

Active Stereo Based Compact Mapping*

Diego Viejo, Juan Manuel Sáez, Miguel Ángel Cazorla and Francisco Escolano

Robot Vision Group

Departamento de Ciencia de la Computación e Inteligencia Artificial

Universidad de Alicante, Ap.99, E-03080, Alicante, Spain

Email: {dviejo, jmsaez, miguel, sco}@dccia.ua.es

Abstract—In this paper we propose a method for extracting the planes from a 3D dense map. Three-dimensional data is acquired using active stereo in order to fill texture gaps which are typical in indoor environments. Then, a randomized SLAM algorithm recently proposed by the authors is applied to compute the 3D map by tele-operating a mobile robot. A 3D mesh generation algorithm, specially designed to triangulate not solids but open objects, is the basis for computing the main planes in the map after clustering the normals of the vertices in the mesh. Finally, we present our experimental results in indoor environments.

Index Terms—Compact Mapping, Active Stereo, SLAM, Mesh generation

I. INTRODUCTION

In this paper we present a method for extracting the planes of an environment with *active stereo*. Stereo-vision cameras are interesting 3D sensors for many reasons: (i) their lower cost in comparison to multiple range finders, (ii) their bio-compatibility, allowing body scans, and (iii) they provide both range information and also appearance information, and the latter is very useful, not only for improving the navigation/exploration algorithms but also for texture mapping. As appearance information is not provided by range finders, some recent approaches for 3D mapping with mobile robots [1] [2] [3] add an additional calibrated camera in their settings.

Active stereo has its origin in the concept of *structured light* and it relies on projecting light patterns for dealing with the lack of texture information in certain parts of the environment. This mechanism improves the robustness of the stereo system yielding more dense and exact 3D data. In contrast to structured light systems [4] [5] [6][7], where specifically-coded patterns are required for solving the correspondence problem, active-stereo patterns are not constrained provided that they cover the complete field of view. In this work we will use patterns built in lines randomly oriented and colored which allows to reduce the natural ambiguity of the environment.

Given a set of observations performed with active stereo, our goal is to obtain the planes of the environment once the SLAM problem has been solved and provided that the

*This research is partially funded by the project TIC2002-02792 of the Spanish Government



Fig. 1. Tele-operated RWI Mobile robot endowed with a trinocular stereo system (Digiclops) and a LCD projector.

data density yielded by the sensor is good enough. In order to compute the dense 3D map we may apply, for instance, EM-like approaches, [2], and ICP-like approaches [8][9]. In this paper we will apply the randomized global rectification approach that we have recently proposed for solving the SLAM problem using stereo vision [10].

In order to obtain the planes from the huge 3D point cloud (map) returned by our SLAM method, here we propose a robust 3D mesh generation algorithm which is not constrained to deal with closed surfaces but specifically designed for dealing with planar surfaces. Our algorithm is rooted in the DeWall algorithm [12] which does not assume that 3D data comes from closed volumes and computes the 3D Delaunay triangulation, in contrast with other methods, like [13], [14] which compute the triangulation of a 3D solid, or like [15], [16] which are designed to build topographic maps through 2D triangulations. Once the triangulation is performed, we obtain the 3D normals of the vertices using the approach described in [17] and cluster them for finding planes.

The rest of the paper is organized as follows. In Section II we present the technical details of the approach: (i) 3D acquisition using active stereo, (ii) SLAM using the randomized global rectification algorithm, (iii) 3D mesh generation and the finding of planes. In Section III we present several experimental results obtained in indoor environments, and in Section IV we outline our conclusions and future work.

