VISUAL NAVIGATION USING THE MONTE CARLO ALGORITHM

A. Gil, A. Vicente, O. Reinoso, L. Payá, R. Puerto
Industrial Systems Engineering Department
Universidad Miguel Hernández
Elche, Alicante (Spain)
arturo.gil@umh.es

Abstract

In order to be fully autonomous, a mobile robot must possess the skill of finding its location in a particular environment. In other words, while navigating, a mobile robot must be capable of finding its location in a map of the environment (i.e. its pose < x, y, θ >), otherwise the robot will not be able to complete its task. The problem becomes specially challenging if the robot does not possess any external measure of its global position. Naively, the position/orientation of the robot can be determined using odometry sensors or inertial systems. However, these sensors lack of accuracy when used for long periods of time, due to wheel slippage, drifts and other problems. Localization techniques are used instead, in order to find the position of the mobile agent in the space. The great majority of localization methods rely on finding salient characteristics sensed by the robot and relating them with a map of the environment. These salient characteristics of space are usually called landmarks, and constitute the basis that enables the robot to deduce its location in a previously built map of the environment. In this paper we present a localization method based on the Monte Carlo algorithm.

Keywords:
Mobile robotics, Map Building, SLAM, computer vision.

1. INTRODUCTION

A mobile robot must be capable of navigating through a given environment while it tries to accomplish a particular task. Frequently, the space traversed by the robot will be unstructured and with objects or people moving around. First, to navigate through the space, the mobile robot must plan a trajectory starting at a point *A* in space and ending at point *B*. To plan a path, the robot must know the structure of the environment, that is, a map. Second, the robot needs to follow this trajectory (path). While the robots moves along the planned path, it needs to know its position and orientation. Naively, the position and orientation can be obtained using dead-reckoning, that is, using odometry sensors. However, these sensors lack of accuracy when used for long time periods, due to wheel slippage, drifting and others.

Thus, the mobile robot cannot rely uniquely on its odometry information to find its situation in the environment, instead, it must use a Localization technique: It must use the information provided by its sensors to observe the space that surrounds it and relate those observations with a map of the environment. In a general way, the process of estimating the pose of a mobile agent requires the extraction of distinctive characteristics of the environment, usually called landmarks. Different kind of sensors can be used for sensing the space surrounding the robot, being SONAR, laser rangefinders and cameras the most common. It is worth mentioning the early work shown in [1], where SONAR data is used to extract salient features from the environment (mainly vertical structures, i.e. corners). Vision sensors have been used by different groups for localization tasks. For example, in [2] Neira et al. use the information provided by a CCD camera and a laser rangefinder and then extract corners and walls from the space surrounding the robot. Using an EKF, those characteristics are matched with a previously built map of the environment, thus permitting the estimation of the robot's location.

Olson [3], proposes the use of salient points in stereo images extracted using the Förstner interest operator. Afterwards, the 3D position of each point is calculated and an ego-motion measure is estimated by matching the points accross successive images. This approach reduces significantly the error in tracking robot's position, however, it does not provide a solution for the global localization problem. In the work presented in [4] and [5] SIFT features are extracted from images from the environment, working as visual landmarks.

A different problem arises when a map of the environment is not available. This represents a much harder problem: The robot must find its location while, simultaneously, builds the map. This concept is often called

Simultaneous Localization and Mapping (SLAM). In [6] and [7], a Kalman Filter approach to the SLAM problem is shown.

Section 2 explains the use of SIFT features. Next, in section 3 we will explain the basics of Monte Carlo localization. Section 4 describes the experimental setup used to test the MCL algorithm together with the use of visual SIFT features. Finally, in section 5 we analyze the main results that obtained and propose future work related to our investigation.

2. SIFT FEATURES

SIFT (Scale Invariant Feature Transform) features were developed by Lowe for image feature generation, and used initially in object recognition applications (See [8] and [9] for details). Lately, SIFT features have been used in robotic applications ([4], [5]), showing its suitability for localization and SLAM tasks. The features are invariant to image translation, scaling, rotation and partially invariant to illumination changes and affine projection. Thus, this enables the same point in space to be viewed from different poses of the robot, which may occur while the robot moves around its workplace, thus providing information for the localization process.

