Homework 3: Inference & Learning for Temporal State Space Models

Brown University CS242: Probabilistic Graphical Models, Fall 2014

Official Solutions

Question 1:

a) Figure 1 shows the Kalman filter estimates with two standard deviations of error.

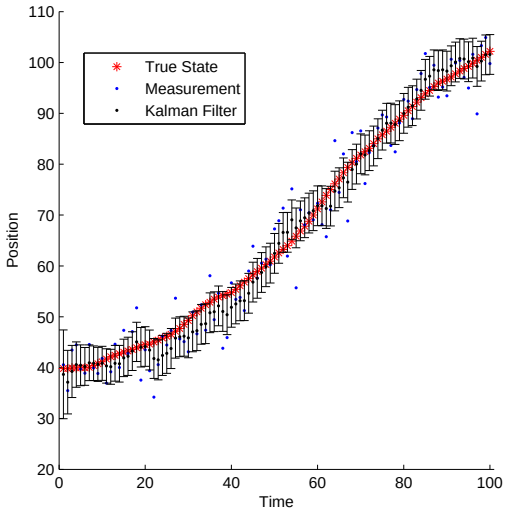
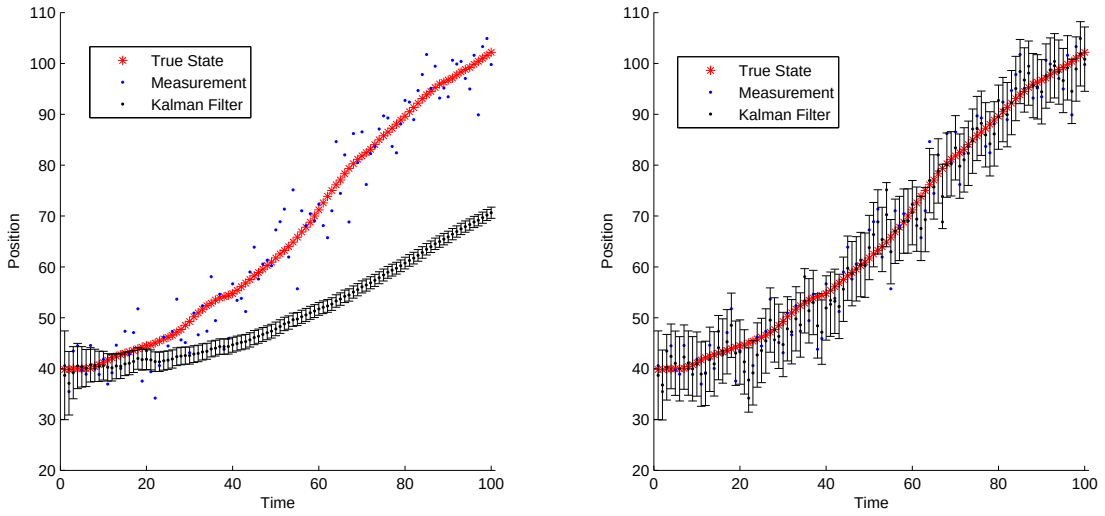


Figure 1: Kalman filter output for constant velocity model.

b) Figure 2 shows the Kalman filter estimates with two standard deviations of error for $\sigma_x^2 = 0.01/3$*, and* $\sigma_x^2 = 10$*. The primary difference between this and Figure 1 is that when using the correct dynamics noise (*$\sigma_x^2 = 0.01/3$*), but incorrectly assuming that the velocity is zero, the filter produces inaccurate state estimates. Using a dynamics model that allows for bigger movements (*$\sigma_x^2 = 10$*), the estimates produced by the Kalman filter follow roughly the true state, but the standard deviation on the estimates is larger. Intuitively, the overly simplified constant-position model must treat the unmodeled velocity as noise.*

