Solution to analysis in Home Assignment 3

Lizi Teng + lizi(cid)

Analysis

In this report I will present my independent analysis of the questions related to home assignment 3. I have discussed the solution with Qun Zhang, but I swear that the analysis written here are my own.

1 Approximations of mean and covariance

a)&b)&c)

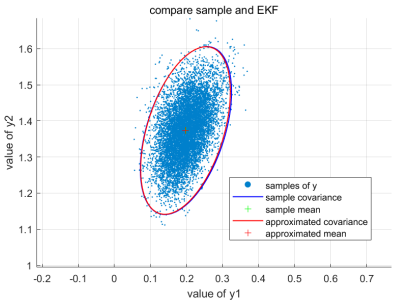
Firstly we get 10000 samples from x distribution and r distribution and then apply nonlinear function $y = h(x) + r$ to get 10000 y samples. Due to the law of large numbers, the sample mean/covariance can be consider as the great approximations for the three cases. Then we can compare the performance of three different Kalman filters.
Then I just appply the prediction step in three kind nonlinear Kalmen filter to approximate the mean/covariance. For EKF, it's basicly compute the jacobean and affine the distribution. For UKF and CKF, we firstly get sigma point with two different strategies then apply nonlinear function and calculate mean/covariance of y.
Here are the table for three cases with different method and the figures:
For the means:

	Samples	EKF	UKF	CKF
case1	[0.1981;1.3741]	[0.1974;1.3734]	[0.1983;1.3743]	[0.1983;1.3743]
case2	[2.3291;2.3552]	[2.3562;2.3562]	[2.3269;2.3550]	[2.3265;2.3550]
case3	[-0.5948;2.1524]	[-0.5880;2.1588]	[-0.5949;2.1524]	[-0.5948;2.1523]

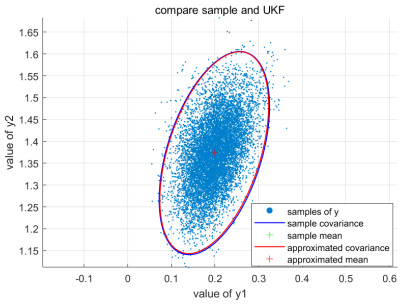
For the covariances:

	Samples	EKF	UKF	CKF
case1	$\begin{bmatrix} 0.0017 & 0.0014 \\ 0.0014 & 0.0058 \end{bmatrix}$	$\begin{bmatrix} 0.0017 & 0.0015 \\ 0.0015 & 0.0060 \end{bmatrix}$	$\begin{bmatrix} 0.0017 & 0.0015 \\ 0.0015 & 0.0059 \end{bmatrix}$	$\begin{bmatrix} 0.0017 & 0.0015 \\ 0.0015 & 0.0059 \end{bmatrix}$
case2	$\begin{bmatrix} 0.0549 & 0.0103 \\ 0.0103 & 0.0020 \end{bmatrix}$	$\begin{bmatrix} 0.0500 & 0.0100 \\ 0.0100 & 0.0020 \end{bmatrix}$	$\begin{bmatrix} 0.0600 & 0.0108 \\ 0.0108 & 0.0020 \end{bmatrix}$	$\begin{bmatrix} 0.0566 & 0.0105 \\ 0.0105 & 0.0020 \end{bmatrix}$
case3	$\begin{bmatrix} 0.0097 & -0.0111 \\ -0.0111 & 0.0149 \end{bmatrix}$	$\begin{bmatrix} 0.0092 & -0.0111 \\ -0.0111 & 0.0148 \end{bmatrix}$	$\begin{bmatrix} 0.0099 & -0.0112 \\ -0.0112 & 0.0151 \end{bmatrix}$	$\begin{bmatrix} 0.0097 & -0.0112 \\ -0.0112 & 0.0150 \end{bmatrix}$

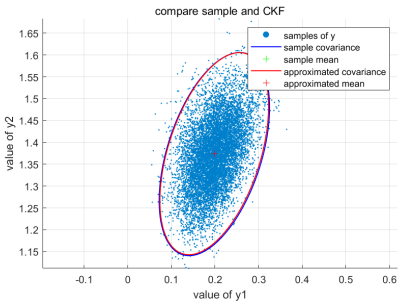
And here are the figures:



