Global Localization by Soft 3D Object Recognition

Fernando Frazão Leiras Ribeiro
fernandolribeiro@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal

October 2014

Abstract

In this paper we present a global localization method for mobile robots based on soft 3D object recognition, GLSOR-3D. A depth sensor is used to acquire a partial view for each observed object. From the partial view, GLSOR-3D extracts a position relative to the robot and an object descriptor, the Partial View Heat Kernel. Probabilistic methods are used, namely a particle filter, to infer the robot localization from the sensor observations. Our contributions hinges on exploiting the inherent characteristics of the descriptor in scenes with multiple objects to obtain localization without explicitly classifying or registering the 3D data.

Prior to recognizing them, objects must be extracted from the surrounding environment. Imposing simplifying assumptions, we present a segmentation approach based on commonly used methods in the literature. This approach obtains accurate and fast segmentation results. Another original contribution is presented in a contrasting second segmentation approach using spectral clustering methods. These methods provide a way to automatically segment the acquired data based on the underlying geometry and color.

Keywords: localization, RGB-D sensor, particle filter, object recognition, segmentation, spectral clustering

1. INTRODUCTION

For robots to interact with humans on a daily basis and safely operate in environments common to both, they must be able to navigate in an environment using natural landmarks, which are complex and difficult to identify. The most interesting landmarks in human environments are often medium sized objects, such as sofas, whose shape and texture yields them unique, hence valuable for localization in large environments. Nonetheless, localizing the robot based on such objects is not a trivial matter as it implies recognizing both the object and its pose. However, errors in either of these tasks can strongly affect the localization.

In this work, we contribute by developing a method that provides a coarse pose estimation of the robot without performing classification nor registration. This method is used in the core of our algorithm, Global Localization by Soft 3D Object Recognition (GLSOR-3D), which estimates a robot position and orientation in a global coordinate system using as landmarks multiple medium size objects.

Prior to recognition, segmentation of the scene is necessary in order to extract the objects from the surrounding background. We address this subject by presenting two segmentation approaches. The first employs methods often used in the literature to create a real-time and reliable algorithm specific for the case in hands. The second constitutes a novel approach suitable for more general cases, using spectral clustering methods to segment based on the underlying geometry and color of the acquired 3D data.

To infer the localization, GLSOR-3D assumes that some prior knowledge is available concerning both the landmarks distribution and their appearance. Namely, the robot must have access to object appearance models. We consider explicit object models, where observations from different view angles are labeled and stored in a dataset. The models allow to estimate coarse poses in the object intrinsic coordinates. GLSOR-3D must also have access to an environment map, containing each object label and pose in global system. The map allows to estimate a coarse global pose from the object coordinates.

While the robot is navigating in an environment where multiple objects are present, as illustrated in Figure 1, GLSOR-3D provides an estimate of the global pose of the robot from data collected by a depth sensor. In particular, the sensor provides the position of the object with respect to the sensor and object observations in the form of a partial view. Partial views correspond to the visible 3D surface of the object as seen from the sensor view angle. By comparing the partial view with the information in the object appearance models, the algorithm extracts information on the object identity and robot pose in the object intrinsic coordinates. Using the map, it then estimates the global pose.

However, the information about the relative pose and object identity is coarse as it can only be obtained by
Figure 1: Robot navigating a scenario where multiple objects are present. GLSOR-3D integrates the observations with prior knowledge encoded into a map, inferring the robot localization.

comparing the observed partial view with all the labeled partial views in the object dataset. This task is difficult due to ubiquitous sensor noise, symmetric objects, similarity between different objects and view angle discretization in the dataset. GLSOR-3D addresses this problem by:

1. using a state-of-the-art partial view descriptor, the Partial View Heat Kernel (PVHK) [5], which allows for easy comparison between partial views and varies smoothly with the view angle;

2. accumulating several observations from different positions, while estimating the displacement between observations through odometry; and

3. performing soft object detection, i.e., by computing the probability associated with each object class instead of performing hard classification.

Finally, to integrate the information from the sequence of observations and odometry readings with the environment map, GLSOR-3D uses a particle filter framework. Particle filters are especially suitable when the observations are ambiguous and are not linearly related with the dynamics, as is the case for the PVHK descriptor.

We validate GLSOR-3D by performing several experiments with different number of objects. Furthermore, we assess the impact of using the dataset to retrieve the landmark class and view angle by solving the same problems with and without partial view information.

