Efficient Entropy-Based Action Selection for Appearance-Based Robot Localization

J. M. Porta, B. Terwijn, and B. Kröse

IAS, Intelligent Autonomous Systems, Informatics Institute
University of Amsterdam, Kruislaan 403, 1098 SJ, Amsterdam, The Netherlands
{porta,bterwijn,krose}@science.uva.nl

***Abstract*— In this paper, we extend our appearance-based localization system [8] moving from a passive approach to an active one where the robot can execute actions with the only purpose of gaining information about its location in the environment. We present a general framework for entropy-based action selection and we describe how this framework can be implemented in our localization system. The result is an action evaluation process more efficient in memory and in execution time than previously existing ones. The experiments we present show that the action selection mechanism effectively decreases the error on localization in environments with a high degree of aliasing. This can be of great help to improve the performance of our localization system in dynamic environments.**

I. Introduction

An autonomous mobile robot must be able to find out by itself its position in the environment. This information is very useful for navigation that is one of the most basic abilities for a mobile robot.

Robot localization from odometry is affected by large error in the long term. Consequently, other sensors have to be used to deduce the robot position. In some cases [2], range sensors (sonar/laser) has been used for this purpose. However, it is widely recognized [3] that the proper solution to the robot localization problem requires the use richer sources of information such as cameras (alone or, most probably, in combination with simpler sensors).

In previous work at our group, an omnidirectional camera has been used to determine the location of the robot [13]. However, omnidirectional vision has some disadvantages: any change in the environment affects the images taken by the camera making matching against images in a training set difficult (or even impossible). So, the system exhibits some problems in highly dynamic environments. To solve this problem, we decided to use a normal camera mounted on a pan-and-tilt device. Normal cameras have a more restricted field of view than omnidirectional cameras and, so, a small change in the environment can affect some of the images the robot can get form a certain position, but not all of them. In the case the robot gets lost, the camera can be readily rotated using the pan-and-tilt device to get a new image. With an adequate sequence of camera movements, the localization in dynamic environments can be significantly improved.

In this paper, we present an entropy-based criterion to evaluate all the possible camera movements. The theory of entropy-based action selection is not new. The key issue here is how to efficiently approximate all the probability distributions involved in the formulation.

The paper is organized as follows. First, we review some work related with our approach. Then, we present our probabilistic framework for robot localization. Next, we extend this framework with the entropy-based criterion for camera movement selection and we describe the experiments that validates its utility. Finally, we summarize our work and we outline lines for further research.

II. Related work

In the last years, there has been a large effort on the development of robust robot localization systems and, therefore, there are many works that have points of contact with that of ours.

Works on Markov localization had established a formal probabilistic framework for robot localization [12]. However, in most of these works, the robot is modeled as a passive agent that gets observations from the environment while performing other tasks. In our work, we address the *active localization* problem where the robot can execute actions with the only purpose of gaining information about its location.

The active localization problem has been addressed by some authors before [4], [5], [6], [7], [9]. Despite the general approach of these works is the same as that of ours (action selection based on entropy), they left large room from improvement. For instance, a problem that remains open is how to efficiently obtain a sensor model of the environment. In some of the existing works, this is done by exhaustive sensor simulation. However, this is only possible if a geometric model of the environment is available (which is not always the case) and if simple range-based sensors have to be simulated. In our framework, the sensor model can be directly extracted from the supervised training set used for the appearance-based localization.

Another open problem is how to represent the probability on the robot's position. In our work, a particle filter is used for this purpose and this makes the resulting system more efficient in the use of memory and in execution time than previous works mainly based on dense probabilistic grids discretizing the whole configuration space of the robot.

Despite the popularity of particle filters they have not been used for active localization and few times for localization using vision. In the cases in which vision is used [3], it is done in a very simple way (for instance, just checking for the brighter point in a small area of the image). Our appearance-based framework extracts more information from each image and, as already suggested in [3], the resulting localization system would converge in less iterations.

Finally, another field closely related with that of ours is that of object recognition from images where active appearance-based systems have been developed [1].

