Expectation-Based Vision for Precise Self-Localization on a Mobile Robot

Daniel Stronger and Peter Stone
Department of Computer Sciences, The University of Texas at Austin
{stronger,pstone}@cs.utexas.edu
http://www.cs.utexas.edu/~{stronger,pstone}

Abstract

This paper presents and empirically compares two solutions to the problem of vision and self-localization on a mobile robot. In the commonly used *particle filtering* approach, the robot identifies regions in each image that correspond to landmarks in the environment. These landmarks are then used to update a probability distribution over the robot's possible poses. In the *expectation-based* approach, an expected view of the world is first constructed based on a prior camera pose estimate. This view is compared to the actual camera image to determine a corrected pose. This paper compares the accuracies of the two approaches on a test-bed domain, finding that the expectation-based approach yields a significantly higher overall localization accuracy than a state-of-the-art implementation of the particle filtering approach. This paper's contributions are an exposition of two competing approaches to vision and localization on a mobile robot, an empirical comparison of the two methods, and a discussion of the relative advantages of each method.

Introduction

This paper presents and empirically compares two solutions to the problem of vision and self-localization on a mobile robot in a known, fixed environment. Specifically, given a series of visual images produced by a camera on-board the robot, how can the robot effectively use those images to determine its position and orientation (pose) over time? This paper discusses two contrasting approaches to this problem, which we denote the *particle filtering* approach and the *expectation-based* approach. Although the particle filtering approach has been commonly used on mobile robots, we demonstrate that the expectation-based approach yields higher localization accuracy on a representative task.

The particle filtering approach, also known as Monte-Carlo localization, represents a probability distribution over the robot's pose with a set of sample poses, or particles (Dellaert *et al.* 1999). This distribution is continually updated by a series of *motion updates*, which take the robot's motion into account, and *observation updates*, which update the pose and relative importance of each particle depending on the robot's observations. Much work has been done in particle filtering on vision-based robots. This includes work in vision-based legged robots (Lenser & Veloso 2000; Kwok & Fox 2004; Rofer & Jungel 2003; Sridharan, Kuhlmann, & Stone 2005; Hoffman *et al.* 2006) and wheeled robots (Stachniss, Burgard, & Behnke 2006; Thrun *et al.* 2001). The particle filtering algorithm is described in the following section.

In vision-based robots, the particle filtering approach involves recognizing a number of discrete objects in the environment, or landmarks, each of which has a known location, shape, and appearance. The robot scans each incoming image to determine if any area of the image matches the appearance of any of the objects in the environment. When matches are found, they are used to determine the robot's relative angle and distance to the corresponding object. These landmark observations comprise the observation update used by particle filtering.

Instead of using discrete objects, the expectation-based approach starts with a line model of the environment. This model represents the three-dimensional locations and orientations of the edges in the environment. Additional information may include colors or textures appearing on either side of any given edge. Furthermore, in analyzing each image, the expectation-based approach makes use of its prior knowledge about the robot's pose in the environment. Specifically, it combines a running estimate of the three-dimensional pose of the camera with its line model of the environment to compute an *expected view* for each image frame. The expected view is constructed by projecting each edge in the world model onto the image plane.

Once the expected view is found, the camera image is analyzed by first applying an edge detector, such as (Canny 1986), and then collecting consecutive, collinear edge points into line segments (Forsyth & Ponce 2003). The resulting line segments are collected as potential matches to the edges in the expected view. For each pair of an observed edge and an expected edge, if they are close enough in position and orientation, they are identified with each other. These line segment identifications are then used to update the robot's camera pose estimate, based on the robot's model mapping line segment positions to a pose. This process is repeated until the camera pose stabilizes. Between each image frame and the next, the robot's pose is updated in accordance with its odometry estimate. This method is described in full detail in the Expectation-Based Approach section.

Although this paper considers the expectation-based approach from the point of view of the vision and localization problem, the methodology used is based on work originally developed in the context of a fixed camera tracking a moving object. In particular, the implementation used in the experiments reported in this paper is based in part on the algorithm presented by Lowe (Lowe 1991). In the context of mobile robot self-localization, the expectation-based approach has been previously employed on a wheeled robot by Kosaka and Pan (Kosaka & Pan 1995) and on a legged robot by Gasteratos et al. (Gasteratos *et al.* 2002).

Both approaches to the vision and localization problem are implemented on a Sony Aibo ERS-7 and evaluated in a test-bed domain. Experimental results demonstrate that the expectation-based method achieves a significantly higher average localization performance while the robot walks to a series of points.

