Assignment 4 report - advanced tracking

Luka Boljević

I. Introduction

Object tracking is an important task in computer vision that has many practical applications, such as surveillance, autonomous driving, and human-computer interaction. The goal of object tracking is to locate and follow a target object in a sequence of images or video frames, despite variations in its appearance and motion. In this assignment, we will implement and test two things: a simplified version of the Kalman filter, and a color-based particle tracker. Kalman filter is tested on artificial trajectories using random walk (RW), nearly constant velocity (NCV), and nearly constant acceleration (NCA) motion models. The particle tracker we implemented uses a color histogram as its visual model, and can use various motion models, out of which we will test the aforementioned three. The tracker will be tested on sequences from VOT (Visual Object Tracking) 2014 challenge [1].

II. Experiments

A. Kalman filter

We tested the Kalman filter on 3 different artificial trajectories - a spiral, a star and a trajectory outlining the letters "CV". Kalman filter matrices are determined with 3 parameters: motion model (RW, NCV or NCA), and (power) spectral densities q and r, of the system covariance matrix $\mathbf{Q}$ and observation covariance matrix $\mathbf{R}$ respectively. We tested all three motion models, and a few combinations of spectral densities q and r. The results for the spiral trajectory can be seen on Figure 1, for star trajectory on Figure 2, and for "CV" trajectory on Figure 3.

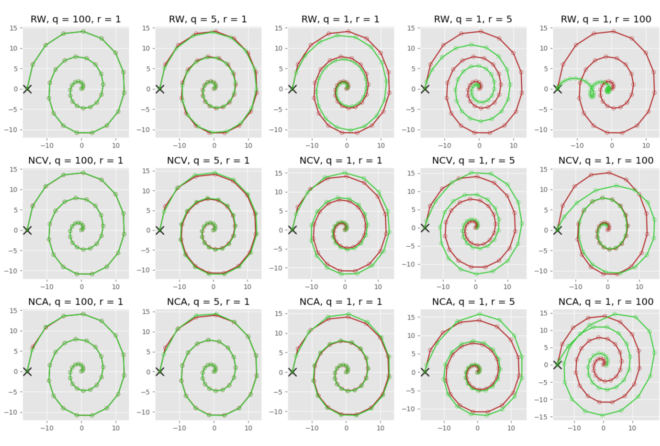

Figure 1. Kalman filter with different motion models and combinations of spectral densities q and r applied to an artificial spiral trajectory. Red trajectory is the original trajectory, while green represents Kalman observations/measurements of it. The black "X" indicates the starting position.

Spectral density q affects the motion model's uncertainty, while spectral density r affects the measurements uncertainty. From the three figures, we can see that when $q = 1$ and we increase r, we aren't following the trajectory very well. On the contrary, when $r = 1$ and we increase q, the trajectory is followed quite nicely. While a perfect match between the original trajectory and the measurements doesn't necessarily mean the best performance, these simple examples indicate that we can expect better performance if the motion model's uncertainty is higher than the observation uncertainty.

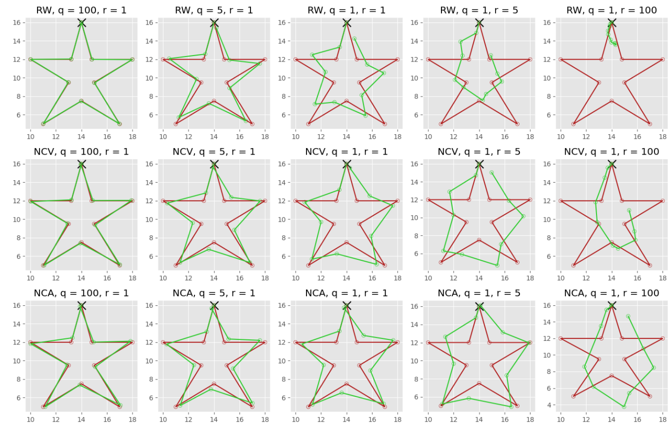

Figure 2. Kalman filter with different motion models and combinations of spectral densities q and r applied to an artificial star trajectory. Red trajectory is the original trajectory, while green represents Kalman observations/measurements of it. The black "X" indicates the starting position.

