EE-638 Term Project Report

Interacting Multiple Sensor Unscented Kalman Filter for Accelerating Object Tracking

Harsh Deshpande, 16D070011

November 27, 2019

Contents

1 Introduction

1.1 Tracking

Tracking the path of a moving object has been studied in quite detail and Kalman filters and its variants have been widely used to attain estimates of the object's state.Tracking can either be performed by a device placed on the moving object itself or by stationary sensors (one or many) which communicate with the object. Distributed tracking is the method in which a set of sensors placed at various positions work together to estimate the position of a moving object.

1.2 Ways of Distributed Tracking

Distributed tracking can be achieved in various ways.They can be divided into 2 major categories: **centralized** and **distributed**. In centralized tracking all the sensors convey their reading to a centralized node where all calculations take place. On the other hand, in a distributed tracking system, all nodes perform their individual computations and convey a limited amount of information to the central node for co-ordination.
The number of nodes is usually used to decide which method is more effective.The limited energy, computational power and communication resources of a sensor node require the use of a large number sensor nodes in a wider region.This warrants the use of distributed computation as in general, the cost of computing a bit is much lesser as compared to transferring a bit. The methods adopted until include a Distributed Kalman Filter(DKF) decentralized kalman filters. The drawback of these techniques is they do no work well for non-linear systems.

1.3 Decentralized tracking for non-linear systems

This project tries to tackle both these problems at the same time.The problem of non-linear systems is solved by using Unscented Kalman Filter (UKF) and distributed computing on all nodes are fused together using Interacting Multiple Sensor (IMS) algorithm. Hence, the entire filtering process is called Interacting Multiple Sensor Unscented Kalman Filter (IMSUKF).

2 The Setting

2.1 State evolution

The state at any time k evolves according to the state space formula:

$$x(k) = f(x(k-1)) + w(k) \tag{1}$$

where, x(k) is the state of the moving object at time k and w(k) is white and uncorrelated gaussian noise with fixed variance Q.The sensor reading are modelled by a non-linear function (z) of the current state and are given by the equation:

$$z(k) = h(x(k)) + v(k) \tag{2}$$

v(k) is the measurment noise and is modelled as a white noise with variance R.
In our case, the measurement of each node is the strength of the RSSI signal each node receives when the object sends the signa;. The strength of the signal depends on the distance between the sender and the receiver. This is modelled in the following way:

$$h(d) = -20 - 23 * log((x_i - x_{obj})^2 + (y_i - y_obj)^2) \tag{3}$$

2.2 Initial State and other Assumptions

The state of the object is considered to have its position along the x-axis and its velocity in x-direction. Since its movement is restricted along the x-axis, this is sufficient.The object is considered to have a constant acceleration and starts at $[11]^T$ with variance **I**. Thus state space equation is given by:

$$[x_1(k); x_2(k)]^T = [x_1(k-1) + x_2(k-1); x_2(k-1) + 0.01]^T + w(k) \tag{4}$$

2.3 Sensors

The object is assumed to be moving along the x-axis with a constant acceleration. Keeping this in mind, the sensors are placed uniformly on a line parallel to the x-axis. Each sensor receives an RSSI value according to the function h() , performs some computations and relays the information to a centre node.

3 The IMSUKF Algorithm

The IMSUKF algorithm can be decomposed into 2 integral parts, the UKF running at each node and the IMS algorithm running at the central node. This section describes each of these in further detail.

3.1 Unscented Kalman Filter

The problem of non-linear function estimation is performed using the Unscented Kalman Filter. In this, the state of the object is estimated from the previous state using what are known as sigma points. Sigma points are points such that they have a mean and variance the same as that of the previous estimate. It is shown, that for Gaussian densities, $(2n+1)$ sigma points can accurately model non-linear functions of a state model with n dimensions.This is known as the **Unscented Transformation** and the UKF integrates this in the Kalman Filter structure.
At the start of each cycle, we first estimate the sigma (s_0 to s_{2n+1} points and their weights

(mean u and variance C) from the previous state estimate:

$$s_0 = u \tag{5}$$
$$S = \sqrt{n * C} \tag{6}$$
$$s_i = u + S(i) \forall 0 < i < n + 1 \tag{7}$$
$$s_i = u - S(i) \forall n < i < 2n + 1 \tag{8}$$
$$w_i = \frac{1}{2n+1} \tag{9}$$

Next,we estimate the current state from the sigma points:

$$s_j(k) = f(s_j(k-1)) \tag{10}$$
$$x(k|k-1) = \sum_{j=0}^{2n+1} w_j * s_j(k) \tag{11}$$

$$K_x(k|k-1) = Q + \sum_{j=0}^{2n+1} w_j * (s_j(k) - x(k|k-1)) * (s_j(k) - x(k|k-1))^T \tag{12}$$

$$z_j(k) = h(s_j(k)) \tag{13}$$
$$z(k|k-1) = \sum_{j=0}^{2n+1} w_j * z_j(k) \tag{14}$$

$$K_z(k|k-1) = R + \sum_{j=0}^{2n+1} w_j * (z_j(k) - z(k|k-1)) * (z_j(k) - z(k|k-1))^T \tag{15}$$

$$K_{xz}(k|k-1) = \sum_{j=0}^{2n+1} w_j * (s_j(k) - x(k|k-1)) * (z_j(k) - z(k|k-1))^T \tag{16}$$