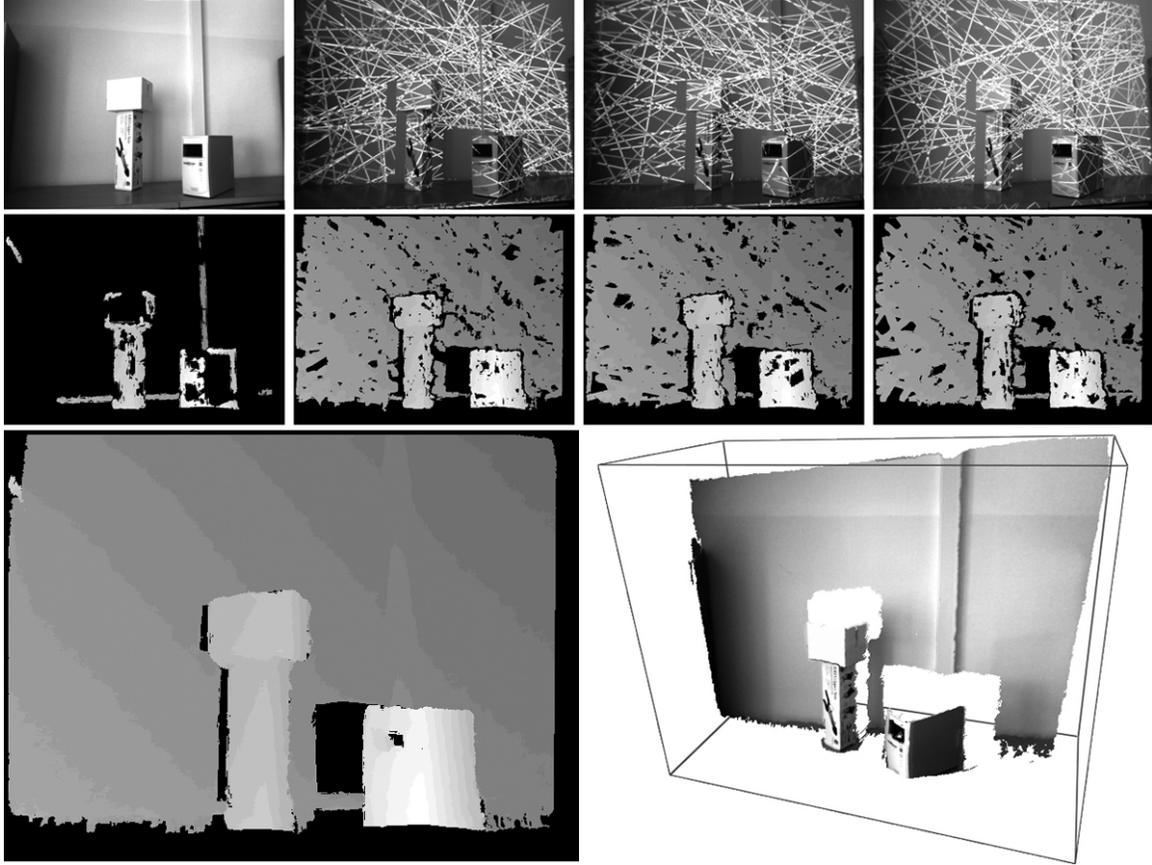


Fig. 2. Acquisition using active stereo. First row: Reference image and three views corresponding to different projected patterns. Second row: The initial disparity image (low textured) and the three disparity maps associated to the projected patterns. Bottom: The fused disparity map (left) and the 3D model of the scene (right).

II. APPROACH DESCRIPTION

A. 3D Image Acquisition

In this work we have tele-operated a RWI Magellan pro mobile robot equipped with a Trinocular Digiclops stereo system which is able of taking 640×480 pixel images at 14 fps (see Fig. 1). The system includes a LCD projector with a resolution of 1024×768 pixels for projecting line patterns where both the orientation and intensity of the lines is selected randomly. Although in structured light approaches classical patterns are built in vertical bars with different frequencies, such patterns are devoted to simplify the correspondence problem. Here we are interested in producing textures with enough heterogeneity to feed the correspondence algorithm of the stereo system in those areas without texture (there are many of them in a snapshot of a typical indoor environment). Line patterns must be random in order to avoid causal coincidences, that is, ambiguity in the stereo process. Furthermore, an unique pattern is not enough to obtain good results. In order to obtain a compact data of the environment it should be convenient to fuse many disparity maps and thus

we need to project many line patterns (experimentally, we have found that 10 patterns/views yield an acceptable result).

