24

Robot Localisation Using a Distributed Multi-Modal Kalman Filter

Oleg Sushkov and William Uther
University of New South Wales and National ICT Australia
Australia

1. Introduction

It is extremely important for any mobile robotics system to know where it is. If an agent does not know where it is or where the objects around it are, meaningful actions become very difficult to perform. The main obstacle to efficient and accurate robot localisation is noise. Noise is present in every part of a robotics system, both in the sensors as well as in the actuators. In a world without noise, with perfect sensors and actuators, localisation would be a relatively simple task.

It is also important to note that localisation is not necessarily restricted to determining the pose of a robot, but can also include tracking the state of other objects. Taking the Robocup domain as an example, localisation could include discovering the ball position and velocity and team-mate robot poses as well as the robot's own position.

The core concept of robot localisation is estimating the world state through sensor data. In most situations, the world state is not directly observable in its entirety – the world is only partially observable. In such cases the state of the world must be inferred from the given sensor data, and integrated over time. Different algorithms for robot localisation provide varying ways of incorporating the partial observations of the world state into an internal representation of the complete world state.

In this chapter we discuss one particular method of Robot localisation as applied to the 4-Legged League as part of Robocup, developed by the rUNSWift team for the 2006 competition. We base our system on Bayesian probability theory. The agent keeps a distribution over possible states of the world, and updates that distribution as it moves about and observes the world.

The Bayesian foundation for localisation is extremely general, and hence leaves many choices in implementation. What is the space over which the state is assumed to vary? How is the probability distribution over that space represented? Which observations and actions are used to update the distribution.

2. Bayesian Localisation

As an initial example of Bayesian localization we'll use a small robot in a world made up of a 5 by 5 grid of states. Initially we'll assume that the robot is completely uncertain about its

Source: Robotic Soccer, Book edited by: Pedro Lima, ISBN 978-3-902613-21-9,
pp. 598, December 2007, Itech Education and Publishing, Vienna, Austria

location, and so all states in the world are given equal probability. As the agent moves, we shift the probability distribution to account for the movement, and blur the distribution slightly as the movement is noisy. The exact amount of movement and blur is recorded by a motion model.

When the agent makes an observation, for example it notes that a wall appears two units north, that observation is processed through a sensor model. This sensor model records the probability of seeing each possible observation in each possible state $P(O|S)$. This model is discovered prior to attempting to localise by conducting experiments on the sensors.

With our prior distribution over states, and our sensor model, we can calculate a posterior distribution over states using Bayes' rule. For each state:

$$P(S|O) = \frac{P(S)P(O|S)}{P(O)} \tag{1}$$

We do not usually know the probability of a given observation, $P(O)$, but luckily that is a constant. As we know that the resulting probability distribution must be normalised, we can ignore $P(O)$ and simply renormalise the resulting distribution.

2. State of the Art

Representing probability distributions as tables of probabilities and doing these calculations individually for each state can be very slow. Luckily, there are common representations for probability distributions which allow efficient updates.

The currently preferred methods of localisation in the 4-legged Robocup league include Monte Carlo Particle Filters, and uni-modal Extended Kalman Filters. By far the most popular of these is the particle filter method. Particle filters approximate a probability distribution with a sample of points drawn from that distribution. Updates are then only required for the sampled points. Some reasons for the popularity of particle filters (Fox et al., 1999) include their ability to handle non-linear observations (where the mapping from observation space to state space is a non-linear function), their quick convergence, and their multi-modal nature (being able to track many different possible positions at the same time, termed Global Localization). The other popular method of robot localisation is the Kalman Filter (Kalman, 1960). This is a state estimation filter which uses a Gaussian probability distribution to approximate the current state of the world. The main advantage of this method is that it is very computationally efficient, being able to compute the updates algebraically in closed form.

In this chapter we describe a system for robot localisation which is a hybrid of the Extended Kalman Filter and the Monte-Carlo particle filter. Our representation for our probability distributions is a weighted sum of Gaussians – similar to having a small set of particles, where each particle is itself a Gaussian. We apply observation and motion updates to each Gaussian particle in the same way as for a standard Kalman Filter. Each Gaussian particle is then weighted according to how well the observation matched the hypothesis. This method combines the advantages of both the particle filter method and the Kalman Filter method. We are able to approximate arbitrary probability distributions (given enough Gaussian

particles), handle high dimensional state spaces, ambiguous observations, and at the same time keep computational costs reasonable.

3. Issues in the Localization Domain

The task of robot localization can be seen as calculating the belief distribution of the robot position with given observations and control updates. Measurement updates are observations of landmarks, in our case distance and heading measurements to the coloured beacons. Control updates are a prediction of the belief state after the robot performs some movement.

The task of robot localisation can vary in difficulty depending on various factors. Each of these factors will influence the choice of algorithm used for the particular situation, since some algorithms are adept at handling certain classes of problems but not others.

We can categorise localisation problems based on the type of measurements available and knowledge of the initial state of the system. Based on these we can classify the problem into either Local or Global localisation. In the case of Local localisation, the probability distribution function has only a single mode, meaning that the localisation algorithm is only required to deal with a small pose error, with the uncertainty confined to a small region around the robot's true pose. In the case of Global localisation, the probability distribution function needs to be able to handle multiple modes, and the algorithm used must be able to handle high uncertainty in the robot pose. This may be due to large errors in robot motion and measurements, or due to the presence of non-unique landmarks. For example, in the situation where there are two identical rooms connected by a corridor, the localisation algorithm must be able handle a probability distribution function which has a mode in each room.

The "kidnapped robot" problem is related to the problem of Global localisation. This arises if at some point in time the robot is taken from its current location and placed in a completely different location. In the case of the Robocup domain, this is especially prevalent, since robots are frequently taken off the field or placed back on the field in a new location. It is very important that the chosen algorithm can deal with "kidnapping" quickly and efficiently. This problem is magnified even further if the place where the robot is replaced has almost symmetrical landmarks when compared to its belief location.