The remainder of this paper is organized as follows. The following two sections present the two competing approaches to vision and localization on a mobile robot: the particle filtering approach and the expectation-based approach. The next section introduces the test-bed domain and presents the adaptations of the two methods to this domain. Finally, there are experimental results, a discussion of the relative advantages of each method, and a concluding section.

The Particle Filtering Approach

This section presents the particle filtering approach to the problem of vision and localization. Particle filtering localization addresses the question of how to represent the robot's uncertainty in its own pose. For instance, an alternative approach, Kalman filtering (Kalman 1960), represents the probability distribution explicitly, as a functional multivariate normal distribution. By contrast, particle filtering represents the distribution as a collection of sample poses, or particles. Each particle has a possible pose, (x, y, θ), and a relative probability, p. The collection of particles is maintained by a series of updates: the motion update, the observation update, resampling, and reseeding. The remainder of this section describes these processes.

In the motion update, each particle's pose is updated according to the robot's odometry estimate. Note that, for example, if the robot believes it has walked straight forward, this may move all of the particles in slightly different directions from each other due to the differences in the particles' orientations. The update is applied to each particle according to its own orientation. Furthermore, to model random noise in the motion model, a random perturbation is added to each particle's position and orientation.

An observation update happens for every landmark observation that is registered by vision. An observation consists of a landmark identity and its distance and relative angle to the robot. For each observation, every particle's probability p is updated according to an observation likelihood. This quantity is the likelihood that the given observation would occur, assuming the particle's pose. This likelihood is computed by predicting what the landmark's distance and angle would be if the particle's pose were correct. The difference between these predicted values and the observed values is converted to a likelihood in accordance with a Gaussian model of the observation noise. To temper the effects of false observations, the change in the particle's probability towards the computed likelihood is limited by a constant.

After every vision frame, the particles are resampled according to their probabilities. That is, multiple copies are made of the highest probability particles, and the lowest ones are deleted. The purpose of this resampling is to concentrate the processing performed by the localization algorithm on the most likely areas of the state space. The number of copies made of each particle is roughly proportional to its probability.

Additionally, a small number of reseeding particles are added, replacing low-probability particles, in accordance with completely new estimates of the robot's pose derived from the current observations (e.g., by triangulation). This process is based on the idea of Sensor Resetting (Lenser & Veloso 2000). Reseeding enables the robot to recover from large, unmodeled movements, also known as the kidnapped robot problem (Engelson & McDermott 1992). Without reseeding, there can easily be no particles in the area of the robot's new actual pose, in which case it is very difficult for the particles to recover.

One potential enhancement to reseeding, which is used in the experiments reported in this paper, is *landmark histories*. A landmark history is an account of all of the recent sightings of a given landmark. The histories are updated in accordance with the robot's odometry. Then, at any time, all of the recent observations of a given landmark can be averaged to yield a more accurate estimate of that landmark's relative distance and angle. These accumulated estimates enable the robot to construct reseeding estimates, even when there is not enough information in any one image to triangulate a position. Because of the accumulation of errors in the robot's odometry, observations are removed from the histories after enough time has passed or enough movement has occurred.

The particle filtering approach requires visual observations that provide information about the robot's location. As described below, the observations in our test domain are based on color-coded fixed landmarks. Another approach is to use scale-invariant features (Stachniss, Burgard, & Behnke 2006).

The Expectation-Based Approach

This section presents the expectation-based approach to the vision and localization problem. For each input image, the robot localizes by continually maintaining an accurate estimate of its camera's three-dimensional pose. This pose is represented by a six-dimensional state vector X, with three dimensions for the camera's position and three for its orientation, the pan, tilt, and roll. Note that in our test-bed domain, the robot's head is actuated with three degrees of freedom, so that the robot can control the direction in which the camera is pointing. However, the method described in this section is applicable regardless of the degree of the camera's mobility with respect to the robot's body. Since the robot walks on a two-dimensional surface, it simultaneously maintains its two-dimensional body pose, (x, y, θ), along with its camera pose X.

For each camera image that is captured, the robot starts with a prior estimate of its camera pose, X^0. This pose is used to generate an expected view, as discussed in the introduction. To construct an expected view, the robot goes through each of the edges in the three-dimensional line model.[1] Each of these model edges is projected onto the image plane. If the line segment is behind the image plane or no part of it falls within the image rectangle, it is ignored.