Each observation at a given pose of the robot is acquired by projecting $N = 10$ random light patterns each one with typically $L = 200$ lines (Fig. 2). For each projected pattern we compute its associated disparity using the stereo camera. Then we compute a simple robust statistic (median) and retain for each pixel a representative disparity along the N disparity maps. Then, we compute the 3D coordinates associated to each representative disparity. In Fig. 3 we show two examples of 3D acquisitions corresponding to the indoor corridor mapped in our experiments. As shots are taken at night the practical field of view is conditioned by the aperture of LCD projector. Such aperture has been corrected by means of a set of lenses.

B. Map Building

Given a low-textured environment (indoor), in order to build a dense 3D map while we are tele-operating a robot, we must fuse the N 3D observations C_0, C_1, \dots, C_{N-1} performed from different poses $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{N-1}$. Assuming

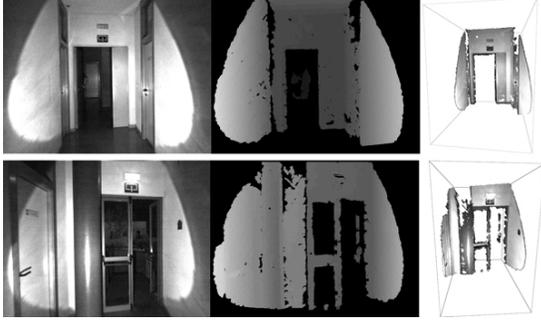


Fig. 3. 3D acquisition in a low-textured corridor. Reference images (left), fused disparities (center), and 3D views after mapping the textures of the reference images (right).

that the robot is confined to the XZ horizontal plane and also that the stereo system is fixed, each pose can be described as $\mathbf{p}_t = [x_t, z_t, \theta_t]$ (coordinates in the horizontal plane and rotation around the vertical axis Y). Given the pose \mathbf{p}_{t-1} , the next pose \mathbf{p}_t is obtained through the incremental action $\mathbf{a}_t = [\delta x_t, \delta z_t, \delta \theta_t]$ between two consecutive observations (3D acquisitions) \mathbf{C}_{t-1} and \mathbf{C}_t .

The mapping problem consists of estimating the incremental actions between two consecutive poses and then use these actions to take all 3D acquisitions to a common reference system. The result is a 3D dense map of the environment. The mapping approach used in this work consists of the two steps (see more details in [10] [11]):

1) *Egomotion/Action Estimation*: The goal of egomotion/action estimation is to estimate the action \mathbf{a}_t between each pair of consecutive observations \mathbf{C}_{t-1} and \mathbf{C}_t of the trajectory, using only visual cues. Here, and for the sake of completeness, we summarize the three stages of our algorithm:

- *Feature Extraction and Matching*. Given the 3D point cloud \mathbf{C}_t observed from the t -th pose, and for the sake of both efficiency and robustness, we retain only those points $\mathbf{M}_t = [x_t, y_t, z_t]^T \in \mathbf{C}_t$ whose projections in the t -th right stereo image (reference image) are associated to strict local maxima of the image gradient. These points define the constrained cloud $\tilde{\mathbf{C}}_t$ that will be used in this algorithm. However, in order to find the correspondences between points of different constrained clouds, we exploit the correlations of their local appearance considering two types of invariances: illumination (Pearson coefficient) and texture deformation (Log-polar transform).
- *Matching Refinement*. Matching candidates with low strength, low distinctiveness or non-bidirectionality are discarded. However, despite considering the three later conditions, the matching process is prone to outliers which must be identified and removed. For instance, if the i -th point \mathbf{M}_t^i of $\tilde{\mathbf{C}}_t$ matches the j -th point

\mathbf{M}_{t-1}^j of $\tilde{\mathbf{C}}_{t-1}$, and similarly \mathbf{M}_t^k matches \mathbf{M}_{t-1}^l , we will remove the first match if the following quantity is high enough:

$$D_{ij} = \frac{\sum_k \sum_l D_{ikjl}}{|\mathcal{M}|}, \quad (1)$$

where \mathcal{M} is the current set of matches, and D_{ikjl} is the maximum ratio between $\|\mathbf{M}_t^i - \mathbf{M}_t^k\|$, and $\|\mathbf{M}_{t-1}^j - \mathbf{M}_{t-1}^l\|$, which must be close to the unit in good matches (similar relative distances). Then, in the *leaving-the-worst-out* process, matches with higher D_{ij} are iteratively considered for removing.