Whether the environment is static or dynamic will also affect the choice of suitable algorithm. In the case of a static environment, the objects which are of concern for the robot are stationary during the operating cycle. A dynamic environment, however, may have moving objects which can affect the robot, or which the robot must interact with. In this case the algorithm must track not only the robot pose, but also the position of the moving objects. When applied to the Robocup domain, we can clearly see that it is a dynamic environment. Specifically, the orange ball is a moving object which we must track. The team-mate robot and opponent robots may also be considered as dynamic objects, but often, as in our case, they are ignored because they are very hard to observe.

The final distinguishing feature of robot localisation that we will consider is the issue of single-robot and multi-robot localisation. The simple case is single-robot localisation, in which case observations are all made by the one robot, only one pose needs to be tracked, and there is no need for communication between robots. Multi-robot localisation offers the advantage of the availability of a greater number of sensors and thus a greater number of

observations. We can use this increased observational power to improve the accuracy of the system as a whole. However, multi-robot localisation also brings with it many challenges. The localisation algorithm used must be able to incorporate the observations of other robots in the system, which in itself brings about issues such as communication lag. In our case, we can use the observations of other robots on the team to better track the position and velocity of the ball, and can also set up correlations in order to be able to use the ball as a beacon, thus improving not only the accuracy of the ball location, but also of the robots pose itself.

4. Kalman-Bucy Filter Algorithm

At its core the Kalman-Bucy Filter is a recursive solution to the discrete-data linear filtering problem. It allows us to estimate the state of a process minimising the squared error. Surprisingly, as already noted, this turns out to be equivalent to a Baysian tracking system when both prior and observation probability distributions are Gaussian.
In the case of Robot localisation, the process is the movement of the robot around the field. The process to be estimated is governed by the stochastic difference equation:

$$x_k = \mathrm{A}x_{k-1} + \mathrm{B}u_{k-1} + w_{k-1} \tag{2}$$

And measurement update:

$$z_k = Hx_k + v_k \tag{3}$$

In the equations above, A is an nxn matrix which relates the process state at the previous time step to the current state in the absence of control input. The nxm matrix B relates the control input u to the process state, and w is the process noise, which is assumed to behave as a Gaussian Distribution.

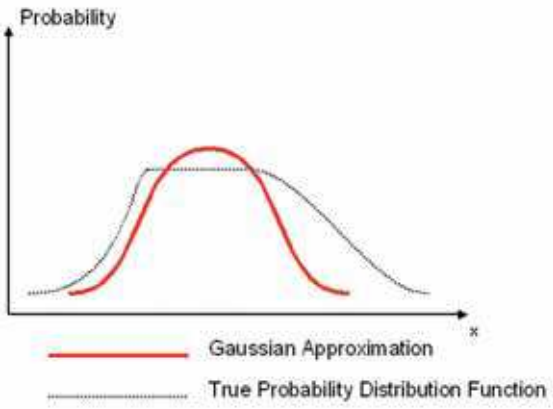

Fig. 1. We approximate a probability distribution function with a uni-modal Gaussian distribution

The Discrete Kalman Filter consists of two distinct steps, the time update and the measurement update. The time update uses the control input to update the belief state of the system, and the measurement update uses possibly noisy observations of the system state to improve the state belief estimate. In terms of the Robocup domain, the time update is derived from the odometry data from the actuators, and the measurement update is derived from the visual sighting of various landmarks around the field such as beacons. The variables which we must keep track of between time steps are the mean vector, and the covariance matrix. The mean vector is the best estimate of the world state, and the covariance matrix is the multi-dimensional measure of the uncertainty of the current estimate.

The time update equations:

$$\begin{aligned} x_k &= Ax_{k-1} + Bu_{k-1} \\ P_k &= AP_{k-1}A^T + Q \end{aligned} \tag{4}$$

The Measurement update equations:

$$\begin{aligned} K_k &= P_k^- H^T (HP_k^- H^T + R)^{-1} \\ x_k &= x_k^- + K_k(z_k - Hx_k^-) \\ P_k &= (I - K_k H)P_k^- \end{aligned} \tag{5}$$

In the time update step we must do two things, we must update the mean state estimate based on the control input, and we also need to update the covariance matrix P, that is, we need to increase our uncertainty estimate, due to the noise present in the control input.
The measurement update step is a more complicated process. Firstly we compute the Kalman Gain K, this is a measure of how much influence the observation z_k will have on the mean state estimate. For example, if we are very certain of our current estimate and we judge that the observation z_k is very unreliable, then the Kalman Gain K will be close to 0. However, if our uncertainty estimate is very high, that is, we consider the current mean to be very unreliable, and at the same time we consider the measurement to be very accurate, then the Kalman Gain will be close to 1.
After we have computed the Kalman Gain, we can adjust the mean state estimate x_k by moving it in the direction of the Innovation Vector $(z_k - Hx_k)$. The Innovation Vector can be seen as the direction in state space in which the mean vector needs to be shifted in order for it to more closely agree with the current state observation z_k. The more the currently observed state disagreed with the mean estimate, the greater will be the magnitude of the Innovation Vector, while if the mean estimate is in complete agreement with the observation then the Innovation Vector will be 0.
Finally, we must recompute the covariance matrix P, the uncertainty estimate. In general, observations tend to decrease the uncertainty estimate, while control updates tend to increase the uncertainty estimate. The derivation of the update of the covariance matrix is beyond the scope of this report. See the seminal paper by Rudolf E. Kalman (Kalman, 1960).

