Monte Carlo Localization Using SIFT Features

Arturo Gil, Óscar Reinoso, Asunción Vicente, César Fernández, and Luis Payá

Área de Ingeniería de Sistemas y Automática
Universidad Miguel Hernández
Avda. del Ferrocarril s/n
03202 Elche (Alicante) SPAIN
arturo.gil@umh.es
WWW home page: http://lorca.umh.es/isa/es

Abstract. The ability of finding its situation in a given environment is crucial for an autonomous agent. While navigating through a space, a mobile robot must be capable of finding its location in a map of the environment (i.e. its pose $< x,\ y,\ \theta >$), otherwise, the robot will not be able to complete its task. This problem becomes specially challenging if the robot does not possess any external measure of its global position. Typically, dead-reckoning systems do fail in the estimation of robot's pose when working for long periods of time. In this paper we present a localization method based on the Monte Carlo algorithm. During the last decade this method has been extensively tested in the field of mobile Robotics, proving to be both robust and efficient. On the other hand, our approach takes advantage from the use of a vision sensor. In particular, we have chosen to use SIFT features as visual landmarks finding them suitable for the global localization of a mobile robot. We have succesfully tested our approach in a B21r mobile robot, achieving to globally localize the robot in few iterations. The technique is suitable for office-like environments and behaves correctly in the presence of people and moving objects.

1 Introduction

The skill of navigating through an environment is a key aspect for a mobile robot. A mobile robot must be capable of travelling from a starting point in space, say A to a final point B. Frequently, the space traversed by the robot will be unstructured, changing and with people moving around. First, the mobile agent must plan a trajectory that starts at point A and ends at point B. The set of algorithms that solve this problem are often referred as path planning techniques. These techniques use a map of the environment to find the best path that reaches a particular destination from a given start location. While the robot moves along the planned path it needs to know its position/orientation, otherwise the mobile agent would not be able to follow it. Naively, the position/orientation of the robot can be determined using dead-reckoning, that is, using odometry sensors. However, these sensors lack of accuracy when used for long periods of time, due

to wheel slippage, drifts and other problems. GPS and inertial systems offer an alternative to dead reckoning, but have a drawback: Often mobile robots working in indoor environments cannot receive the GPS signal properly. Localization techniques are used instead, in order to find the position of the mobile agent in space. Most of them try to match salient characteristics of the space sensed by the robot with the same characteristics in the map, thus localizing the robot. For example, in [1] a CCD camera and a laser rangefinder are used to find vertical and horizontal structures in the space that surrounds the robot (i.e. corners and walls). Those structures and then matched against a map of the environment. As a result, the robot can localize itself in the map.

Our approach to localization is based on the Monte Carlo algorithm. During the last decade this method has been extensively tested in the field of mobile Robotics, proving to be both robust and efficient. On the other hand, our work is based on a stereo vision system, which allows us to compute the relative distance of the robot to significant points of the space. These significant points of space are usually called landmarks, and are the basis that enables the robot to deduce its location in a previously built map of the environment. In particular, we have chosen to use SIFT features as visual landmarks finding them suitable for the global localization of a mobile robot.

In section 2 we relate our work with previous implementations. Section 3 explains the use of SIFT features, which have been used previously in robotics applications, such as [2] and [3]. Next, in section 4 we will explain the basics of Monte Carlo localization. The integration of SIFT features in Monte Carlo localization will be also explained in section 4. Section 5 describes the experimental setup used to test the MCL algorithm together with the use of visual SIFT features. Finally, in section 6 we analyze the main results that we have obtained and propose future work related to our investigation.

2 Relation to prior work

As stated previously, a mobile robot must not rely uniquely on its odometry information to estimate its position: 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.

Vision sensors have been used by different groups for localization tasks. For example, in [4] Neira et al. use the information provided by a CCD camera and a laser rangefinder and then extract the significative characteristics 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 [5], proposes the use of salient points in stereo images extracted using the Förstner interest operator. Afterwards, the 3D position of each point is calculated 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 [2] and [3] stereo vision is used to track 3D visual land-

marks extracted from an unstructured environment, in particular, SIFT features are used as visual landmarks. During an exploration phase, the robot extracts the SIFT features from stereo images (actually a trinocular stereo system), calculating their 3D position in space, and stores them in a database (which conforms the map). Later, when the robot is navigating, the stored SIFT features are found in the environment and the relative distance to them is computed. Finally, its pose is calculated by means of a Hough transform.

