BUILDING 3D MAPS WITH SEMANTIC ELEMENTS INTEGRATING 2D LASER, STEREO VISION AND IMU ON A MOBILE ROBOT

Luca Iocchi and Stefano Pellegrini

Dipartimento di Informatica e Sistemistica
Sapienza University of Rome, Italy
{iocchi,pellegrini}@dis.uniroma1.it

KEY WORDS: 3D mapping, sensor integration, autonomous robots

ABSTRACT:

Building 3D models is important in many applications, ranging from virtual visits of historical buildings, game and entertainment, to risk analysis in partially collapsed buildings. This task is performed at different scales: city, buildings, indoor environments, objects and using different sensors: cameras, 2D and 3D laser, etc. Moreover, different map representation have been considered: metric (or feature based) 3D maps represented as a set of 3D points (plus color information), in contrast with maps represented as a set of semantic structural elements (i.e., floors, walls, steps, stairs, etc.). In this paper we describe an approach to generate visually realistic 3D maps formed by semantic structural elements. The approach is suitable for indoor environments and integrates three different sensors: a 2D laser range finder, a stereo camera, an inertial measurement unit (IMU). Data acquisition is automatically performed by an autonomous mobile robot mounting such sensors on board. Model building is then achieved by using 2D SLAM techniques for building 2D consistent maps, stereo vision and inertial navigation system to detect semantic elements. Stereo vision is also used to extract textures of such elements. While the main objective of our research is to represent a 3D map as a set of 3D elements with their appropriate texture, and to use a generative model to build the map starting from such primitives, in this paper we outline a system doing this task and present some experiments to evaluate the impact of human interaction in the modelling process on increasing the semantic modelling and the visual realism of the maps.

1 INTRODUCTION

Building 3D models is important in many applications, ranging from virtual visits of historical buildings, to game and entertainment, to risk analysis in partially collapsed buildings. Existing systems for building 3D representation of environments have been developed at different scales: city, buildings, indoor environments, objects, presenting many differences in the sensors and in methods used to acquire data, in the techniques used to process the data, and in the kind of result computed.

Many different sensors have been used for data acquisition. Cameras are the main sensors, since they provide images that contain a very high amount of information: geometry of the scene, colors, textures, etc. However, these data are very difficult to analyze, since Computer Vision problems are still very challenging in real, unstructured environments. To retrieve information about the geometry of the environment, 2D and 3D Laser Range Finders (LRF) are very useful since they provide very precise measurements of the environment. In fact, mapping 2D or 3D environments with LRF has been an active research topic in the last year (this problem is also known as Simultaneous Localization and Mapping (SLAM)) and many systems have been demonstrated to be very effective in this task (specially for 2D environments). However, the use of 3D Laser Scanners is very expensive, while using 2D LRF mounted on pan-tilt unit allows for scanning 3D data, but it requires some time due to the movement of the pan

The methods used for data acquisition are mostly human driven. Typically, sensors are mounted on a mobile vehicle (e.g., a car driven by human around a city (Früh, 2004), an aerial vehicle, or a mobile robot (Thrun et al., 2004)) that navigates through the environment to acquire data and reconstruct it. In some cases, some form of autonomy is required by the mobile platforms, since the operating scenario may be difficult to access by human operators. For example, in a Search and Rescue mission, robots may need to enter places that are not accessible to humans (for example, for safety reasons) and the communication may not be available at all times. Such robots thus need to acquire data autonomously and to come back to the base station after the acquisition has been completed.

One additional and important characteristic of these systems is the way in which they represent the output map: metric (or feature based) 3D maps represented as a set of 3D points (plus color information), in contrast with maps represented as a set of semantic structural elements (i.e., floors, walls, steps, stairs, etc.).

The approach followed in this paper is to generate visually realistic 3D maps formed by semantic structural elements. The approach has been tested in indoor multi-level planar environments and integrates three different sensors: a 2D laser range finder, a stereo camera, and an inertial measurement unit (IMU). Data acquisition is automatically performed by an autonomous mobile robot mounting such sensors on board, model building is then achieved by using 2D SLAM techniques for building 2D consistent maps, stereo vision and inertial navigation system to detect semantic elements, stereo vision to extract textures of such elements.

The paper is organized as follows. Section 2 describes related work and compares our approach with previous research in this field. Section 3 presents an overview of the proposed system, and Sections 4 to 7 describe the main components of our system, namely, 3D SLAM, semantic structure extraction, texture extraction and model generation. Section 8 shows some results of the proposed system, and, finally, Section 9 draws some conclusions and present ideas for future work.