We can apply the Kalman filter to the problem of Robot Localization as follows. The time update step is triggered by the walking module when the robot makes a step. We can use the noisy odometry data to update the mean robot pose of the form (x, y, θ). Already we have violated the strict definition of a Kalman filter in that the motion of the robot is not a strictly linear change in state. The direction of the robot relates to the position of the robot through various non-linear trigonometric functions as shown in Fig. 2.
The measurement update is triggered when the vision system detects an object. Objects that can be detected include the four unique landmarks, or beacons, placed around the field, the ball, and the two goals. The vision system returns distance and angle estimates from the robot to the object detected. We could use the noisy distance and heading to the landmark to form an observed state estimate and then update the mean pose estimate. However, at this point the algorithm breaks down again. This is because the standard Kalman Filter assumes that the mapping between the state vector and any observation is linear, in the above case, represented by the matrix H. If we were able to observe directly, albeit with noise, the robot pose in (x, y, θ) form, then we would be fine. When observing a beacon, however, the information given implies that the robot pose can be anywhere on a helix in (x, y, θ) space. Knowing the distance to a landmark places you on a circle of a given radius around that landmark, and knowing the heading to it gives you a certain heading at every point on that circle, but they do not provide a single point. The helix is non-linear and cannot be represented by a Gaussian. In order to deal with this issue, we need to move to the Extended Kalman Filter, which can handle non-linear mappings between observations and state.

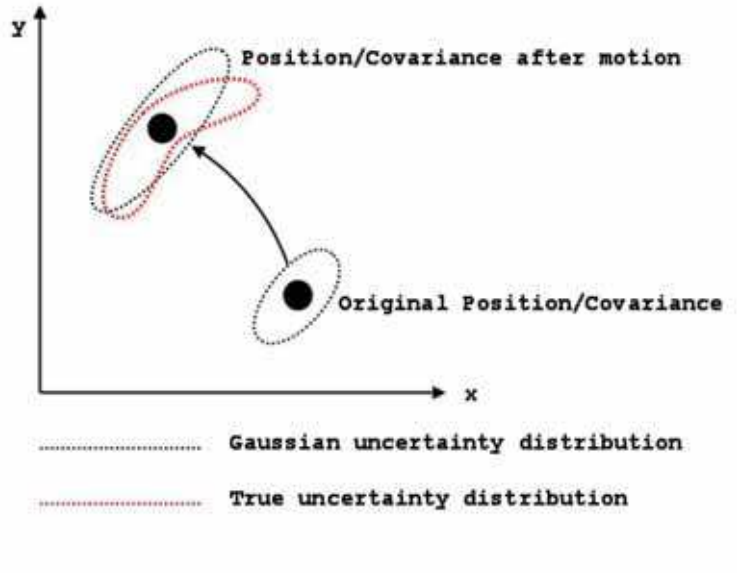

Fig. 2. Gaussian approximation to a non-linear motion update

5. Extended Kalman Filter

A Kalman Filter uses linear Gaussians over the state space to estimate the probability distribution function. However, as noted in the previous section, if a measurement or motion update has a non-linear nature, the classic Kalman Filter algorithm cannot handle this kind of situation. A solution to this problem is to linearise the function from state space to observation space around certain point such that we would now have a linear Gaussian approximation. We use the tangent line (or hyper-plane) which passes through the point x as the linear approximation. When applied to the Extended Kalman Filter, we must compute the multi-dimensional derivative of the non-linear function - the Jacobian Matrix. Take the function F which maps an n-dimensional state space onto an m-dimensional observation space:

$$F = \begin{pmatrix} f_1(x_1,\ldots,x_m) \\ \vdots \\ f_n(x_1,\ldots,x_m) \end{pmatrix} \tag{6}$$

The corresponding Jacobian Matrix would be:

$$J = \begin{pmatrix} \frac{\delta f_1}{\delta x_1} & \cdots & \frac{\delta f_1}{\delta x_m} \\ \vdots & \ddots & \vdots \\ \frac{\delta f_n}{\delta x_1} & \cdots & \frac{\delta f_n}{\delta x_m} \end{pmatrix} \tag{7}$$

Each of the elements of the Jacobian Matrix is a partial derivative of the non-linear function F. We can now use this linearisation to be able to incorporate non-linear observations and time updates into our Kalman Filter. The time update equations become:

$$\begin{aligned} x_k &= f(x_{k-1}, u_k) \\ P_k &= AP_{k-1}A^T + Q \end{aligned} \tag{8}$$

In the above equation, f is the function which updates the mean state estimate position, it may be non-linear, and the matrix A is the Jacobian of this function. The measurement update equations become:

$$\begin{aligned} K_k &= P_k^- J^T (JP_k^- J^T + R)^{-1} \\ x_k &= x_k^- + K_k(z_k - Jx_k^-) \\ P_k &= (I - K_k J)P_k^- \end{aligned} \tag{9}$$

and we can now assume that the state to observation equation is no longer linear, becoming:

$$z_k = j(x_k) + u_k \tag{10}$$

If we consider beacon observations to be two dimensional observations, being heading and distance to the beacon, we can derive a Jacobian matrix which is a derivative of the mapping between robot pose state space and observation space. The mapping between state space and observation space is as follows:

$$\begin{aligned} \text{distance} &= \sqrt{(x_{robot} - x_{beacon})^2 + (y_{robot} - y_{beacon})^2} \\ \text{heading} &= \tan^{-1}\frac{(y_{beacon} - y_{robot})}{(x_{beacon} - x_{robot})} - \theta_{robot} \end{aligned} \tag{11}$$

If we now compute the first derivative of this function from state space to observation space, we get the following Jacobian matrix:

$$J = \begin{pmatrix} -\dfrac{k*(y_{rbt} - y_{bcn})}{(x_{rbt} - x_{bcn})^2 + (y_{rbt} - y_{bcn})^2} & \dfrac{k*(x_{rbt} - x_{bcn})}{(x_{rbt} - x_{bcn})^2 + (y_{rbt} - y_{bcn})^2} & -1.0 \\ \dfrac{(x_{rbt} - x_{bcn})}{\sqrt{(x_{rbt} - x_{bcn})^2 + (y_{rbt} - y_{bcn})^2}} & \dfrac{(y_{rbt} - y_{bcn})}{\sqrt{(x_{rbt} - x_{bcn})^2 + (y_{rbt} - y_{bcn})^2}} & 0.0 \end{pmatrix} \tag{12}$$

A more thorough derivation and explanation of the Extended Kalman Filter can be found in other papers (Thrun et al., 2005) (Zarchan et al., 2005).