III. General Framework

In the following subsections we introduce the three basic elements of our localization system: the Markov localization model, the auxiliary particle filter, and the appearance-based paradigm for localization.

A. A Probabilistic Model for Robot Localization

The Markov localization method aims at improving the estimation of the position and orientation of the camera (from which the position and orientation of the robot can be readily determined) at time t (denoted as x_t) taking into account the movements of the robot (and the pan-and-tilt) $\{u_1,\ldots,u_t\}$ and the observations of the environment taken by the robot $\{y_1,\ldots,y_t\}$ up to that time[1]. Formally, we want to estimate the posterior $p(x_t|\{u_1,y_1,\ldots,u_t,y_t\})$. The Markov assumption states that this probability can be updated from the previous state probability $p(x_{t-1})$ taking into account only the last executed action (u_t) and the last observation (y_t). Thus we only have to estimate $p(x_t|u_t,y_t)$. Applying Bayes we have that

$$p(x_t|u_t,y_t) \propto p(y_t|x_t)\, p(x_t|u_t), \tag{1}$$

where the probability $p(x_t|u_t)$ can be computed propagating from $p(x_{t-1}|u_{t-1},y_{t-1})$

$$p(x_t|u_t) = \int p(x_t|u_t,x_{t-1})\, p(x_{t-1}|u_{t-1},y_{t-1})\, dx_{t-1}. \tag{2}$$

Note that what we are actually computing in equation 2 is $p(x_t|u_t,u_{t-1},y_{t-1})$ but, due to the Markov assumption, terms u_{t-1} and y_{t-1} have no influence on the probability on x_t.

[1] In our notation, the Markov process goes through the following sequence: $x_0 \xrightarrow{u_1} (x_1,y_1) \xrightarrow{u_2} \ldots \xrightarrow{u_t} (x_t,y_t)$.

The probability $p(x_t|u_t,x_{t-1})$ for any couple of states and for any action is called the *action model* and it is assumed as known (i.e., inferred from odometry). On the other hand, $p(y_t|x_t)$ for any observation and state is the *sensor model* that has to be defined for each particular problem.

Equations 1 and 2 define a recursive system to estimate the position of the robot.

B. The Auxiliary Particle Filter

The previous is just theoretical framework and, for each particular case, we have to devise how to approximate all the probability distributions involved in this framework and how to update them. In our case, we use a particle filter [11].

In this approach, the continuous posterior $p(x_{t-1}|u_{t-1},y_{t-1})$ is approximated by a set of I random samples, called particles, that are positioned at points x^i_{t-1} and have weights π^i_{t-1}. Thus, the posterior is given by:

$$p(x_{t-1}|u_{t-1},y_{t-1}) = \sum_{i=1}^{I} \pi^i_{t-1}\, \delta(x_{t-1}|x^i_{t-1}),$$

where $\delta(x_{t-1}|x^i_{t-1})$ represents the delta function centered at x^i_{t-1}. Using the above, the integration of equation 2 becomes discrete

$$p(x_t|u_t) = \sum_{i=1}^{I} \pi^i_{t-1}\, p(x_t|u_t,x^i_{t-1}). \tag{3}$$

The central issue in the particle filter approach is how to obtain a set of particles (that is, a new set of states x^i_t and weights π^i_t) to approximate $p(x_t|u_t,y_t)$ from the set of particles approximating $p(x_{t-1}|u_{t-1},y_{t-1})$. In [11] you can find details of how to perform this step in the auxiliary particle filter.

C. Appearance-based Localization

The open problem in the just described framework is how to compute the sensor model $p(y|x)$.

In general [14], non-parametric models are used to derive the distribution from the supervised training set. In our case, however, the observations y are images and their high dimensionality makes direct use of non-parametric models unfeasible. To alleviate this problem, the appearance-based paradigm [10] propose to compress images to few feature detectors using a lineal projection. The projection matrix is obtained by principal component analysis (PCA) on a supervised training set ($T = \{(x_i,y_i)|\, i \in [1,N]\}$) including observations y_i obtained at known states x_i. After the PCA computation, we only have to keep the subset of eigenvectors that represent most of the variance of the images.

