Exercise 4: Advanced tracking

Klemen Škrlj

I. Introduction

In this exercise we are working on implementation of different motion models that are used with Kalman filter. We test them on 3 different trajectories and comment on their performance based on q and r values. In the second part we implement Particle filter, which uses these ideas. We evaluate performance on VOT14 dataset and compare how performance changes when we use different hyper parameter values. We evaluate tracker in terms of average overlap, number of failures and speed of tracking.

II. Experiments

In the first part we implemented 3 different motion models: Random Walk (RW), Nearly Constant Velocity (NCV) and Nearly Constant Acceleration (NCA). For each of them, we defined all needed matrices as seen in A. We used this motion models together with Kalman filter to test how well they work on artificial trajectories with different hyper parameter values. On Figures 1, 2 and 3 we can see tracking results on spiral, square and random path respectively.

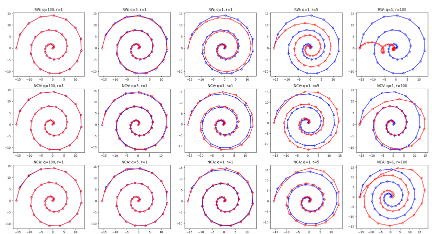

Figure 1. Tracking performance based on motion model and hyper parameter values on spiral trajectory

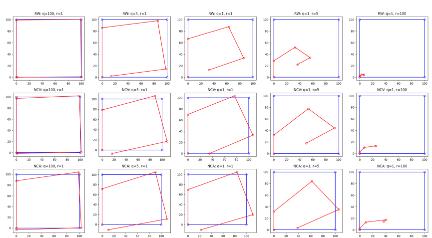

Figure 2. Tracking performance based on motion model and hyper parameter values on square trajectory

When q value is higher our model trusts more our observations while a higher r value means that motion model is more important for prediction. From images we can see that higher q values result in our tracking more closely following the original line. When q starts lowering we quickly start to loose original line, especially in Figure 2. The opposite happens with r where higher values start to degrade tracking. In Figures 1 and 3 we see that NCA model seems to be the best while Figure 2 causes problems for all motion models.

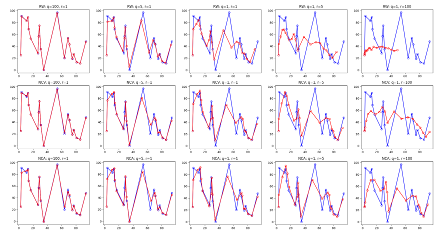

Figure 3. Tracking performance based on motion model and hyper parameter values on random trajectory

In the second part of the exercise we implement Particle filter. In the initialization we construct a visual feature vector based of ground truth bounding box. For this we use RGB patch of the image and represent it with a histogram. We also initialize the Kalman motion model and N particles with weights. In the update function we resample N particles with repetitions. From each particle a new visual feature vector is extracted and based on Hellinger distance from ground truth new particle weight is calculated. At the end the new object position is calculated as the weighted mean between all the particles and template histogram is updated accordingly depending on forgetting factor α.

For hyper parameter testing we choose "ball", "basketball", "bicycle", "bolt" and "car" sequences from VOT14 challenges. They incorporate tough tracking examples with fast changes in movement speed, occlusions and similar objects in the surroundings. We have 5 main parameters that we want to optimize: number of histogram bins (nbins), number of particles(N), α for updating template, motion model and q in covariance matrix. For calculating q we use Equation 1, which we defined by trial and error. Since q is dependent on the amount of movement of the tracked object, we make it scale based on bounding box size.

$$q = 0.1 * min(bbox_{width}, bbox_{height}) \tag{1}$$

In Table II we see tracking results when using NCV motion model with nbins = 8, N = 80 and α = 0.01

Seq. name	**#Frames**	**Overlap**	**#Failures**
ball	602	0.537	2
basketball	725	0.453	4
bicycle	271	0.409	3
bolt	350	0.490	3
car	252	0.291	1
Average overlap		0.43	
Total #Failures		13	

Table I
Results using NCV motion model

We also test how performance changes usign RW or NCA model and present results in Table II and II respectively.

Seq. name	#Frames	Overlap	#Failures
ball	602	0.594	2
basketball	725	0.551	1
bicycle	271	0.480	2
bolt	350	0.425	0
car	252	0.350	1
Average overlap		0.48	
Total #Failures		6	

Table II
Results using RW motion model

Seq. name	#Frames	Overlap	#Failures
ball	602	0.434	5
basketball	725	0.405	8
bicycle	271	0.359	5
bolt	350	0.422	5
car	252	0.308	2
Average overlap		0.39	
Total #Failures		25	

Table III
Results using NCA motion model

The best average overlap and minimal number of failures is acquired when using Random Walk motion model while NCA is the worst for every sequence. Since every sequence features moving objects like cars, runners or balls, we would intuitively expect that something like NCV model would be the best. But here the catch is that on all sequences camera isn't still so not only does the tracked object move but also camera. This somewhat cancels out the object's movement and thus RW motion model fits the best.