6. Multi-Modal Localization

The Extended Kalman-Bucy Filter is a powerful algorithm for robot localisation. However, one of its major downfalls is the fact that it can only approximate a uni-modal probability distribution function. So, for example, if the robot knew it was in one of two positions, say, it knew it was next to one of the two goals, the probability distribution function for the pose of the robot would have two local maxima, each centred near one of the goals. The Extended Kalman Filter, which uses a single Gaussian to represent the pose and uncertainty, would be unable to model this situation sufficiently well. This hypothetical situation would be much better modelled by the sum of two separate Gaussians, each of which is centred on one of the modes. In fact, it is possible to approximate any probability distribution function arbitrarily accurately using a weighted sum of an arbitrary number of Gaussians. To incorporate multiple modes into the localisation algorithm we use an array of uni-modal Gaussians, each with an associated weight. We can then view the full probability distribution over the world state as a weighted sum of Gaussians.

$$P(\underline{x}) = \sum_{i=0}^{N} \omega_i G_i(\underline{x}) \tag{13}$$

This weight ranges between 1.0 and 0.0 and is a measure of the probability that that particular Gaussian represents the state of the system. A Bayesian interpretation allows us to update the weights when an observation is made. In practice, the Gaussians which match the observation well have a higher weight than the Gaussians which disagree with the observation. Given an observation covariance R, Jacobian J, and the covariance C of the Gaussian prior, we can calculate the combined covariance E. Combining this with the innovation vector v we calculate the weight scalar S, which allows us to update the weight of the Gaussian distribution.

$$\begin{aligned} E &= R + J^T C J \\ S &= \exp(-\frac{1}{2} v^T E^{-1} v) \\ \omega_i &= S\omega_{i-1} \end{aligned} \tag{14}$$

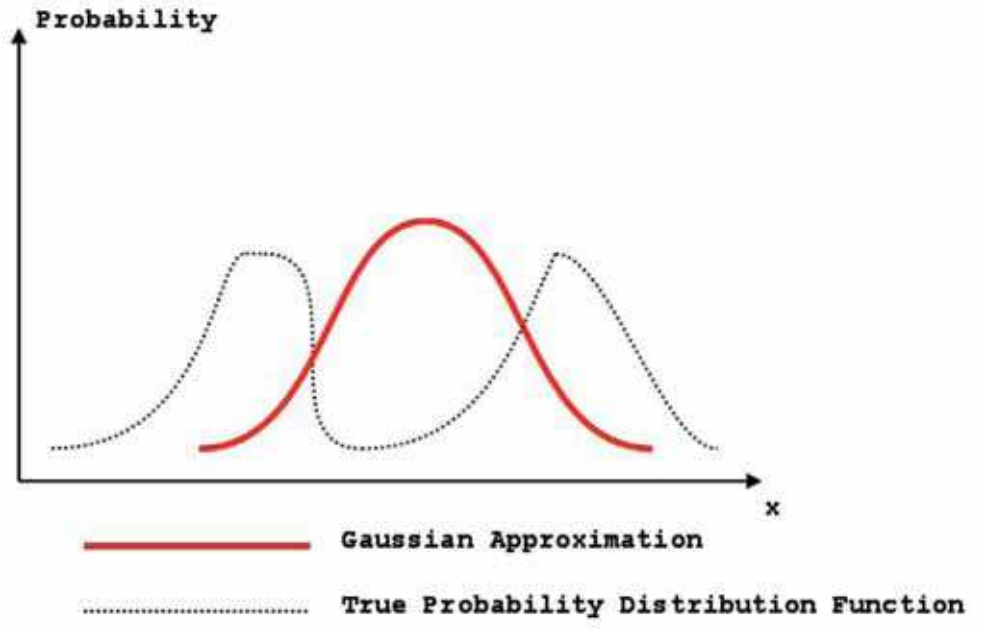

Fig. 3. A single Gaussian approximates a multi-modal probability distribution poorly

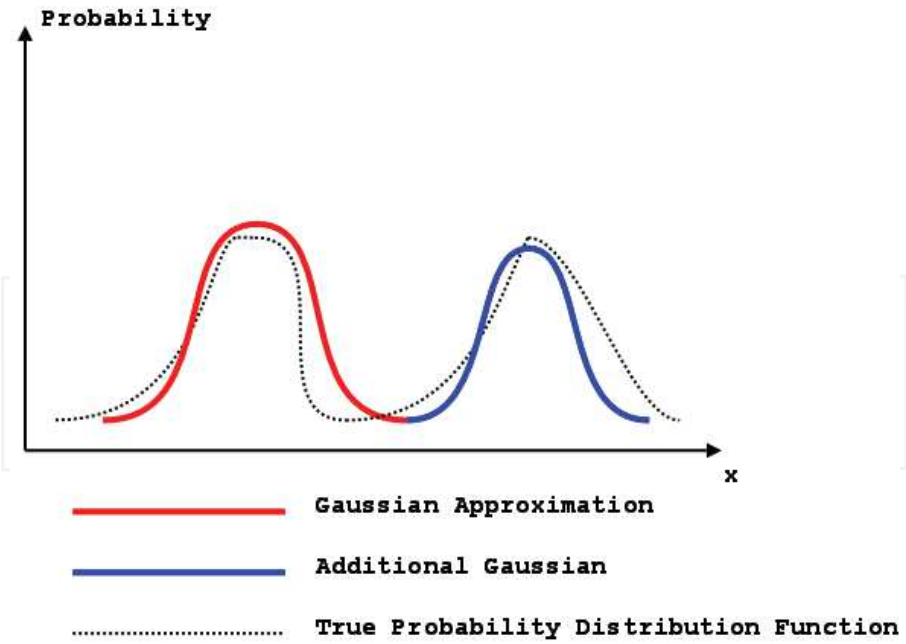

Fig. 4. A weighted sum of Gaussians provides a far closer approximation to the true distribution

The sum of the weights of all of the Gaussians in the distribution must sum to 1.0 in order to make the entire distribution a valid probability distribution (one which integrates to 1.0 from -1 to 1). In order to maintain this property a renormalisation must happen every time the weight of a Gaussian is modified. In addition to this, Gaussians which have a very low weight are removed from the distribution array. This is because they represent extremely unlikely modes, and for performance reasons we cannot keep track of unlikely modes. Another method used to reduce the number of Gaussians that form the distribution is merging similar Gaussians. If two Gaussians have a similar mean and similar covariance matrices, then one of them is removed and the other becomes the average of the two. To calculate the global maxima of the weighted sum of Gaussians distribution a simple approach is taken, the maxima is assumed to be the mean of the Gaussian of highest weight. This is not strictly correct, but is a good enough approximation, seeing as a correct solution for finding the global maximum of a sum of Gaussians is a lot more involved and does not provide enough of a benefit for it to be used in our system.