2. RELATED WORK
Here we present both the state of the art in Localization and the partial view representation used.

2.1. Localization
In recent years we have seen robots performing more and more complicated tasks in human environments, for which they rely on navigation in global coordinate systems. The resilience of current localization methods, such as [4], which relies on 3D information as GLSOR-3D, ensures that robots can follow a map through numerous tasks. However, most algorithms rely only on static landmarks such as walls and corners. The ambiguity of such landmarks means that the algorithms must be initialized by other methods, including manual labeling. GLSOR-3D, by making use of more singular landmarks, performs global localization without requiring any other sources of information.

Also using 3D active cameras, the state-of-the-art SLAM++ algorithm [17], provides impressive results for localization. SLAM++ tracks the camera pose while creating an object pose map of the unknown environment. Our work addresses the opposite case, where we start with a map and an unknown initial position and then infer the localization. Nevertheless, similar methods could be used to solve our problem. However, despite the outstanding results of SLAM++, this method suffers from important drawbacks. For one, it is computationally demanding due to several complex steps and iterations performed to achieve a pose estimation through registration against dense and complete 3D models acquired with KinectFusion [14]. For such reason, an highly parallel GPU implementation is required to obtain real time performance. Furthermore, this method requires classification upon object recognition, risking an erroneous decision that might affect the rest of the process. To avoid this problems our approach does not perform decision nor registration, which is achieved through probabilistic methods.

The use of joint localization and object recognition has also been addressed previously in probabilistic terms, either for the purpose of solving a localization problem [1] or for the purpose of object recognition [6], [21]. In [1] the authors introduce the concept of hard and soft object detection, whether explicit classification is made or not. The devised approach performs soft recognition in a omni-directional image. It computes a per-pixel vector of detection scores, using local image features, for each object class present in the dataset. This results in a heatmap which represents the probability of a given object appearing in a particular position of the image. The localization is then estimated by integrating the observations with a map annotated with the position and label of the objects using a particle filter framework [7]. GLSOR-3D has a very similar approach, however we make full use of a 3D sensor to both estimate the probability of each object class, and also retrieve the relative position between robot and objects. Furthermore, comparing with [1], we have more than one template associated to each object, which allows GLSOR-3D to obtain a coarse orientation estimation. The coarse pose estimation is one of the central inputs to GLSOR-3D.

Finally, the PVHK [5] was previously used in [6] also in the context of joint localization and recognition. GLSOR-3D leverages on the approach proposed on [6] for modeling the similarity between partial views and estimating the probability of each partial view to be associated with a given object and view angle. However, [6] can be seen as a dual of ours. Their goal is to, through multiple hypothesis about the object class,
obtain a final correct classification using the minimum number of particles. In this work, the multiple hypothesis and various observations are used to obtain the most likely robot location. Furthermore, the algorithm proposed in [6] only handles a single object, while GLSOR-3D handles an arbitrarily number of objects.

For the benefit of the unfamiliar reader, in the following we provide a brief overview of the descriptor, highlighting its main characteristics that guided our choice.

2.2. Partial View Representation
The PVHK represents a partial view by the geodesic distance between a reference point and the ordered set of points in the partial view boundary. The reference point is chosen in a systematic way that depends on the relative position between sensor and object. As the distance between a point and the boundary uniquely defines the surface, apart from isometric transformations, the PVHK is unique for any given pair of objects and view angles.

To handle the impact of noisy 3D data on the geodesic distances, PVHK uses diffusive geometry concepts, which are considerably more robust [19]. In particular, it relies on the heat propagation, which is a diffusive process, to obtain proxies for the distance between the reference point and the boundary. The propagation, illustrated in Figure 2, considers an instantaneous heat source at the reference point, s, and can be interpreted as a sequence of locally averaging operations applied at each time step to the complete surface temperature. The operations dilute spikes and discontinuities in the function, diffusing the temperature from hot regions near the source to neighboring points. By stopping the propagation when the heat reaches the boundary, the PVHK obtains a temperature profile that is strongly correlated with the desired geodesic distance, but strongly resilient to noise.

The PVHK stability is relevant to our choice, as it ensures that the descriptor changes smoothly with the view angle, as illustrated in Figure 3. As long as the silhouette of the object does not change radically, the source point and the boundary will change smoothly between similar view angles and so will the descriptor, reducing errors related to the view angle estimation. Therefore, the descriptor captures the information on the partial view in the shape of a graphic of the temperature as a function of the boundary.