c) See solution code in `particle_filter.m`*.*

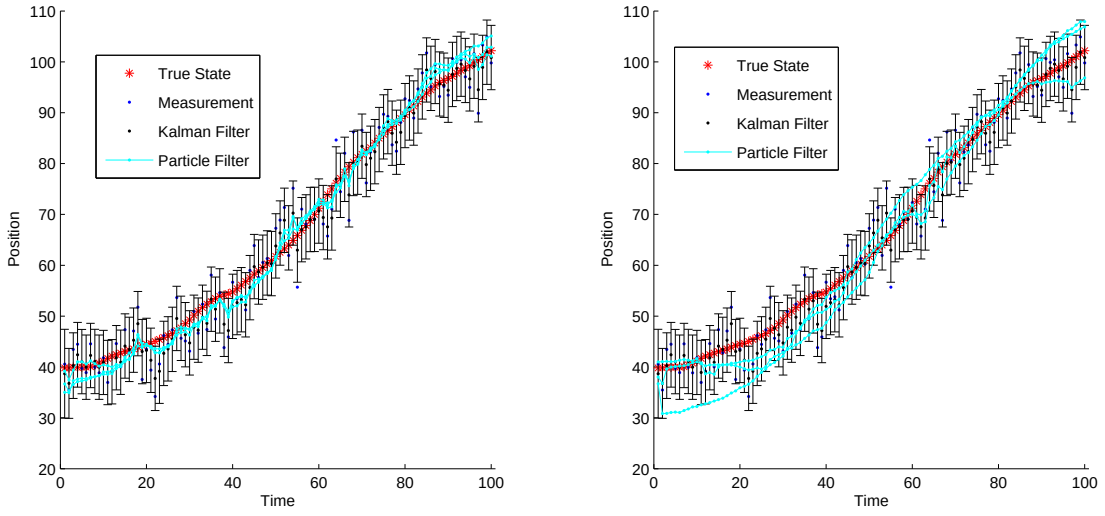
d) Figure 3(a) shows the particle filter compared to the Kalman filter for three random initializations with 100 *particles. Particle filter state estimates are generally close to the true posterior means.*



(a) (b)

Figure 2: (a) Kalman filter output for constant position model with $\sigma_x^2 = 0.01/3$. (b) Kalman filter output for constant position model with $\sigma_x^2 = 10$

e) Figure 3(b) shows the particle filter compared to the Kalman filter for three random instances with 20 *particles. In many cases, the state estimates deviate significantly from the true posterior means. The particle filter is only robust if sufficiently many particles are used; for this model and data,* 20 *appears too small.*



(a) (b)

Figure 3: Particle filter output for three random initializations using (a) 100 particles, and (b) 20 particles.

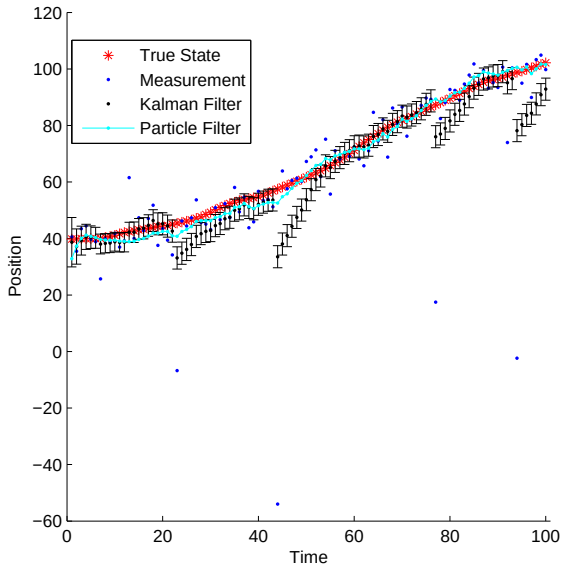


Figure 4: Comparison of Kalman Filter and Particle Filter with 1000 particles on constant velocity model with outliers. The observation likelihood of the Kalman filter does not account for outliers, whereas the particle filter is matched to the model.

f) Figure 4 shows the results of tracking on the constant velocity model with outliers. We see that the particle filter, with matched observation model, is more robust to the noisy data.