If a model line does project onto a line segment in the image rectangle, this segment is compared to each of the detected edge segments in the image. Four matching criteria are computed between the two line segments: length of overlap, effective distance, angle difference, and appearance match. The overlap length and effective distance are depicted in Figure 1. The angle difference is the angle between the two lines. The appearance match reflects whether or not model information corresponding to the line matches the features of the observed line (e.g., such as colors on each side of the line). If there is an appearance match, any positive length of overlap, and the angle difference and effective distance are both sufficiently low, the model line and image line are considered to be a match.

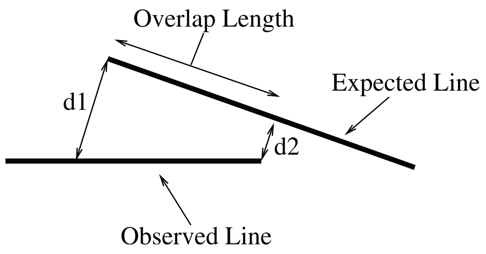

Figure 1: For each pair of an expected view line and a line detected in the image, the robot computes a length of overlap, effective distance, and angle difference. The effective distance is defined as the average of d_1 and d_2, which are measured perpendicular to the expected line.

All of the matches between a model line and an observed line are collected into a list of size n. The next step is to determine a new camera pose vector X, such that when the model lines are projected onto the image plane assuming a camera pose of X, their distances from the corresponding observed lines are as small as possible. That is, considering the observed lines to be fixed, each of the distances $d1$ and $d2$ depicted in Figure 1 can be thought of a function of X: $F_i(X^0) = d_i$, where i runs from 1 to $2n$ and d_{2k-1} and d_{2k} are the two distances for the kth line match out of n. If the fit were perfect, the following $2n$ equations would all be satisfied:

$$F_i(X) = 0, \qquad i = 1 \text{ to } 2n \tag{1}$$

[1]Although this paper considers only models with line segment edges, the methodology is easily extended to curved surfaces (Lowe 1991).

This system of equations is nonlinear and frequently overdetermined. Nevertheless, an approximate solution can be found by using Newton's Method and least squares fitting. This process consists of repeatedly linearizing F and applying least squares. F is linearized by computing the Jacobian J, given by $J_{i,j} = \partial F / \partial X_j$, taken at X^0. The partial derivatives can be computed either analytically (Lowe 1991) or by using a direct approximation. The latter approach relies on the approximation $J_{i,j} \approx (F(X^0 + \epsilon e_j) - F(X^0))/\epsilon$, where e_j is the unit vector with all components 0 except for the jth component of 1, and ϵ is a small non-zero constant. Given these partial derivatives, the linear approximation of F_i at X_0 is given by

$$F_i(X) = d_i + \sum_{j=1}^{6} J_{i,j}(X_j - X_j^0) \tag{2}$$

If we define C to be the correction $X - X^0$, then combining Equations 1 and 2 yields

$$\sum_{j=1}^{6} J_{i,j} C_j = -d_i, \qquad i = 1 \text{ to } 2n \tag{3}$$

The sum of the squares of the errors in these equations is minimized by the least squares fit, which can be computed, for example, by the normal equations (Strang 1998). One potential problem is that if there are not enough line matches, the system of equations is underdetermined, and there is no unique least squares fit. Another is that it is possible for the system to be highly ill-conditioned and therefore unstable. To circumvent these problems, six stabilizing equations, $C_j = 0$, are added to the system, one for each dimension of the state space. Each stabilizing equation should theoretically be weighted inversely with the standard deviation of the prior estimate of that dimension (Lowe 1991), and the system is then solved by weighted least squares fitting (Strang 1998). The choice of stabilizing weights used for the experiments reported in this paper is discussed in the following section.

Newton's method consists of repeatedly computing and solving the linear approximation to the system of equations. The line matches, partial derivatives, and least squares fit are computed for each iteration of Newton's method. Although it is possible in theory for Newton's method to diverge, in practice the authors find that it always converges after few iterations.

After a camera pose X has been extracted from the image, the robot computes its body pose, (x, y, θ), updates that pose according to the robot's odometry estimate, and then estimates its camera pose again from the resulting body pose. This final camera pose, is used as the starting point, X^0, for the following frame.

The body position, (x, y), is defined as the center of the robot's body, and the orientation, θ represents the direction the body is pointing. The relationship between the body pose and the camera pose is represented by a homogeneous transformation matrix from the robot's camera-centered coordinate system to its body-centered coordinate system. This transformation, denoted as T^{cam}_{body}, is a function of the robot's body tilt and roll, and any joint angles connecting the camera to the head. The transformation matrix T^{cam}_{body} can be computed as the product of a series of DH-transformations over the joints from the body to the camera, as described by Schilling (Schilling 2000).