EKF compared with samples

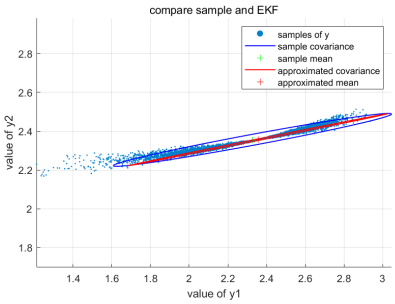


UKF compared with samples

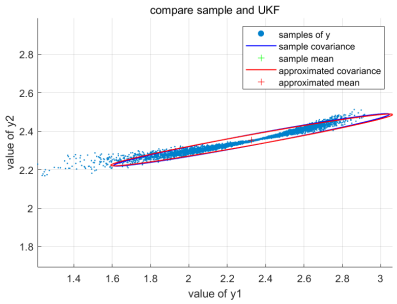


CKF compared with samples

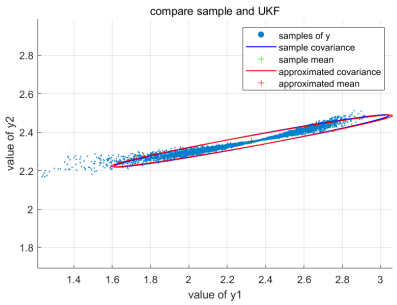
case1



EKF compared with samples

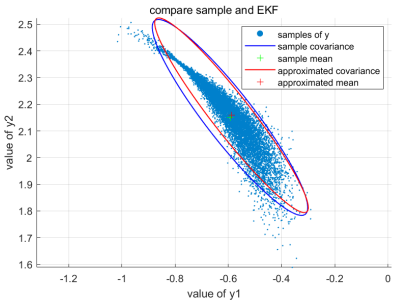


UKF compared with samples

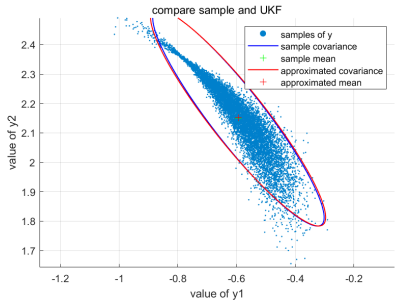


CKF compared with samples

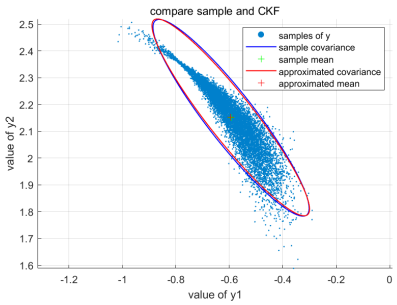
case2



EKF compared with samples



UKF compared with samples



CKF compared with samples

case3

d)

We can see that the EKF performs well in case1 and almost as well as UKF and CKF. But in case3 and case2, the accuracy of EKF is not good enough and act a lot worse than UKF and CKF. That is because the EKF ignores high order terms when doing derivation and when the non-linearity is too strong, it can not describe the nonlinear transition very well. However the UKF and CKF use sigma point to approximate the distribution, could have a better result when dealing with strong nonlinear functions.

Here is the trade-off. When using UKF and CKF, we have a better accuracy when capturing non-linearity. However, it might have higher computational complexity. When we use EKF, it shows bad accuracy with strong non-linearity, but the computational complexity is lower.

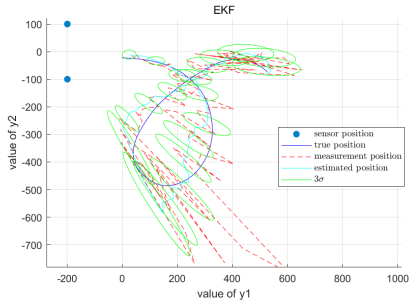
As an engineer, I would consider the application scenario and the non-

linearity of the function. When we are facing a weak non-linearity, we can choose the EKF to have a higher computational speed. But with strong non-linearity, we need to use UKF or CKF to have an acceptable accuracy. When the speed of the system is limited, we also prefer EKF to fasten the computational speed.