SIFT features are located at maxima and minima of a difference of Gaussian function applied in scale space. They can be computed by building an image pyramid with resampling between each level. The input image is first convolved with a Gaussian function of $\sigma = \sqrt{2}$, resulting in image A. Next, the image is further convolved with a Gaussian function, yielding image B. SIFT locations are extracted as maxima and minima from the image $C=A-B$. SIFT locations extracted by this procedure can be understood as significant points in space that are highly distinctive. The next step needed is to describe that point in space, so that the robot can be capable of recognising it in a later stage, while it navigates through the environment. One simple solution would be to sample the image around the key location and store the values in a matrix. Then, a correlation measure could be used in order to identify the feature. However, this descriptor is very sensitive to illumination and 3D viewpoint changes, hence this solution does not produce valid results. In our application, we used a descriptor similar to the one proposed in [9], based on local image gradients, which behaves correctly with illumination and viewpoint changes. Once the SIFT location is calculated, we assign an orientation to each feature, based on local image properties. By doing this we can represent the descriptor relative to this orientation, thus achieving invariance to image rotation.

3. MOBILE ROBOT LOCALIZATION

In robot localization we are interested in estimating the pose of the vehicle (typically, the state $\xi=(x, y, \theta)$) using a set of measurements $Z_k=\{z_k, i=1...k\}$ from the environment and a set of actions u_k performed. This can be stated in a probabilistic way, that is: Localization aims at estimating a belief function $p(\xi)$ over the space of all possible poses, conditioned on all data available until time k, that is: $p(\xi_k \mid Z_k)$. The estimation process is usually done in two phases, which are repeated recursively:

- Prediction phase: In this phase, a motion model is used to calculate the probability density function (PDF) $p(\xi_k \mid Z_{k-1})$, taking only motion into account. Usually it is assumed that the current state x_k is only dependent on the previous state ξ_{k-1} and a control input u_k. The motion model is specified in the form of the conditional density: $p(\xi_k \mid \xi_{k-1}, u_{k-1})$. The prediction is then obtained by integration:

$$p(\xi_k \mid Z_{k-1}) = \int p(\xi_k \mid \xi_{k-1}, u_{k-1}) p(\xi_{k-1} \mid Z_{k-1}) d\xi_{k-1} \quad (1)$$

- Update phase: In the second phase, a measurement model is used to incorporate information from the sensors and obtain the posterior PDF $p(\xi_k \mid Z_k)$. The measurement model is given in terms of a probability $p(z_k \mid \xi_k)$ which provides the likelihood of the state ξ_k supposing that a particular measurement z_k was observed. The posterior density $p(\xi_k \mid Z_k)$ can be calculated using Bayes' Theorem as follows:

$$p(\xi_k \mid Z_k) = \frac{p(z_k \mid \xi_k) p(\xi_k \mid Z_{k-1})}{p(z_k \mid Z_{k-1})} \quad (2)$$

Knowledge about the initial state at time t_0 is represented by $p(\xi_0)$. In the case of global localization, where the pose of the vehicle is totally unknown, $p(\xi_0)$ is represented by a constant function over the space of all possible poses, indicating that there is no previous knowledge about robot's position. Note that in expressions (1) and (2) nothing is said about the representation of the PDF. This fact leads to a series of different algorithms that are based on the above prediction-update scheme, mainly: The Kalman filter, Markov grid-based localization and Monte Carlo localization.

3.1 Kalman filter

In the Kalman filter approach both the measurement model and the motion model are described using a Gaussian density. In this case, the expressions (1) and (2) can be evaluated in a closed form, yielding the classical Kalman filter. The Kalman filter has been used with success in some applications (i.e. [1] and [2]), proving to be robust and accurate while keeping track of the robot's pose. However, Kalman Filters do not correctly handle non-Gaussian distributions, which may appear in the measurement and motion models. On the other hand, Kalman filter does only manage uni-modal distributions, in consequence, it cannot cope with possible ambiguities in landmark observation and the global localization problem.

3.2 Markov Grid-Based localization

The PDF $p(\xi_k \mid Z_k)$ can be aproximated using a discretization of the state space. Thus, the equations (1) and (2) can be performed by a numerical integration over a grid of points. This forms the basis of the Markov grid-based localization approach. This method has been employed with success in dynamic environments (i.e. [10]). Nevertheless, if the state space is large, this method can suffer from computational overload. This fact limits the resolution which can be used to represent space, thus it settles the maximum accuracy of the algorithm.

3.3 Monte Carlo localization

Monte Carlo localization can be included in a set of algorithms called particle filters, which have had a great development during last decade (e.g. [11] and [12]). In Monte Carlo localization (MCL for short), the PDF $p(\xi_k)$ is represented by a set of M random samples $\chi_k = \left\{ \xi_k^i, i = 1 \ldots M \right\}$ extracted from it. Each particle can be understood as a hypothesis of the true state of the robot (i.e. its pose $\xi = (x, y, \theta)$). The algorithm is calculated in a prediction-update fashion, implementing equations (1) and (2) as stated before. The prediction-update phases are repeated recursively. To localize the vehicle globally, the initial set of particles is spreaded randomly over the entire state space. See [12] and [11] for details.