2 RELATED WORK

Several approaches have been presented for 3D environment reconstruction, using different sensors (cameras, stereo cameras,

multiple 2D LRF, 3D LRF, and combinations of them). For example, (Diebel et al., 2004) use active stereo vision for building a 3D metric map of the environment, (Thrun et al., 2004, Früh, 2004) use two orthogonal 2D LRF to build 3D maps of indoor and outdoor environments, while (Nüchter et al., 2005) use a 2D LRF mounted on a tilt unit that is able to acquire a very precise 3D scan of the environment with a relative cheap sensor, but it requires a higher acquisition time due to the rotation of the laser. The generation of large 3D maps of portions of a city is considered in (Früh, 2004); data acquisition is performed through a truck equipped with a horizontal 2D laser scanner (for localization), a wide angle camera and a vertical 2D laser scanner for reconstructing the building's facades. Obstacles, such as trees, cars or pedestrians, are removed considering their relative depth, while holes in the facades arising from the presence of obstacles and from the presence of specular surfaces, are filled through interpolation. The localization was achieved with the help of aerial images, thus increasing the cost requirements of such a system. On the other hand, approaches based on feature extraction and computer vision techniques have been proposed (e.g., MonoSLAM (Davison et al., 2007)), providing for 3D feature-based maps. Outdoor mapping has also been investigated. For example, in (Konolige et al., 2006) the use of stereo vision and visual odometry has been proposed for long distances outdoor navigation of a mobile robot. All these approaches are focused on building metric or feature based maps, either considering relative small environments to map or focussing on the navigation capabilities of an autonomous platform.

Another set of works have instead focused on extracting semantic features (mostly walls) from 3D data. Maps composed by semantic structures are also called *object maps* (Thrun, 2002). A first example has been given in (Iocchi et al., 2001), where textured planes of the floor and the walls of an office-like environment have been extracted from stereo vision data; this approach exploits planar assumption and deals with loop closures by assuming all orthogonal walls. The acquisition of multi-planar maps has also been investigated in (Thrun et al., 2004), by using 2 orthogonal 2D laser range finders and Expectation Maximization Algorithm to extract planar features from 3D data. Also this approach assumes that the robot is well localized. A similar approach has been proposed in (Nüchter et al., 2003), where planar features such as walls, doors or floors are detected using a method that mixes ICP and RANSAC. Labels are assigned to such planes with a predefined semantic net that implements the general knowledge about the scene. Also, a set of relational constraints such as those of parallelism, or perpendicularity, are produced by the labelling. After a simplification of the plane set, where neighboring planes are merged, the set of planes is globally refined at the same time optimizing the point (i.e. data) to plane distance and enforcing the semantic constraints introduced in the labelling step. This method offers a solution that is suited for indoor simple environments. However, scenes without many regularities to exploit in the semantic labelling and in the successive optimization may represent a problem for this kind of solution. Finally, in (Biber et al., 2004) a mobile robot is equipped with a panoramic camera and a 2D laser range finder. The processing starts from the 2D map obtained with the laser scanner, and features such as walls and the floor are extracted and finally augmented with their own textures.

The approach described in this paper aims at combining the robustness and efficiency of 2D SLAM techniques with the need of building 3D visually realistic maps of an environment. The main idea is to consider a multi-level planar environment and to perform an off-line analysis of the 3D data, in order to cluster them in many sets each belonging to a single plane. On each of these sets of data coming from a planar environment 2D SLAM techniques are applied and then these sub-maps are merged together using visual odometry techniques. 3D semantic structures are extracted from the 2D maps and from stereo vision data, while color images are used to retrieve texture of relevant parts of the environment (e.g., the floor). Exploiting robustness and efficiency of state-of-the-art 2D SLAM methods and of computer vision techniques, the proposed approach can provide a 3D visual realistic representation of large environments.

3 OVERVIEW

The system we have developed is based on a mobile robot that carries different sensors for 3D data acquisition. After an exploration phase, in which the robot collects and stores data from the environment, these data are processed off-line to build a 3D representation of the environment.

The robot used in the experiments is a Pioneer 3 equipped with an on-board PC, a wireless connection to a base station, and 3 different sensors: a 2D SICK Laser Range Finder, a Videre Stereo Camera, and an XSens IMU. The robot has software components for autonomous exploration based on an on-line fast mapping approach (Calisi et al., 2005), and thus it can be operated in three modalities: fully autonomous (i.e., the robot runs the autonomous exploration algorithm), partial autonomous (i.e., the user can specify target locations to reach and the robot is able to go there), fully tele-operated (i.e., the robot is controlled by a human operator through a wireless network connection).