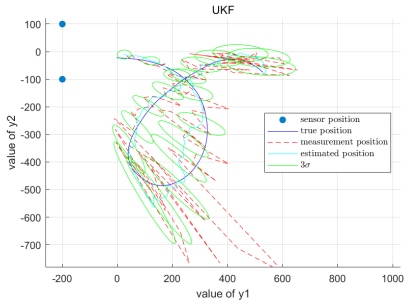
2 Non-linear Kalman filtering

a)b)

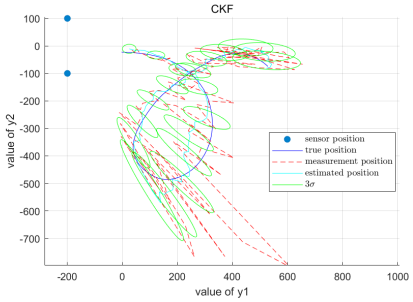
In order to evaluate the results of different filters, we compare the estimated positions to the true positions. I plot the sensor positions, the measurements and the true position sequence, all in Cartesian coordinate system. For each of the filters, plot the estimated position sequence, together with 3σ-contours at every 5 estimate. And here are the result for the 3 cases:



EKF estimated position compared with true position

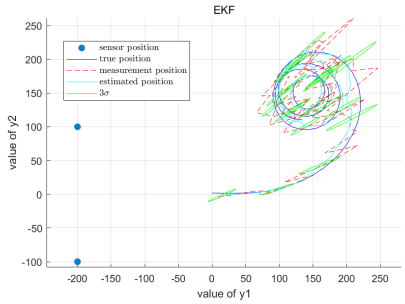


UKF estimated position compared with true position

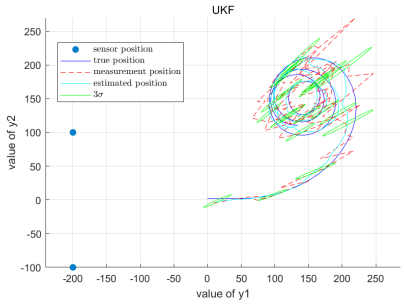


CKF estimated position compared with true position

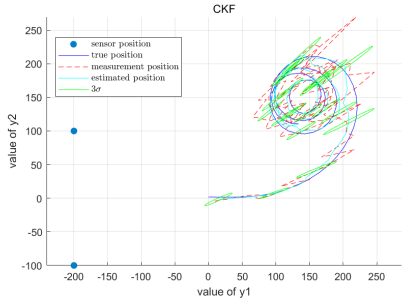
case1



EKF estimated position compared with true position



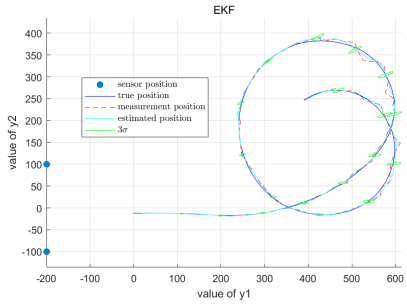
UKF estimated position compared with true position



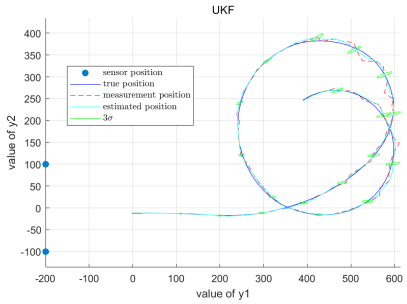
CKF estimated position compared with true position

case2

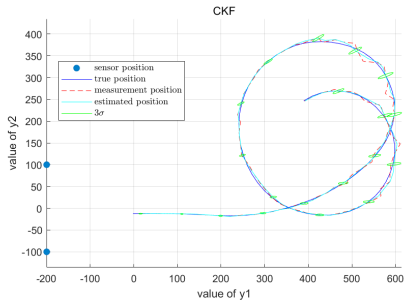
For the first two cases, the measurement noises are big and it even hard to tell the real trajectory from the measurement with eyes. But the nonlinear kalman filters performs good and can almost follow the true trajectory but still with a lot noise. And the error covariances can represent the uncertainty well, because for all six figures the true trajectories are in the area of 3σ level ellipses. And all the three nonlinear Kalman filters have almost the same performance, that might because the scale is too big and we can hardly find the difference. But due to the next question, the CKF and UKF are a little bit better than EKF.