The translational component of T^{cam}_{body}, in particular the horizontal components, are used to determine the horizontal displacement between the camera and body centers. This displacement is subtracted from the horizontal translational components of the camera pose, X, to yield the robot's body position. A horizontal overall pan angle is also extracted from the transformation matrix, and subtracts from the camera pose pan angle to yield the body orientation. After the odometry update is applied to the body pose, the entire transformation matrix is used to compute the new starting estimate for the camera pose, X^0. This process is performed in between every two consecutive image frames. Pseudocode for the entire expectation-based method is presented in Algorithm 1. The algorithm is executed once for each incoming camera image.

Algorithm 1 The expectation-based method.

```
Given: $X^0$, camera image, 3-D line model, $numIterations$
Perform edge detection on camera image
Collect consecutive, collinear edge pixels into line segments
for $k = 1$ to $numIterations$ do
  $equationList \leftarrow \emptyset$
  for each model line do
    Project line onto image plane, for camera pose $X^0$.
    if part of the line projects onto the image rectangle then
      for each observed line do
        if observed and model lines match then
          Compute Jacobian of distances
          with respect to $X$.
          Add two equations to $equationList$,
          one for each distance.
        end if
      end for
    end if
  end for
  Add stabilizing equations to $equationList$.
  $X \leftarrow$ least squares fit of $equationList$.
  $X^0 \leftarrow X$
end for
Compute transformation matrix, $T^{cam}_{body}$.
Compute body pose, $(x, y, \theta)$ from final $X$.
Update body pose according to robot odometry.
Compute next starting camera pose, $X^0$, from body pose.
```

Test-Bed Domain

The methods described in this paper are implemented on a Sony Aibo ERS-7.[2] The robot is roughly $280mm$ tall and $320mm$ long. It has 20 degrees of freedom: three in each of four legs, three in the neck, and five more in its ears, mouth, and tail. At the tip of its nose there is a CMOS color camera that captures images at 30 frames per second in YCbCr format. The images are 208×160 pixels giving the robot a field of view of $56.9°$ horizontally and $45.2°$ vertically. The robot's processing is performed entirely on-board on a 576 MHz processor. Notably, legged robots, while generally more robust than wheeled robots to locomotion in various terrains (Wettergreen & Thorpe 1996; Saranli, Buehler, & Koditschek 2001), pose an additional challenge for vision, as the jagged motion caused by walking leads to unusually sharp motion in the camera image.

The robot's environment is a color-coded rectangular field measuring 4.4×2.9 meters, whose components are geometrical shapes: two colored goals composed of rectangular panes, four color-coded cylindrical beacons, and inclined walls surrounding the field. These components are distinguished primarily by their color; every part of the field is one of five discrete colors: green, white, blue, yellow, and pink. The two goals are blue or yellow, respectively, on the inside, and white on the outside. The cylindrical beacons are each composed of white, pink, and blue or yellow. Whether the third color is blue or yellow, and the relative positions of the colors on the cylinder, correspond to which corner of the field the beacon is in. Additionally, the field is surrounded by a white inclined border that is 10 cm high, and there are 2.5 cm thick white field lines on the green field surface. The robot and its environment are depicted in Figure 2.

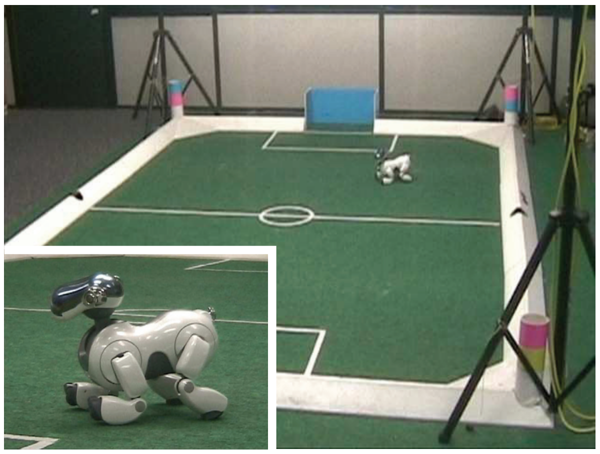

Figure 2: The Aibo robot and its environment. Noise in the robot's joints and a relatively low resolution camera make accurate localization a difficult task.

Because this domain has been used in the four-legged league of the annual RoboCup robot soccer competitions (Stone, Balch, & Kraetszchmar 2001), many researchers have addressed the problem of vision and localization on the Aibo robot on this field. Nevertheless, we are not aware of any that have taken the expectation-based approach.