- Prediction phase: A set of particles χ_k is generated based on the set of particles χ_{k-1} and a control signal u_k. This step uses the motion model $p(\xi_k \mid \xi_{k-1}, u_{k-1})$ and applies it to every particle in set χ_k. As a result, a new set of particles χ'_k is generated, which represents the density $p(\xi_k \mid Z_{k-1})$.
- Update phase: In this second phase, we take into account an observation z_k made by the robot. For each particle in the set, a weight ω_k^i is computed (frequently called Importance Factor). This weight is calculated using the observation model $\omega_k^i = p(z_k \mid \xi^{ui}_k)$ resulting in the set $\overline{\chi}_k = \left\{ \xi_k^i, \omega_k^i \right\}$. Finally the resulting set χ_k is calculated by resampling with replacement from the set $\overline{\chi}_k$, where the probability of resampling each particle is given by its importance weight ω_k^i. Finally, the set χ_k represents the distribution $p(\xi_k \mid Z_k)$.

4. VISUAL NAVIGATION

In this section we describe our approach to robot localization. Mainly, it is based on the Monte Carlo algorithm and uses SIFT features as landmarks. A B21r robot equipped with a calibrated stereo head was used for the experiments, shown in Fig. 1 (a). The environment is characterized for being frequently traversed by people. The experiment can be divided in two phases: A) Environment exploration and map creation, and B) Localization.

4.1 Environment exploration and map creation

The purpose of this step is to create a map of the environment. This map is constituted by the interesting points extracted from the images taken. Each interesting point (landmark) is characterized by a SIFT descriptor, as stated in section 1. In this first phase, the robot was commanded to move along the environment, varying its

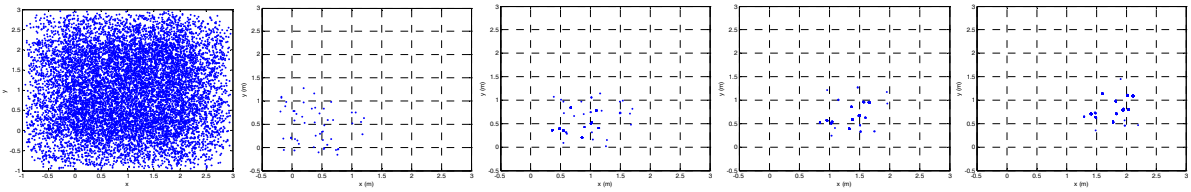
position and orientation. Simultaneously, images were captured with both cameras and processed to extract SIFT features. Next, features extracted in the left image were matched with the ones found in right image. The following restrictions were applied during this process:

- Epipolarity restriction: The feature location in the right image must be placed in the same row as the in the left image. In practice, this condition was relaxed, permitting a maximum ±2 pixel displacement.
- SIFT restriction: The euclidean distance between two SIFT descriptors must not surpass a certain threshold (determined experimentally).
- Maximum and minimum disparity restriction: The disparity *d* between two corresponding features must be in a specified interval. This restriction avoids finding false correspondences and accelerates the matching process. In addition, the maximum and minimum distance where a SIFT feature can be placed is limited too, thus avoiding large errors in distance calculation.

In Fig. 1 (b) and (c) two stereo images from the environment are shown. Aside, in Fig. 1 (d) and (e), the SIFT features are extracted and a matching between corresponding points is made. Note that some features found in one image appear in the other. However, some features can only be found in one image and consequently must not be matched. Each time a SIFT feature is matched correctly in both images, its position relative to the robot is calculated using stereo vision. In addition, the position of the SIFT feature in space is determined relative to a global frame. Besides, in order to minimize the error in robot's odometry, a tracking procedure of the landmarks was used (similar to the exposed in [4]). The information gathered is stored in a database, which constitutes the map.

a) b) c) d) e)

Fig. 1. Image (a) shows our B21r robot. Figures (b) and (c) show two images taken from our environment. Figures (d) and (e) show SIFT features extracted from them.



a) b) c) d) e)

Fig. 2. A global localization process with the Monte Carlo algorithm.

4.2 Visual localization

When the robot needs to find its pose in the environment, it captures images with its cameras, processes them and finds SIFT features. Next, a matching procedure between images is taken, in order to find the relative position of the feature. Afterwards, the robot tries to match any feature found with its correspondence in the database using the euclidean distance between SIFT descriptors. Some issues appear during this process:

- The correspondence between the observed feature and a stored landmark is not straight forward. The robot may confuse one landmark for another, thus providing erroneous information for the localization algorithm. The MCL algorithm behaves robustly against data association errors and permits the robot to be localized correctly, as long as some SIFT features are correctly identified.
- The use of visual landmarks allows us to localize the robot in populated environments. It is not necessary for the robot to visualize all SIFT points in the scene.

