Abstract
Several works deal with 3D data in mobile robotic problems. Data come from any kind of sensor providing a huge amount of unorganized 3D data. In this paper we detail an efficient method to build complete 3D models including planar surfaces and creases from a 3D scene. This information can be used to compute the movement performed by a mobile robot by means of a 3D models registration algorithm. Some promising results are shown for both outdoor and indoor environments.

I. Introduction
One of the central research themes in mobile robotics is the determination of the movement performed by the robot using its sensors information. The methods related with this research are called pose registration and can be used for automatic map building and SLAM [1][2][3]. Our main goal is to perform six degrees of freedom (6DoF) pose registration and SLAM in semi-structured environments, i.e., man-made indoor and outdoor environments. We use dense raw 3D data as input sets. Our method is developed for managing 3D point sets collected with any kind of sensor. For our experiments, we use two main data sources: a sweeping unit with a 2D laser Sick and a Digiclops stereo camera, mounted on a mobile robot. Sweeping laser provides 3D data with a low error and a higher range compared to stereo systems, but data from this sensor is slower to retrieve than a stereo system. We are also interested in dealing with outliers, i.e., environments with people or not modeled objects. This task is hard to overcome because classic algorithms, like ICP and its variants, are very sensitive to outliers. Furthermore, we will not use odometry information.

Nevertheless, handling raw 3D data is not suitable for the most of the mobile robot methodologies. In this paper we propose a new method for extracting and modeling planar patches and creases (3D features) from the 3D raw data. There are two main reasons for that: first, a complexity reduction (when comparing with raw data) is done and time and memory consumptions are improved (we obtain over 500 features from 100000 3D points); second, outliers are better overcomed using those features, as points not supported by a planar patch or a crease are deleted. Planar patches are useful features as a man-made environment is easily described with them. Nevertheless, the approach of representing a 3D scene only by means of planar surfaces may not be enough in some situations. We need extra information about environment in order to find robot movements such as crease surface information. Surface creases have numerous applications in geometric modeling [4][5][6], image processing [7][8], other fields [9], and also for the resolving the SLAM problem [10].

The rest of the paper is organized as follows: first, a section describing the physical systems used for experiments; then, our planar patches and creases extraction method is described; the experimental section will show our modeling results and its application on egomotion, finishing with our conclusions and future work in the last section.

II. Physical systems
One of the goals of our work is the independence of our algorithm: it can be applied to any robot platform, any 3D measurement device and can be used in indoor and outdoor handmade environments. We present here the physical systems used in our experiments.

We have used several robot platforms, depending on the perception system used. In Figure 1 the two platforms are shown. The left one is an indoor platform, a Magellan Pro from iRobot. It is used in indoor experiments, given its dimensions (diameter: 40cm, height: 24cm). For outdoor we have used a PowerBot from ActiveMedia. It has a battery life of 5 hours, which is necessary for long experiments. Furthermore, PowerBot can carry the 3D sweeping laser unit, that is very heavy. Both come with an onboard computer.

We are going to manage 3D data and these data can come from different devices (see Figure 2). First, we use a stereo camera Digiclops from Point Grey. It has a range of 8 meters and is ideal for indoor environments. It can provide 24 images per second with grey level information for each point. However it suffers the lack of texture: areas in the environment without texture, can not provide 3D data. Furthermore, there is a measurement error of 10%. For outdoor environments we use a 3D sweeping laser unit. It is a LMS-200 Sick laser mounted on a sweeping unit. Both can provide 3D data but without color information. It doesn’t suffer the lack of texture and its range is 80 meters with an error of 1mm for every meter. The main disadvantage of this
unit is the data capturing time: it takes more than one minute in one shot.

