

---

# Using odometry and invariant visual features for a Monte-Carlo based robot localization method

---

**Jesús Martínez-Gómez**  
University of Castilla-La Mancha, Spain  
jesus.martinez@dsi.uclm.es

**Alejandro Jiménez-Picazo**  
University of Castilla-La Mancha, Spain  
ajimenez@dsi.uclm.es

**Ismael García-Varea**  
University of Castilla-La Mancha, Spain  
ivarea@dsi.uclm.es

**Jose A. Gámez**  
University of Castilla-La Mancha, Spain  
jgamez@dsi.uclm.es

## Abstract

This paper presents a new approach to robot localization in indoor environments. The system presents a Monte Carlo localization method using odometry for prediction step and SIFT for update step. The scope of the system is indoor environments where no artificial landmarks are necessary.

## 1 Introduction

Self-localization is one of the harder problems in mobile robot research. Intelligent robots need to perform different tasks depending on their own pose. If a robot's pose is not well estimated, the behaviour obtained will not be correct.

Different approaches have been successfully implemented to deal with real-world problems (noise, uncertainty, low-quality images, etc) [2, 3, 8, 6]. Most of these approaches combine the information obtained from two different sources: perception and odometry. Perception uses all the robot's sensors to retrieve information from the environment. Main sensors are distance sensors and vision cameras. Odometry can be used to estimate the pose of the robot using the last pose and the set of movements the robot has performed so far. Both sources (perception and odometry) are noisy and with a high uncertainty, so the combination of these sources of information must be performed appropriately.

Localization algorithms have to be developed taking into account the environment where the robot will perform its actions. Some environments have landmarks artificially added. These landmarks have specific colour and size features that make them easy to be recognized using image processing techniques. If the environment where the robot is placed can not be extended with artificial landmarks, other techniques have to be used. These techniques combine the vision information with the information obtained from other kind of sensor such as sonar, laser, GPS, digital compass, etc.

Visual processing techniques, necessary to cope with these environments, have to extract features from training frames and compare these features with that extracted from test frames in real-time. At instant  $t$ , the robot will be located close to positions related to training frames similar to test frames captured by the robot's camera at the instant  $t$ .

The approach presented here carries out localization by using odometry and visual information. Our proposed localization system follows the principles of the particle-filter-based Monte Carlo [1] method, which is applied (in our proposal) with some variations. The combination of this localization method with a robust image processing algorithm allows the robot to reduce the uncertainty about its pose when captured images have a good resolution. The algorithm keeps information about the robot's pose even when the quality of the images decreases.

Scale-invariant feature transform[4] is the technique selected for the visual processing. Odometry is performed by using the linear and angular velocity of the robot, estimated from the training frames.

## 2 Scale-Invariant Feature Transform

Scale-Invariant Feature Transform (SIFT) is a computer vision algorithm, developed to detect key features in images. The main idea of the algorithm is to apply different transformations and study the points of the image which are invariant under these transformations. These extracted points can be used to perform object recognition by carrying out a matching between images representing the same object or scenario.

Features extracted are invariant to image scale and rotation, and they are also robust to noise and changes in viewpoint. An important characteristic of systems developed to perform object recognition using SIFT is that they are robust to partial object occlusion.

The main problems of using this algorithm are the long execution time, the high demand on memory, and the lighting variations that affect the algorithm's performance in a negative way.

## 3 Monte Carlo

Monte Carlo is a probabilistic localization method based on particle filters. This method spread particles over the environment and each one of these particles represents a robot's pose  $\langle x, y, \theta \rangle$ . An iterative process performs an prediction and an update step. The prediction step applies a movement model to each one of the particles. The update step introduces the information obtained from robot sensors to the particle population. This information is used to evaluate the goodness of the particles. After all the particles have been evaluated, a sampling step is applied obtaining a new population. The particles are sampled with replacement using their goodness value as selection probability. Best candidates should be duplicated, and worst particles should disappear from the environment. After several iterations, the algorithm will converge and all the particles will be around the real robot pose.