In a color-coded domain such as this one, a common

[2]http://www.aibo.com

first step of the visual processing is *color segmentation*. Color segmentation is carried out via a *color table*, a three-dimensional array with a color label for each possible combination of Y, Cb, and Cr values. The color table is created off-line, by manually labeling a large suite of training data and using a Nearest Neighbor learning algorithm to learn the best label for each YCbCr combination. The full details of our color segmenting algorithm are presented in (Stone *et al.* 2004).

After the image has been segmented, the remaining processing depends on which approach to vision and localization is being used. The adaptations of these two approaches to a four-legged robot in a color-coded domain are described in the following two subsections.

Adapting the Particle Filtering Approach

Our group has previously adapted the particle filtering approach to vision and localization to this domain. Our implementation has been highly optimized, and it incorporates recent research contributions (Sridharan & Stone 2005; Sridharan, Kuhlmann, & Stone 2005).

In applying particle filtering to the test-bed domain, the observations consist of a visual identification of a landmark in the environment. When a landmark is observed, its distance and horizontal angle are computed from its size and position in the image. These quantities are the inputs to particle filtering.

As discussed above, the landmarks are recognized by a process of object detection. This process relies on the fact that, after segmentation, the goals and beacons have distinct, and relatively consistent, appearances in the image. For example, a goal usually appears as a blue or yellow rectangle, and a beacon will appears as a pair of squares, one directly above the other, one pink and the other blue or yellow, depending on which beacon it is.

The object recognition used in the experiments reported in this paper starts by segmenting every pixel in the image into a color category. As all of the pixels are segmented, adjacent pixels of the same color are combined into a *bounding box*, a rectangular structure consisting of the rectangle's coordinates and the color inside. Heuristics are used to determine if some adjacent boxes should not be merged, and also if some boxes should be deleted because they contain a density of the desired color that is too low. These bounding boxes are then used to determine which objects are present in the image.

The robot first attempts to detect goals in the image, because they are generally the largest objects found. Thus the blue and yellow bounding boxes are sorted from largest to smallest, and tested for being the goal in that order. The goal tests consist of heuristics to ensure that the bounding box in question is close enough to the right height to width ratio, and that there is a high enough density of the desired color inside the bounding box for a goal, as well as other heuristics. To find the beacons, the robot finds bounding boxes of pink and blue or yellow that satisfy appropriate beacon heuristics, and then combines them into beacon bounding boxes, labeled by which beacon is inside. The resulting goal and beacon bounding boxes are converted into observations for particle filtering.

Additionally, the lines on the field can provide useful information for localization, but they do not form useful bounding boxes, so they are recognized by a separate process. Following the approach of Rofer and Jungel (Rofer & Jungel 2003), we examine a series of vertical scan lines, testing for green/white transitions. These transitions represent field line pixels, and nearby collinear line pixels are collected into image lines. If the image contains two lines that meet at right angles when projected onto the field, their intersection is matched to a field line intersection and used as an observation input to particle filtering. Full details of our implementation of the particle filtering approach to vision and localization are presented in (Stone *et al.* 2004).

Adapting the Expectation-Based Approach

This section describes our adaptation of the expectation-based approach to the test-bed domain, which is new to this paper. For this approach, the model of the world is represented as a collection of three-dimensional line segments. These segments are each associated with two colors, one on each side. For line segments that border the edge of the environment, like the top of the goal, one of the two colors is left unspecified, as it is not known a priori what color the background will be. Additionally, some edges are surrounded by different colors depending on the robot's position. In these cases, the robot's estimated position is used to assign the edge colors.

As described in the Expectation-Based Approach section, the expectation-based method matches observed and model lines based on four criteria: length of overlap, effective distance, angle difference, and appearance match. In a color-coded domain, the two lines have an appearance match if the two colors around the observed line are the same as the two colors in the expected line. An unspecified (background) color is allowed to match any observed color.

The white field lines in the test-bed environment, which have a thickness of one inch, are represented in the model as pairs of lines, one inch apart. If the robot is too far away from these lines, they become invisible because they are too thin. In these cases, it is dangerous to include them in the expected view, because they are more likely to match incorrect lines than correct lines. Therefore, while computing the expected view, the robot computes the expected width of these field lines. If it is lower than a threshold width (given in Table 1), it is not included in the expected view.

One other aspect of the expectation-based approach that is dependent on the robotic platform is the computation of the coordinate transform from the robot's body to its camera. This depends on the head and neck joint angles, as well as the body's tilt, roll, and height off the ground. For the experiments reported in this paper, body tilt, roll, and height are relatively constant over the range of walking motions performed. They are therefore treated as constants and estimated manually beforehand.