7. The State

An important part of building a tracking system is deciding which state to track. It is possible to track the pose of a single robot as a three dimensional system (x, y, θ). This is very effective, but it is possible to do better.

The current rUNSWift system tracks a 16 dimensional state, rather than a 3 dimensional one.

$$MeanVector = \begin{pmatrix} robotXpos \\ robotYpos \\ robot\theta \\ ballXpos \\ ballYpos \\ ball\delta a \\ ball\delta a \\ teammate_1Xpos \\ teammate_1Ypos \\ teammate_1\theta \\ \vdots \\ teammate_3Xpos \\ teammate_3Ypos \\ teammate_3\theta \end{pmatrix} \quad (15)$$

This state incorporates tracking the ball as well as tracking all four robots on the team. This enables information from all four robots to be combined when determining the state. The mechanism for distributing this information is discussed below.

In addition to this, we use more complex motion update than the simple "shift-and-blur" model mentioned above. A Kalman Filter can handle any linear transform on the state as a motion update. If the robot is standing still, this update would normally be an identity matrix for a single robot – the robot stays where it is. With the addition of a ball that moves when the robot is not moving, a non-identity transition matrix is required: the velocity of the ball is added to the location of the ball at each time step. This simple change induces a correlation between the location of the ball and its velocity in the state distribution. When we see the ball a second time, the new information about its position will also give information about its velocity.

$$MotionMatrix = \begin{pmatrix} I_{3,3} & & & & \\ & 1.0 & 0.0 & 1.0 & 0.0 \\ & 0.0 & 1.0 & 0.0 & 1.0 \\ & 0.0 & 0.0 & friction & 0.0 \\ & 0.0 & 0.0 & 0.0 & friction \\ & & & & I_{9,9} \end{pmatrix} \quad (16)$$

8. Noise Filtering

Using the power of a multi-modal filter we can improve the robustness of the system to spurious observations. In the 4-legged league it is very likely that the vision system will make false classifications of objects which are part of the background, such as spectators wearing coloured t-shirts. These may be classified as beacons, or goals, or the ball. This is a different type of noise in the system to what a standard filter is designed to handle, that is, noise that is centred on the mean. We need a way to be able to reject spurious observations, otherwise they will significantly reduce the accuracy of the localisation system, more so than standard "noisy" observations. Take as an example a seeing a spectator wearing an orange t-shirt in the crowd, which the vision system classifies as a very large (and thus close) ball. The variance of this observation will be small because the closer the ball is the more certain we are about its observed distance. This results in our estimate of the ball position shifting significantly towards the observed "phantom" ball position, despite the observation being false.

Our solution to this problem is to allow that the observation may or may not be correct, and as such when we apply the observation to every Gaussian in the weighted sum distribution we also make a copy of the Gaussians which do not have the observation applied, but we scale the associated weights down by a constant factor. This constant factor can be seen as the probability of a false observation. This means that for every observation, the system doubles the number of Gaussians which make up the distribution. These are later culled if there are too many or their weights are too small. An observation is said to be a phantom observation if the weight of the Gaussian with it applied is lower than the weight of the Gaussian without the observation applied.

This technique works because the more an observation disagrees with the current state, the lower the weight will be of the resulting Gaussian after applying the observation. So if an observation is made which is extremely unlikely, and so is probably a false one, the resulting Gaussian will have a lower weight than the Gaussian without the observation applied.

The effectiveness of this approach was demonstrated to us when we accidentally swapped two of the beacons around and asked the robot to position itself at a kick-off position. This results in 4 valid landmarks (2 goals and 2 beacons), and 2 invalid landmarks (the 2 swapped beacons). Despite expecting the robot to localise poorly due to the contradictory observations, the robot localised extremely well, rejecting almost all observations which were of the switched beacons. This robustness to false observations was extremely important at the competition due to the fact that there were spectators close to the field at "eye level" who were wearing coloured shirts. Without this noise filtering in the localisation system, the performance of the system would have been far lower. It is important to note that unlike some previous work (Browning et al., 2002), out system does not reject "bad" observations, but rather tracks all possibilities as different Gaussians, rejecting them only when they become extremely unlikely.

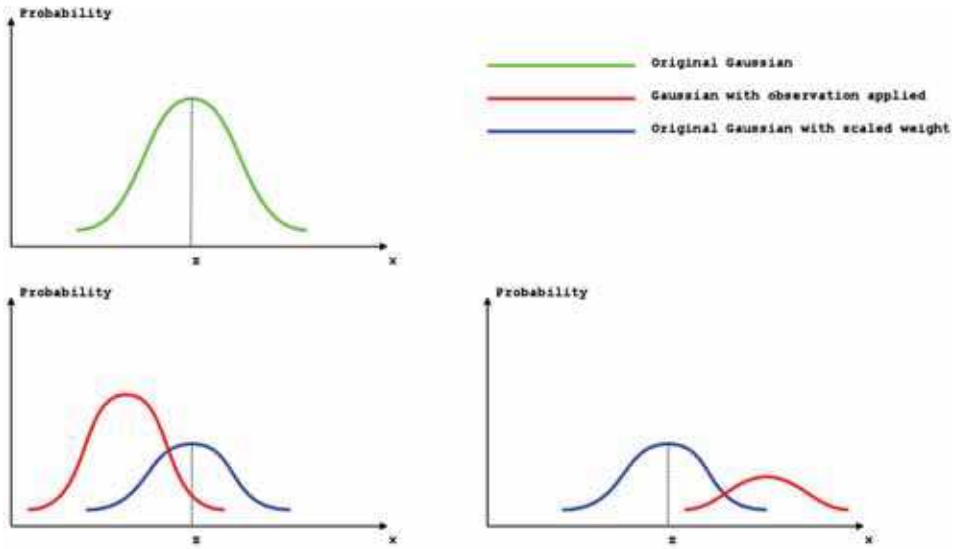

Fig. 5. shows two possible scenarios. One is where the original Gaussian (top left) is split into two, the resulting distribution is such that the Gaussian with an observation applied has a higher weight than the scaled down Gaussian with no observation applied (bottom left). The other is that the observation is spurious and thus the Gaussian with the observation applied has a lower weight than the scaled down Gaussian (bottom right)