After this we perform the update step where the current observation z(k) is used to get current state estimate:

$$\Phi(k|k-1) = K_{xz}(k|k-1) * K_z(k|k-1)^{-1} \tag{17}$$
$$\tilde{z}(k) = z(k) - z(k|k-1) \tag{18}$$
$$x(k) = x(k|k-1) + \Phi(k|k-1)\tilde{z}(k) \tag{19}$$

$$K_x(k) = K_x(k|k-1) - \Phi(k|k-1) * K_z(k|k-1) * \Phi(k|k-1)^T \tag{20}$$

This finishes the UKF algorithm cycle. This is preformed at each node and the 4-tuple including the state, variance, z-innovation ($\tilde{z}$) and its variance are sent to the centre node for further computation.

3.2 Interacting Multiple Sensors Algorithm

The Interacting Multiple Sensor Algorithm is based on a markov process. Each node has a probability of contributing to the total estimate which changes according to the probability transition matrix **P** (diagonal terms much higher). Suppose the initial probability of each node is given by the vector ct(k-1). Nodes are indexed by i:

$$c(k) = ct(k-1) * P \tag{21}$$

$$L^i(k) = \mathcal{N}(\tilde{z}_i(k); 0, K_z^i(k|k-1)) \tag{22}$$

$$ct_i(k) = \frac{L^i(k) * c_i(k)}{\sum_i L^i(k) * c_i(k)} \tag{23}$$

This is then used to get the overall estimate which is used as the output:

$$\hat{x}(k) = \sum_i ct_i(k) * x_i(k) \tag{24}$$

$$K(k) = \sum_i ct_i(k) * (K_x^i(k) + (\hat{x}(k) - x_i(k)) * (\hat{x}(k) - x_i(k))^T) \tag{25}$$

This gives us the estimate for this cycle. For next cycle, smoothing is performed according to:

$$c_{ji}(k) = \frac{P_{ji} * ct_j(k)}{c_i(k+1)} \tag{26}$$

$$\hat{x}_i(k) = \sum_j x_j(k) * c_{ji}(k) \tag{27}$$

$$\hat{Kx}_i(k) = \sum_j c_{ji}(k) * (K_x^j(k) + (\hat{x}_(k) - \hat{x}(k)) * (\hat{x}_(k) - \hat{x}(k))^T) \tag{28}$$

Each node is then updated with the mixing estimates $\hat{x}_i(k)$ and $\hat{Kx}_i(k)$ and the next cycle begins.

4 Simulations

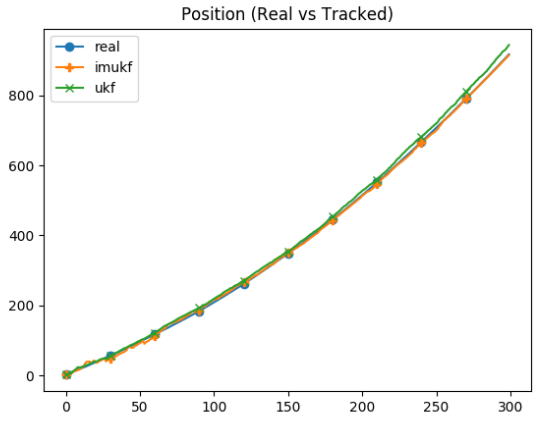
4.1 Parameters

The parameters used were:

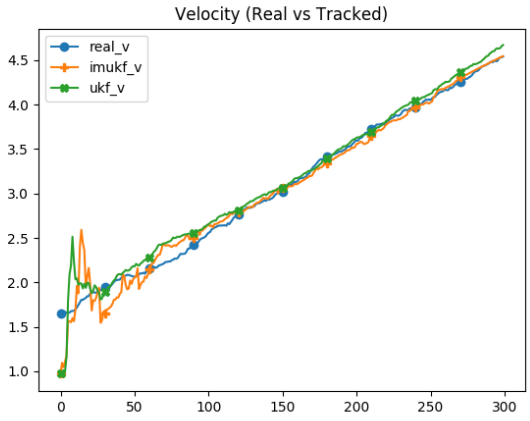
$$Q = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.0001 \end{bmatrix} \tag{29}$$

$$R = 100 \tag{30}$$

$$\tag{31}$$



(a) Actual Position vs Tracked Position



(b) Actual Velocity vs Tracked Velocity

$$P = \begin{bmatrix} 0.95 & 0.01 & 0.01 & 0.01 & 0.01 & 0.01 \\ 0.01 & 0.95 & 0.01 & 0.01 & 0.01 & 0.01 \\ 0.01 & 0.01 & 0.95 & 0.01 & 0.01 & 0.01 \\ 0.01 & 0.01 & 0.01 & 0.95 & 0.01 & 0.01 \\ 0.01 & 0.01 & 0.01 & 0.01 & 0.95 & 0.01 \\ 0.01 & 0.01 & 0.01 & 0.01 & 0.01 & 0.95 \end{bmatrix} \tag{32}$$