Vlassis *et al.* in [13] compute $p(y|u)$ using the J points in the training set that are closer to y (after the

corresponding PCA dimensionality reduction). Thus, we have that

$$p(y|x) = \sum_{j=1}^{J} \lambda_j(y_j)\, \phi(x|x_j(y)),$$

with $x_j(y)$ the nearest neighbors (i.e., the subset of training points x_i with an observation y_i more similar to y), $\lambda_j(y_j)$ a set of weights that favor closer nearest neighbors, and ϕ a Gaussian.

IV. Active Localization

The previous model offers good results for *passive* localization using an omnidirectional camera [8]. However, when using a camera mounted on a pan-and-tilt, as it is our case, the robot can decide by itself where to look to increase the certainty on its position estimation. Next, we describe a general criterion for action selection and how to implement it in our localization framework.

A. Entropy-based Action Selection

At a given moment, the robot can execute a set of actions $\{u_1, \ldots, u_n\}$. The usefulness of one of these actions u, as far as localization is concerned, can be determined examining the probability $p(x_{t+1}|u, y_{t+1})$. An ideal action would allow the robot to find out its position without any doubt, that is, would produce a probability $p(x_{t+1}|u, y_{t+1})$ with a single peak, hopefully centered at the correct position of the camera. Such a probability distribution would have a very low entropy $H(u, y_{t+1})$ defined as:

$$H(u, y_{t+1}) = -\int p(x_{t+1}|u, y_{t+1}) \log p(x_{t+1}|u, y_{t+1})\, dx_{t+1}$$

To compute the entropy of a given action u, we integrate over all possible observations

$$H(u) = \int H(u, y_{t+1})\, p(y_{t+1}|u)\, dy_{t+1}.$$

The posterior involved in $H(u, y_{t+1})$ can be written as

$$p(x_{t+1}|u, y_{t+1}) = \frac{p(y_{t+1}|x_{t+1})\, p(x_{t+1}|u)}{p(y_{t+1}|u)}.$$

Consequently, the entropy for a given action becomes

$$H(u) = -\int\int p(y_{t+1}|x_{t+1})\, p(x_{t+1}|u) \log \frac{p(y_{t+1}|x_{t+1})\, p(x_{t+1}|u)}{p(y_{t+1}|u)}\, dx_{t+1}\, dy_{t+1}. \quad (4)$$

At any moment, the action u to be executed next is the one with lower $H(u)$ since, the lower the entropy $H(u)$, the more informative the action u is likely to be.

B. Implementation

The entropy-based action selection formalism just described is quite general and similar to that described, for instance, in [4]. However, the localization framework presented in section III allows an efficient implementation of this action-selection theory.

The basic idea is to exploit the particle filter and the appearance-based training set to discretize the double integral of equation 4.

First, we discretize the probability $p(x_{t+1}|u)$. Using equation 3 we have that

$$p(x_{t+1}|u) = \sum_{i=1}^{I} \pi_t^i\, p(x_{t+1}|u, x_t^i).$$

In the absence of any other information (i.e., new observations) the probability on the position of the camera at time $t+1$, after executing action u, ($p(x_{t+1}|u)$) can be approximated applying the action model $p(x_t|u_t, x_{t-1})$ to each one of the particles approximating the current position of the camera. In general, this results in a shift and a blur of the set of particles. In the particular case in which we only move the pan-and-tilt device, particles only have to be shifted since pan-and-tilt movements do not add error on the position of the camera. We denote the state for particle i at time t after applying action u as $x_t^i(u)$. So, using the shifted (and possibly blurred) particles according to action u, we have that

$$p(x_{t+1}|u) = \sum_{i=1}^{I} \pi_t^i\, \delta(x_{t+1}|x_t^i(u)),$$

and we can re-write equation 4 as

$$H(u) = -\int \sum_{i=1}^{I} \pi_t^i\, p(y_{t+1}|x_t^i(u)) \log \frac{\pi_t^i\, p(y_{t+1}|x_t^i(u))}{p(y_{t+1}|u)}\, dy_{t+1}.$$

Now, we have to discretize the integration over the observations.