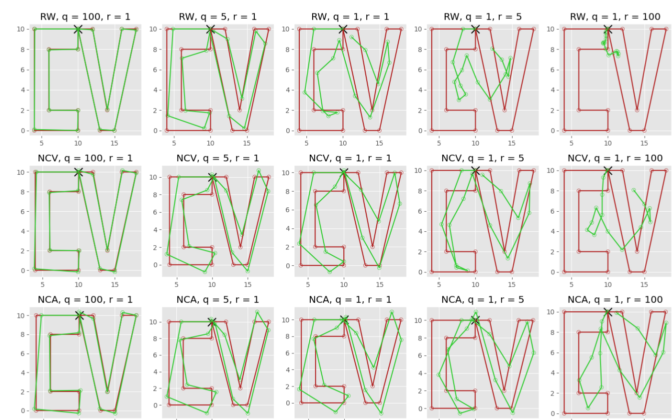

Figure 3. Kalman filter with different motion models and combinations of spectral densities q and r applied to an artificial trajectory outlining the letters CV. Red trajectory is the original trajectory, while green represents Kalman observations/measurements of it. The black "X" indicates the starting position.

B. Particle tracker

Other than manually, we also tested the particle tracker using the lite version of the toolkit used in previous assignment. We tested it on VOT2014 sequences. As stated, we used color histograms as our visual model. We will evaluate the tracker in terms of accuracy (average overlap across all sequences), robustness, number of failures, and average speed.

The tracker has a number of hyperparameters: (1) motion model (RW, NCV or NCA), (2) number of particles N, (3) (power) spectral density q of the system covariance matrix $\mathbf{Q}$, (4) update speed α of our target's color histogram, (5), number of histogram bins m, (6) σ_E for Epanechnikov kernel, used when extracting the color histogram from an image patch, (7) covariance σ_H^2 when converting Hellinger distance to a probability.

During all of our tests, we kept the Epanechnikov kernel sigma $\sigma_E = 1$, and the Hellinger distance sigma $\sigma_H^2 = 0.1$. The reason we used $\sigma_H^2 = 0.1$ was because the curve $\exp\left(-\frac{1}{2}(d_H^2/\sigma_H^2)\right)$ maps Hellinger distance between two histograms d_H to probability the most accurately, based on the following demo graph.

We also empirically established that the most successful trackers used $q = 3$ or $q = 5$, update speed $\alpha = 0.01$ or $\alpha = 0.05$, and number of histogram bins $m = 16$.

We will first show some results when using NCV as the motion model. We will first show the best trackers, based on number of failures, that use 75, 100, 125 and 150 particles respectively. Other than showing us how the number of particles affects tracking performance, those results should give us an idea about the price (speed) to performance ratio of the particle tracker. Results, and the specific hyperparameters of each tracker, can be seen in Table I.

N	Accuracy	Robustness	Num. failures	Avg. speed (FPS)
75	0.4632	0.63	48	**82.8**
100	0.4667	0.64	46	64.8
125	**0.4747**	0.62	49	51.6
150	0.4723	**0.68**	**39**	43.5

Table I
BEST PARTICLE TRACKERS THAT USE NCV MOTION MODEL. FOR ALL TRACKERS, $\sigma_H^2 = 0.1$, $\sigma_E = 1$. IT JUST SO HAPPENED THAT NUMBER OF HISTOGRAM BINS IS $m = 16$ FOR ALL OF THEM TOO. FOR 1ST TRACKER, $q = 3, \alpha = 0.01$. FOR 2ND TRACKER, $q = 5, \alpha = 0.01$. FOR 3RD TRACKER, $q = 5, \alpha = 0.05$. FOR 4TH TRACKER, $q = 3, \alpha = 0.01$.

We can immediately see from Table 1 that apart from average speed, the results aren't drastically different for different N as would perhaps be initially excepted. Accuracy is within 1% for all 4 trackers, and the largest difference in number of failures is 10, which is on average less than a failure per sequence (there are 25 sequences in VOT2014). Overall though, we can say the best tracker was the one with $N = 150$, but average speed decreases the more particles we have. This is normal since the number of particles we have affects how long it takes to move them all after resampling and how long it takes to recompute all of their weights. Other than the number of histogram bins m, other hyperparameters shouldn't affect speed of the tracker that much, or at all.

Summa summarum, based on Table I, we can say that the tracker which offers the best price (speed) to performance ratio would be the one that uses around $N = 100$ particles. It isn't that much worse than the overall "best" tracker in terms of performance (accuracy, number of failures, robustness), but on the other hand, it's at least 50% faster. Now, let us now take the "best" NCV tracker, and its hyperparameters, and just blindly change the motion model to RW or NCA to see what happens. The results for this can be seen in Table II.