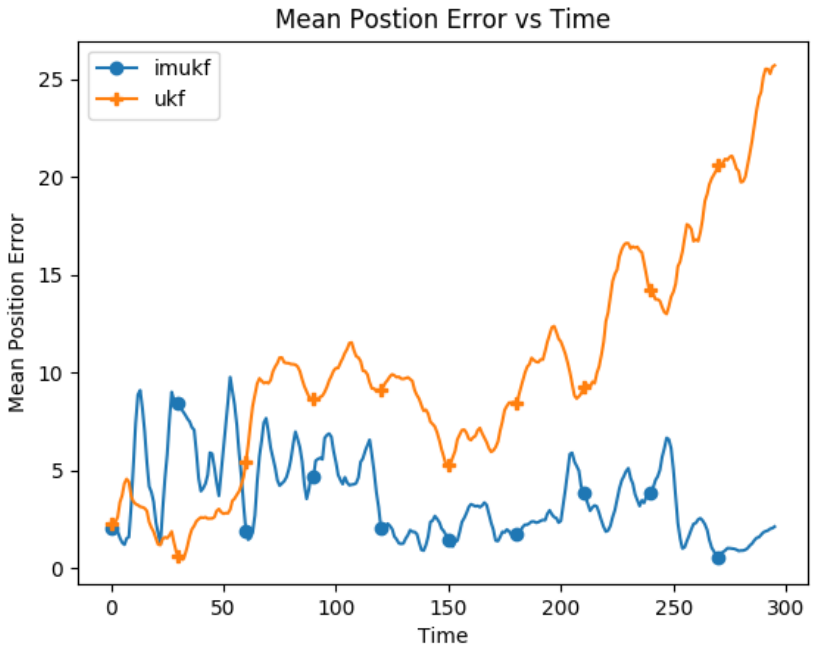
Node positions were fixed to be (-1, 20), (150,20), (300, 20), (450, 20), (600, 20) and (750, 20).

4.2 IMSUKF/UKF comparision

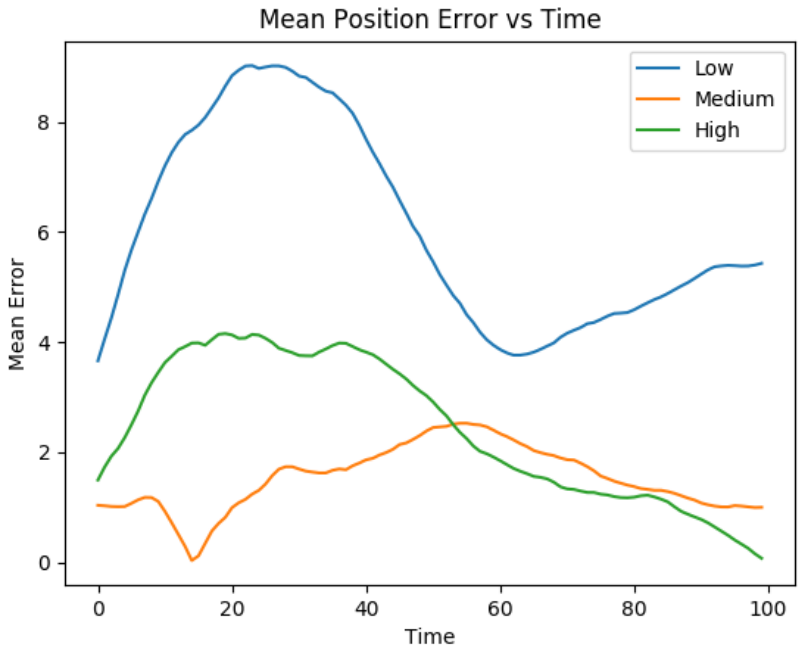
For validating the IMSUKF, its performance was compared to that of UKF applied on an individual node alone. Figure 2a shows simulations averaged over 50 independent runs. As you can see, the error of UKF alone increases linearly as variance R is in range of the a function observations whereas IMSUKF error saturates. Initial error is high due to initial assumptions but joint tracking reduces error of IMSUKF over time. Thus IMSUKF is indeed better than normal UKF on a single node. To justify the number of nodes Figure 2b shows the error in position estimate with three different node densities (high,medium and low).The error is high initially for high density due to wrong starting estimates but reduces with time as long as object is in that cluster range

5 Conclusion and Code Instructions

Thus, we can conclude that IMSUKF performs better than UKF and its preformance increases with increasing node density. To run the code follow the README or visit: `https://github.com/harshsd/Estimation_Project`.
You can view the paper at: `https://ieeexplore.ieee.org/document/5461518`



(a) Mean error for IMSUKF and UKF



(b) Mean error for IMSUKF for different node densities

Figure 2: Figure 2