To compare two partial views, we compute the distance between descriptors using the Modified Hausdorff Distance (MHD), which is often used to compare lines and shapes. To use the distance, we first represent the descriptor \( d \in \mathbb{R}^L \) as a set of points \( \eta = \{[1/L, d_1], [2/L, d_2], \ldots, [1, d_L]\} \) and then compute \( \rho(d, d') = \text{MHD}(\eta, \eta') = \min \{\sum_{x \in \eta} \text{inf}_{y \in \eta'} \|x - y\|^2, \sum_{y \in \eta'} \text{inf}_{x \in \eta} \|x - y\|^2\} \).

We establish the probability that a descriptor \( d \) corresponds to an object class \( c \) and view angle \( \vec{v} \), \( p(d|c, \vec{v}) \), by computing the distance between \( d \) and \( d^{c, \vec{v}} \) and assuming an exponential distribution:

\[
p(d|c, \vec{v}) = \exp \left(-\rho(d, d^{c, \vec{v}})/\lambda\right) / \lambda 
\]  

where \( \lambda \) represents the average inner distance between descriptors in the dataset.

3. SCENE SEGMENTATION
The goals of this task are to segment each point cloud acquired with the RGB-D sensor into a set of clusters that belong only to the objects of interest and to remove, as much as possible, any trace of the background so that GLSOR-3D can perform the recognition and consequent localization.

3.1. Fast Segmentation Approach
In the literature is common to find segmentation approaches for specific indoor tasks. Features commonly present in indoor environments are walls, floor and table-tops which can be easily identified in 3D due to their large planar areas [2], [3], [18], [10]. To immensely simplify the complexity of the segmentation algorithms the approaches present in the literature often start by identifying these planes with efficient methods and then segment the remaining objects of interest based on some additional criterion.

Similarly to the literature, we admit that: a) the floor of the scene is planar and occupies a greater area than any of the present objects; b) the 3D active camera used for data acquisition has a stable, aligned, position regarding to the floor; and finally c) the objects are distanced from on another by some small space.

These assumptions greatly simplify the process of identifying the objects in the scene by discarding the need to solve segmentation puzzles which often require classification right from the start. Considering all of the
mentioned points, the algorithm devised for the segmentation task is briefly summarized in the following steps:

1. Down-sample the acquired point cloud using a voxel grid;
2. Find, by RANSAC, the largest plane that is parallel to the \( x - y \) axis within a certain angle threshold;
3. Compute the convex-hull of the identified plane, i.e., the smallest convex set of points of the plane that contains all of the remaining points;
4. Define a prism with the convex-hull and a given height, then extract all points from the original point cloud, with higher resolution, that lay inside it;
5. Apply euclidean clustering to the remaining points, i.e., cluster points by comparing the distance between them with a predefined value;
6. Filter clusters according to their properties such as number of points, approximated size, proportions and distance to camera;
7. Triangulate and build mesh, which is necessary for the descriptor computation.

Results can be seen in Figure 4. Our implementation employed algorithms present in the Point Cloud Library (PCL) [16]. However, many adaptations were made to obtain more efficient results exploiting the organized structure of the clouds acquired with our sensor.

![Figure 4: Segmentation process. Left: Original point cloud. Middle: Plane from the down-sampled cloud in green. Convex-hull as a white polyline enclosing the plane. Points extracted from the original cloud in orange. Right: Final results after clustering and filtering.](image)

### 3.2. Spectral Clustering Segmentation

Spectral clustering is a widely used technique for data analysis and low dimensional embedding, mainly with the purpose of interpreting the data and identifying groups of similar properties. In the literature, spectral methods have been commonly applied in the decomposition of highly detailed and complete meshes into morphologically meaningful shapes [22], [15], [12]. Usually the objective is to identify perceptual relevant features such as symmetries, limbs or elementary pieces of a more complex object. Opposite to the literature, the problem in hands is to extract objects from a noisy and incomplete point cloud \( P \) of a scene acquired with an RGB-D sensor. On the plus side and contrary to the literature, we can explore the use of color to guide the segmentation.

The clustering is based on similarity between points, therefore we create an adequate similarity graph to represent and solve our problem. Since the RGB-D sensor resolution decreases with distance, we start by excluding all the points from \( P \) that are further than a certain distance, as their sparse presence might interfere with the clustering process. Then, exploiting the organized structure of the point cloud, we use a fast triangulation method to build a mesh from the remaining points. This mesh represents all the connections \( E \) and nodes \( V \) of our graph.