Question 2:

a) Figure 5 shows a comparison between the Kalman Filter and Smoother. In general the smoother reduces covariance and produces a better estimate of the latent state.

b) We can introduce, and integrate over, the latent state x_{t+1} as,

$$p(y_{t+1} \mid y_1, \ldots, y_t) = \int_{\mathcal{X}_{t+1}} \int_{\mathcal{X}_t} p(y_{t+1} \mid x_{t+1}) p(x_{t+1} \mid x_t) p(x_t \mid y_1, \ldots, y_t) \, dx_t dx_{t+1}$$

Observe that $p(y_{t+1} \mid x_{t+1}) = N(y_{t+1} \mid Cx_{t+1}, R)$ is given by the observation model, $p(x_{t+1} \mid x_t) = N(x_{t+1} \mid Ax_t, Q)$ the dynamics, and $p(x_t \mid y_1, \ldots, y_t) = N(x_t \mid \hat{x}_{t|t}, P_{t|t})$ is computed by the Kalman filter. Using standard Gaussian identities we can solve,

$$\begin{aligned} p(y_{t+1} \mid y_1, \ldots, y_t) &= \int_{\mathcal{X}_{t+1}} \int_{\mathcal{X}_t} N(y_{t+1} \mid Cx_{t+1}, R) N(x_{t+1} \mid Ax_t, Q) N(x_t \mid \hat{x}_{t|t}, P_{t|t}) dx_t dx_{t+1} \\ &= \int_{\mathcal{X}_{t+1}} N(y_{t+1} \mid Cx_{t+1}, R) N(x_{t+1} \mid A\hat{x}_{t|t}, Q + AP_{t|T}A^T) dx_{t+1} \\ &= N(y_{t+1} \mid CA\hat{x}_{t|t}, R + C(Q + AP_{t|t}A^T)C^T). \end{aligned}$$

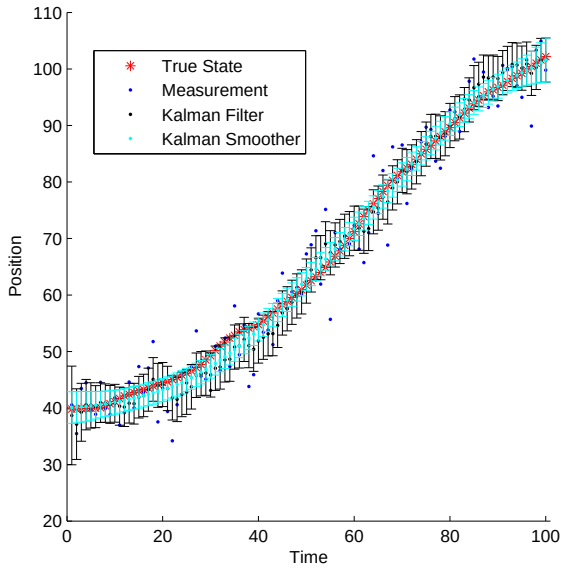


Figure 5: Kalman filter vs. Kalman smoother for the constant velocity model.

With prior mean μ_0 and covariance P_0 we can write the full marginal log-likelihood as,

$$\log p(y) = \log N(y_1 \mid C\mu_0, R + CP_0C^T) + \sum_{t=1}^{T-1} \log N(y_{t+1} \mid CA\hat{x}_{t|t}, R + C(Q + AP_{t|t}A^T)C^T).$$