9. Multiple Linearisation Points

One of the sources of error in an Extended Kalman Filter comes from the fact that we are approximating a possibly non-linear probability distribution with a linear Gaussian function. This is one of the advantages of a particle filter approach to localisation, particle filters do not have to linearise a non-linear distribution, since they can approximate any probability distribution function. However, using a sum of multiple weighted Gaussians to represent our function, we can reduce the error from the linearisation process by better approximating non-linear function. In effect we have a simple, and efficient, unscented Kalman Filter or Rao-Blackwellised particle filter. Figure 6 is a representation of the linearisation process for the standard Extended Kalman Filter approach. The true probability distribution function is not a linear Gaussian one, so in order to approximate it, a linear Gaussian distribution is used which is tangential to the true probability distribution at the current mean point.

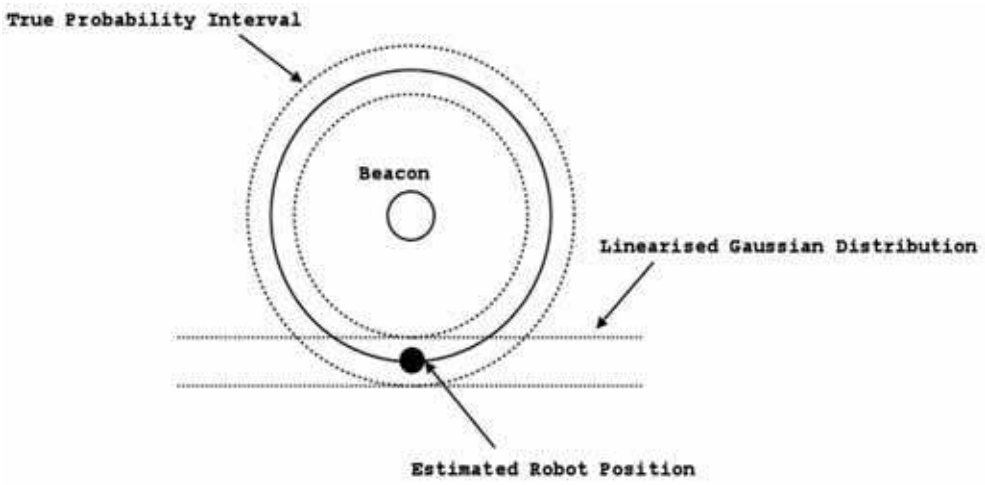

Fig. 6. The standard Extended Kalman Filter linearises around only one point, the estimated mean of the world state

The way in which we do this is every time a single beacon (note that for the purposes on this section, the ball should be considered as being a beacon) is observed, a copy is made of every Gaussian in the distribution, with the mean of each copy being offset such that it is the same distance from the observed beacon, but is rotated around by a given angle. The weights of these new displaced Gaussians are also scaled down. It is important to note that the displaced Gaussians can only be generated when only one landmark is observed. This is because it is not consistent with the observations to rotate a Gaussian around a beacon if there are multiple observed beacons. Every time there is a single beacon observation, only one displaced copy is made per existing Gaussian, whereas we would need 2 to maintain symmetry. This is done with the aim of improving the speed of the localisation module, spawning two additional Gaussians would have been too expensive, so instead we alternate whether to rotate the added displaced Gaussian clockwise or anti-clockwise around the beacon. In our implementation we chose to rotate the displaced Gaussian by 16 degrees, and the weights are multiplied by 0.1.

The end result of this is a better approximation of the probability distribution function through a reduction in the inherent error introduced by the linearisation process. Figure 7 shows how the two additional linearisation points are placed in relation to the mean, and how the multiple linear Gaussians are a closer approximation to the true probability distribution function.

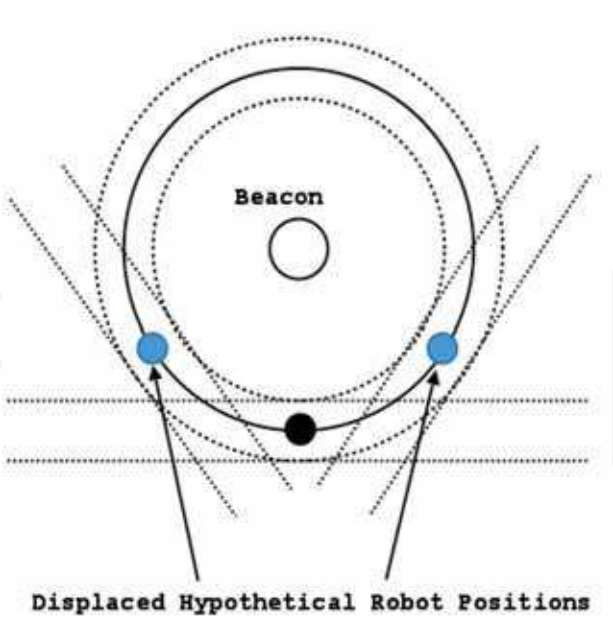

Fig. 7. Our system adds low weight Gaussians rotated around the beacon, meaning we linearise around multiple points

10. Incorporating Teammate Observations

The previous sections have all described ways in which the increased representational power of a sum of Gaussians representation allows us to more accurately model a probability distribution, and hence more accurately localise. In this section we discuss the distribution of information between robots on a team rather than the representation of information on a single robot.

