

# Global Robot Localization using Principal Component Analysis on Vision Data

3rd Project, CS7001  
Michael Kaess  
Sponsor: Frank Dellaert

Dec 1, 2001

## Overview

Localization is one of the main problems in autonomous robotics. Relying on the odometry of the robot usually works only for short periods of time, since small errors quickly build up to big discrepancies. A number of algorithms exist to approach this problem, using different sensors from ultrasonic over laser range finders to cameras.

Usually there is a distinction between local and global localization. The global version tries to locate a robot in a big environment, possibly without any knowledge of the robot position at all, while the local version has some information about the location (from the global version or from odometry) and tries to get a better estimate.

Localization algorithms are often based on probabilistic models. Kalman Filters are commonly used in this area, but due to their restriction to Gaussian distributions they are not suited for global localization where multimodal distributions are required. An extension without this restriction is called Markov localization, which uses a discrete, grid based version of the probability distribution. The drawback is the computational complexity for high grid resolutions, necessary for suitable accuracy in big environments.

To avoid this problem while still taking advantage of the discrete form, Monte Carlo localization uses sampling of the probability distribution. Samples can be compared with a grid which uses higher resolutions in areas where it is more likely to find the robot. Therefore computational power is concentrated on these areas resulting in higher accuracy. The algorithm is described in [1] including experimental results with different sensors. One of the sensors used is a camera pointing upwards, and only the average over a few pixels in the center was used as input to the algorithm.

## Approach

The idea for this project is to use more information from the image than the average of a few pixels. The problem is that the image contains too much information and has to be reduced in some way to information which helps discriminating between images



Figure 1: ATRV-Jr mobile robot platform with an Omnicam mounted on top.

taken at different locations. Principal Component Analysis allows the calculation of a projection matrix for this task. Sample images are taken from different locations in the environment.

Besides the location of the robot, its orientation is also of interest. An advantage of the omnicam is that sample images can just be rotated to a number of different angles without having to acquire all the different images in the environment.

All the images and their rotations are treated as vectors, and all these vectors are combined into a matrix. Then the eigenvectors are calculated, they are sorted by the corresponding eigenvalues and finally only the first  $n$  of them are used as a projection matrix. Using this matrix, an image can be projected to a vector of size  $n$ . For all the input images and their rotations this projection is done and the resulting values are plotted into a map considering the position each image was taken at. The remaining areas of the map are filled by interpolation of the samples.

Once the map is created, the robot takes an image at an arbitrary position. Again the projection matrix is applied to get a vector of size  $n$ . This vector is compared with every position in the map, the similarity corresponds to an energy function, which is related to the probability distribution of the location of the robot according to the sensor information. This can be combined with the prior distribution of the robot location using the Monte Carlo method to get a more accurate estimation.

## Implementation

For the experiments we mounted a camera with a 360 degree round view, also called omnicam, on top of an ATRV-Jr mobile robot (Figure 1). The camera points upwards



Figure 2: Picture from omniam

to the ceiling, that way less non-static objects like people get into the view. All hallways in the third floor of the MaRC building were scanned by remotely controlling the robot. The covered area is H-shaped, with side lengths of 85 and 36 meters. From the resulting video footage one image about every meter was taken, resulting in 380 single images. The location where each image was taken from is known. It should be noted that offline algorithms exist which are capable of creating such maps automatically without knowledge of the position of the robot; these algorithms are computationally very expensive, and the implementation was outside of the scope of this project.

Analysis of the images obtained from the omniam (an example is given in Figure 2) showed that they are stretched in  $x$ -direction by a factor of 1.11 and have to be corrected accordingly. Furthermore, the black disc in the middle, where the camera sees itself, is not in the center of the image of the environment. The correct location of this center is the intersection of images of lines that are vertical in the environment, like doorframes and windows. Since the camera is off-center, the images are additionally distorted, but the effect is minimal and should not have a big influence on the experiments.

After finding the correct center, the image was cropped to be square-shaped, and a mask was constructed to hide the black disc in the center and the black regions outside of the image. That way rotation of the image gives (approximately) the same result as rotating the camera in the environment. To restrict the complexity of the following computations, the images are converted to gray scale. The possible advantages of including color information has to be examined in later work.

All  $k$  sample images  $I_1, I_2, \dots, I_k$  of size  $n \times n$  each are converted to 1-dimensional vectors of size  $n^2$  and put together into one matrix  $M$  of dimension  $n^2 \times k$ . The calculation of the eigenvectors cannot be done in the conventional way, since this would require to



Figure 3: Map of one corridor ( $32.5m$  by  $1.7m$ ) calculated from 58 sample images. Each sample is projected using each of the first three eigenimages. The resulting three values are assigned to the colors red, green and blue. Points between the samples are obtained by interpolation.



Figure 4: Energy function using one of the sample images. The brightest point at the top, near the center, is the actual location the image was taken from.

calculate a covariance matrix with a side-length of  $n^2$ , which in our case is over 200,000 for the gray scale images. [2] describes a way to calculate the first  $l$  eigenvectors  $v_1, v_2, \dots, v_l$  of a  $n \times m$  matrix  $M$  with  $l < m \ll n$  by calculating the eigenvectors of its transpose  $M^T$ . These  $l$  eigenvectors can be combined to form a projection matrix  $P$ , which can then be used to project any input image to a vector of size  $l$ .

To create a map, each sample image is projected using  $P$ . The result is plotted into a map at the corresponding location the sample was taken from. Empty areas between the samples are filled by interpolation. Due to memory restrictions results for only one corridor were calculated, using 58 sample images without rotation. To cover all 380 sample images the algorithm has to be changed to calculate only parts of the equations at once, since the complete matrix of sample images would use over  $20GB$  of memory when using 8 steps of rotation. The resulting map for the projection using the three most significant eigenimages for one rotation angle is shown in Figure 3. The three maps are combined assigning each of them to a separate color in the RGB color model.

This map can now be used to localize the robot. At run-time, the robot takes an omniscam image from its current position, projects the image to an  $l$ -dimensional vector using  $P$ , and calculates the sum of square differences for each point of the map to generate the energy function  $E$  (Figure 4). This energy function serves as input to the Monte Carlo Localization algorithm.

## References

- [1] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. (1999). Using the Condensation Algorithm for Robust, Vision-based Mobile Robot Localization. IEEE Computer Society Conference on Computer Vision and Pattern Recognition
- [2] M. Turk, and A. Pentland. (1991). Eigenfaces for recognition. The Journal of Cognitive Neuroscience, 3(1): 71-86.