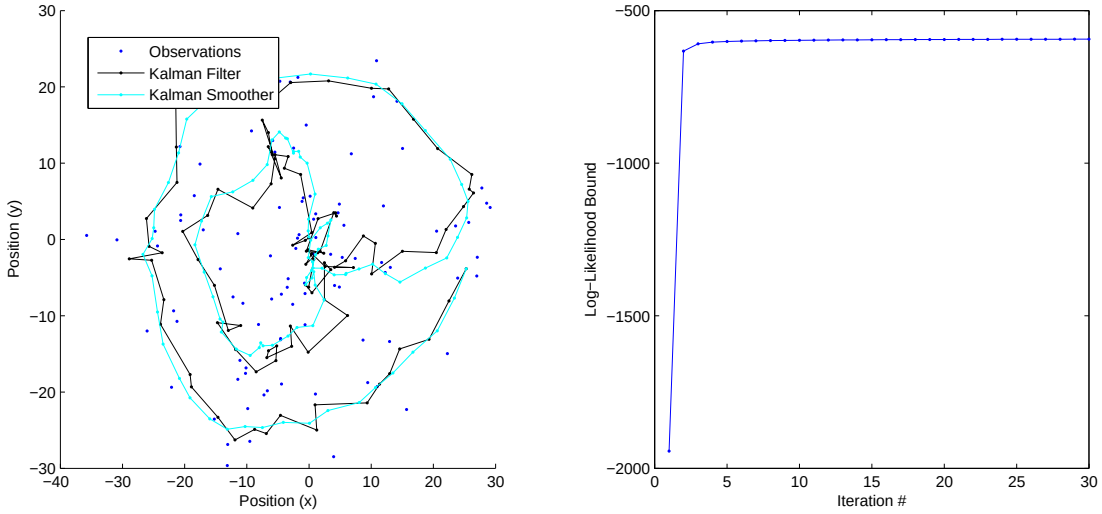
c) *See solution code in* `em_lds.m`. *Denote the Gaussian posterior distribution computed by the Kalman smooother by $p(x_t \mid y) = N(x_t \mid \hat{x}_t, P_t)$. From the course textbook or other resources, the M-step update of the covariance matrix R equals*

$$R = \frac{1}{T} \sum_{t=1}^{T} (y_t - C\hat{x}_t)(y_t - C\hat{x}_t)^T + CP_tC^T.$$

The first term in the sum would be the ML estimate of R if x_t were observed to exactly equal its posterior mean $\hat{x}_t$. Because there is posterior uncertainty in the E-step, the covariance is increased by an amount proportional to the posterior covariance P_t.

d) *Figure 6(a) shows the tracking output for the* `spiral` *data based on parameters learned by EM. The log-likelihood bound versus EM iteration is in Figure 6(b), and the learned covariance is*

```
R =
   18.7408   -1.5461
   -1.5461   21.0857.
```



(a) (b)

Figure 6: (a) Tracking output based on learned parameters for `spiral` data. (b) Log-likehood bound vs. iteration. Note that it is monotonically non-decreasing.

Question 3:

a) Let N be the number of sequences in the training set and T_n the number of timescans in the n^{th} sequence. The transition parameters $\pi \in \mathbb{R}^{K \times K}$ are computed by summing the total number of transitions from one state to another,

$$\widetilde{\pi}_{ij} = \sum_{n=1}^{N} \sum_{t=2}^{T_n} \mathbb{I}(z_{n,t-1} = i)\mathbb{I}(z_{n,t} = j),$$

and then normalizing across columns,

$$\pi_{ij} = \frac{\widetilde{\pi}_{ij}}{\sum_{k=1}^{K} \widetilde{\pi}_{ik}}$$

Figure 7 shows the transition probabilities learned for each train/test split.

b) Given state space model parameters the different training sequences are conditionally independent. We can then collect sufficient statistics for the E-step by running the Kalman smoother on each sequence with the current parameter estimates.

The M-step will depend on the sufficient statistics of all sequences. Where there is originally a summation over the length of a sequence $\sum_{t=1}^{T}$ there will appear the double summation over sequences $1, \ldots, M$ and the corresponding length of that sequence T_m as in $\sum_{m=1}^{M} \sum_{t=1}^{T_m}$. For example, the dynamics matrix update would be,

$$A^{new} = \left(\sum_{m=1}^{M} \sum_{t=2}^{T_m} \mathbb{E}_m[x_t x_{t-1}^T] \right) \left(\sum_{m=1}^{M} \sum_{t=2}^{T_m} \mathbb{E}_m[x_{t-1} x_{t-1}^T] \right)^{-1},$$

where $\mathbb{E}_m[\cdot]$ denotes the expected sufficient statistics of the m^{th} sequence.