- *Action/Transformation Estimation*. Once we perform the latter matching refinement, which in practice is robust enough to avoid a interleaved EM-like process, we proceed to estimate the rotation \mathbf{R}_t and translation \mathbf{t}_t associated to action \mathbf{a}_t . Given that each point $\mathbf{M}_t^i \in \tilde{\mathbf{C}}_t$ matches $\mathbf{M}_{t-1}^j \in \tilde{\mathbf{C}}_{t-1}$ the optimal action is the one minimizing the usual quadratic energy function

$$E(\mathbf{R}_t, \mathbf{t}_t) = \sum_i \sum_j \mathbf{B}_{ij} \|\mathbf{M}_{t-1}^j - (\mathbf{R}_t \mathbf{M}_t^i + \mathbf{t}_t)\|^2, \quad (2)$$

being \mathbf{B}_{ij} binary matching variables (1 when \mathbf{M}_t^i matches \mathbf{M}_{t-1}^j and 0 otherwise). To perform the optimal transformation, we use a conjugate gradient descent, using equation 2 like a energy function.

2) *Map Building and Rectification*: Given a robot trajectory $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{N-1}$ of size N , and a sequence of estimated actions $\mathbf{a}_0, \mathbf{a}_1, \dots, \mathbf{a}_{N-1}$ with length N , being $\mathbf{a}_0 = \mathbf{p}_0$, an initial approximation of the 3D map comes from superposing all the point clouds with respect to the referential pose \mathbf{p}_0 . We define the initial map, $\mathbf{A} = [p = (x, y, z) | p \in \mathbf{M}_t]$ to be the union of all the 3D measurements in a common reference frame. However, as this map accumulates the errors produced at local action estimations (errors due to the latter algorithm, or even to the absence of 3D cues when non-textured parts of the environment are observed), it is desirable to provide a consistency criterion and an updating strategy that exploits it in order to obtain a globally-consistent map.

Our information maximization criterion for measuring the global consistency of the complete (not reduced) point cloud of the current map \mathbf{A} is the minimization of the energy:

$$E(\mathbf{A}) = H(\mathbf{q}_{XZ}) + \mu(H(\mathbf{q}_X) + H(\mathbf{q}_Z)), \quad (3)$$

where $H(\cdot)$ denotes the entropy of the argument. In the latter criterion we consider three probability distributions. We discretize the XZ plane into a grid of rectangular bins, and define $q(x, z)$ to be the number of 3D points in \mathbf{A} that lie within bin x, z . $\mathbf{q}_{xz}(x, z)$ is then defined as the probability distribution $\mathbf{q}_{xz}(x, z) = q(x, z) / \sum_x \sum_z q(x, z)$ associated to normalizing, after a proper discretization of the XZ plane, and for each pair $x \in X$ and $z \in Z$, the sum $q(x, y)$ of all points $\mathbf{M}^i = [x^i, y^i, z^i]$ in the map \mathbf{A} satisfying both $x^i \approx x$

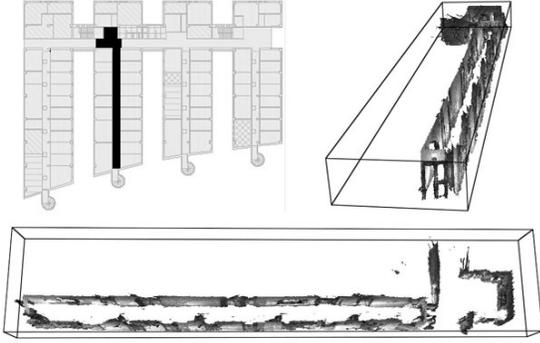


Fig. 4. Mapping example. Top-left: Real plane of the scanned corridor. Top-right and bottom: Different perspectives of the obtained 3D map.

and $z^i \approx z$. In practice, and prior to computing \mathbf{q}_{XZ} , the sum $q(x, z)$ should be smoothed by a convolution with a Gaussian kernel of variance σ_{XZ} in order to combine the projections of walls that are close enough with respect to that variance.