The relative distance from the robot to a SIFT feature in the map permits us to select, with high probability, some areas of the map where the robot could be, and reject some others. That is implemented by the sensor model $p(z_k \mid \xi_k)$, which is used by the MCL algorithm described in section 3. In Fig. 2 a global localization process is shown. First, in Fig. 2 (a) a random set of particles is spreaded over the entire space state (we show

only the (x, y) components for clarity). In the following figures, a series of prediction-update phases are shown. Finally, in Fig. 2 (e) the particles gather around the last robot position, hence localizing it. Odometry can be used as a good measure of the robot's pose when used in short displacements, thus, we have compared it with the result of the MCL estimation. Table 1 shows a comparison between odometry data and the results obtained with the Monte Carlo algorithm. The algorithm was tested for using several trajectories through the environment. We obtained similar results to the ones showed, achieving to localize the robot quite accurately in a few steps.

Table 1: Odometry vs MCL comparison.

	Odometry			**MCL**		
	x (m)	y (m)	θ (rad)	x (m)	y (m)	θ (rad)
Fig. 2 (b)	0.307	0.342	0.403	0.275	0.367	0.397
Fig. 2 (c)	0.802	0.541	0.403	0.974	0.608	0.391
Fig. 2 (d)	1.268	0.732	0.403	1.347	0.802	0.394
Fig. 2 (e)	1.714	0.885	0.403	1.753	0.923	0.394

5. DISCUSSION AND FUTURE WORK

This paper describes a localization method based on the Monte Carlo algorithm in combination with visual landmarks. In particular, SIFT features have been used as visual landmarks, finding them suitable for the global localization problem. Our approach has been implemented on a mobile platform and tested in a real environment. Good results have been achieved, proving the effectiveness of our solution. However, we plan to further test the algorithm for longer periods of time. During our experiments, we found that some SIFT features found in the environment lacked of stability: They were found from a robot's pose, but could not be detected from elsewhere. To solve this, we plan to track features for consecutive frames, hence ensuring that the feature found is stable and can be detected from different viewpoints.

6. AKNOWLEDGEMENT

This research is sponsored by the spanish Ministerio de Educación y Ciencia (Project reference: DPI2004-07433-C02-01. Title: HERRAMIENTAS DE TELEOPERACION COLABORATIVA. APLICACION AL CONTROL COOPERATIVO DE ROBOTS).

References

[1] J. J. Leonard, H. F. Durrant-Whyte, “Mobile Robot Localization by Tracking Geometric Beacons”. *IEEE Transactions on Robotics and Automation*, vol. 7 no. 3, 1991.

[2] J. Neira, J. D. Tardós, J. Horn, G. Schmidt, “Fusing Range and Intensity Images for Mobile Robot Localization”. *IEEE Transactions on Robotics and Automation*, vol 15, no 1, pp 76-83, Febrero 1999.

[3] C. F. Olson, L. H. Matthies, M. Schoppers, M. W. Maimone,“Rover Navigation using stereo ego-motion”. *Robotics and Autonomous Systems*, no. 43, pp. 215-229, 2003.

[4] S. Se, D. Lowe, J. Little, “Vision-based Mobile Robot Localization and Mapping using Scale-Invariant Features”, *Proceedings of the IEEE International Conference on Robotics and Automation* (ICRA), pp. 2051-2058, Seul, Mayo 2001.

[5] S. Se, D. G. Lowe, “Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks”. *The International Journal of Robotics Research*, vol. 21, no. 8, 2002.

[6] M. W. M. Gamini Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, M. Csorba, ”A solution to the Simultaneous Localization and Map Building (SLAM) Problem”. *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, 2001.

[7] J. E. Guivant, F. R. Masson, E. M. Nebot, “Simultaneous localization and map building using natural features and absolute information”. *Robotics and Autonomous Systems*, no. 40, pp. 79-90, 2002.

[8] D. G. Lowe, “Object Recognition from Local Scale-Invariant Features”. *International Conference on Computer Vision*, vol. 2, 1999.

[9] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints”, *International Journal of Computer Vision*, no. 60, vol. 2, pp. 91-110, 2004.

[10] W. Burgard, D. Fox, D. Henning, “Fast Grid-based Position Tracking for Mobile Robots”, *Proceedings of the 21th German Conference on Artificial Intelligence* (KI-97), Freiburg, Germany, 1997.

[11] F. Dellaert, D. Fox, W. Burgard, S. Thrun, “Monte Carlo Localization for Mobile Robots”. *IEEE International Conference on Robotics and Automation* (ICRA99), 1999.

[12] S. Thrun, D. Fox, W. Burgard, F. Dellaert, “Robust Monte Carlo Localization for Mobile Robots”. *Artificial Intelligence*, vol 128, nº 1-2, pp 99-141, 2000.