This incorporation of distributed observations of the world state into Kalman Filter localisation is one of the most important improvements of our system over a standard extended Kalman Filter. Our technique involves each robot keeping track of a separate traditional Kalman Filter, which we update as per normal, but does not form part of the main weighted sum of Gaussians probability distribution function. We refer to the data stored in this Kalman Filter as the "shared Gaussian". Periodically, this is sent to all team-mates. After being sent, the shared Gaussian is reset to a 'uniform' state, with high variance and a mean equal to our best estimation of our pose. This avoids any observations being incorporated into a team-mate's state estimate multiple times. In addition to sending the Shared Gaussian mean vector and covariance matrix, we also send the cumulative odometry information, which we also reset every time a wireless packet is broadcast.

When a wireless packet is received, we can incorporate the team-mate observation into the main probability distribution as a direct observation. There is no need to linearise the function mapping observation to state space, since it is already a linear one - all observations have already been linearised by the sending robot. The data sent is of the form of a 7

dimensional mean vector and a covariance matrix. This reduced form is used to save communications bandwidth. The receiving robot must have a matrix which maps the team-mate pose and ball position/velocity estimates into its own world estimate.

$$A = \begin{pmatrix} 0_{7,3} \\ 0 \\ I_{3,3} \\ 0 \end{pmatrix} \tag{17}$$

$$B = \begin{pmatrix} 0_{3,4} \\ I_{4,4} \\ 0_{9,4} \end{pmatrix}$$

$$H = (A \mid B)$$

The matrix H is the mapping between the wireless team-mate observation and our world state estimate. The matrix B is constructed such that the placement of the 3x3 identity matrix corresponds to the index of the robot id from which the packet was received, such that our idea of where that team-mate robot is positioned is updated accordingly inside the mean vector.

The inclusion of team-mate observations greatly increases the accuracy of ball position and velocity tracking. With this change it is possible for a robot to grab a ball under its chin with its own local distance observations turned off, and relying on team-mate robots on the field to obtain the distance to the ball.

In addition, the accuracy of the robot pose itself is greatly improved by this communication. Our scheme allows for the ball to act as a moving landmark from which the robots can localise. For example, if a very well localised robot is looking at the ball, it can transmit a very accurate and certain position of the ball to its team-mates. Following this, a poorly localised team-mate robot can look at the ball, and knowing the ball's position, gain information about its own location.

Before introducing this information sharing, our robots required an active localisation behaviour. If a robot is chasing the ball for a prolonged period of time it will have seen few, if any, landmarks and would previously become severely mis-localised. The active localisation behaviour involves a robot looking away from the ball to glance at a landmark to re-localize itself. The downside of active localisation is that while the robot is looking away from the ball, there is a chance the ball could be moved by an opponent and our robot will lose track of it. With information sharing in place, the ball itself helps the robot localise, allowing it to focus more on the ball and less on looking around for beacons.

11. Results

We evaluated the effectiveness of our system by setting up various configurations of landmarks, balls and team-mates, and then by running a robot between a set of waypoints

in order. The robot would stop when it considered itself to be close to a waypoint, and then the distance in centimetres between the robots position and the true waypoint position was recorded. We performed this test with several different field layouts and between several different version of the localisation module. The field layouts for the various tests are shown in Figure 8. The results are presented in tabulated form, listing the average measured distance between the robot position and the waypoint position for every waypoint. The total average distance error is also recorded.

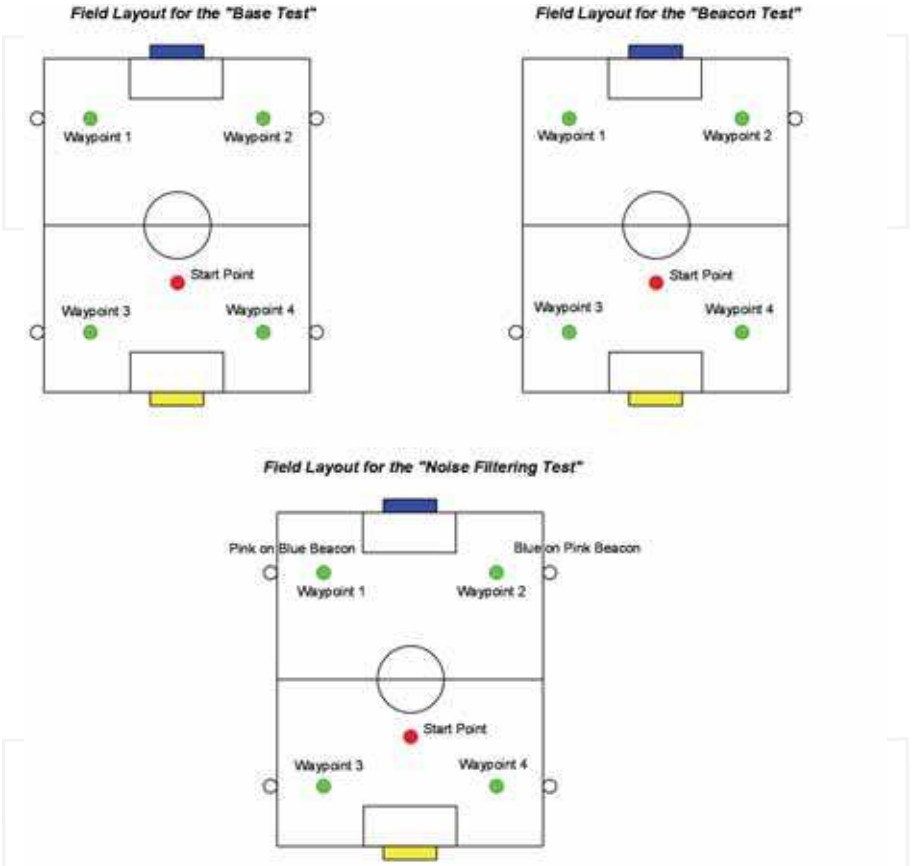

Fig. 8. This figure shows the respective field layouts for the tests that were run in order to evaluate the performance of the system

The first test which we ran is a base test. This involves 4 waypoints, each at a quadrant of the field, and all of the beacons and goals. The robot starts near the centre circle and runs between the waypoints in increasing numbered order. We use this test to compare the accuracy of the previous localisation system, and the new localisation system which has all of the enhancements mentioned in this report.