On the other hand, the second term of Eq. 3, $\mu(H(\mathbf{q}_X) + H(\mathbf{q}_Z))$ where $\mu \geq 0$ is a scaling factor, is only applicable to plane-parallel environments, that is, in environments where the main planes (walls, doors, floor, ceiling, and so on) are either parallel or orthogonal. This allows, for instance, to correct a typical straight corridor that appears slightly curved if we only minimize the sum of local energies. In order to do so, we compute the marginal distributions $\mathbf{q}_X(x) = \sum_z \mathbf{q}_{XZ}(x, z)$ and $\mathbf{q}_Z(z) = \sum_x \mathbf{q}_{XZ}(x, z)$. Intuitively, if a corridor is perfectly orthogonal to the X axis, we will observe two well-defined peaks in the projection and $H(\mathbf{q}_X)$ will be minimal with respect to any other rotation. Furthermore, the sum $H(\mathbf{q}_X) + H(\mathbf{q}_Z)$ is maximal when either the corridor is orthogonal to X or to Z . Therefore, this second term, that we call alignment term prefers maps yielding peaked projections and this is why curved corridors may be rectified (the starting pose should be aligned with the main building directions).

To estimate the set of actions that minimize equation 3, we use an entropy minimization algorithm based on a voting schema. The underlying idea is to modify a subset of actions (not only one) simultaneously, modifying each action proportionally to the changes that this action has produced on the global energy in the past (see [11] for a complete description of this algorithm).

In Fig. 4 we show an example of a map estimated with the algorithm (egomotion and global rectification) described above. This map is built in 106 observations. The complete 3D point cloud has 14.697.233 points, that is, each observation has 138.653 on average.

C. Plane Extraction

Stereo techniques are unable to recover untextured areas, as shown in Figure 2. The use of structural light helps to recover

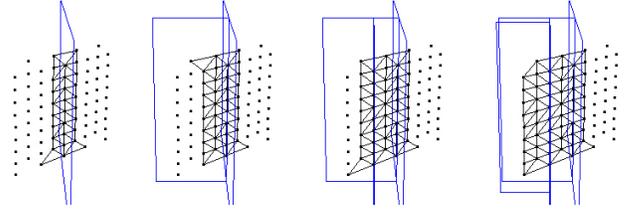


Fig. 5. Mesh generation process. Beginning on the left, first splitting plane divide the space. After a recursive call a new splitting plane is generated, orthogonal to the first one. The last two images show two new planes. In each iteration the triangles intersecting the new splitting plane are generated.

these areas. On the other hand, the error in measurements (3D point position) are high enough which makes useless approaches based on principal vectors in order to compute normals in points. Thus, in order to extract planes from the points in 3D dense map we perform two steps:

- 1) *Mesh Generation/3D Triangulation* The first one is mesh generation. We estimate a mesh from the 3D data. Our approach is rooted on the DeWall algorithm [12] that follows a divide-and-conquer strategy by reversing the order between the solutions of sub-problems and the merging phase.

In order to build our mesh, we use this idea and then we incorporate the constraints imposed by the problem. The DeWall algorithm uses a tetrahedron as geometric basic entity to build a mesh from a set of 3D points that form a closed volume. The assumption of a closed volume is not fulfilled in our case and it would be a source of problems due to the high noise levels that we must handle. For this, we are going to use the triangle as geometric basic entity to calculate the triangulation, in spite of the tetrahedron. This idea arises from the fact that we are triangulating points from 2D planes inside a 3D environment and, in general, to build a triangulation from 2D points the basic geometric entity is the triangle. In addition, we have to consider the fact that the triangle size constraint has a consequence: it might happen that certain parts of the space remain unconnected and, therefore, not be triangulated. For this reason, we compute clusters of connected points which will be the input to this triangulation process.

