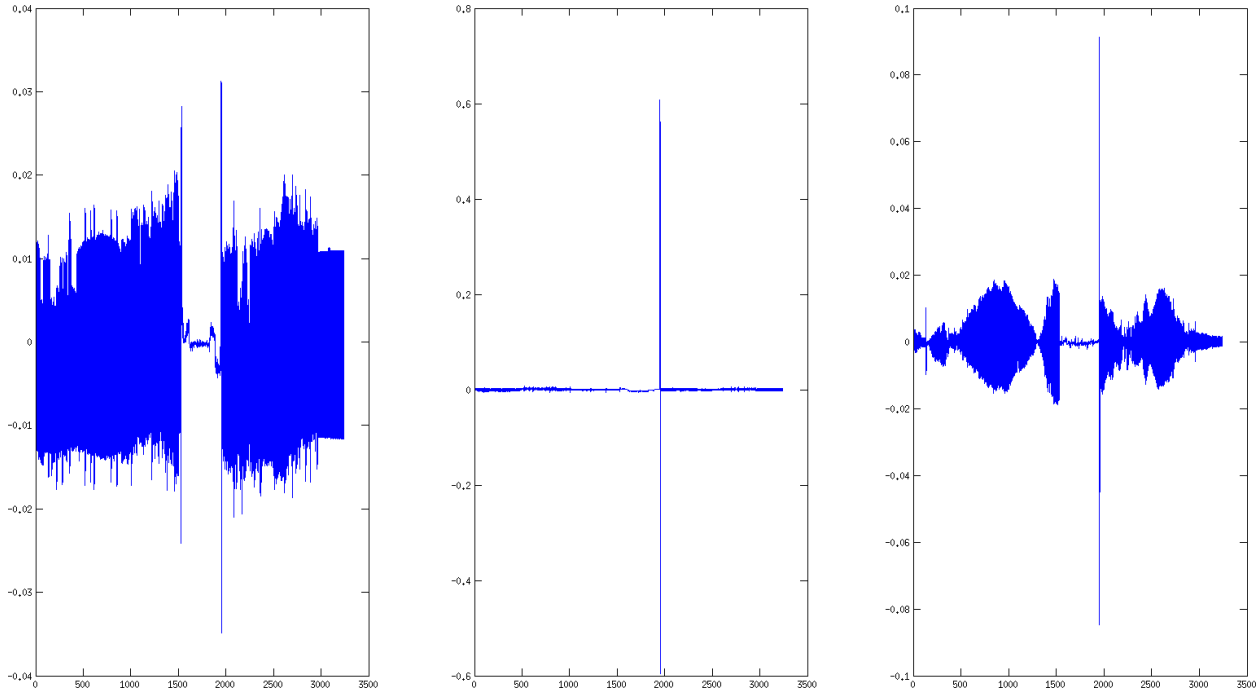


Figure 6: From left to right, innovation process for p_x, p_y and p_z.

- Closed-loop tracking: `http://youtu.be/2p36vw06Pcc`.
- Closed-loop tracking (computer view): `http://youtu.be/mkBCuoJLSSo`.

6 Conclusions

The most important lesson learned in this project came from the study of state-constrained filtering papers, where the authors address the issues of how to incorporate state constraints into filtered estimates in a principled way. In the linear constraint case, the authors in [1] derive the optimal constrained state estimates and show that they only help the filter by reducing the total amount of uncertainty. They show that the projection step is an instance of the conventional measurement update step where a "perfect" (no sensor noise) measurement corresponding to the constraint is used. Motivated by this observation, the same authors propose in [2] to use the same idea to deal with nonlinear state constraints, giving rise to the MAEKF and MAUKF implemented in this project.

In terms of performance, the MAEKF and the MAUKF behaved almost identically. This is probably due to the fact that pose measurements were frequently sampled from the cameras and that the robotic platform being tracked did not move very fast, rendering the EKF linearizations reasonably accurate. Concerning difficulty of implementation, the MAEKF was slightly easier to implement, since the Jacobian in (9) was very easy to derive and implement when compared to the Unscented Transform (UT) operations necessary for the UKF. Nevertheless, implementing the UT in C++ was almost the same effort as doing so in Matlab, since the Eigen matrix library was used.