EKF estimated position compared with true position



UKF estimated position compared with true position



CKF estimated position compared with true position

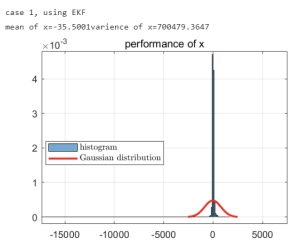
case3

For the case3, the measurement noise is a lot smaller, and the measurement value can be more believable and we have even better performance for the three Kalman filter, and the covariance of the estimate is smaller due to the update step.And the error covariance can represent the uncertainty well, because the true trajectories are in the area of 3σ level ellipses. We can hardly see the difference between different Kalman filters, that might because the nonlinear function is smooth.

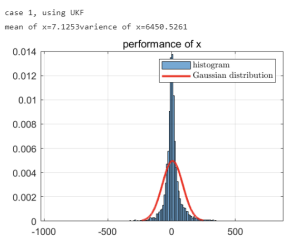
c)

From the result of the histograms of the three cases, we can see that the EKF has a much worse performance compared to the UKF and CKF. The mean

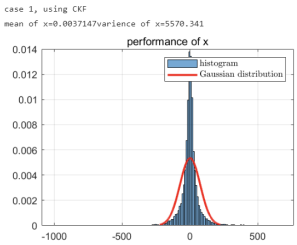
of the EKF is far from zero and the standard deviation is higher. And the UKF and CKF have similar and better performance. (The CKF performs slightly better than the UKF sometimes.)
And the Gaussian distribution generated using the mean and variance of the position errors does not match with its normalized histogram. The x and y are not normal distribution after applied motion model which is a nonlinear function. And so the errors are also not normal distribution.
We can find that the estimated x are more accurate than y in most cases(the means and covariance of errors are smaller). That is because the two sensor are set on a vertical line. And the y position of the object can be hard to detect when it is aligned with the two sensor, the errors are higher here.