Our mesh generation approach is as follows: First, we use a plane α to split the space in two half spaces. Then, we compute the merging triangulation Σ_α using the splitting plane α . The technique used to build the Σ_α is a slight variation of an incremental construction algorithm: a starting triangle is founded and then Σ_α is built by adding a new triangle at each step. To find the first triangle we select the nearest point p_1 to plane α . Then, we select p_2 such that it is the nearest point to p_1 on the other side of α . From p_1 and p_2 we search the point p_3 such that the *circum-*

circle around p_1 , p_2 and p_3 has the minimum radius r_i . The center of the *circum-circle* are extracted and we can compute a sphere with center c_i and radius r_i . Finally, to accept p_3 as the point which completes the triangle, it has to fulfill a pair of conditions. First, the Delaunay condition: no one point is inside the sphere; second, the triangle size constraint: point p_3 is near enough from p_1 and p_2 .

```

function Triange_Builder (P: point_set,  $L_\alpha$ : side_list)
return: triangle_list
var f: side; ,  $L_l$ ,  $L_r$ : side_list;
    t: triangle;  $\Sigma$ : triangle_list;  $\alpha$ : splitting_plane;
begin
  if  $L_\alpha = \emptyset$  then
    t:=FindFisrtTriangle(P,  $\alpha$ );
     $\Sigma := \Sigma \cup t$ ;
    for each  $f'$ :  $f' \in$  Sides(t) do
      if IsIntersected( $f'$ ,  $\alpha$ ) then Insert( $f'$ ,  $L_\alpha$ );
      if InHalfSpaceL( $f'$ ,  $\alpha$ ) then Insert( $f'$ ,  $L_l$ );
      if InHalfSpaceR( $f'$ ,  $\alpha$ ) then Insert( $f'$ ,  $L_r$ );
  while  $L_\alpha \neq \emptyset$ 
    f:=Extract( $L_\alpha$ );
    t:=FindTriangle(f, P);
    if t  $\neq$  null then
       $\Sigma := \Sigma \cup t$ ;
      for each  $f'$ :  $f' \in$  Sides(t) AND  $f' \neq f$  do
        if IsIntersected( $f'$ ,  $\alpha$ ) then Insert( $f'$ ,  $L_\alpha$ );
        if InHalfSpaceL( $f'$ ,  $\alpha$ ) then Insert( $f'$ ,  $L_l$ );
        if InHalfSpaceR( $f'$ ,  $\alpha$ ) then Insert( $f'$ ,  $L_r$ );
  /*Recursive Triangulation*/
  if  $L_l \neq \emptyset$  then  $\Sigma := \Sigma \cup$  Triange_Builder(P,  $L_l$ );
  if  $L_r \neq \emptyset$  then  $\Sigma := \Sigma \cup$  Triange_Builder(P,  $L_r$ );
  Triange_Builder:= $\Sigma$ ;
end.

```

Fig. 6. Algorithm Triange_Builder computes 3D Delaunay triangulation.

The rest of Σ_α is built from the first triangle. We label each triangle side as whether it is completely contained in one of the two half-spaces (left or right) defined by α or whether it intersects with the plane. We use three lists L_l , L_r and L_α to insert the triangles sides depending on their label. We extract a side from L_α and search for a new triangle, but now we must consider that this side comes from an exiting triangle. Each side of a triangle has a plane β which is perpendicular to the triangle. The plane β divides the space into two half-spaces, therefore, we just search the next triangle in the valid half-space. The sides of the new triangle are now inserted in the corresponding list. The algorithm is shown in Fig. 6. In Fig. 5, we show the process of applying the method to a set of points, and in Fig. 7 we show result obtained in a single view of the mapped corridor.

- 2) *Estimation of 3D Planes.* Once, we have obtained the triangulation we estimate the normals following the

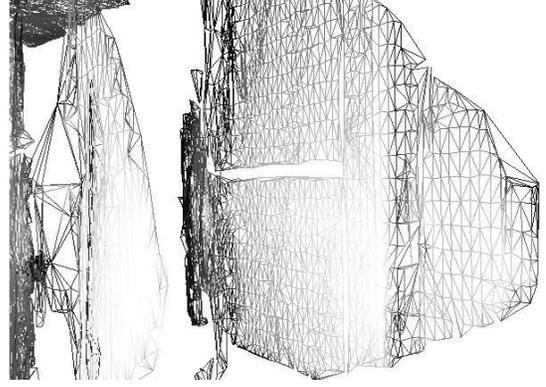


Fig. 7. A mesh generated with the points from a single view.