To obtain a similarity value between neighboring vertices \( v \in V \) an affinity function \( a(v_i, v_j) \) is defined. As core for the affinity metric, we use a combination of the distance between points, difference between colors and difference between normal vectors associated with each point. The first can be easily obtained by computing the euclidean distance \( d_{ec} \). Let \( p_i \in \mathbb{R}^3 \) be the 3D coordinates of \( v_i \):

\[
d_{ec} = \| p_i - p_j \| \tag{2}
\]

To compute the color difference a change of color space is recommended since the RGB representation mixes color with light information. Following the choice used in [11], we opt for the Lab color space which is divided into lightness (\( L \)) and two color-opponent dimensions (\( a \) and \( b \)). The Lab space has the particularity that a difference between colors has an euclidean distance correspondence similar to how humans perceive it. This is useful for combining geometry and color distances in a consistent fashion. Thus, we define \( d_{Lab} \) as the color distance between vertices in the Lab color space. Let \( C \) be a 3-dimensional space where each dimension is contained within the limit \([0, 1] \) and \( c_i \in C \) be the color values in the Lab space associated with \( v_i \), then:

\[
d_{lab} = \| c_i - c_j \| \tag{3}
\]

Finally, we devise a metric for evaluating the affinity between normal vectors. RGB-D sensors do not provide surface and normal information, thus we use a method based on integral images [9] to estimate per-point normals. This method is suited to deal with the noisy nature of the acquired point clouds. To efficiently compute an affinity value between adjacent normal vectors we simply use the dot-product. This way, since the vectors are unitary, the similarity will depend on the angle between them. To obtain small values when the vectors are similar, i.e. close to parallel, and high otherwise, we define \( d_{norm} \) as:

\[
d_{norm} = 1 - |\vec{n}_i \cdot \vec{n}_j|, \tag{4}
\]

where \( \vec{n}_i \) and \( \vec{n}_j \) are the unit normal vectors associated with \( v_i \) and \( v_j \), respectively.

With these metrics we completely define \( a(v_i, v_j) \):
where \( \mathcal{N}_i \) is the set of vertices connected by an edge \( E_{ji} \in E \) to \( v_j \). With this function small distances will result in great affinity values, connecting the vertices firmly, while great distances will produce frail connections, becoming more likely candidates to be cut upon segmentation. The constants \( \alpha, \beta \) and \( \gamma \) are used as importance factors for each of the parcels and \( \delta \) is just a small constant to condition the result. In the experiments performed, setting all importance factors equal provided good results without the need for further tweaking.

The most important step in spectral clustering is the eigendecomposition of a graph Laplacian as the eigenvectors are the elements that can reveal the underlying partitions. To build the Laplacian of a graph with equal provided good results without the need for further tweaking.

By:

\[
\mathbf{a}(v_i,v_j) = \begin{cases} 
1 & \text{if } v_j \in \mathcal{N}_i \\
0 & \text{otherwise}
\end{cases}
\]

(5)

\[
\mathbf{W} = \begin{pmatrix} \mathbf{a}(v_1,v_1) & \cdots & \mathbf{a}(v_1,v_n) \\
\vdots & \ddots & \vdots \\
\mathbf{a}(v_n,v_1) & \cdots & \mathbf{a}(v_n,v_n) \end{pmatrix}
\]

where \( \mathcal{N}_i \) is the set of vertices connected by an edge \( E_{ji} \in E \) to \( v_j \). With this function small distances will result in great affinity values, connecting the vertices firmly, while great distances will produce frail connections, becoming more likely candidates to be cut upon segmentation. The constants \( \alpha, \beta \) and \( \gamma \) are used as importance factors for each of the parcels and \( \delta \) is just a small constant to condition the result. In the experiments performed, setting all importance factors equal provided good results without the need for further tweaking.

The most important step in spectral clustering is the eigendecomposition of a graph Laplacian as the eigenvectors are the elements that can reveal the underlying partitions. To build the Laplacian of a graph with \( N \) nodes we define the weight matrix \( \mathbf{W} \in \mathbb{R}^{N \times N} \) where \( W_{ji} = \mathbf{a}(v_i,v_j) \) and the degree matrix \( \mathbf{D} \in \mathbb{R}^{N \times N} \), which is a diagonal matrix with entries \( D_{ii} = \sum_{j=1}^{N} W_{ij} \). From this matrices, the normalized graph Laplacian is given by:

\[
\mathbf{L}_n = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{W})\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2},
\]

(6)

where \( \mathbf{I} \) is the identity matrix. \( \mathbf{L}_n \) is symmetric and positive-definite, hence all eigenvalues will be real and non-negative. If we colorize each point according to the respective graph degree we can obtain a visual interpretation of the resulting Laplacian and how tightly connected is each vertex to its neighbors(Figure 5). Following the approach in [13],the spectral clustering problem can now be solved, for a number \( k \) of clusters, as stated in the following steps:

1. Compute the Laplacian’s first \( k \) eigenvectors \( v_1, \ldots, v_k \), where \( v_i \) is the eigenvector associated with the \( i \)th smallest eigenvalue.