Finally, early experimental results showed the importance of analyzing the closed-loop dynamics

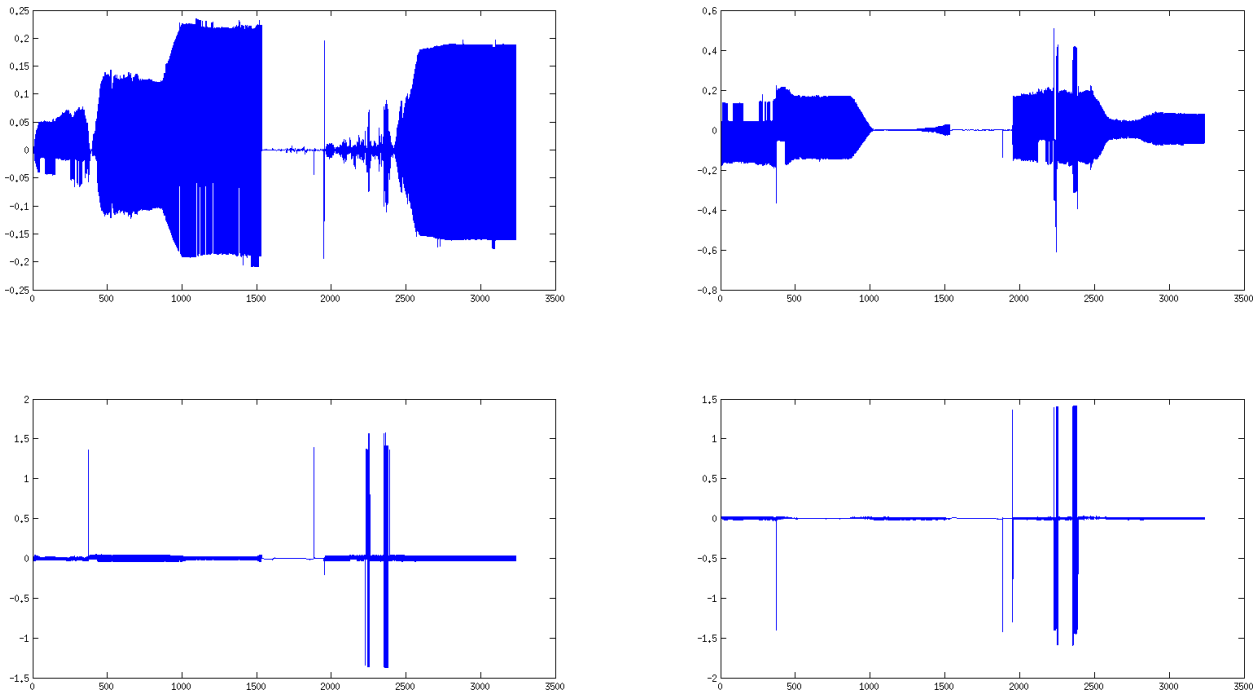


Figure 7: Innovations for each one of the four components of the orientation quaternion.

of systems with state estimators and controllers. In an early setup of the filters, the process covariance matrices in (6) had been chosen to be small due to the estimates oscillation effect caused by the jitter in the pose measurements coming from *ar_pose*. This, however, caused the filter to lag behind the true state of the system while it was been acted upon by the pose controller, generating instability. The solution was to speed up filter convergence by increasing the covariance matrices for the time propagation equations, which cause the Kalman gain to be higher. Unfortunately, this had the negative effect of accentuating the jitter coming from *ar_pose* and propagating it to the state estimates, an effect that we studied analytically in class and that can be clearly observed on the zig-zag pattern seen in Figure 5(b).

Bibliography

[1] B. O. S. Teixeira, J. Chandrasekar, L. a. B. Torres, L.A. Aguirre, and D. S. Bernstein. State estimation for equality-constrained linear systems. *2007 46th IEEE Conference on Decision and Control*, pages 6220–6225, 2007.

[2] B. Teixeira, J. Chandrasekar, H.J. Palanthandalam-Madapusi, L. Torres, L.A. Aguirre, and D. S. Bernstein. Gain-Constrained Kalman Filtering for Linear and Nonlinear Systems. *IEEE Transactions on Signal Processing*, 56(9):4113–4123, September 2008.

[3] Bruno O. Soares Teixeira, J. Chandrasekar, Leonardo a. Borges Torres, L.A. Aguirre, and D. S. Bernstein. Unscented filtering for equality-constrained nonlinear systems. *2008 American Control Conference*, pages 39–44, June 2008.

[4] I.Y. Bar-Itzhack and J. Deutschmann F.L. Markley. Quaternion normalization in additive EKF for spacecraft attitude determination. In *Proceedings of the Flight Mechanics/Estimation Theory Symposium*, pages 403–421, October 1991.