In *International Conference on Robotics and Automation* (ICRA 07),
pp. 3915-3920, Rome, April 2007.

A Comparison of Two Approaches for Vision and 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 considers two approaches to the problem of vision and self-localization on a mobile robot. In the first approach, the perceptual processing is primarily bottom-up, with visual object recognition entirely preceding localization. In the second, significant top-down information is incorporated, with vision and localization being intertwined. That is, the processing of vision is highly dependent on the robot's estimate of its location. The two approaches are implemented and tested on a Sony Aibo ERS-7 robot, localizing as it walks through a color-coded test-bed domain. This paper's contributions are an exposition of two different 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.**

I. Introduction

Despite evidence that perceptual processing in humans makes use of a combination of bottom-up and top-down processing [1], the standard approach to many problems in robotics continues to be a primarily bottom-up approach. That is, information flows in only one direction, from the low-level sensory inputs towards progressively higher-level knowledge and concepts. This discrepancy suggests analyzing the relative advantages and disadvantages of incorporating top-down information in a robot's perceptual processing.

This paper explores this issue in the context of 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 considers two contrasting approaches to this problem, distinguished primarily by the extent to which localization information is used during the vision processing. In one method, which we call Approach A, the visual processing completely precedes the localization computations. By contrast, in Approach B, the robot's localization and vision are intermixed, with vision processing relying on the robot's prior estimate of its pose. While there are many possible algorithms that take either approach, this paper presents representative examples of the two approaches, comparing them empirically as a case study.

In Approach A, the problem of vision and localization decomposes neatly into a vision problem followed by a localization problem. The input to vision is the raw camera input, and the output is a collection of identified objects and their locations in the image. This, in turn, is the input to the localization problem, along with the previous pose estimate.

Solving the vision problem in general, or being able to accurately attach labels to objects in an arbitrary visual scene, is currently far beyond the state of the art in computer vision. However, when the visual scene is restricted to containing a small set of familiar objects with known visual footprints, the problem can be solved with a relatively high degree of accuracy. The general method involves scanning each incoming image to determine if any area of the image matches the appearance of any of the objects in the environment. This approach to object detection has been applied successfully in many different domains [2], [3].

When matches are found, they are used to determine the robot's relative horizontal angle and distance to the corresponding object. As the robot moves through its environment, it must continually incorporate its noisy odometry estimates, as well as its current landmark observations, into its running pose estimate. One method that has been shown to be very effective on state estimation problems with noisy observations and actions is Monte Carlo Localization (MCL) [4]. Much work has been done in MCL on vision-based robots. This includes work in vision-based legged robots [5], [6], [7], [8], [9] and wheeled robots [10], [11]. The two components of Approach A, object recognition and MCL, are presented in the following section.

Approach B is based on the idea that knowledge about the robot's location can be used to inform its processing of its visual input. The location information is useful because, in a fixed environment, knowledge of the camera pose can be used to predict what is appearing in each image frame. For example, the authors have previously shown that predicting the locations of objects in the image can dramatically speed up a robot's visual processing [12]. In this paper, instead of using prediction to speed up vision, the instance of Approach B discussed uses prediction to perform a qualitatively different kind of vision, towards the goal of localization accuracy. Specifically, the implementation of Approach B does not perform object recognition at all. Instead, it represents the environment in terms of the three-dimensional locations, orientations, and appearance of the edges in the environment. This information is combined with the robot's three-dimensional camera pose estimate to yield 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 [13], and then collecting consecutive, collinear edge points into line segments [14]. 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. 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 Section III.

The instance of Approach B described in this paper is based in part on the object tracking algorithm presented by Lowe [15]. More recent object tracking methods are presented by Harris [16] and by Drummond and Cipolla [17]. In the context of mobile robot localization, Approach B has been employed both on wheeled robots [18], [19], [20] and on a legged robot by Gasteratos et al. [21].

Finally, there has been some previous work comparing localization methods. Gutmann and Fox compare a number of such methods [22], but these methods all use the same vision process and so do not compare methods for vision and localization taken as a whole. Lauer et al. compare examples of Approach A and B [20], but do not provide a quantitative comparison based on ground truth measurements. We are not aware of any previous work that provides such a comparison.