EKF

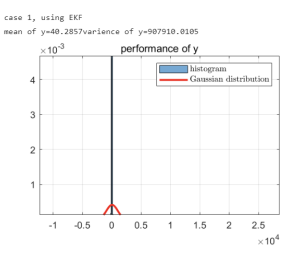


UKF

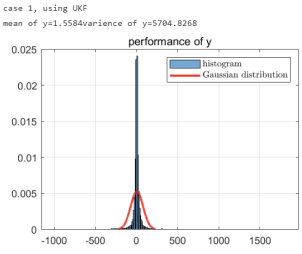


CKF

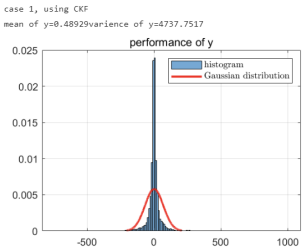
the performance of x for case1



EKF

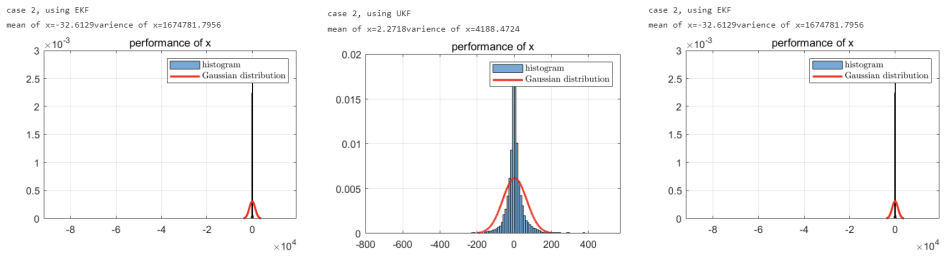


UKF



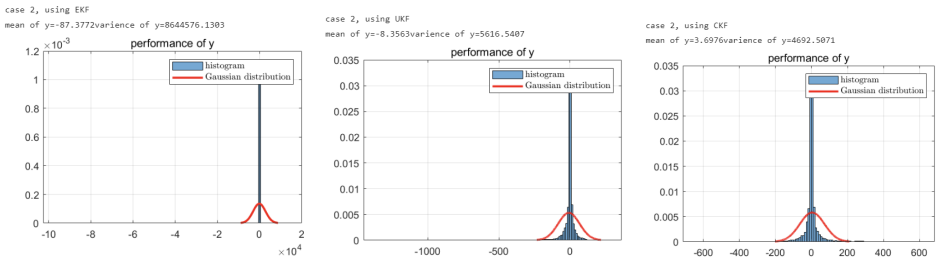
CKF

the performance of y for case1



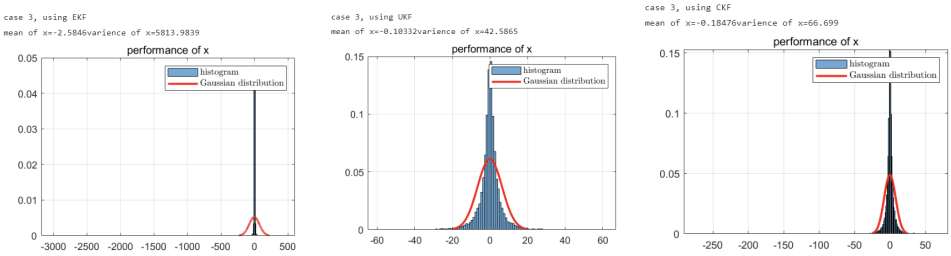
EKF UKF CKF

the performance of x for case2



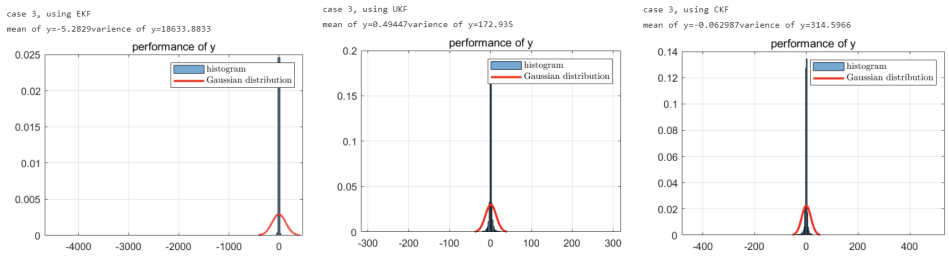
EKF UKF CKF

the performance of y for case2



EKF UKF CKF

the performance of x for case3



EKF UKF CKF

the performance of y for case3

3 Tuning non-linear filters

a)

The CKF has the best performance in previous,so I choose CKF to be used in the following tasks. The parameters of Q were increased/decreased by several orders of magnitude.

When only σ_v is increased, the estimated position becomes very noisy and the velocity is changing randomly too much. And that is not reasonable for a moving car.

When only σ_w is increased, the car's orientation changes all the time even though it's following a straight line. However it will believe the measurement more, and can better adapt higher angular acceleration in true sequence.

When both σ_v and σ_w are increased, the filter tends to trust more the measurements.But with the noisy measurement, we will have a noisy vehicle trajectory.

When only σ_v is decreased, we will have a clear trajectory with less noise of the position, because the car is moving with a constant speed and we can trust the model a lot.

When only σ_w is decreased, when the angle velocity start to change, the estimate value is not able to follow the true trajectory fast enough. For example when the car is leaving straight line to curve or when leaving curve to straight line.

When both σ_v and σ_w are decreased, the car is not able to follow fast enough

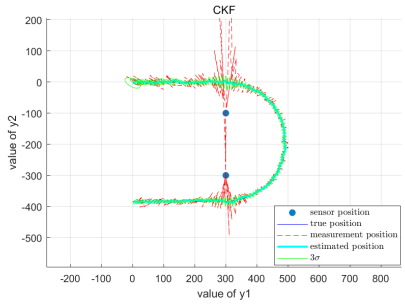
when the angle velocity is changing, the performance is similar to only decrease σ_w.

b)