In our work, we take an approach similar to that of [2] and [3]. That is, we extract SIFT landmarks from the environment and store them in a database. SIFT landmarks are characterized using a descriptor. This means that the same landmark can be recognised by the robot when it navigates through the environment, hence, this enables the use of SIFT features for the global localization problem. However, our localization algorithm differs greatly from the one cited previously. Indeed, we use a particle filter approach inspired in the work exposed in [6], [7] and [8], which has proved to be both fast and robust.

3 Sift Features

SIFT (Scale Invariant Feature Transform) features were developed by Lowe for image feature generation, and used initially in object recognition applications (See [9] and [10] for details). Lately, SIFT features have been used in robotic applications ([2], [3]), 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. SIFT locations are extracted by means of successive filtering. 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 $C = A - B$.

The 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 [10], 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 variance to image rotation.

4 Monte Carlo SIFT localization

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

Prediction phase In this phase, a motion model is used to calculate the probability density function (PDF) $p(\mathbf{x}_k \mid Z^{k-1})$, taking only motion into account. Usually it is assumed that the current state $\mathbf{x}_k$ is only dependent on the previous state $\mathbf{x}_{k-1}$ and a control input $\mathbf{u}_{k-1}$. The motion model is specified in the form of the conditional density: $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1})$. The prediction is then obtained by integration:

$$p(\mathbf{x}_k|Z^{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) p(\mathbf{x}_{k-1}|Z^{k-1}) d\mathbf{x}_{k-1} \tag{1}$$

Update phase In the second phase, a measurement model is used to incorporate information from the sensors and obtain the posterior PDF $p(\mathbf{x}_k|Z^k)$. The measurement model is given in terms of a probability $p(\mathbf{z}_k \mid \mathbf{x}_k)$ which provides the likelihood of the state $\mathbf{x}_k$ supposing that a particular measurement $\mathbf{z}_k$ was observed. The posterior density $p(\mathbf{x}_k|Z^k)$ can be calculated using Bayes' Theorem as follows:

$$p(\mathbf{x}_k|Z^k) = \frac{p(\mathbf{z}_k \mid \mathbf{x}_k) p(\mathbf{x}_k|Z^{k-1})}{p(\mathbf{z}_k|Z^{k-1})} \tag{2}$$

The process is repeated recursively after update phase. The knowledge about the initial state at time t_0 is represented by $p(\mathbf{x}_0)$. In the case of global localization, where the pose of the vehicle is totally unknown, $p(\mathbf{x}_0)$ is represented by a constant function over the space of all possible poses.

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. The Kalman filter does not solve for the global localization. On the other hand, Markov grid-based localization requires large amounts of memory and computation time. Hence, our approach relies on the Monte Carlo localization method.

4.1 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. [7], [6] and

[11]). In Monte Carlo localization (MCL for short), the PDF $p(\mathbf{x})$ is represented by a set of M random samples $\chi_k = \{x_k^i, i = 1 \dots M\}$ extracted from it. Each particle can be understood as a hypothesis of the true state of the robot (i.e. its pose $\langle x,\ y,\ \theta \rangle$). The algorithm is calculated in a Prediction-update fashion, as stated before.

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 $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1})$ 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 $p(\mathbf{x}_k|Z^{k-1})$.

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 $\omega_k^i = p(z_k|x_k'^i)$ resulting in the set $\overline{\chi}_k = \{x_k^i, \omega_k^i\}$. 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 $p(\mathbf{x}_k|\mathbf{Z}_k)$.

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 [11], [6] and [7] for details.

5 Experimental results

In this section we report the Monte Carlo localization technique that has been tested together with SIFT features in an office-like environment. A B21r robot equipped with a calibrated stereo head was used for the experiments. In Fig. 1, an image of the environment is shown. 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.

5.1 Environment exploration and map creation

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).

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

Fig. 1. B21r robot during data acquisition in the environment.

of the SIFT feature in space is determined relative to a global frame. In order to minimize the error in robot's pose, the exploration phase was performed in several runs. Besides, in order to minimize the error in robot's odometry, a tracking procedure of the landmarks was used (similar to the exposed in [2]). The information gathered is stored in a database, which constitutes the map.

5.2 Localization

During localization, the robot navigates along the environment. Meanwhile, the robot captures images with its cameras, processes them and finds SIFT features. Next, a matching procedure 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. The euclidean distance between SIFT descriptors is used in order to find corresponding features. Some issues appear during this process:

- The correspondence between the observed feature and a stored landmark is not direct. The robot may confuse one landmark for another, thus providing erroneous information for the localization algorithm.
- 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 robot may only detect a few points and still will be capable of finding its pose in the map.

The parameters in the observation model were adjusted by experience. It is worth mentioning that the parameters in motion and observation models have a great influence in the convergence properties of MCL algorithm. A general trend is to inflate both motion and observation, which speeds up the localization process and avoids the possibility of loosing track of the robot ([6], [11]).