approach described in [17]. Then, it is possible to know the normal vector at a given vertex by exploiting the underlying geometry of the scene. Then, we proceed to adjust planes to 3D points. Each pair (p_i, n_i) , defined by a point and a normal represents a hypothesis h_i for a possible plane. For each hypothesis h_i from the set H we test whether there is evidence enough in the available data in order to include a new plane. In order to check such a consistency: we search in H the hypotheses $\{h_j\}$ ($i \neq j$) supporting h_i , that is: (i) those hypotheses where the distance of their associated points to p_i is below the threshold $W_d = 0.8m$; (ii) those hypotheses that have been considered before as part of the plane, whose difference with respect to n_i is bounded by the threshold $W_n = \pi/8rad$, and whose distance to the plane is less than $W_t = 0.25m$. Once we find a hypothesis h_j supporting h_i we recompute the parameters of the plane by considering all its supporting hypotheses. Finally, we will include a new plane when its number of supporting hypotheses is greater than $W_h = 50$.

Finally, we must map the texture of the images to the planes found in the previous step. In order to do so, we project the intensity of the 3D points to their closest plane provided that such a distance is below $W_d = 0.3m$.

III. EXPERIMENTAL RESULTS

In this section we show the planes found in a mapped corridor with the approach described above. We present both the result after and before texture mapping. In Fig. 8 we show the reconstruction from the point of view of the robot. The texture mapping process is highly influenced by the limits of the LCD range of illumination. The tele-operating trajectory was designed to see each wall two times. However, certain planes in the ceiling are not identified due to the lack of texture (they were sometimes out of the practical field of view). In Fig.s9 and 10 we see the corridor from the outside. What is interesting of this approach is that it is possible to

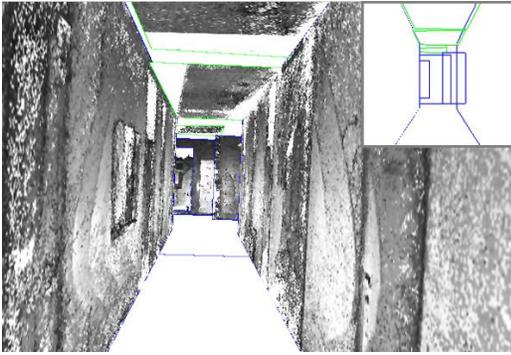


Fig. 8. First result of the reconstruction with the texture mapped on the planes. On the top right hand, we show the planes without texture.

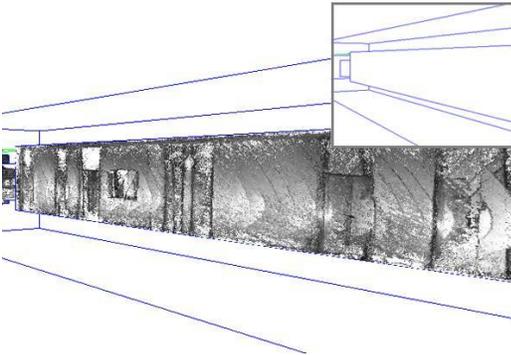


Fig. 9. Second result.

find the main planes of an environment by exploiting stereo data provided that we introduce artificial texture. However, when applying the SLAM algorithm described above to the original stereo data, the obtained map was so sparse that it was very difficult to extract meaningful planes from it.

IV. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a method for extracting 3D planes from dense maps obtained with active stereo (stereo camera with LCD projector), which yields a lower cost alternative with respect to using 3D range scanners. We have proved that the projection of artificial texture may improve notably the quality of the maps of indoor environments obtained with stereo cameras, and also that such a quality is good enough to infer the main planes in these environments. Our future work includes: (i) the design of better mechanisms for fusing multiple disparity maps (statistical consistence measures, and so on), and (ii) the identification of other geometric primitives (cylinders, boxes) with 3D deformable templates.