2. Construct \( \mathbf{V} \in \mathbb{R}^{N \times k} \) as the matrix containing the eigenvectors \( v_1, \ldots, v_k \) as columns. The rows of \( \mathbf{V} \) will now be interpreted as data points \( y_i \in \mathbb{R}^k \) represented by vectors;

3. Individually normalize each vector \( y_i \) by doing \( y_i = \frac{y_i}{\sqrt{\sum_{j=1}^{N} y_{ij}^2}} \).

4. Use a clustering method to group the points \( y_i \) for \( i = 1, \ldots, n \) into clusters.

In the experiments effectuated two different approaches were used to cluster the points into groups, k-means and a voting technique.

**K-means** | K-means always results in convex clusters, for which reason it is not adequate for clustering the initial data points. However, in the embedded space, the points become better separated making the task easier. Figure 6 presents the results given by k-means, for \( k = 3 \), using \( y_i \) for \( i = 1, \ldots, N \) as data points.

An important aspect of k-means is the necessity of choosing the correct value for \( k \). Interestingly, a useful property resulting from the spectral clustering process is that the most stable partitioning is usually given by the value \( k \) that maximizes the eigengap, which is the difference between consecutive eigenvalues:

\[
\Delta_k = |\lambda_{k-1} - \lambda_k|
\]

(7)

where \( \lambda_k \) is the \( k \) smallest eigenvalue. This heuristic proven correct for the experiments made with simple scenarios. When the scene contained two objects of interest and the ground plane, the resulting largest eigen-gap was obtained for \( k = 3 \), while on scenes where there was only one object and the ground plane it led to \( k = 2 \).

![Figure 6: Resulting k-means clusters for k = 3. Left: 3D color-labeled data points. Right: Embedded data points.](image)

**Voting** | In the studies presented in [15], for each eigenvector \( v_i \) the authors decompose the mesh into regions where \( v_i \) has either positive or negative value. By segmenting the surface into regions of constant sign they can extract shape features from the mesh. Following this approach, we compute the \( k \) first eigenvectors and produce the respective \( k \) segmentations.

As seen in Figure 7, the obtained partitions are mostly far from perfect, but many of the visible cuts correspond to the intended solution. Furthermore, with increasing \( k \), smaller details are captured and frequently the same cuts are repeated. Thus, we can vastly improve the segmentation by devising a simple voting scheme where each edge cut is counted and given an importance based on
the order of the eigenvector from which it originated. Using this method we obtain the intended partitions as the most voted cuts, along with many of the smaller details which are also featured but with smaller importance (Figure 8).

Comparing with k-means, this approach has the advantage of achieving better results for more complex scenes where smaller details are available. On the other hand, it often results in over-segmentation and consequently requires a method for recognizing the object by classification.

4. COARSE POSE ESTIMATION

GLSOR-3D estimates a coarse robot pose by comparing data acquired during execution with prior knowledge.

The prior knowledge corresponds to: i) the set of possible states, represented by the state space $\mathcal{S}$; ii) the layout of the $K$ objects present in the scenario, $O = \{o_1, \ldots, o_K\}$, where each object $o_k$ is characterized by its class $c_k$ and pose in the global coordinate system $(\tilde{x}_k, \tilde{y}_k, \tilde{\theta}_k)$; iii) the object dataset $\mathcal{D}$ that contains a set of descriptors from each object $o_k$, indexed by their view angle $v_{om}$ in the object intrinsic coordinate system. All this information is given to the robot in the form of a knowledge map $m = (\mathcal{S}, O, \mathcal{D})$.

The information gathered during execution corresponds to the set of observations $Z = \{z_1, \ldots, z_N\}$, $N < K$, associated with each of the $N$ visible objects at a given time instant. As illustrated in Figure 9, our observations are both a translation vector $T_o \in \mathbb{R}^2$ in the sensor coordinate system, and a partial view descriptor $d_o \in \mathbb{R}^L$ with $L = 80$.