	LT	RT	W
LT	0.97	0.00	0.027
RT	0.00	0.98	0.021
W	0.021	0.021	0.96

	LT	RT	W
LT	0.98	0.00	0.021
RT	0.00	0.98	0.020
W	0.014	0.013	0.97

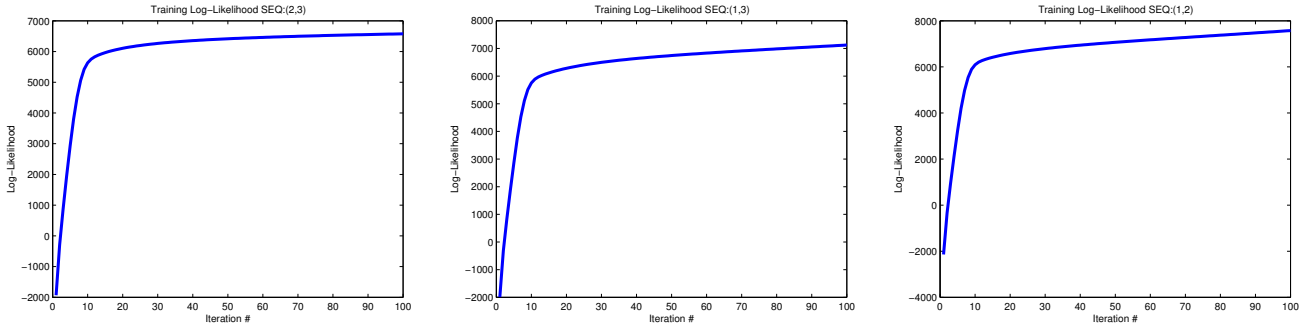
	LT	RT	W
LT	0.98	0.00	0.025
RT	0.00	0.98	0.022
W	0.02	0.0172	0.96

Figure 7: Transition probabilities for each train/test split between "Left Turn" (LT), "Right Turn" (RT), and "Waggle" (W). Entry (i, j) in the matrix represents $P(z_{n,t+1} = i \mid z_{n,t} = j)$.

c) This is done in the code for you; it is the second output out of load_bees.

d) Figure 8 shows the log-likelihood bounds per iteration for each of the train/test splits. Learned parameter R for the first train/test split is,

```
R_1 =
    0.0005    0.0002   -0.0027    0.0008
    0.0002    0.0004    0.0010    0.0001
   -0.0027    0.0010    0.0995   -0.0157
    0.0008    0.0001   -0.0157    0.0093
R_2 =
    0.0005   -0.0000    0.0008   -0.0006
   -0.0000    0.0006    0.0066   -0.0023
    0.0008    0.0066    0.1287   -0.0451
   -0.0006   -0.0023   -0.0451    0.0255
R_3 =
    0.0324   -0.0080   -0.0174   -0.0036
   -0.0080    0.0134   -0.0003    0.0034
   -0.0174   -0.0003    0.0139    0.0006
   -0.0036    0.0034    0.0006    0.0011
```



(a) (b) (c)

Figure 8: Log-likelihood bound per EM iteration on each of the three train/test splits of the dancing honeybees dataset.