III. Features extraction method

We can reduce the amount of information contained in a 3D scene by modelling object surfaces included in it. Normal vectors estimated from a local area around each 3D point into the scene is a good starting point for obtaining surfaces descriptions. Some methods, such as [11] or [12] were developed for handling noisy input data sets. The basic idea consists in analyzing each point in the local neighbourhood by means of a robust estimator. In [13] a singular value decomposition (SVD) based estimator is used for obtaining surface normal vectors. Using this method, when the underlying surface is a plane, the minimum singular value is quite smaller than the other two singular values, and the singular vector related to the minimum singular value is the normal vector of the surface at this point. On the other hand, creases usually come from building corners, window and door frames, trunk trees, etc. In these situations, the maximum singular value is quite higher than the others and the singular vector related to the maximum singular value is tangent to the surface and its direction coincides with the direction of the crease. An example is shown in Figure 3.

From this information we can label each point in a 3D scene as belonging to a planar surface, when one of the singular values is much smaller than the others; belonging to a crease, when one of the singular values is much higher than the others; or not defined objects in other case. In Figure 4 we can see an example of applying this segmentation for both outdoor and indoor scenes. Despite the segmentation of the scene points, we need to do some extra work to extract planar patches and creases in the scene. We use a template matching for fitting the labeled points into a planar patch or a crease model.
This process retrieves the underlying surface normal vector of a given set of points. Furthermore, a threshold called thickness can be defined from singular values in order to determine in which situations a point, as well as its neighbourhood, belong to a planar surface or not (crease candidate or noise). This thickness value can be used to measure the fitting of a 3D point set to a plane. The lower thickness value we find, the better fitting between points and planar surface is. The size of the window used for obtaining neighbour points has an important impact on the results. As it is considered in [10], sample density of 3D laser range finder data presents large variations due to the divergence of consecutively sampled beams. If we take constant size neighbourhood we can not obtain planar patches at a certain distance where points are too far from each other. Also, its necessary to ensure a minimum number of points inside the window in order to make reliable the Singular Value Decomposition result. [13] uses a minimum number of points of nine. Furthermore, the window has to be large enough to be able to understand the spatial points organization inside it. In [13] and [14] adaptative online window radius methods are proposed. Nevertheless, we propose a dynamic size window that just depends on the distance between a point in the 3D scene and the coordinate origin (viewpoint). A factor is used here to ensure an optimal window size. This factor has been empirically stated. Figure 5 shows the results of using different factor values. From these results, a value of 0.03 represents the best choice.

Using SVD based normal vector estimation method we can obtain a model that represents the planar surfaces in the scene. We propose an optimal method that can obtain a planar patch model from a 3D point set in $O(\log n)$. This method is based in automatic seed selection methods [15] [16]. The idea consists in performing a selection of the most representative points in the whole 3D scene. These selected points must belong to a planar surfaces. To ensure this we use the thickness value, that can be obtained from the SVD based estimation method as we described above. In order to find out the most representative points we select points in the scene, in a random order, until all points are visited. For each point visited we compute its normal vector and thickness value. If its thickness value is low enough, the point is inserted into the most representative points list and its neighbours inside the window used for computing its normal vector are marked as visited. When this process ends, planar patches model is directly computed from the most representative points and its normal vectors. The size of the planar patches depends on the size of the window used to compute normal vectors. The overall procedure for fast planar patches estimation can be found in Figure 6.

In Figure 7 we can observe the result of applying this method for computing planar patches models from 3D scenes captured by a 3D range laser. Since our method is supposed to work using any data source, the result of applying it to a stereo 3D image can be observed in Figure 8. At the same time we look for planar patches, we use a merging algorithm in order to obtain the creases of a scene. A crease $C_i(p_i, \bar{d}_i, l_i, w_i)$ is described by four parameters: its position vector, its director vector, its length and its width. The width measures the mean of the distance from the crease

```
function Fast_Patch_Estimation (II: 3DPoint_set; thick : R)
return: planar_patch_set
var p_i: 3DPoint; Q, neighs: 3DPoint_set; w : R;
< n_i, γ_i >: normal-and-thickness_tuple; N : N
begin
Q := II
while (Q ≠ Ø) do
N := ||Q||
p_i := Remove(Q, Random(N))
w := ComputeWindowSize(p_i)
neighs := GetNeighbors(II, p_i, w)
< n_i, γ_i > := NormalSVD(p_i, neighs)
if (γ_i < thick) then
Add(result, newPlanarPatch(p_i, n_i, w))
RemoveAll(Q, neighs)
endif
endwhile
Fast_Patch_Estimation := result;
end.
```