GLSOR-3D performs a coarse pose estimation for each possible robot state, $s = \{x, y, \theta\}$, by estimating the probability of a given observation, $z$, without making any hard decision on the object class, $c$, nor position $(\tilde{x}, \tilde{y})$ of the object in the observation. It first estimates the observations that the robot would have received from all objects if it was in that position. Then it compares them with the true observations and estimates if it is possible, i.e., it computes $p(z|s, m) = \sum_{o_k} p(o_k|s, m) p(d, T|s, o_k, m)$.

The term $p(o|s, m)$ represents the visibility of an object $o$ from the state $s$. However, we assume that any object with significant probability of generating the translation vector $T$ has the same visibility probability. Furthermore, given the robot pose, the map and classification of each object, we consider that $d$ and $T$ are conditionally independent from each other. Therefore, our objective is to compute

$$p(z|s, m) \approx \sum_{o} p(T|s, o, m)p(d|s, o, m) \tag{8}$$

To estimate the term $p(T|s, o, m)$ we first estimate the translation vector $T_o$ in the global coordinate system using the orientation $\theta_o \in s$ and the object position $(\tilde{x}, \tilde{y})$. Assuming that the sensor error is Gaussian, we estimate $p(T|s, o, m) \propto \exp\left(-\frac{\| (x, y) - \tilde{T} \|^2}{\sigma_T^2} \right)$.

To estimate the term $p(d|s, o, m)$, we use Eq.1. However, as the observation models for each object are represented in their intrinsic coordinate system, we first estimate the view angle $\nu_o$ associated with the position $(x, y)$, and the object orientation $\theta_m$ in the global map. Then use the indexed dataset associated with the class $c$ of the object $o$ to retrieve $d^{(c)}$ and estimate $p(d|\nu, c)$.

5. LOCALIZATION

GLSOR-3D refines the coarse pose estimation by considering a sequence of observations retrieved from different positions in the map, assuming that the robot has access to the changes in position $U$ through the odometry system. In particular, it estimates the state $s$ at each instant $t$ by integrating the information contained in all the previous observations $Z_{1:t}$ and actions $U_{1:t}$ as:
\[
\hat{s} = \arg \max_s \{ p(s|Z_{1:t}, U_{1:t}, m) \}. \tag{9}
\]

In a Markovian setting, the posterior in (9) can be defined recursively as:

\[
p_{\text{target}}(s_t) = p(s_t|Z_t, U_t, m) = \eta p(Z_t|s_t, m) q_{\text{proposal}}(s_t)
\]

where \( \eta \in \mathbb{R} \) is a normalization constant and

\[
q_{\text{proposal}}(s_t) = \int_{s_{t-1}} p(s_t|s_{t-1}, U_t, m) p(s_{t-1}|Z_{t-1}, U_{t-1}, m) ds_{t-1},
\]

is the proposal distribution. However, the computation of (10) has a complexity of \( O(|\mathcal{S}|^2) \), where \(|\mathcal{S}| \) is the number of possible states in the map. Hence, GLSOR-3D reduces the complexity of the problem by using a Monte Carlo Approach, namely it estimates the robot state recursively using a particle filter [7] framework.

5.1. Recursive State Estimation

A particle filter approximates the target probability (10) at any time instant \( t \) by a set of weighted particles \( S_t = \{ [s_{t,j}^{|j|}, w_{t,j}^{|j|}] \}_{j=1,.., J} \). The recursive estimation is done by sampling a set \( S_{t-1} \) from \( S_{t-1} \) according to a motion model \( p(s_{t}|s_{t-1}, U_t, m) \). Consequently, \( S_t \) follows the proposal distribution \( q(s_t|Z_{t-1}, U_{t-1}, U_t, m) \). To correct the mismatch between the proposal and the target distribution the particles are weighted according to:

\[
w_{t,j}^{|j|} = \frac{p_{\text{target}}(s)}{q_{\text{proposal}}(s)} \propto p(Z_t|s_{t,j}^{|j|}, m). \tag{12}
\]

The weighted particles are then resampled according to their importance, effectively approximating (10). The resampling step favors samples in regions of high posterior probability, and thus focus the computational power only in relevant regions of the state space. The sequential iterations should progressively reduce the final error and refine the estimations as new observations are acquired.