The remainder of this paper is organized as follows. The following two sections present the two approaches to vision and localization on a mobile robot. Section IV presents an experimental comparison of the two approaches implemented on a Sony Aibo ERS-7 in a color-coded test-bed domain. Finally, Section V discusses the relative advantages of each method and Section VI concludes.

II. Approach A

Approach A can be divided into two components. Section II-A discusses object recognition in a color-coded domain, and Section II-B describes Monte Carlo Localization.

A. Object Recognition

The implementation of object recognition used by Approach A for this paper was previously developed by our research group. These algorithms are described in complete detail by Sridharan and Stone [3] and in our technical report [23]. They are summarized in this section.

Using object recognition for localization relies on the environment having a set of landmarks: distinct objects at fixed, known positions. For example, if the objects are a set of differently colored spheres, they will appear as differently colored solid circles in the image, and their sizes and colors can be used to determine the robot's distances from the corresponding objects in its environment.

In a color-coded domain like the one used in the experiments reported in this paper, a common first step of the visual processing is *color segmentation* (see, e.g., [24]). Color segmentation is carried out via a *color table*, a three-dimensional array with a color label for each possible combination of camera pixel color components. The color table can be created off-line by manually labeling a large suite of training data and using the Nearest Neighbor algorithm to learn the best label for each color combination.

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 whether boxes should be merged or removed.

Finally, object recognition itself consists of using these bounding boxes to determine which objects are present in the image. This process is highly dependent on the details of the set of landmarks in the environment and their appearances. Each object is recognized by a routine that identifies bounding boxes, or combinations of bounding boxes, that indicate the presence of the object in question. These recognition routines are heuristically designed based on the details of the environment. The heuristics used in Approach A have been manually tuned for the test-bed domain [23].

Once each object in the image has been identified, it's identity and image location is used to inform the robot's knowledge of its own location in the environment. Combining this information with the robot's odometry into a running estimate of its pose is performed by Monte Carlo Localization, described in the following section.

B. Monte Carlo Localization

After identifying the objects present in the image, Approach A proceeds by incorporating the objects' positions in the image into a running estimate of the robot's two-dimensional body pose. MCL addresses the question of how to represent the robot's uncertainty in its own pose. MCL 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. The update is applied to each particle in its own reference frame, causing differently oriented particles to move in different directions. Furthermore, to model random noise in the motion model, a random perturbation is added to each particle's pose.

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 localization algorithm's processing 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 (i.e., by triangulation). This process is based on the idea of Sensor Resetting [5]. Reseeding enables the robot to recover from large, unmodeled movements, also known as the kidnapped robot problem [25]. 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* [8]. 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.

III. Approach B

This section presents the implementation of Approach B used in this paper. Recall that in this approach, the vision and localization processes are intertwined. That is, the visual interpretation of each image is heavily influenced by the robot's prior knowledge about its location, and the result of the visual processing yields direct information about the robot's three-dimensional camera pose, which in turn informs the pose of the robot itself.[1]

For each input image, the robot begins by segmenting the entire image into color categories, just as in Approach A. It further analyzes the image by performing edge detection on the segmented image, and combining collinear edge points into line segments, as discussed in Section I. The remainder of the robot's visual processing, however, is intertwined with its localization. This process consists of the robot 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. Since the robot walks on a two-dimensional surface, it also 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, each edge in the three-dimensional line model is projected onto the image plane.[2] If no part of the edge projects to 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. In a color-coded domain, these features are the colors on either side of the line. Certain lines represent the boundary between the known environment and the unknown background. The side of the line corresponding to the background is permitted to match any color. Additionally, some edges may be 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. 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.

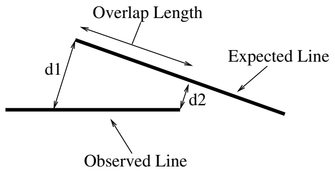

Fig. 1. For each pair of an expected view line and a line detected in the image, the robot computes a length of overlap, the angle difference, and the effective distance, defined as the average of d_1 and d_2.