The implementation of the expectation-based approach requires choosing values for a few constants, presented in Table 1. The first, $numIterations$, is the number of iterations of Newton's Method that are executed each vision frame. A few smaller numbers were tried first, but it was found that the method did not always converge in so few trials. For the line width threshold, the value of four pixels was the only value tried. The remaining constants are the weights of the stabilizing equations added to the least squares fit, one for each of the six dimensions of the state vector (measured in millimeters and radians). As mentioned above, the weight for each dimension should be inversely proportional to its standard deviation. These values were chosen with that rule of thumb in mind, but each of the three values had to be adjusted a few times (generally by multiplying or dividing by a factor of 10) based on preliminary experimentation. If the occasional false matches caused a given component of X to fluctuate wildly, the corresponding weight needed to be increased. If correct matches were unable to move a component, its weight needed to be decreased.

Constant	Value
$numIterations$	7
Line width threshold	4 pixels
x and y weight	0.002
z weight	0.01
Orientation weight	6

Table 1: Constants used by the expectation-based method.

The expectation-based method is depicted in Figures 3 and 4. Figure 3 shows the edges in the expected view. After Newton's method has been applied, the robot's new camera pose estimate is updated so that the lines are projected into the correct locations. This final situation is depicted in Figure 4. A video of the robot walking with superimposed model lines is available anonymously online.[3]

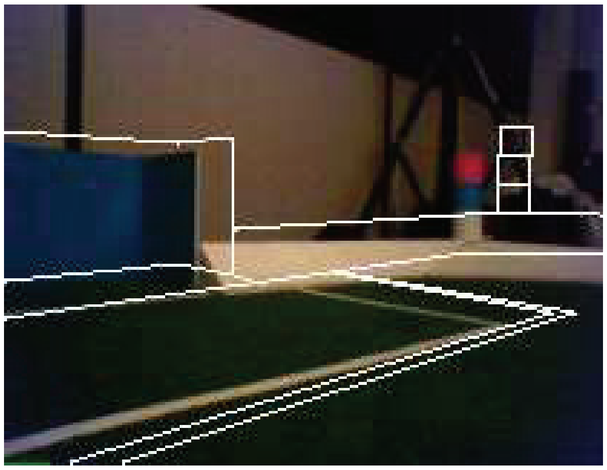

Figure 3: The white lines depict the expected view.

[3]www.cspapers.com/expectation-vision/

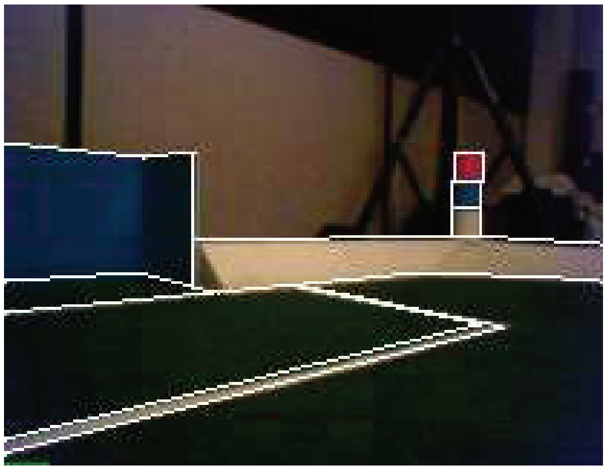

Figure 4: The white lines depict the final line placements.

Experimental Results

Both of the approaches to the vision and localization problem described above have been implemented and evaluated in our test-bed domain. The goal of both approaches is for the robot to be able to move freely throughout its environment and continually maintain an accurate estimate of its two-dimensional pose: (x, y, θ). This ability is tested by having the robot slowly walk to a series of poses in its environment. For each pose, the robot walks towards it, and stops when its location estimate is within a threshold distance (2 cm) of the desired position. It then rotates in place, until its orientation estimate is within a threshold distance (10°) of its intended orientation. It then stops moving and suspends vision and localization processing so that its position and orientation accuracy can be measured manually. These measurements are performed with a tape measure and a protractor, and have an estimated average measurement error of one centimeter and one degree respectively.