## 4 Proposed localization method

Our proposal is to develop a particle-filter-based localization method, using SIFT features to evaluate the goodness of the particles.

This method use the information obtained from two different sources: robots sensors (mainly vision cameras and distance sensors) and robot's odometry. The experimental evaluation was performed on the IDOL2 database <sup>1</sup>. Using this database, the system can be trained using the frames captured by the robot's camera under three different lighting conditions. All training frames are labelled with the absolute pose  $\langle x, y, \theta \rangle$  from where the frame was captured.

The learning process will be performed offline, using all the available training frames. We will retrieve information from the visual information stored in the images and also from the odometry information, stored in the frame's labels.

Firstly, SIFT features are extracted from the training frames (these features are named SIFT points). Secondly, the expected robot's movement is estimated using the position and orientation information stored in the labels of the training frames. We assume that robot's movements for the test sequences will be similar to those obtained with the training sequences.

The expected robot's average velocity can be obtained from the difference between the poses of the training sequence. We obtain the average linear and angular velocity, defined in centimetres and degrees per frame.

### 4.1 Algorithm

A particle-filter-based method is defined using two main steps: prediction and update. It is necessary to define some parameters, such as the particle representation and the environment boundary. The particles represent robot poses and so, they are defined using three parameters (with continuous numeric values):  $x$  and  $y$  for position and  $\theta$  for orientation. The boundary of the environment is defined by the limits for the  $x$  and  $y$  component.

---

<sup>1</sup><http://cogvis.nada.kth.se/IDOL2/>

Particle-filter methods present some problems when the evaluation for the particles is not correctly performed. This happens with noisy or low resolution frames and with false positives. The main drawback, facing that situation, is that the system's performance decreases, increasing the error rate. Another important problem happens when most of particles converge to a wrong environment area. The system will need a high number of iterations to escape from this situation, and robot's pose will be improperly estimated during these iterations.

#### 4.1.1 Prediction Step

A movement model has to be applied to each one of the particles. This movement model, based in odometry, is applied taking into account the average linear and angular velocity, obtained from the training frames. In this step we add some uncertainty, commonly termed white noise, to the process.

#### 4.1.2 Update Step

Each particle goodness is obtained by performing a matching between the SIFT points extracted from the current test frame and particle SIFT points. All the frames included in the training frames database have a set of SIFT points associated and extracted offline. In order to obtain the SIFT points of a particle we look for the closest frame in the training frames database. We will use as the particle SIFT points those associated to the closest training frame to the pose of the particle.

The result of matching the frame link to a particle and the closest frame in the training database will be the percentage of common points between both frames. This matching is performed using a kd-tree [7] structure.

#### 4.1.3 Algorithm Stability

Some preliminary tests were performed using a standard Monte Carlo method with training and test frames acquired over different lighting conditions. These tests presented several problems related to wrong convergences, because some test frames were not represented by any training frame. This happens when there are no training frames from the pose where the test frame was taken or when matching fails (noisy frames or different lighting conditions). The consequence of both cases is that the goodness of the particles decreases continuously and they are spread over the environment. If this happens and after finding false positives, the system will converge at wrong environment positions.

In order to escape form wrong convergences, we propose to spread a new particles population. This restart should be performed only when the algorithm stability decreases. To control this situation, we estimate the stability of the algorithm by studying the particles goodness and the variation obtained for the last  $n$  pose estimations. The process will be stable if the variation obtained for the last  $n$  pose estimations is sufficiently small using particles with high goodness. This happens when most of the particles are close to the real robot's pose, without high variations between frames.

Most reliable robot position will be stored while the algorithm is stable. This position  $p$  will be used in future restarts, where particles will be spread over a restricted area. This area is defined as a circumference centred at  $p$ , which radius depends on the algorithm instability level. This level is obtained with the best particle's for the last  $n$  iterations.

If the algorithm has been unstable for a big number of iterations, this circumference will be expanded to the whole environment.