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 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 as 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 match out of n. If the fit were perfect, the following $2n$ equations would all be satisfied:

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

This system of equations is nonlinear and frequently overdetermined. An efficient and numerically stable way of approximating a solution to such a system is Levenberg-Marquardt optimization [26]. This process relies on the Jacobian, J, of F, given by $J_{i,j} = \partial F_i / \partial X_j$, taken at X. The partial derivatives can be computed either analytically [15] or by using a direct approximation. The latter approach relies on the approximation $J_{i,j} \approx (F(X + \epsilon e_j) - F(X))/\epsilon$, where e_j is the unit vector with all components 0 except for the

[1] To our knowledge, this method has not been previously implemented on the Aibo.

[2] Although this paper considers only models with line segment edges, the methodology is easily extended to curved surfaces [15].

jth component of 1, and ϵ is a small non-zero constant. Given these partial derivatives, Levenberg-Marquardt iteratively solves for p in the equation:

$$(J^T J + \lambda I)p = -J^T F(X) \tag{2}$$

where p points in the direction of the correction applied to X from one iteration to the next and λ is a parameter that is typically set heuristically. In the instance of Approach B presented here, a slight variant of Levenberg-Marquardt is used, where λ is constantly set to one and the applied adjustment to X is equal to p in both direction and magnitude. Notably, the impact of the λI term is heavily dependent on the units of the coordinate system in which X is represented. These units are chosen manually, each one proportional to an estimate of the variance, or average inaccuracy, of X in that dimension.[3] Furthermore, in each iteration, the line matches are recomputed in accordance with the latest estimate of X, before Equation 2 is applied again.

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, 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 [27].

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 subtracted 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 all of Approach B is presented in Algorithm 1. The algorithm is executed once for each incoming camera image.

Approach B is depicted in Figure 2. The left side shows the edges in the expected view. After Levenberg-Marquardt has been applied, the robot's new camera pose estimate is updated so that the projected lines match the observed ones (right). A video of the robot's point of view with superimposed model lines is available online.[4]