As the robot walks from each pose to the next, its head scans from left to right, to increase the diversity of visual information the robot receives over time. The two approaches being tested were developed for different head scan speeds, and therefore each method is evaluated with the scan speed for which it has been optimized. The expectation-based approach uses a significantly slower head scan than the particle filtering approach. This difference was based on the intuition that the particle filtering approach would be less effective with a slower head scan, because the observations in the landmark histories would grow stale before they could be used for reseeding. On the other hand, the expectation-based method performs better with the slower head scan, because it is less prone to error when consecutive image frames are as similar to each other as possible. An informal comparison confirmed that the particle filtering method worked at least as well with the faster head scan as it did with the slow one. Aside from this difference, the testing scenario is identical between the two vision and localization methods. The walking module used in the experiments reported in this paper has been developed previously as part of a larger project (Stone *et al.* 2004).

For each pose that the robot attempts to visit, its position error and angle error are recorded. In each trial, the robot attempts to visit a pre-set series of 14 poses on the field. Each trial begins with the robot's pose estimate being as accurate possible. The poses visited are depicted in Figure 5. The sequence of poses is designed to cover a wide range of difficulties for vision and localization. The average position and angle errors for a given trial are considered as one data point. Each approach was evaluated over ten trials. The means and standard deviations of the average errors are presented in Table 2.

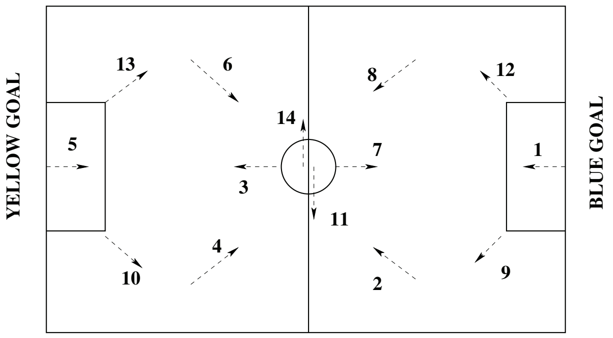

Figure 5: The series of poses traversed by the robot.

Method	Particle Filtering	Expectation-Based
Position Error	11.81 ± 0.82 cm	7.55 ± 0.63 cm
Angle Error	6.49 ± 1.61 degrees	7.18 ± 2.22 degrees

Table 2: Comparison of the particle filtering and expectation-based methods.

The position errors attained by the particle filtering approach were, on average, 56% more than those attained by the expectation-based approach. This difference is statistically significant in a one-tailed t-test, with a p-value of less than 10^{-9}. Although the particle filtering approach achieved a lower average angle error, this difference was not statistically significant, with a p-value of 0.44 for a two-tailed t-test.

Discussion

The experiments reported in the previous section indicate that the expectation-based method is able to maintain a more accurate estimate of its pose, compared to particle filtering, as the robot walks to a series of points. Nevertheless, there are advantages to each method that are not reflected in this test.

An important difference between the two algorithms is their robustness. Particle filtering is robust to the jagged camera motions caused by walking and scanning the head quickly, large unmodeled movements, and collisions (Sridharan, Kuhlmann, & Stone 2005). By contrast, the expectation-based method, as described above, is relatively brittle. If the robot's pose estimate becomes sufficiently inaccurate, the computed expectation views will not be close to the observed line segments. Because the line matching process assumes that the expected positions of model lines are close to the corresponding actual lines in the image, if this assumption is violated, it is difficult for the algorithm to infer useful information about the robot's pose from the observed image. In this case, the algorithm may not be able to recover. An interesting problem for future research is to augment the expectation-based method with a recovery method that would enable it to be simultaneously highly accurate and robust.

On the other hand, one potential advantage to the expectation-based method is the ease with which it could be extended to different environments. For example, consider the task of implementing the two approaches in a different domain with color-coded geometrical shapes. For the expectation-based approach, all that is needed is a line model of the new domain. By contrast, the particle filtering approach relies on object detection, which employs different algorithms in identifying differently shaped objects. For example, in the adaptation to the test-bed domain described above, three different algorithms are used to recognize goals, beacons, and lines. In a new domain with objects with different appearances, new algorithms would need to be developed to recognize those objects.

In this paper, we restrict our attention to a robot localizing in a known, fixed environment. One common approach to localizing in an unknown environment is (vision-based) Simultaneous Localization and Mapping (SLAM) (Sim *et al.* 2005), which frequently incorporates a particle filtering localization algorithm. One challenge for future research is to similarly extend the expectation-based approach to have simultaneous mapping capabilities.

Both the particle filtering and expectation-based approaches are adaptable, to some extent, to a partially dynamic environment. For example, if we add a moving ball to our test-bed domain, the state space can be represented as a concatenation of the robot's pose and the ball's location. In particle filtering, the particles might represent a joint distribution over the ball and the robot, while in the expectation-based method, the ball would be added to the expected view based on the prior estimate of its location. Further empirical tests are required to determine if the two methods would continue to be effective under these conditions.