The complete processing scheme can be observed in Fig.1.

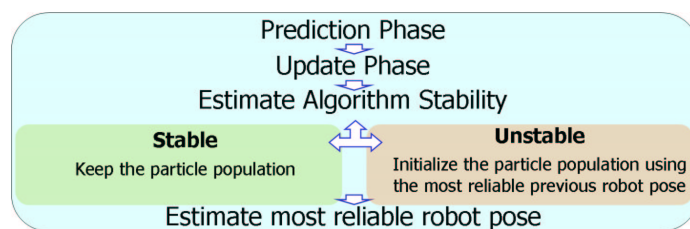


Figure 1: General Processing Scheme

## 5 Experiments and Results

All the experiments were performed with the IDOL2 [5] database. Using this database, we have to choose a set of training frames and a different set of test frames. The system was trained offline, extracting all the SIFT points from all the training frames. This process has a high demand on space and processing time, but has to be performed only one time.

In order to perform a complete system evaluation, we trained three systems using three different lighting conditions for the training frames. All these systems were tested using three test frames (acquired under three different lighting conditions). For all the experiments the training and the test frame set were different, even having the same lighting conditions. We used the proposed partition for training and test frames of the IDOL2 database.

Robot most reliable pose is obtained from the particle population at the end of each iteration. This pose is compared with the real robot's pose to obtain the error value.

Table 1 shows the mean absolute error obtained for the three pose components:  $x$ ,  $y$  and  $\theta$ . These results illustrate the key importance of the lighting conditions. Best results for sunny and night test frames were obtained using the same lighting conditions for the training sequence. Cloudy test frames were not so dependent on training illumination conditions and smallest mean absolute error was obtained using sunny training frames.

Training Seq	Test Sequences								
	Night			Cloudy			Sunny		
	x (cm)	y (cm)	$\theta$ (rad)	x (cm)	y (cm)	$\theta$ (rad)	x (cm)	y (cm)	$\theta$ (rad)
Night	<b>42.8</b>	<b>42.9</b>	<b>0.27</b>	115.6	368.7	0.91	73.5	185.3	0.85
Cloudy	69.0	196.8	0.71	84.5	146.7	0.67	62.5	155.4	0.45
Sunny	56.0	153.5	0.57	<b>72.2</b>	<b>140.4</b>	<b>0.36</b>	<b>24.3</b>	<b>21.8</b>	<b>0.27</b>

Table 1: Mean Absolute Error (MAE) for different combinations of training and tests sequences

The mean errors obtained were considered acceptable taking into account that robots tours were performed inside a 1500 x 2500 cm environment. The worst obtained results were always smaller than 7.70% for  $x$ , 14.75% for  $y$  and 14.48% for  $\theta$ , with respect to the total size of the environment.

## 6 Conclusions

According to the experimental results, our proposal becomes a robust alternative to traditional localization methods for indoor environments. It uses the principles of particle filter and SIFT to estimate the pose of the robot. The system works properly with variable lighting conditions, and, as expected, best results were obtained with unchanging lighting conditions.

## References

- [1] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE International Conference on Robotics*, 1999.
- [2] D. Fox, W. Burgard, and S. Thrun. Active markov localization for mobile robots, 1998.
- [3] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11(391-427):27, 1999.
- [4] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [5] J. Luo, A. Pronobis, B. Caputo, and P. Jensfelt. Incremental learning for place recognition in dynamic environments. In *Proc. IROS*, volume 7. Citeseer.
- [6] Rudy Negenborn. Robot localization and kalman filters. 2003.
- [7] J.T. Robinson. The KDB-tree: a search structure for large multidimensional dynamic indexes. In *Proceedings of the 1981 ACM SIGMOD international conference on Management of data*, pages 10–18. ACM New York, NY, USA, 1981.
- [8] Z. Wasik and A. Saffiotti. Robust color segmentation for the robocup domain. *Pattern Recognition, Proc. of the Int. Conf. on Pattern Recognition (ICPR)*, 2:651–654, 2002.