IV. Experimental Results

A. Test-Bed Domain

The methods described in this paper are implemented on a Sony Aibo ERS-7.[5] 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 onboard on a 576 MHz processor. Notably, legged robots, while generally more robust than wheeled robots to locomotion in various terrains [28], [29], pose an additional challenge for vision, as the jagged motion caused by walking leads to unusually sharp motion in the 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.
          Add two equations to $equationList$,
          one for each distance.
        end if
      end for
    end if
  end for
  $X \leftarrow p$ from Equation 2.
  $X^0 \leftarrow X$
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.
```

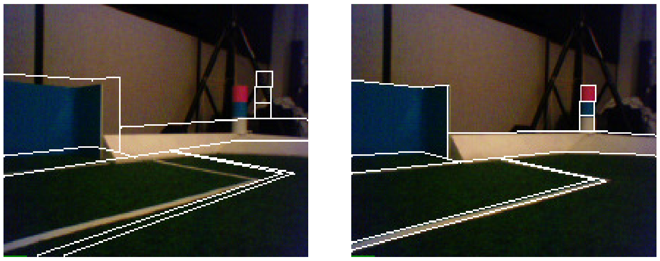

Fig. 2. Left: The white lines depict the expected view. Right: The final line placements.

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

[3] Lowe [15] uses "stabilizing equations" that have the same effect.

[4] www.cs.utexas.edu/~AustinVilla/?p=research/model-based_vision

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

surrounding the field. Additionally, the field is surrounded by an inclined border that is 10 cm high, and there are 2.5 cm thick lines on the field surface. There are five discrete field colors total. The robot and its environment are depicted in Figure 3.

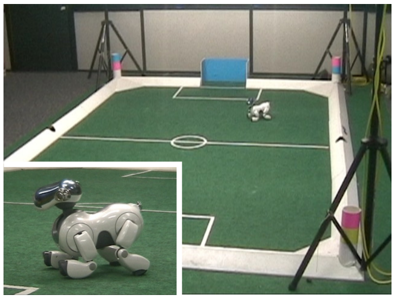

Fig. 3. The Aibo robot and its environment. Noise in the robot's joints and a relatively low resolution camera make accurate localization a difficult task.

Because this domain has been used in the four-legged league of the annual RoboCup robot soccer competitions [30], many researchers have addressed the problem of vision and localization on the Aibo robot on this field. However, all the approaches we are aware of have been in the spirit of Approach A.

One prominent feature of this test-bed domain is the presence of lines on the field. From the point of view of Approach A, these lines 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 [7], 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. Intersections of observed lines can then be used as observation inputs to MCL. Full details of our implementation of Approach A are presented in [23].

The field lines also affect the way Approach B is implemented. They are represented in the environment model as pairs of parallel lines, 2.5 cm 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 (the value used is four pixels), it is not included in the expected view.

One other aspect of Approach B 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.

B. 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. Preliminary testing revealed that each of the two approaches worked best at a different head scan speed, with Approach B working best with a significantly slower head scan. Approach A is better suited to a faster head scan, because if the head moves too slowly the landmark histories would grow stale before they could be used for reseeding. On the other hand, the Approach B 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. Each approach was evaluated with the head scan speed at which it worked the best. Aside from this difference, the testing scenario is identical between the two vision and localization methods.

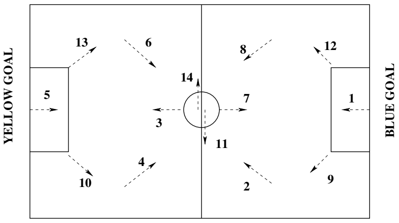

Fig. 4. The series of poses traversed by the robot.

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 4. The sequence of poses is designed to cover a wide range of difficulties for vision and localization. The robot's position error and angle error are recorded for each pose attempted. 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 I.

Method	Approach A	Approach B
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 I

EXPERIMENTAL RESULTS.

The position errors attained by Approach A were, on average, 56% more than those attained by Approach B. This difference is statistically significant in a one-tailed t-test, with a p-value of less than 10^{-9}. Although Approach A 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. Additionally, both approaches were similarly computationally efficient, running in real time on-board the robot at frame rate (30 Hz), along with the robot's motion processing.

V. Discussion

Of the two approaches to vision and localization presented in this paper, each has a number of relative advantages and disadvantages. The experimental results presented in the previous section demonstrate that, in the test-bed domain used in this paper, a representative instance of Approach B achieves more accurate localization than one of Approach A. This result suggests that incorporating top-down information in a robot's perceptual processing can be used to yield an advantage in certain robotic settings.

However, one factor that was not evaluated above was the robustness of the two approaches. For example, if a robot is suddenly moved to a new location without its knowledge, how quickly can the robot recover from its sudden localization inaccuracy? The instance of Approach A presented in this paper has a built-in measure designed to enable a smooth recovery from such an occurrence, namely the reseeding particles described in Section II-B. On the other hand, the implementation of Approach B has no such mechanism, and preliminary experiments showed it to be quite brittle to large unmodeled movements. Lauer et al. achieve robustness to such movements in Approach B by periodically testing a number of random poses to search for matches [20].

Furthermore, both methods rely on complete prior knowledge of the fixed geometrical structure of the environment. In Approach A, this information was represented as a set of discrete objects and lines, each with its own location, appearance, and complex set of heuristics needed to properly recognize it. In Approach B, knowledge of the environment was encoded as a set of three-dimensional line segments with color labels. In considering transferring either approach to new domains, one important question is how easy it would be for human programmers to codify the prior knowledge needed by the robot in those domains. The authors' intuition is that the object recognition heuristics needed by Approach A are often more difficult to specify than the CAD-type model information used by Approach B. However, in a novel, complex environment, either one could be a daunting task.