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(f) the particles gather around the last robot position, hence localizing it. In Table 1 we show a comparison between odometry

Table 1. Odometry vs MCL comparison

	Odometry			MCL		
	x_{rob} (m)	y_{rob} (m)	θ_{rob} (rad)	x_{rob} (m)	y_{rob} (m)	θ_{rob} (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
Fig. 2(f)	2.072	1.065	0.403	2.093	1.071	0.395

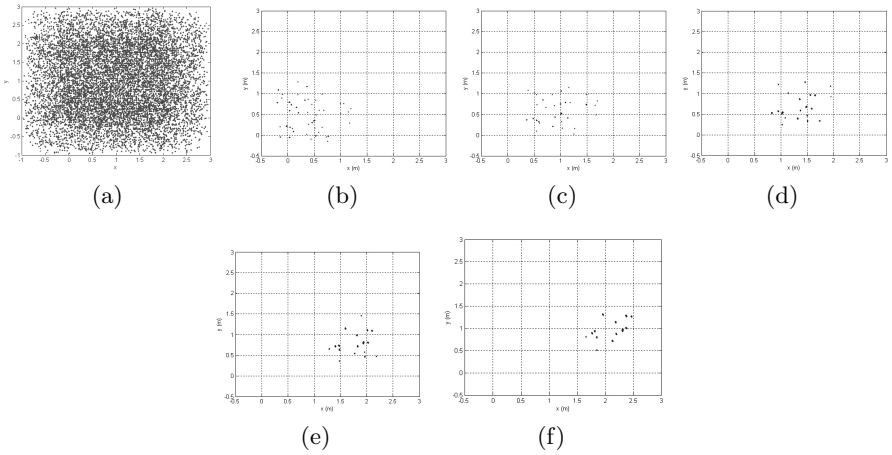

Fig. 2. MCL localization progress. Image a) shows the set of particles spreaded around the entire state space. From image and b) to f) the localization progress is presented. Finally, the robot is localized in image f).

and the MCL estimation. Odometry can be used as a good measure of the robot's pose when used in short displacements, thus permits us to compare it with the result of the MCL estimation.

The algorithm was tested during several runs through the environment. We obtained similar results to the ones showed, achieving to localize the robot quite accurately in a few steps.

6 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.

Note that, in case a sudden change in robot's position may occur (i.e. the robot hits an obstacle), the algorithm could fail in localizing the robot. If the particles concentrate around a certain position and the observation model suggests that the robot is elsewhere, then the weight ω_k^i may be zero for every particle in the set. This can be avoided by the injection of a random set of particles in each iteration of the MCL algorithm. We plan to add this feature in a future approach. On the other hand, as a future work we plan to apply a version of the MCL algorithm for multi-robot localization. We consider that the localization problem can be solved efficiently using several robots in combination with the method exposed in this paper.

References

1. J. Neira J. D. Tardós J. Castellanos, J. M. Martínez. Experiments in multisensor mobile robot localization and map building. *3rd IFAC Symposium on Intelligent Autonomous Vehicles*, 1998.
2. J. Little S. Se, D. Lowe. Vision-based mobile robot localization and mapping using scale-invariant features. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2051–2058, 2001.
3. J. Little S. Se, D. Lowe. Global localization using distinctive visual features. *Proceedings of the 2002 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems EPFL*, 2002.
4. J. Horn G. Schmidt J. Neira, J. D. Tardós. Fusing range and intensity images for mobile robot localization. *IEEE Transactions on Robotics and Automation*, 15(1):76–83, 1999.
5. M. Schoppers M. W. Maimone C. F. Olson, L. H. Matthies. Rover navigation using stereo ego-motion. *Robotics and Autonomous Systems*, (43):215–229, 2003.
6. F. Dellaert S. Thrun D. Fox, W. Burgard. Monte carlo localization: Efficient position estimation for mobile robots. *Proc. of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, 2000.
7. W. Burgard S. Thrun F. Dellaert, D. Fox. Monte carlo localization for mobile robots. *IEEE International Conference on Robotics and Automation (ICRA99)*, 1999.
8. J. Guivant H. Durrant-Whyte E. Nebot, F. Masson. Robust simultaneous localization and mapping for very large outdoor environments. *Proceedings of the 8th International symposium on Experimental Robotics (ISER '02)*, 2002.
9. D. Lowe. Object recognition from local scale-invariant features. *International Conference on Computer Vision*, 2, 1999.
10. D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2(60):91–110, 2004.
11. W. Burgard F. Dellaert S. Thrun, D. Fox. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141, 2000.