REFERENCES

[1] C. Martin and S. Thrun. *Real-time acquisition of compact volumetric maps with mobile robots*. In Proceedings of ICRA'02: IEEE International Conference on Robotics and Automation (2002).

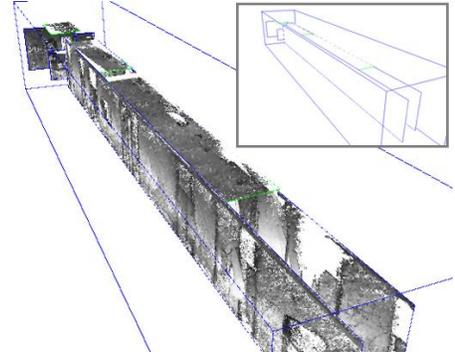


Fig. 10. Third result.

[2] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, and S. Thrun. *Using EM to learn 3D models with mobile robots*. In Proceedings of ICML'01: International Conference on Machine Learning (2001).

[3] D. Hähnel, W. Burgard, and S. Thrun. *Learning compact 3D models of indoor and outdoor environments with a mobile robot*. Robotics and Autonomous Systems, 44 (2003) 15-27.

[4] J. Pagés, J. Salvi, R. García and C. Matabosh. *Overview of coded light projection techniques for automatic 3D profiling*. ICRA'03: IEEE International Conference on Robotics and Automation (2003)

[5] S. Rusinkiewicz, O. Hall-Holt and M. Levoy. *Real-time 3d model acquisition*. SIGGRAPH (2002)

[6] C. Rocchini, P. Cignoni, C. Montani, P. Pingi and R. Scopigno. *A low cost 3D scanner based on structured light*. Eurographics 2001, Volume 20, Number 3.

[7] D. Scharstein and R. Szeliski. *High-Accuracy Stereo Depth Maps Using Structured Light*. Proceedings on CVPR'03: IEEE Conference on Computer Vision and Pattern Recognition (2003)

[8] H. Surmann, A. Nüchter, J. Hertzberg. *An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments*. Robotics and Autonomous Systems, number 45, pp 181-198, 2003.

[9] A. Nüchter, H. Surmann, S. Thrun. *6D SLAM with an Application in Autonomous Mine Mapping*. Proceedings of ICRA'04: IEEE International Conference on Robotics and Automation (2004).

[10] J.M. Sáez, F. Escolano. *A Global 3D Map-Building Approach Using Stereo Vision*. Proceedings of ICRA'04: IEEE International Conference on Robotics and Automation, New Orleans, May 2004.

[11] J.M. Sáez, F. Escolano. *Entropy Minimization SLAM Using Stereo Vision*. Proceedings of ICRA'05: IEEE International Conference on Robotics and Automation, Barcelona, April 2005.

[12] P. Cignoni, C. Montani and R. Scopigno: *DeWall: a fast divide and conquer Delaunay triangulation algorithm*. Ed. Computer-Aided Design 30 (1998) 333-341

[13] E. Mücke: *A Robust Implementation for Three-dimensional Delaunay Triangulations*. In Proceedings of the 1st International Computational Geometry Software Workshop (1995)

[14] K. Hormann and M. Reimers: *Triangulating Point Clouds with Spherical Topology*. Curve and Surface Design (2003) 215-224

[15] M. de Berg, M. van Kreveld, M. Overmars and O. Schwarzkopf: *Computational Geometry, Algorithms and Applications*. Ed. Springer (1991) 181-183.

[16] G. Petrie and T.J.M. Kenne: *Terrain modelling in Survey and Civil Engineering*. Computer Aided Design, 19, number 4 (1987).

[17] D. L. Page, Y. Sun, A. F. Koschan, J. Paik and M. A. Abidi: *Normal vector voting: crease detection and curvature estimation on large, noisy meshes*. Graphical Models, Special Issue on Large Triangle Mesh Models, 64 (2002) 199-229

[18] H.K. Nishihara and T. Poggio: *Stereo Vision for Robotics*, In: Proceedings of the First International Symposium of Robotics Research, Brady and Paul (eds.), MIT Press, Cambridge, MA, 489-505, 1984.