5.1.1 Prediction Step

The motion model implemented in the prediction step is the odometry-based sampling method presented in [20]. The motion between sequential states is uncoupled into a unique combination of three elementary movements: a initial rotation \( \delta_{rot} \), followed by a straight line translation \( \delta_{trans} \), and a final rotation \( \delta_{rot2} \). Each movement is corrupted by random noise normally-distributed yielding \( \delta'_{rot1}, \delta'_{trans} \) and \( \delta'_{rot2} \), which represent the uncertainty in the odometry measurements. Thus, the particles in \( S_t \) are sampled according to \( p(s_t|s_{t-1}, U_t, m) \) by transforming the particles of \( S_{t-1} \) with:

\[
x_t = x_{t-1} + \delta'_{trans} \cos(\theta_{t-1} + \delta'_{rot1})
\]
\[
y_t = y_{t-1} + \delta'_{trans} \sin(\theta_{t-1} + \delta'_{rot1})
\]
\[
\theta_t = \theta_{t-1} + \delta'_{rot1} + \delta'_{rot2}
\]

The noise is sampled per-particle to guarantee the distribution is correctly approximated.

5.1.2 Update Step

Given a new set of observations, \( Z_t \), GLSOR-3D updates the particles according to (12). Following the soft recognition approach, we make no commitments on the nature of the observed object. Instead we assume that the observation could have been generated by any object in the map, and marginalize over all the possible object combinations, as introduced in 4.

However, now we can have more than one partial view descriptor as an observation. Thus, instead of marginalizing over all possible objects in the map \( o \in \mathcal{O} \), as in 4, GLSOR-3D marginalizes over sets of possible object assignment \( y_{1:N} \in \Gamma = \{ o_1, \ldots, o_N \} \), so that \( o_i \neq o_j \) if \( i \neq j \). Thus, we first compute

\[
\tilde{w}_{t,j}^{|j|} \propto p(Z_t|s_{t,j}^{|j|}, m)
\]
\[
\propto \sum_{j \in \Gamma} \prod_{n} p(d_n, |s_{j}^{|j|}, o_n, m) p(T_n|s_{j}^{|j|}, o_n, m),
\]

and then normalize so that \( \sum_j \tilde{w}_{t,j}^{|j|} = 1 \).

5.2. Resample Step

To avoid the computational load from low probability particles, we resample a new set of particles from \( S_t \). The new sample is chosen so that particles with higher weight are more probable of being selected and thus the new weight of all the particles becomes a constant.

5.2.1 State Estimation

At the end of each iteration, GLSOR-3D estimates the state using an approach similar to RANSAC. We randomly choose one percent of the particles and, for each, we compute the number of particles that are closer than a given neighborhood of radius \( \tau \). Our state estimation corresponds to the particle with more neighbors. This approach allows to obtain better states estimates that converge to the solution.

6. EXPERIMENTAL SETUP

We validate our localization algorithm in a diversified set of experiments, taking place in our office, using every day objects such as sofas and chairs. We designed the experiments in order to:

1. provide empirical evidence that GLSOR-3D effectively estimates a correct final object by disambiguating and reducing the errors as new observations are integrated;
2. evaluate the impact of the soft object recognition versus the use of purely geometric terms obtained with the 3D information.

3. show how the inclusion of multiple objects leads to better estimations from the observations.

6.1. Data collection
To construct both the object datasets as the map we used augmented reality markers. These markers [8], resembling QR codes, are easy to identify in RGB images and provide the means to estimate both the 3D position and orientation of the observer with respect to the marker. The markers can then be combined to define an exterior coordinate system, where the observer can localize himself as long as one of the markers is in view.

Thus, we placed a set of markers attached to and around each object, as illustrated in top row images in Figure 10. We then acquired several partial views completing a full circle around the object. To extract the partial views from the surrounding background we employed the segmentation method described in Section 3.1. From the partial views we computed the descriptors as represented in the bottom row of Figure 10. The descriptors were then stored in the dataset together with the object label and the respective view angle.

Using another set of markers, we constructed a global reference system for our experiments. The markers previously attached to each object provided the respective pose in the global map, allowing for an easy creation of different maps, with different object layouts. Namely, we created a total of 9 maps, from where we gathered information over 22 different paths. The maps differed with respect to the number of objects as: a) a single object - total of seven paths, one per object, except for the maple that had 3; b) pairs of objects - total of six paths, two per each of three different layouts; and finally c) all objects - total of seven paths in a single layout.

We run the experiments using a hand-held Kinect camera, moving around the objects. We obtained the odometry data using the markers [8] to evaluate changes in position, which we then corrupted by adding Gaussian noise. Furthermore, observational evidence showed that the sensor had an error in the object position vector of 15 cm. Thus, all our results assume a $\sigma_T = 15$. The experiments were run using $J = 2000$ particles.