In Table IV we compare performance when using different number of bins and α values for our template, while we keep other parameter values as: motion model = RW and N = 80.

nbins	α	Average overlap	#Total failures
8	0.01	0.48	6
16	0.01	0.46	9
32	0.01	0.39	11
8	0	0.50	7
8	0	0.50	7
8	**0.01**	**0.48**	**6**
8	0.05	0.47	8

Table IV
Results with different values for nbins and α parameters

We get the best results when using only 8 bins for template histogram. Larger amount of bins also means larger computational requirements in the update function which causes slower speed overall. When looking at α values, we get the best overlap when we don't update our template over duration of the tracking with currently best values. But with this we also get one more failure, so there is a trade-off between this two measures. We choose nbins = 8 and $\alpha = 0.01$ for our best hyper parameter values.

Lastly we tested out how number of particles affects tracking performance and speed. We present results in Table V.

More particles generally means better results in terms of less number of failures and larger overlap. But there is a point of diminishing return, where the performance doesn't increase substantially. This is around 80 to 100 particles. But we can also see that more particles causes slower tracking speed since we have to extract and evaluate more patches in the update function. At the end we choose 80 particles as the best value for parameter N.

N	Average overlap	#Total failures	Average FPS
20	0.47	9	253.60
40	0.48	8	134.37
60	0.47	7	90.89
80	**0.48**	**6**	**69.36**
100	0.49	8	55.96 FPS
120	0.46	5	46.45 FPS

Table V
Results with different number of particles

In Table II we can see overall performance on the whole VOT14 dataset using parameter values: nbins = 8, $\alpha = 0.01$, N = 80 and RW as our motion model.

Seq. name	Overlap	#Failures
ball	0.593	2
basketball	0.550	1
bicycle	0.479	2
bolt	0.424	0
car	0.350	1
david	0.476	2
diving	0.352	0
drunk	0.396	0
fernando	0.392	1
fish1	0.313	6
fish2	0.352	3
gymnastics	0.581	0
hand1	0.377	5
hand2	0.362	9
jogging	0.534	1
motocross	0.330	3
polarbear	0.626	0
skating	0.344	1
sphere	0.378	1
sunshade	0.399	6
surfing	0.464	0
torus	0.515	4
trellis	0.362	2
tunnel	0.243	5
woman	0.524	1
Average overlap		0.429
Total #Failures		56

Table VI
Results on the whole VOT14

III. Conclusion

We looked at 3 basic motion models and test out how do they affect trajectory tracking using Kalman filter. Then we implemented Particle filter and evaluated it on VOT14 dataset. We compared hyper parameter values on smaller subset and lastly reported the results over the whole dataset.

References

Appendix

Matrices for RW:

$$x_{state} = \begin{bmatrix} x \\ y \end{bmatrix} \qquad F = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\Phi = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} Tq & 0 \\ 0 & Tq \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Matrices for NCV:

$$x_{state} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \qquad F = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\Phi = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} \frac{T^3q}{3} & 0 & \frac{T^2q}{2} & 0 \\ 0 & \frac{T^3q}{3} & 0 & \frac{T^2q}{2} \\ \frac{T^2q}{2} & 0 & Tq & 0 \\ 0 & \frac{T^2q}{2} & 0 & Tq \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Matrices for NCA:

$$x_{state} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \\ \ddot{x} \\ \ddot{y} \end{bmatrix} \qquad F = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\Phi = \begin{bmatrix} 1 & 0 & T & 0 & \frac{T^2}{2} & 0 \\ 0 & 1 & 0 & T & 0 & \frac{T^2}{2} \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} \frac{T^5q}{20} & 0 & \frac{T^4q}{8} & 0 & \frac{T^3q}{6} & 0 \\ 0 & \frac{T^5q}{20} & 0 & \frac{T^4q}{8} & 0 & \frac{T^3q}{6} \\ \frac{T^4q}{8} & 0 & \frac{T^3q}{3} & 0 & \frac{T^2q}{2} & 0 \\ 0 & \frac{T^4q}{8} & 0 & \frac{T^3q}{3} & 0 & \frac{T^2q}{2} \\ \frac{T^3q}{6} & 0 & \frac{T^2q}{2} & 0 & Tq & 0 \\ 0 & \frac{T^3q}{6} & 0 & \frac{T^2q}{2} & 0 & Tq \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$