# Ground-plane based indoor mobile robot localization using RGB-D sensor[1]

Miguel Vaz
miguelvaz@ist.utl.pt
Rodrigo Ventura
rodrigo.ventura@isr.ist.utl.pt

Institute for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal

## Abstract

This paper addresses the problem of absolute localization in an indoor environment using a RGB-Depth camera. The approach is based on the use of the ground region perceived by the RGB camera to detect and decode its position and edges. The localization system uses this data to match it with a known on-board map. The ground plane detection algorithm is designed to be robust to vibration or disturbances during the robot motion. The localization system is based on the particle filter, fusing odometry with ground region matching where each particle's weight is proportional to the quality of correspondence between the ground edge estimation and the nearest walls. Promising results were obtained and are presented in this article.

## 1 Introduction

In the past years, several publications have addressed the localization problem based on wall detection methods [1]. These systems have some limitation, especially in unstructured environments where walls are sometimes difficult to detect (hidden by furniture or not present). Another issue is that this algorithm can be mislead by a planar obstacle. We tackled the localization problem using the RGB-Depth camera and the Particle filter method. The data acquired by the RGB-Depth camera form a point cloud, which consists in a set of 3D points $(x_c, y_c, z_c)$ in the camera frame. Each 3D point of the set is associated to one pixel on the image plane of the camera. In this work we used the Microsoft Kinect sensor. Since the localization system is based on the ground observation, we assumed that the camera is always pointing down with the floor on a big part of the FOV. First we detected the ground point cloud based on a pre-calibrated ground plane and on a dynamic threshold filter. The filter point cloud edges are then estimated and used to calculate the particle weight on the Particle filter algorithm [5]. The weight of each particle is inversely proportional to the distance of the edges from the nearest wall, as seen by the particle position. While the localization system developed by Biswas and Veloso [1] is based on wall detection and random sampling of the depth images for plane detection, our approach uses ground detection algorithms that are less sensible to planar obstacles and usable in unstructured environments.

## 2 Floor Detection

### 2.1 Floor detection

The floor is modeled as a plane and parametrized according to the following normalized equation [4]:

$$ax + by + cz + d = 0, \quad (1)$$

where $[a, b, c]$ is the normal vector, $d$ the distance to the origin and $(x, y, z)$ are the coordinates of a point on the ground in the camera frame $(X_c, Y_c, Z_c)$. Before starting the localization system we performed a calibration of the floor. This calibration setup is shown in Fig. 1 where a chess board pattern is placed inside the camera's FOV, maintaining the robot still. A corner detection algorithm [6] is used on the colored image to estimate the pixel coordinates of the board's inner corners. With the obtained data and the corresponding depths provided by the camera, we estimated the 3D coordinates of the chess corners.

The calibrated floor parameters $[a', b', c', d']$ are estimated by applying the Least Squares on the estimated 3D inner corners position, providing an approximated ground model. A new frame defined by the orthonormal basis $(X_{cp}, Y_{cp}, Z_{cp})$ is associated to the calibrated ground plane as shown in Figs. 1 and 3, where $X_{cp}$ axis is the normalization of the $Z_c$ axis' projection and the $Z_{cp}$ axis is the normal vector of the plane. The $Y_{cp}$ axis results from the external product of $X_{cp}$ with $Z_{cp}$. This definition assumes that the $Z_c$ axis points forward the robot.

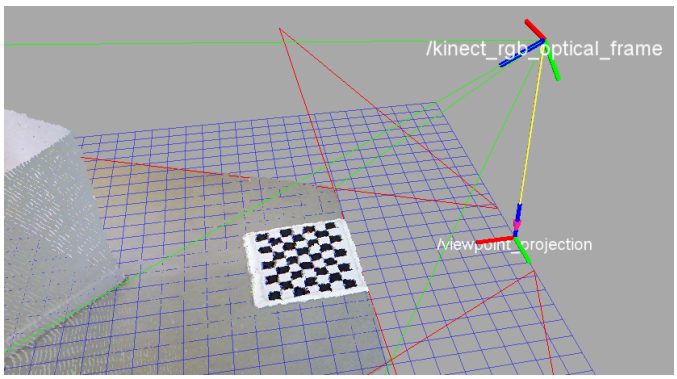


Figure 1: Calibration setup with the FOV (green) and ground plane intersection with FOV planes (red).