7. RESULTS
In Figure 11 we represent a set of steps in the execution of GLSOR-3D. We can see that initially, the particles are scattered around two objects, but as the robot moves and more data is collected, the particles get centered around the ground truth. The same behavior is also noticeable in Figure 12, where we can see that the error on both position and orientation decrease very fast with the first observations, but then remain constant at around 15 cm.

In Table 1 we compare the results obtained with GLSOR-3D, under the different set of objects. As expected, observations containing multiple objects provide more reliable estimations. For single objects, frequently the descriptor is not able to correctly identify the view angle due to geometric symmetries, resulting in estimations fairly off from the truth. This was specially noticeable for the trash can and the robot examples.
which have roughly a square symmetry and thus ambiguous observations. When pairs of objects are used, symmetries are easier to disambiguate and a poor view angle estimation from one object might be corrected by the other. When using all objects, the observations usually either capture just one or two objects at the same time. Therefore the resulting error falls between the two cases.

For comparison purposes, we run the same experiments discarding the information provided by the observed descriptors, $d_n$. Thus, the localization depends only on the relative position provided by the 3D information, $T_n$. In this experiment we use only the sets containing a pair of objects or all of them. As can be seen in Table 2, the errors greatly increase, as expected. Most of the localization errors are due to unsolved ambiguity. For the case of the pairs of objects this becomes common as two landmarks result in a estimation focused in two symmetric regions (Figure 13a). In cases with more objects, initial ambiguity might lead to the deletion of particles in the true location, resulting in a final erroneous estimation (Figure 13b). Furthermore, the lack of the information given by the descriptor leads to a slower convergence of the particles, even when correct estimations are achieved (Figure 14), and often results in sparser concentration of the particles.

With these results, we finally prove that the developed approach is indeed able to perform localization using multiple objects as landmarks and provides better results that a similar approach without object identification would.

8. CONCLUSIONS AND FUTURE WORK

In this paper we presented a method for mobile robot localization using multiple objects as landmarks while avoiding explicit classification and dispensing registration against complete 3D models for pose estimation. This was achieved by exploiting the PVHK descriptor

<table>
<thead>
<tr>
<th>Objects</th>
<th>Mean</th>
<th>Median</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>0.2279</td>
<td>0.2057</td>
<td>7.4610</td>
<td>6.6865</td>
</tr>
<tr>
<td>Pairs</td>
<td>0.1271</td>
<td>0.1243</td>
<td>2.9535</td>
<td>2.5121</td>
</tr>
<tr>
<td>All</td>
<td>0.1909</td>
<td>0.1949</td>
<td>4.3336</td>
<td>4.3128</td>
</tr>
<tr>
<td>Total</td>
<td>0.1847</td>
<td>0.1586</td>
<td>5.0142</td>
<td>4.8695</td>
</tr>
</tbody>
</table>

### Table 2: Final Localization Errors w/o descriptors

<table>
<thead>
<tr>
<th>Objects</th>
<th>Mean</th>
<th>Median</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pairs</td>
<td>2.5232</td>
<td>3.1588</td>
<td>118.33</td>
<td>174.03</td>
</tr>
<tr>
<td>All</td>
<td>1.1274</td>
<td>0.2332</td>
<td>26.97</td>
<td>7.3373</td>
</tr>
<tr>
<td>Total</td>
<td>1.7716</td>
<td>0.2894</td>
<td>69.13</td>
<td>7.3522</td>
</tr>
</tbody>
</table>

Figure 11: Execution of the implemented algorithm. Only a sample of the total particles are presented.

Figure 13: Examples of errors obtained with the particle filter without using descriptors. Left: Pair of object. Right: All objects.

Figure 14: Comparison between the results with (left) and without (right) descriptor after 5 iterations of the particle filter. The final result after more iterations is correct in both cases.
and the MHD as tools to recognize and compare the similarity between objects represented by their partial views. A particle filter was used to deal with the noise and ambiguity, recursively estimating the most likely state.

An interesting point of future research is to merge this work and the one presented in [6] to obtain both a localization and recognition method that simultaneously estimates the most probable solution for both problems, possibly improving the results of both.

Moreover, we also presented a novel segmentation approach based on spectral clustering and similarity graphs. This approach segments the scene based on geometric and chromatic characteristics of the acquired surfaces, partitioning the data into homogeneous clusters. The experimental results obtained are encouraging, opening the door for future, more exhaustive work on extracting objects from a scene using spectral methods.

9. REFERENCES