Fig. 5. Window size factor’s influence. Upper chart shows the number of neighbouring points inside window at different point distances. Bottom chart shows the percentage of actually computed normal vectors.
to its supporting points. First of all, we consider all points previously labelled as crease to be small creases through the whole scene. A pair of creases $C_i$ and $C_j$ will be merged if they fulfill crease constraint, it is said, if the angle between $\vec{d_i}$ and $\vec{d_j}$ vectors is under a threshold $\alpha$ and $\vec{p_i}$ lies on the crease $C_j$ with an error margin under $l_j + \beta$ along the direction of the crease and under $w_j + \gamma$ in the perpendicular direction of the crease. $\alpha$, $\beta$ and $\gamma$ are empirically started to 0.2 rad, 0.15 m and 0.02 m respectively.

The merging algorithm has two steps that are iterated until no more merges can be found. In the first step, all possible merges are performed using the previous criterium. The points that support each new crease are retained. The second step consists in computing creases parameters from its supporting points. When this process finishes, creases with few supporting points are removed. We can observe the results of applying this method in figure 9.
IV. Using 3D models: 6DoF egomotion

In the previous section we described a method for building 3D models from scenes captured with a 3D sensor. Therefore, we want to use these models to achieve further mobile robot applications in real 3D environments. We can use our previous approach on egomotion using 3D models [17] for computing the robot movement from the 3D models extracted at each pose the robot has been. The basic idea is to take advantage of the extra knowledge that can be found in 3D models such as surfaces and its orientations. This information is introduced in a modified version of an ICP-like algorithm in order to reduce the outliers incidence in the results. ICP [18][19][20][21] is widely used for geometric alignment of a pair of three-dimensional points sets. From an initial approximate transformation, ICP iterates the next three steps until convergence is achieved: first, closest points between sets are stated; then, best fitting transformation is computed from paired points; finally, transformation is applied. In the mobile robotics area, the initial transformation usually comes from odometry data.

Nevertheless, our approach does not need an initial approximate transformation like ICP based methods do. We can use the global model structure to recover the correct transformation. This feature is useful for those situations where no odometry is available or it is not accurate enough, such as legged robots. In our case, we are going to exploit both the information given by the normal vector of the planar patches and its geometric position. Whereas original ICP computes both orientation and position at each iteration of the algorithm, we can take an advantage on the knowledge about planar patches orientation for decoupling the computation of rotation and translation. So we first register the orientation of planar patch sets and when the two planar patches sets are aligned we address the translation registration. In Figure 10 we can observe the steps performed for computing the alignment between two sets of planar patches. The top image shows a zenithal view of two planar patches sets computed from two consecutive 3D scenes obtained by a robot during its trajectory. The middle image shows the result of rotation registration. Finally, bottom image shows the result after the translation between planar patches sets is computed.

V. Conclusions and future work

We have presented a new method for computing 3D models from unorganized raw 3D data. We don’t need to know anything about the kind of sensor used for obtaining data so the method we propose can be used with the most of the 3D scanner devices. First, we have explained an algorithm for computing the planar patches that fits with the planar surfaces in the 3D scene. This is a low complexity method that can be used for obtaining online 3D models. Results have been shown for both stereo and 3D range laser. We have also presented a crease estimation method that represents

![Fig. 10. Planar patches matching example. For all the three images patches from the model are painted in dark grey whereas scene paths are represented in light grey. Top, initial situation. Middle, after rotation registration. Bottom, final result after translation registration is completed.](image-url)
a model improvement. We can recover crease information using the spatial analysis used for obtaining planar patches. The usefulness of our models is demonstrated by applying an 6DoF egomotion algorithm that uses those models as input for computations.

As future work we plan to improve the accuracy and performance of our creases estimation method in order to use it together with planar patches sets in 6DoF egomotion or 6D SLAM.

Acknowledgments

This work has been supported by project DPI2005-01280 from Ministerio de Educación y Ciencia (Spain).

References