To estimate the ground model during the robot's movements, a dynamic threshold function is defined to remove from the complete point cloud the points that are further away from the calibrated ground plane. The threshold function is defined as two planes, one down and the other one up symmetrical along a pre-calibrated ground plane model. The function value grows while we walk away from the robot, as one can see in Fig. 2 where in black we have the pre-determined ground plane position, in orange the true position of the plane and in dashed yellow the value of the threshold function. Therefore a point which distance from the calibrated plane is bigger then its corresponding threshold value is discarded (like point $P_1$ in Fig. 2).

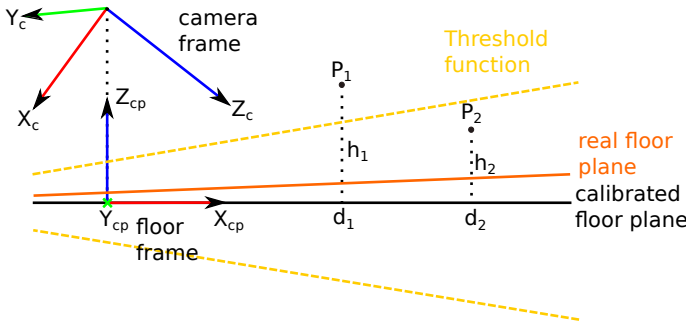


Figure 2: Threshold dynamic filter. Its values depends on the point distance to the floor frame.

The resulting filtered point cloud is formed by the points of the ground and by some outliers, in particular from the obstacles that are further from the robot. To remove these outliers a random sample consensus (RANSAC) algorithm [3] is applied. The result is a point cloud formed just by the floor point cloud and the real ground plane parameters. An example of such point cloud is illustrated in Fig. 3.

We developed this floor detection algorithm because the calibration process by itself is not enough to detect the floor point cloud on-line in a robust way because of the vibration during the robot's motion. This makes the floor parameters change significantly comparing with the calibrated parameters.

### 2.2 Edges estimation

After the floor detection, an edges estimator was designed to detect the edges of the ground seen by the robot. For that we applied the concave hull algorithm [2]. The result is a list of points forming the polygon of the ground seen in the camera FOV that includes the edges created by the end of the FOV. A filter is then applied to remove them. As one can see in Fig. 1, the intersection of the FOV planes (green lines) with the ground plane is estimated (red

[1]This work was supported by the FCT project [PEst-OE/EEI/LA0009/2013]

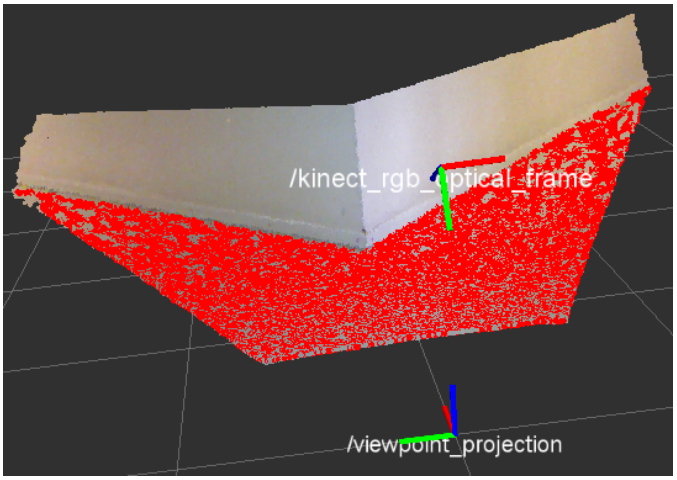


Figure 3: Result of the ground point cloud estimation (red points). The camera frame (named `/kinect_rgb_optical_frame`) and the camera projection frame (named `/viewpoint_projection`).

lines). Then we filtered the points of the polygon that are near the intersection of the two planes.

### 2.3 Cost function

A cost function is defined as the $L^1$ norm of the distances between the ground edges and the nearest walls according with a determined particle, as shown in the following equation:

$$C(L,(x_j,y_j,\theta_j)) = \sum_{i=0}^{n} D(P_i^{(x_j,y_j,\theta_j)}) \quad (2)$$

where $L$ is the list of edge points, $(x_j,y_j,\theta_j)$ are the coordinates of the particle $j$, $D(.)$ is the distance from a point to the nearest occupied pixel and $P_i^{(x_j,y_j,\theta_j)}$ is the $i^{th}$ point in the edges list $L$ transformed to the $(x_j,y_j,\theta_j)$ coordinates, i.e. the edges as seen in the $j$ particle position.