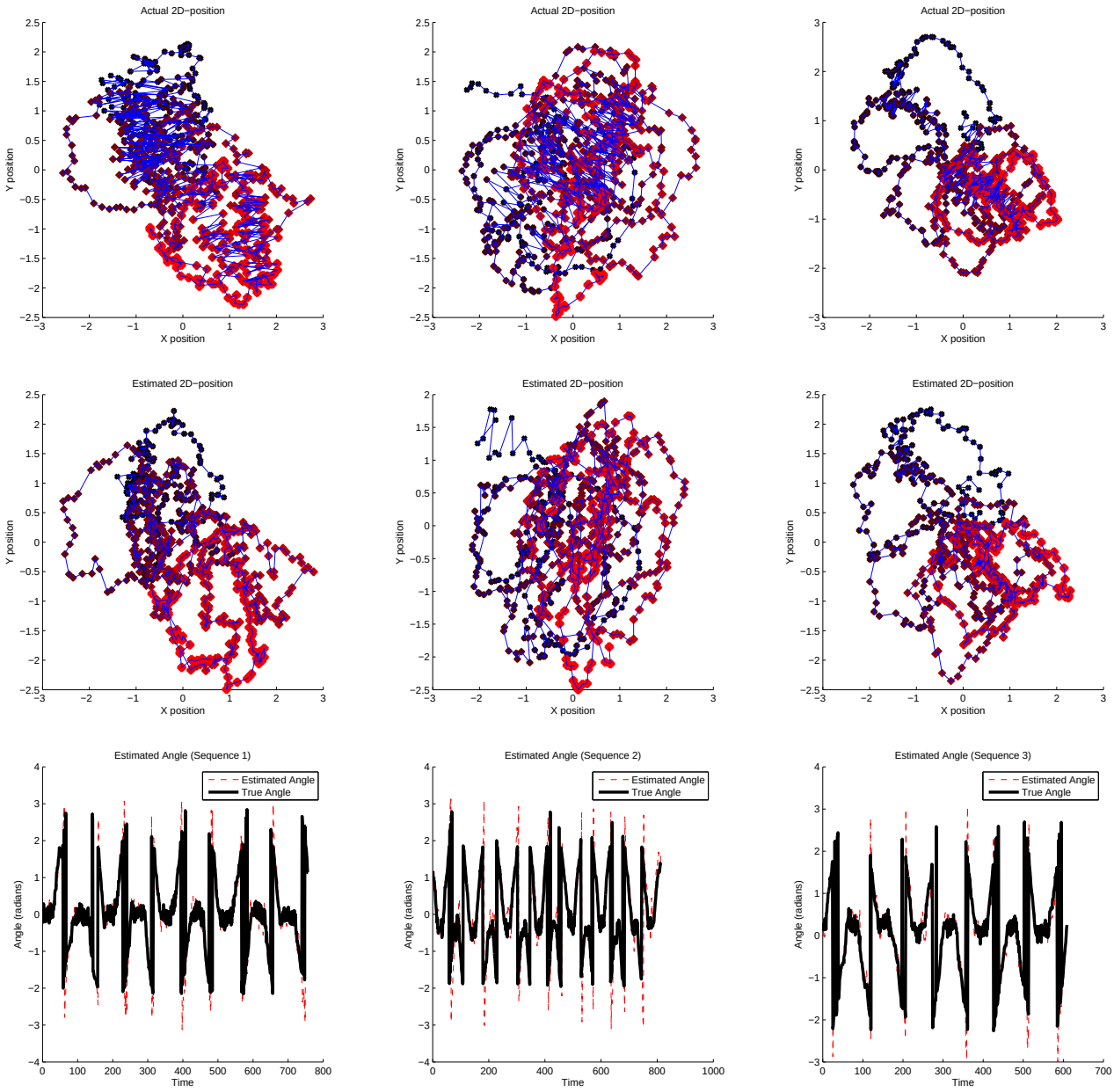
e) The revised observation likelihood is given by,

$$p(\widetilde{y} \mid x_t) = 0.9Norm(\widetilde{y}_t \mid Cx_t, 0.1I_4 + R) + 0.1Norm(\widetilde{y}_t \mid 0, 5I_4)$$

f) See solution code in `particle_filter_slds.m`.