Conclusion

This paper presents and compares two competing approaches to the problem of vision and self-localization on a mobile robot, the commonly used particle filtering approach, and the expectation-based approach. The approaches are implemented and tested on a Sony Aibo ERS-7 robot, localizing as it walks through a test-bed domain. Each method's localization accuracy was measured over a series of trials. In these tests, the expectation-based method achieved a significantly lower average distance error than the particle filtering method, and no significant difference was found in the methods' average orientation errors.

Aside from the methods' performances on the experimental tests presented here, each method has further relative advantages, discussed in the previous section. It is a continuing focus of research to develop a method for vision and localization that simultaneously achieves the advantages of both approaches.

References

Canny, J. 1986. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence* 8(6):679–698.

Dellaert, F.; Fox, D.; Burgard, W.; and Thrun, S. 1999. Monte carlo localization for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Engelson, S., and McDermott, D. 1992. Error correction in mobile robot map learning. In *Proceedings of the IEEE International Conference of Robotics and Automation*, 2555–2560.

Forsyth, D., and Ponce, J. 2003. *Computer Vision: A Modern Approach*. Upper Saddle River, New Jersey: Prentice Hall.

Gasteratos, A.; Beltran, C.; Metta, G.; and Sandini, G. 2002. PRONTO: a system for mobile robot navigation via CAD-model guidance. 26:17–26.

Hoffman, J.; Spranger, M.; Gohring, D.; and Jungel, M. 2006. Making use of what you don't see: negative information in Markov localization. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems*.

Kalman, Rudolph, E. 1960. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering* 82:35–45.

Kosaka, A., and Pan, J. 1995. Purdue experiments in model-based vision for hallway navigation. In *Proceedings of Workshop on Vision for Robots in IROS'95*.

Kwok, C., and Fox, D. 2004. Reinforcement learning for sensing strategies. In *Proceedings of the International Confrerence on Intelligent Robots and Systems (IROS)*.

Lenser, S., and Veloso, M. 2000. Sensor resetting localization for poorly modelled mobile robots. In *The International Conference on Robotics and Automation*.

Lowe, D. 1991. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-13(5):441–450.

Rofer, T., and Jungel, M. 2003. Vision-based fast and reactive monte-carlo localization. In *The IEEE International Conference on Robotics and Automation*.

Saranli, U.; Buehler, M.; and Koditschek, D. 2001. RHex: A simple and highly mobile hexapod robot. *The International Journal of Robotics Research* 20(7):616–631.

Schilling, R. 2000. *Fundamentals of Robotics: Analysis and Control*. Prentice Hall.

Sim, R.; Elinas, P.; Griffin, M.; and Little, J. 2005. Vision-based SLAM using the Rao-Blackwellised particle filter. In *IJCAI Workshop on Reasoning with Uncertainty in Robotics*.

Sridharan, M., and Stone, P. 2005. Real-time vision on a mobile robot platform. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Sridharan, M.; Kuhlmann, G.; and Stone, P. 2005. Practical vision-based monte carlo localization on a legged robot. In *IEEE International Conference on Robotics and Automation*.

Stachniss, C.; Burgard, W.; and Behnke, S. 2006. Metric localiztion with scale-invariant visual features using a single perspective camera. In *Proceedings of the European Robotics Symposium (EUROS)*.

Stone, P.; Balch, T.; and Kraetszchmar, G., eds. 2001. *RoboCup-2000: Robot Soccer World Cup IV*. Lecture Notes in Artificial Intelligence 2019. Berlin: Springer Verlag.

Stone, P.; Dresner, K.; Fidelman, P.; Jong, N. K.; Kohl, N.; Kuhlmann, G.; Sridharan, M.; and Stronger, D. 2004. The UT Austin Villa 2004 RoboCup four-legged team: Coming of age. Technical Report UT-AI-TR-04-313, The University of Texas at Austin, Department of Computer Sciences, AI Laboratory.

Strang, G. 1998. *Linear Algebra and its Applications*. Brooks Cole.

Thrun, S.; Fox, D.; Burgard, W.; and Dellaert, F. 2001. Robust monte carlo localization for mobile robots. *Journal of Artificial Intelligence*.

Wettergreen, D., and Thorpe, C. 1996. Developing planning and reactive control for a hexapod robot. In *Proceedings of ICRA '96*, volume 3, 2718–2723.