As an illustration, for the list $L$ obtained looking at a wall's corner and fixing the robot orientation, one can see the function value for a wide area near the true position of the robot

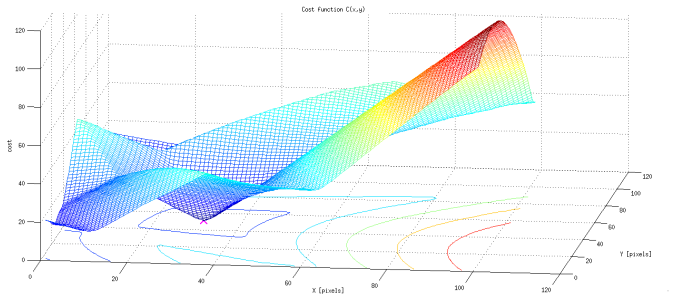


Figure 4: Cost function defined, fixing the orientation, in the red area in Fig. 5 for the blues edges detected (wall corner).

If one tries to estimate the robot position by the minimum of the function, we find that it is consistent with the robot true position as one can see in the Fig. 5 by the good correspondence of the walls (black points) with the edges detected (blue points) at the position estimated by the minimum (magenta cross).

## 3 Robot Localization System

The localization system is formed by the particle filter [5] where the particles' weight is inversely proportional to the particle cost. The particle cost is estimated according to the previous defined cost function. This way, the system at the predict step moves the particles according to the odometry readings and adds zero mean Gaussian noise. At the update step, the system gets the point cloud from Kinect, estimates the floor point cloud, performs the edges detection and calculates each particle's cost value. Then the resulting particle set of the resampling based on the particle weight, i.e inverse of the cost, is used on the next iteration of the particle filter.

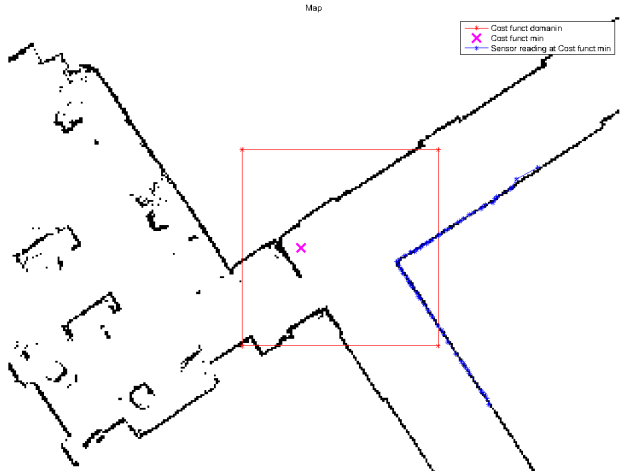


Figure 5: Area where the cost function was calculated (red lines), the position of less cost value (magenta cross) and the measurements in the global frame as seen in the minimum cost position (blue lines).

## 4 Results

As a preliminary result, we ran the particle filter with the robot in a fixed position with a start particle set of 300 uniformly distributed around the real pose of the robot in a 4 by 4 square meter and with $\pm 10^{\circ}$ of orientation range. The system successfully managed to convert to the real position as one can see in Fig. 6.

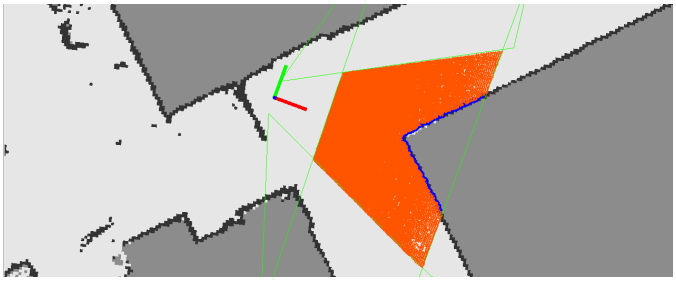

Figure 6: Result of the particle filter algorithm with the ground detection (red points) and edges estimation (blue lines).

## 5 Conclusions and Future Work

In this paper we described a localization system based on a RGB-Depth camera on an indoor environment. The proposed implemented system appears to converge with small errors for the scenario specification used in the experiment. As a future work, we would like to perform further tests on the system in different scenarios and carry out a deeper analysis of the errors in the system including during robot motion.

## References

[1] J. Biswas and M. Veloso. Depth camera based indoor mobile robot localization and navigation. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1697–1702. IEEE, May 2012.

[2] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2000.

[3] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[4] P. J. Schneider and D. Eberly. *Geometric Tools for Computer Graphics*. Elsevier Science Inc., New York, USA, 2002.

[5] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent robotics and autonomous agents series. The MIT Press, 2005.

[6] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, 2000.