The set of states $x_t^i(u)$ is a sample on the possible placements (including position and orientation) of the camera after executing action u. The image observed at each one of these placements can be inferred using the training set: the observation for each position $x_t^i(u)$ would be similar to the observation y obtained in the training point x that is as close as possible to $x_t^i(u)$. We take the set of views (Y_u) obtained in this way from all states $x_t^i(u)$ as a representative sample on the possible views after executing action u.

If the training set is sampled on a uniform grid over the space of configuration of the robot, finding the closest training point to a given state $x_t^i(u)$ is straightforward and can be done in constant time. If this is not the case, a

$h^* \leftarrow \infty$
For each candidate action u
  $Y_u \leftarrow \emptyset$
  For each particle (π_t^i, x_t^i)
    $(x', y') \in T$ with minimum $\|x' - x_t^i(u)\|$
    If $y' \in Y_u$ **then**
      $p(y'|u) \leftarrow p(y'|u) + \pi_t^i$
    else
      $p(y'|u) \leftarrow \pi_t^i$
    Endif
    $Y_u \leftarrow Y_u \cup \{y'\}$
  Endfor
  $h \leftarrow 0$
  For each $y \in Y_u$
    For each particle (π_t^i, x_t^i)
      $g \leftarrow \pi_t^i \sum_{j=1}^{J} \lambda_j(y)\, \phi(x_t^i(u)|x_j(y))$
      $h \leftarrow h - g \log(g/p(y|u))$
    Endfor
  Endfor
  If $h < h^*$ **then**
    $h^* \leftarrow h$
    $u^* \leftarrow u$
  Endif
Endfor
Execute u^*

Fig. 1. Algorithm for entropy-based action selection.

KD-three structure can be used to perform this search in logarithmic time in the number of training points (N).

With the above, we achieve a discretization on the possible observations. Now, for each one of the observations y included in Y_u we have to define $p(y|x_t^i(u))$. This can be done, as in section III-B, using a nearest-neighbor approach. So,

$$p(y|x_t^i(u)) = \sum_{j=1}^{J} \lambda_j(y)\, \phi(x_t^i(u)|x_j(y)), \qquad (5)$$

for $x_j(y)$ the J training points with observations more similar to y. Observe that, we only compute equation 5 for images stored in the training set. Consequently, the process of finding the nearest neighbors $x_j(y)$ and the corresponding weights $\lambda_j(y)$ can be pre-computed saving a large amount of time in the on-line execution of the entropy evaluation algorithm.

Finally, we define for any $y \in Y_u$

$$p(y|u) = \sum_{k=1}^{K} \pi_t^{i_k}$$

with $\{i_1, \ldots, i_k\}$ the set of particles that advocate for observation y. In a situation where particles are spread all over the configuration space of the robot, each particle is likely to propose a different observation y. However, in case where particles concentrate in few clusters, particles are similar each other and, thus, many particles can propose the same observation to be included into Y_u.

With the above approximations, the entropy-based evaluation of an action u becomes

$$H(u) = -\sum_{y \in Y_u} \sum_{i=1}^{I} \left[\pi_t^i \sum_{j=1}^{J} \lambda_j(y)\, \phi(x_t^i(u)|x_j(y)) \log \frac{\pi_t^i \sum_{j=1}^{J} \lambda_j(y)\, \phi(x_t^i(u)|x_j(y))}{\sum_{k=1}^{K} \pi_t^{i_k}} \right].$$

The algorithm to evaluate this equation is shown on figure 1. The cost of this algorithm is $O(U I^2 J)$ with U the number of actions considered, I the number of particles, and J the number of nearest neighbors used to compute the sensor model.

To speed up this procedure, we can replace the point $x_t^i(u)$ by its closest point in the training set (x'). In this way, equation 5 can be fully pre-computed and the cost reduces to $O(U I^2)$.