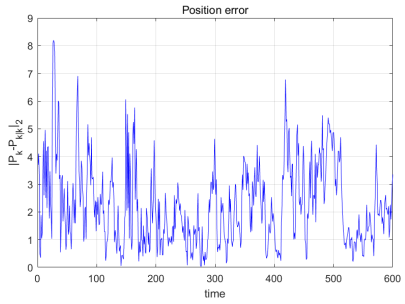
When we think about how the true trajectory is generated, we can find that the the veloity is a constant and the turn-rate changes when the vehicle is entering or leaving the curve. So the motion model with prior can greatly reflect the speed of the car, and the variance of velocity(σ_v) can be really small. So I give it $\sigma_v = 0.1$.
But the model can not represent the trajectory very well when the turn-rate is changing. So angular velocity variance has to be tuned until we have a value such that the estimate trajectory is not too noisy and can follow the true trajectory. And I give it $\sigma_w = \pi/180$.
And $Q = diag([0, 0, \sigma_v^2, 0, \sigma_w^2])$.

c)

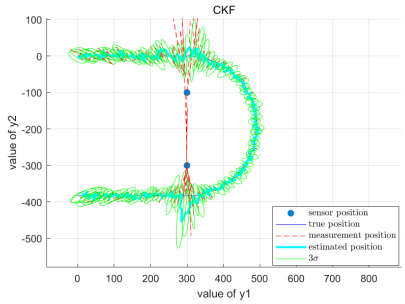
We present typical results by generating sensor positions, the true position sequence, the positions corresponding to the measurements, and the estimated position sequence for three different process noise settings. For well-tuned, we use the value in the last task. For too large, we have $\sigma_v = 100, \sigma_w = 100 * \pi/180$. For too small, we have $\sigma_v = 0.01, \sigma_w = 0.01 * \pi/180$. The result are as followed:
And we also compute the position error and plot vs time. As we can see in the following figures ,we have lowest and most constant errors when using well-tuned variance. And when the prosses noise is too small, we have really large errors when the vehicle is entering and leaving the curve. And when we have large noise, the filter tends to trust more the measurements, leading to bigger and noisier errors.And when the position is aligned with the two sensor locations, the errors are higher.



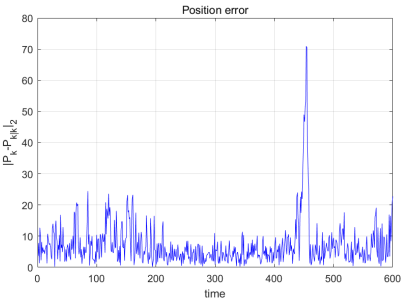
trajectory when well-tuned



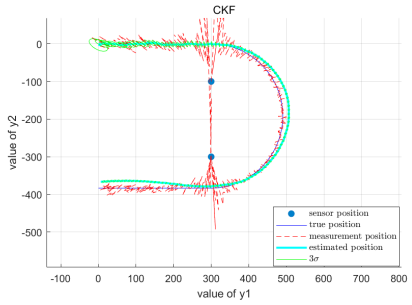
position array when well-tuned



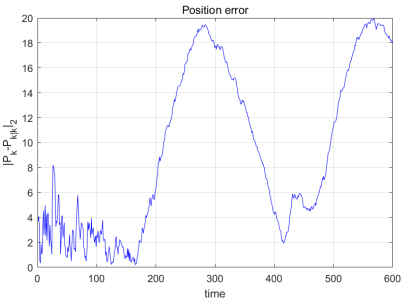
trajectory when too large



position array when too large



trajectory when too small



position array when too small

d)

No it is not possible to tune the filter to have accurate estimate for the whole sequence. That is because we have different best tuned parameters for

straight/curve line and for the transition.
There is conflict for different part of the trajectory. When it's for straight line or for the turn, the motion model can well describe the trajectory because the vehicle is moving with constant velocity and angular velocity. So we should have both σ_v and σ_w very low. But when it comes to the transitions, the estimate can not follow the trajectory fast enough.That is because the angular velocity is changing. So we need a higher value for the σ_w.
Yes, we want the same parameter setting for the straight line as for the turn, since they can all believe in the motion model. And yes, we want the same parameter setting for transitions from straight to turning and from turning to straight.