Motion model	Accuracy	Robustness	Failures	Speed (FPS)
RW	0.4021	0.41	90	42.8
NCA	0.4712	0.15	192	42.9
NCV	**0.4723**	**0.68**	**39**	**43.5**

Table II
"BLINDLY" CHANGING THE MOTION MODEL TO RW AND NCA FOR THE TRACKER FROM TABLE I WHICH USES $N = 150$ (OVERALL "BEST" ONE). ITS RESULTS ARE COPIED FROM TABLE I FOR EASIER COMPARISON. OTHER THAN THE MOTION MODEL, OTHER HYPERPARAMETERS ARE THE SAME.

Table II shows us that, while average speed (expectedly) didn't change much, the performance definitely did. It is clear from the results that RW and NCA motion models are just not accurate representations of target movement - RW is too random, while the assumption that our targets move with near constant acceleration seems very inaccurate. The performance when we use RW or NCA is, simply put, not good, while NCV seems to model movement most accurately. This is a pattern that constantly occurred during testing, and it tells us NCV is, out of RW, NCV and NCA, the most robust motion model to use.

We also tested a few other particle trackers with motion model set to RW and NCA, but results weren't much better than the particle trackers from Table II. If we showed some of those results, we would reach the exact same conclusion as we did for NCV trackers from Table I - average speed depends on number of particles N, while for a fixed motion model, the best RW and NCA trackers don't differ that much in terms of performance (disregarding the fact they are bad, or straight up unusable).

III. CONCLUSION

In this report, we implemented and tested the Kalman filter on 3 artificial trajectories, and implemented a particle tracker that uses a color histogram as the visual model. For Kalman filter, we saw that if the motion model's uncertainty is bigger than the measurement uncertainty, Kalman filter can be quite robust. Regarding the particle tracker, it was tested on VOT2014 sequences. We mostly discussed how varying the motion model affects performance, and arrived at the conclusion that NCV is the most robust one to use. We also saw that the number of particles N increases performance, but degrades tracking speed, and we discussed that the best price to performance ratio is offered at around $N = 100$ particles.

We can say that the particle tracker is quite robust, with the right motion model and hyperparameters. Unfortunately, one of the main drawbacks of this implementation is that the tracker is not scale adaptive. Likewise, tracking performance can degrade quite a lot if q is set too high or too low, so fine tuning the particle tracker for a specific task may be quite cumbersome.

REFERENCES

[1] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin, "A novel performance evaluation methodology for single-target trackers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2137–2155, Nov 2016.

Appendix

For all motion models, T is the difference in time between discrete measurements, q is the spectral density of system covariance matrix $\mathbf{Q}$.

A. *Random Walk*

State $\mathbf{x} = [x,\ y]^T$

$$\mathbf{F} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{L} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{H} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$
$$\mathbf{\Phi} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{Q} = q \begin{bmatrix} T & 0 \\ 0 & T \end{bmatrix}$$

B. *Nearly constant velocity*

State $\mathbf{x} = [x,\ \dot{x},\ y,\ \dot{y}]^T$

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{L} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$
$$\mathbf{\Phi} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{Q} = q \begin{bmatrix} T^3/3 & T^2/2 & 0 & 0 \\ T^2/2 & T & 0 & 0 \\ 0 & 0 & T^3/3 & T^2/2 \\ 0 & 0 & T^2/2 & T \end{bmatrix}$$

C. *Nearly constant acceleration*

State $\mathbf{x} = [x,\ \dot{x},\ \ddot{x},\ y,\ \dot{y},\ \ddot{y}]^T$

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{L} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix},$$
$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix},$$
$$\mathbf{\Phi} = \begin{bmatrix} 1 & T & T^2/2 & 0 & 0 & 0 \\ 0 & 1 & T & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T & T^2/2 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$
$$\mathbf{Q} = q \begin{bmatrix} T^5/20 & T^4/8 & T^3/6 & 0 & 0 & 0 \\ T^4/8 & T^3/3 & T^2/2 & 0 & 0 & 0 \\ T^3/6 & T^2/2 & T & 0 & 0 & 0 \\ 0 & 0 & 0 & T^5/20 & T^4/8 & T^3/6 \\ 0 & 0 & 0 & T^4/8 & T^3/3 & T^2/2 \\ 0 & 0 & 0 & T^3/6 & T^2/2 & T \end{bmatrix}$$