The only thing that remains to decide is when to use the action selection procedure just described. The particle filter allow us to devise a simple criterion for that since particles not only estimate the robot's position but also provide an estimation on the localization error: the variance of the particle centers. Thus, when this variance grows above a threshold, we trigger the action selection procedure to reduce as fast as possible the localization error.

V. Experiments and Results

We test our localization system in an office environment. The training set include images taken rotating the camera every 15 degrees in 16 positions arranged in a grid 2×8 with a resolution of about 75 cm. This makes a total amount of 352 training images[2]. The short distance between training points and the fact we use lenses with a wide field of view (90 degrees) make images taken at close positions/orientations to look very similar increasing the difficulty of the localization task. In the experiments, we compress the images using PCA keeping 5 feature detectors that preserve up to 70% of the variance of the original images, we use 10 nearest neighbors to approximate $p(y|x)$, and the initial distribution $p(x_0)$ is defined uniformly over the configuration space of the robot. At run-time, we considered 22 different actions (with the same orientations as those used to get the training set) and up to 150 particles. Despite these large figures (that are oversizes not only for our small test environment

[2]The range of movements of our pan device is $[-154, 154]$ degrees and, so, we get 22 images per $X-Y$ position.

Fig. 2. An image taken in a testing position (top) and the image taken at the closest training point (down).

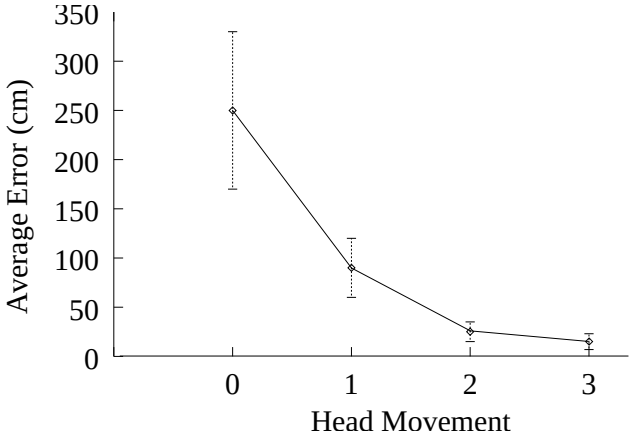

Fig. 3. Evolution of the average error (and the standard deviation) w.r.t. the correct position as we get new images.

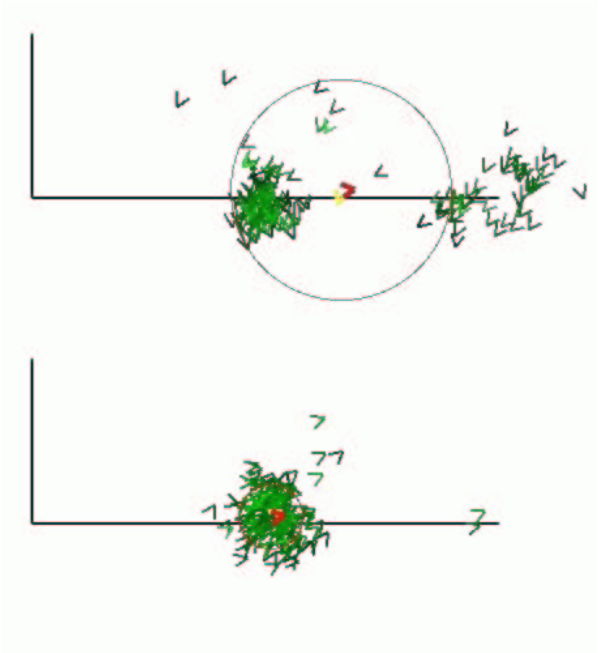

Fig. 4. Convergence toward the correct position estimation: particles around different hypotheses (top) and particles around a single correct hypothesis (down).

but also for localization in relatively large environments), there was no problem to compute the entropy-based action evaluation on-line (the entropy evaluation for all actions takes less than 0.2 seconds in a Pentium *IV* at 1.8GHz).