Full 2005 Localisation System		
	Mean Error (cm)	Standard Deviation
Waypoint 1	32.0	8.86
Waypoint 2	18.8	5.15
Waypoint 3	31.4	23.6
Waypoint 4	62.8	64.74
Average	28.0	25.5

Full 2006 Localisation System		
	Mean Error (cm)	Standard Deviation
Waypoint 1	37.2	9.32
Waypoint 2	14.0	5.3
Waypoint 3	19.0	11.2
Waypoint 4	17.0	7.7
Average	21.8	8.3

11.1 Beacon Test

This test is very similar to the Base Test, except for the removal of 2 beacons. This change makes it much harder for the robot to localise as it receives far fewer observation updates. This shows a larger disparity between the old and new localisation systems.

Multiple Linearisations Disabled		
	Mean Error (cm)	Standard Deviation
Waypoint 1	58	30.5
Waypoint 2	34.2	12.1
Waypoint 3	20.8	10.3
Waypoint 4	28.4	8
Average	35.35	15.22

Multiple Linearisations Enabled		
	Mean Error (cm)	Standard Deviation
Waypoint 1	27.0	12.4
Waypoint 2	13.0	7.0
Waypoint 3	37.0	5.1
Waypoint 4	15.6	2.3
Average	23.15	6.7

11.2 Noise Filtering Test

Our system is multi-modal in nature, allowing it to consider observations as possible false and possibly true. This results in spurious observations having a far lower effect on the accuracy of the system. If the robot observes a landmark which does not make sense for the current estimated world state then it is possible for the system to deal with this by using the Gaussian without the observation applied as the mean Gaussian instead. The setup for this

test is similar to that of the Base Test, except that we swap around the Yellow on Pink and Pink on Blue beacons. This result in any observation of either of these beacons as being "false", and hopefully the localisation system will cull them.

Noise Filtering Disabled		
	Mean Error (cm)	Standard Deviation
Waypoint 1	137.0	31.0
Waypoint 2	191.0	153.0
Waypoint 3	189.0	8.2
Waypoint 4	123	13.5
Average	160.0	51.42

Noise Filtering Enabled		
	Mean Error (cm)	Standard Deviation
Waypoint 1	31.0	8.6
Waypoint 2	27.3	9.8
Waypoint 3	34.0	8.6
Waypoint 4	12.6	1.0
Average	26.22	7.0

We used a one sided Mann-Whitney U test to test the significance of these results. The Beacon and noise filter tests were significant (the beacon test at $p = 0.05$ and the noise filter test at $p = 0.01$). The base test is not significant with this small sample, but the changes do not degrade performance.

In the end, the best test is the performance of the entire system. Our experience is that each of the changes presented in this chapter lead to small, but important, improvements in the level of play of the team as a whole.

Our current experiments demonstrate a small but not statistically significant improvement in accuracy due to the ball tracking. Our more subjective tests with the whole team in a game suggest that this is important for complete game behaviour.

12. Conclusion

The above results show that the documented improvements have had a great effect on the overall accuracy of the localisation system. The effectiveness of the Noise Filtering part of the system is staggering. It should be also noted that these experiments only measured the (x, y) pose error, whereas the localisation system tracks much more data than those 2 dimensions, including the ball position and velocity. All of the mentioned improvements to the localisation system allowed us to great flexibility and power in terms of higher level behaviours. This is shown by the fact that our team came 2nd in the World Open in 2006, and in the same year were the Australian Champions.

National ICT Australia (NICTA) is funded by the Australian Government's Department of Communications, Information Technology, and the Arts (DICTA) and the Australian Research Council through Backing Australia's Ability and the ICT Research Center of Excellence programs.

13. References

Browning, B; Bowling, M & Veloso, M. (2002). Improbability Filtering for Rejecting False Positives. *Robotics and Automation,* Vol. 3, (November 2002) 3038-3043, 0-780-37272-7

Fox, D; Burgard, W & Dellaert, F. (1999). Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. *Proceedings of AAAI,* pp. 343-349, 0-262-51106-1, Orlando Florida, July 1999, MIT Press, Cambridge

Kalman, R. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME – Journal of Basic Engineering,* Vol. 82, (1960) 35-45

Thrun, S; Burgard, W & Fox, D. (2005). *Probabilistic Robotics*, MIT Press, 0-262-20162-3, Cambridge

Zarchan, P & Musoff, H. (2005). *Fundamentals of Kalman Filtering: A Practical Approach SE,* AIAA, 1-56347-694-0, Reston, VA.

Robotic Soccer

Edited by Pedro Lima

ISBN 978-3-902613-21-9

Hard cover, 598 pages

Publisher I-Tech Education and Publishing

Published online 01, December, 2007

Published in print edition December, 2007

Many papers in the book concern advanced research on (multi-)robot subsystems, naturally motivated by the challenges posed by robot soccer, but certainly applicable to other domains: reasoning, multi-criteria decision-making, behavior and team coordination, cooperative perception, localization, mobility systems (namely omni-directional wheeled motion, as well as quadruped and biped locomotion, all strongly developed within RoboCup), and even a couple of papers on a topic apparently solved before Soccer Robotics - color segmentation - but for which several new algorithms were introduced since the mid-nineties by researchers on the field, to solve dynamic illumination and fast color segmentation problems, among others. This book is certainly a small sample of the research activity on Soccer Robotics going on around the globe as you read it, but it surely covers a good deal of what has been done in the field recently, and as such it works as a valuable source for researchers interested in the involved subjects, whether they are currently "soccer roboticists" or not.

InTech Europe

University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China

Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821