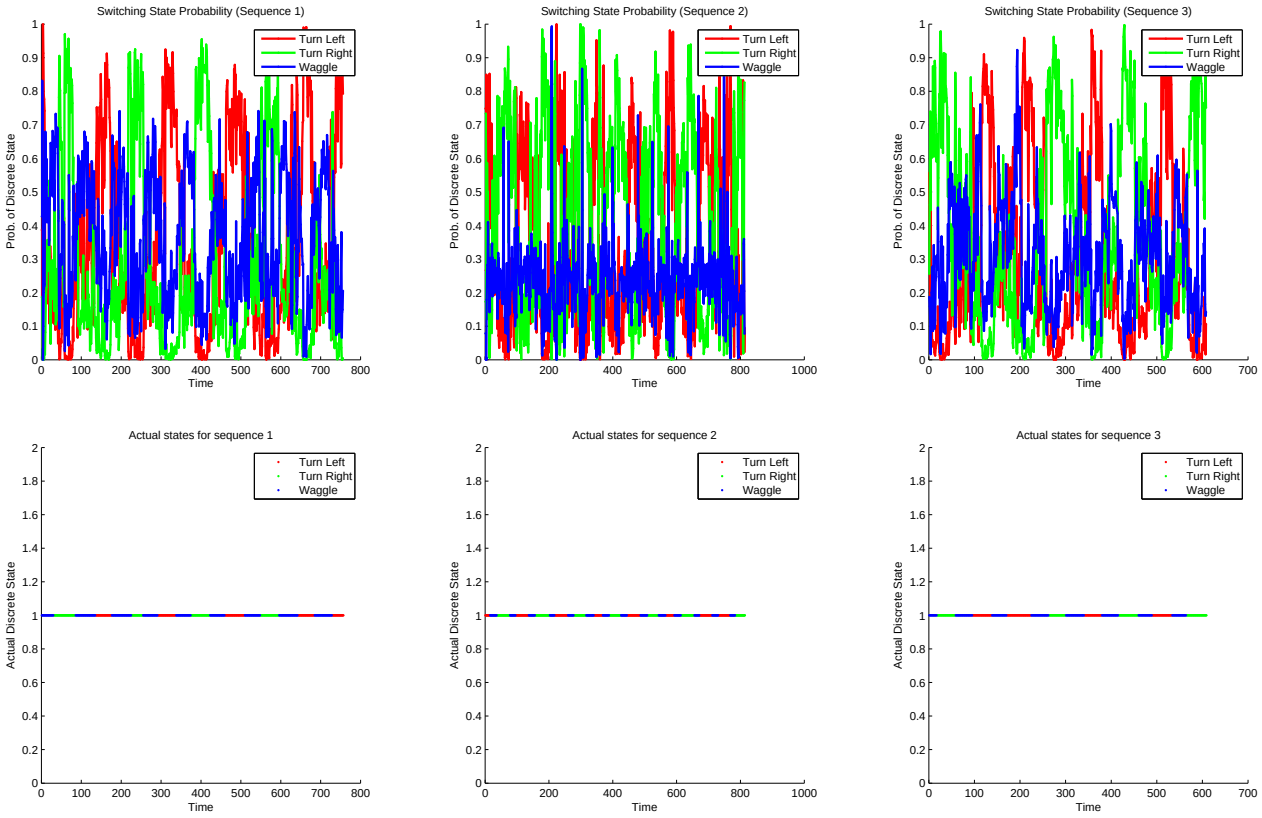
g) Figures 9 and 10 show an analysis of the learning and inference on the dancing honeybee data for each of the three test sequences. In Figure 9 the ground truth (first row) 2D position and estimated 2D position (second row) show reasonable performance in predicting the target position on all test scenarios. On second and third test scenarios the inference underestimates the presence of "Waggle" state. The estimate angle is shown against ground truth (third row).

In Figure 10 the estimated state probabilities at each time step and the ground truth state are shown in the first row and second row, respectively. As can be seen, the ground truth discrete states are generally given high probability in the state sequence estimates.



(a) (b) (c)

Figure 9: Analysis plots for the dancing honeybee data test sequences. (first row) True 2D position. (second row) Particle filter estimated position. (third row) Estimated angle vs. truth vs. time.



(a) (b) (c)

Figure 10: Analysis plots for the dancing honeybee data test sequences. Estimated probability of discrete state vs. time. (second row) Actual discrete state vs. time.