The test were performed placing the camera at a position not included in the training set. The difference between images at a testing point and images at the closest training point could be appreciated in figure 2: although the main elements of the scene are similar (the wall with the poster, etc), items on the desk changed from the moment the training set was collected to the moment the testing was performed. These differences are not so large for other camera orientations and this is why rotating the camera helps to improve the localization of the robot. Figure 3 shows the decrease on the average position error measured as $\|c-a\|$ with c the correct position and a the position estimated by the particle filter[3]. The results shown correspond to the average (and the standard deviation) over ten runs placing the camera in two different testing positions. All distance and, thus, the error are expressed in centimeters. We can see that the entropy-based action selection allows a fast reduction of the localization error as the head is moved and new images are processed. If we consider the estimation a to be correct if the closest training point to a is the same as the closest training point to to the correct position c, then the success ratio in localization after 3 camera movements is over 95%.

Figure 4 shows a typical evolution of particles from a distribution around different hypothesis (top) to the convergence around the correct position (down) achieved as new images are processed. In the figure, each green symbol represents a particle, the yellow one represents the state (position/orientation) of the camera, and the red one represents the state of the robot. The circle represents the standard deviation of particles in the $X-Y$ dimensions.

[3]Computed as the average of all particles taking into account the weight for each particle.

VI. Conclusions

We have presented a general framework for active localization and we have described how it can be included in our localization system resulting in an efficient action selection procedure. The experiments we report show that this mechanism effectively helps to find out the location of the robot. This can be of great help in dynamic environments, where our previous passive localization system exhibited some problems.

The main assumption behind our approach is the existence of a training set obtained off-line and densely sampled over the space where the robot is expected to move. Without this dense training set, the approximations made on section IV-B would be no longer valid. In general, obtaining this training set is not a problem, but it would be desirable the robot to build it on-line. With this extension, our system would become a SLAM (Simultaneous Localization and Mapping) system. To achieve this improvement, we have to explore the use of incremental PCA techniques for compressing the images that are obtained on-line.

Since our robot is equipped with a stereo vision system, another research line we want to explore is the use of depth maps for localization. This would make the localization more robust since depth maps are less sensitive to changes of illumination than plain intensity images. The use of other sensors such as sonars or laser scanners could also help to increase the robustness of the system although, as already mentioned, they provide less information than that provided by the images.

VII. Acknowledgments

This work has been partially supported by the European (ITEA) project *"Ambience: Context Aware Environments for Ambient Services"*.

VIII. References

[1] T. Arbel and F. P. Ferrie. Entropy-based gaze planning. In *Proceedings of the Second IEEE Workshop on Perception for Mobile Agents, in association with the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Fort Collins, Colorado*, June 1999.

[2] W. Burgard, D. Fox, and S. Thrun. Active mobile robot localization. In *15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, 1997.

[3] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, June 1999.

[4] D. Fox, W. Burgard, and S. Thrun. Active Markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(3-4):195–207, 1998.

[5] L. P. Kaelbling, A. R. Cassandra, and J. A. Kurien. Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1996.

[6] J. Kleinberg. The localization problem for mobile robots. In *Proceedings of the 35th IEEE Symposium on Foundation of Computer Science*, 1994.

[7] B. J. A. Kröse and R.Bunschoten. Probabilistic localization by appearance models and active vision. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2255–2260, 1999.

[8] B.J.A. Kröse, N. Vlassis, R. Bunschoten, and Y. Motomura. A probabilistic model for appearance-based robot localization. *Image and Vision Computing*, 19(6):381–391, April 2001.

[9] S. Maeda, Y. Kuno, and Y. Shirai. Active vision based on eigenspace analysis. In *Proceedings of the IROS conference*, pages 1018–1023, 1997.

[10] H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[11] M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *J. Amer. Statist. Assoc.*, 94(446):590–599, June 1999.

[12] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995.

[13] N. Vlassis, B. Terwijn, and B. J. A. Kröse. Auxiliary particle filter robot localization from high-dimensional sensor observations. In *Proceedings of the IEEE International Conference on Robotics and Automation, Washington D.C.*, May 2002.

[14] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman & Hall, London, 1995.