VI. Conclusion

This paper presents and compares two approaches to the problem of vision and self-localization on a mobile robot. In Approach A, processing is strictly bottom-up, with visual object recognition entirely preceding localization. In Approach B, the two processes are intertwined, with the processing of vision being highly dependent on the robot's estimate of its location. The two 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, Approach B attained a significantly lower average distance error than Approach A, and no significant difference was found in the methods' average orientation errors. These differences represent just one of the many relative benefits and drawbacks of the two approaches, which are discussed in Section V.

Looking forwards, it would be desirable for a robot to learn the relevant parameters of its novel environment autonomously, in analogy with the success in Simultaneous Localization And Mapping (SLAM) on wheeled robots with laser range finders. For robots whose primary sensors are cameras, one possibility for future work involves combining the techniques of Approach B with those used in vision-based SLAM, e.g. [31], [32], [33]. Further research towards combining the relative advantages of different methods will continue to bring the field closer to the long-term goal of fully autonomous robotic exploration of novel environments.

Acknowledgments

This research is supported in part by NSF CAREER award IIS-0237699 and ONR YIP award N00014-04-1-0545. Thanks also to the UT Austin Villa team, and especially Mohan Sridharan, for developing the software base used in this paper and our implementation of approach A.

References

[1] J. Wolfe, S. Butcher, C. Lee, and M. Hyle, "Changing your mind: On the contributions of top-down and bottom-up guidance in visual search for feature singletons," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 29, no. 2, pp. 483–502, 2003.

[2] R. Nelson and A. Selinger, "Large-scale tests of a keyed, appearance-based 3-d object recognition system," *Vision Research, Special issue on computational vision*, vol. 38, no. 15–16, 1998.

[3] M. Sridharan and P. Stone, "Real-time vision on a mobile robot platform," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, August 2005.

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

[5] S. Lenser and M. Veloso, "Sensor resetting localization for poorly modelled mobile robots," in *The International Conference on Robotics and Automation*, 2000.

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

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

[8] M. Sridharan, G. Kuhlmann, and P. Stone, "Practical vision-based monte carlo localization on a legged robot," in *IEEE International Conference on Robotics and Automation*, April 2005.

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

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

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

[12] D. Stronger and P. Stone, "Selective visual attention for object detection on a legged robot," in *RoboCup-2006 Robot Soccer World Cup X*, 2007.

[13] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

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

[15] D. Lowe, "Fitting parameterized three-dimensional models to images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-13, no. 5, pp. 441–450, 1991.

[16] C. Harris, *Tracking with Rigid Models*. MIT Press, 1992.

[17] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 932–946, 2002.

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

[19] T. Schmitt, R. Hanek, M. Beetz, S. Buck, and B. Radig, "Cooperative probabilistic state estimation for vision-based autonomous mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 670–684, Oct. 2002.

[20] M. Lauer, S. Lange, and M. Riedmiller, "Calculating the perfect match: an efficient and accurate approach for robot self-localization," in *RoboCup Symposium 2005*, 2005.

[21] A. Gasteratos, C. Beltran, G. Metta, and G. Sandini, "PRONTO: a system for mobile robot navigation via CAD-model guidance," vol. 26, pp. 17–26, 2002.

[22] J.-S. Gutmann and D. Fox, "An experimental comparison of localization methods continued," in *Proceedings of Intelligent Robots and Systems (IROS)*, October 2002.

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

[24] J. Bruce, T. Balch, and M. Veloso, "Fast and inexpensive color image segmentation for interactive robots," in *The International Conference on Intelligent Robots and Systems (IROS)*, 2000.

[25] S. Engelson and D. McDermott, "Error correction in mobile robot map learning," in *Proceedings of the IEEE International Conference of Robotics and Automation*, May 1992, pp. 2555–2560.

[26] P. Gill, W. Murray, and M. Wright, *Practical Optimization*. Academic Press, 1981.

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

[28] D. Wettergreen and C. Thorpe, "Developing planning and reactive control for a hexapod robot," in *Proceedings of ICRA '96*, vol. 3, 1996, pp. 2718–2723.

[29] U. Saranli, M. Buehler, and D. Koditschek, "RHex: A simple and highly mobile hexapod robot," *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 616–631, 2001.

[30] P. Stone, T. Balch, and G. Kraetzschmar, Eds., *RoboCup-2000: Robot Soccer World Cup IV*, ser. Lecture Notes in Artificial Intelligence. Berlin: Springer Verlag, 2001, vol. 2019.

[31] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *Int. J. Robotics Research*, vol. 21, no. 8, pp. 735–738, 2002.

[32] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proceedings of the Ninth IEEE International Conference on Computer Vision*, Oct. 2003.

[33] R. Sim, P. Elinas, M. Griffin, and J. Little, "Vision-based SLAM using the Rao-Blackwellised particle filter," in *IJCAI Workshop on Reasoning with Uncertainty in Robotics*, Edinburgh, Scotland, 2005.