For the purposes of 3D environment reconstruction described in this paper, the main goal of the robot is to gather data from the environment, while exploring it, and to store these data on a local disk. The data will be processed off-line at a later stage, possibly on another machine. More specifically, we store all the sensor readings with a 10 Hz frequency (except for stereo images that are acquired at 1 Hz): for each frame, we memorize 180 readings for the 2D LRF, a pair of images from the stereo camera (color left image and disparity image), and 6 values from the XSens IMU. All these data are synchronized with a time-stamp that is generated by the PC on board the robot.

In these experiments data have been acquired through autonomous exploration, although other modalities would have been adequate too. Further details on the data collected for the experiments are reported in Section 8.

Our approach for off-line data processing is based on four modules that are executed in the described order and are responsible for: 1) computing a 3D map of the environment as a set of 2D maps connected together; 2) recognizing and extracting properties of 3D structural elements; 3) extracting textures of these elements; 4) generating a model of the environment. It is important to notice that off-line execution is exploited by processing the data sets multiple times, refining the environment representation iteratively.

In the following, we will give a brief overview of the main system components that are explained in details in the next sections.

- **3D-SLAM** is realized as composition of two process: a) 1D SLAM on the Z coordinate, under the assumption of multi-planar environment this allows for determining the number of planes present in the environment and to associate sensor readings to each detected plane; b) several 2D SLAM processes, one for each detected plan, using the corresponding

sensor readings as determined in the phase a). The relative poses among the different 2D maps are established with some visual odometry steps that are executed from positions belonging to different and adjacent planes. A *multi-level 2D map* is thus obtained in this phase.

- **Structural element extraction** is performed on the 2D maps in order to extract relevant structural elements (also called *objects*) from the environment. Walls are detected by applying a Probabilistic Hough Transform detection on the 2D maps, while 3D structures, like stairs, steps, ramps are detected using a Neural Network classifier using stereo vision 3D data. The multi-level 2D map is then augmented by placing in it the recognized structural elements. In some cases, such 3D information are used to connect different planes in the map (e.g., stairs).

- **Texture extraction** for the semantic elements composing the map is performed by ad-hoc procedures, possibly involving user intervention. However, it is also possible to use synthetic textures, that still make the world realistic, even though they do not reproduce exactly the real environment. In this step, color images from the stereo camera and the registration of the robot poses in the map, obtained as a result of the SLAM process, are used to associate color information to the geometry of the environment.

- **Model Generation** is achieved by using a simple script language for describing objects and their properties to show in the map. A script file is interpreted by a 3D viewer that allows for navigating in the environment and to interact with it, adding, removing and changing properties of objects.

4 3D SLAM THROUGH 1D + 2D SLAM

In our procedure, the first step to reconstruct a 3D environment is that of acquiring its multi-level map. Having to explore the environment with an autonomous robot, the problem is that of localizing the robot and at the same time acquiring the map. This is the well known SLAM problem. The solutions proposed in these last years efficiently solve the problem in the 2D situation, thus retrieving the robot position and orientation on the plane and acquiring the 2D map of the explored environment. However, the described solution is not suited when dealing with non completely planar scenarios. In fact the SLAM problem has been generalized to the 3D case (see for example (Nüchter et al., 2006)). Due to the increase in complexity, these solutions are difficult to implement and tune, computationally demanding and not always robust.

Not having to deal with a completely planar environment, we cannot use only a 2D SLAM algorithm. Yet we do not need to use a full 3D SLAM algorithm. Indeed, we can exploit the assumption that the environment that we wish to reconstruct is piecewise planar. Our idea is that we can still use a robust 2D SLAM algorithm (Grisetti et al., 2006) to acquire the map in a planar section of the environment. But, to cope with the transitions between different levels, we use the IMU sensor together with the stereo camera. In particular the IMU is used to detect the transitions, while the visual odometry is used to compute the movement of the robot in this transition phase, where, otherwise, the 2D laser range finder would have been useless.

Summarizing, we process the data as follows: 1) IMU is used to detect plane-to-plane transitions; 2) visual odomotery is applied to measure the displacement of two points when a transition occurs; 3) 1D SLAM is performed to extract the number of planes and to cluster data in sub-sets each belonging to a single plane; 4) 2D SLAM is applied for each sub-set of data belonging to a single plane; 5) 2D maps are aligned using visual odometry information computed before. These steps are described in details in the rest of this section.

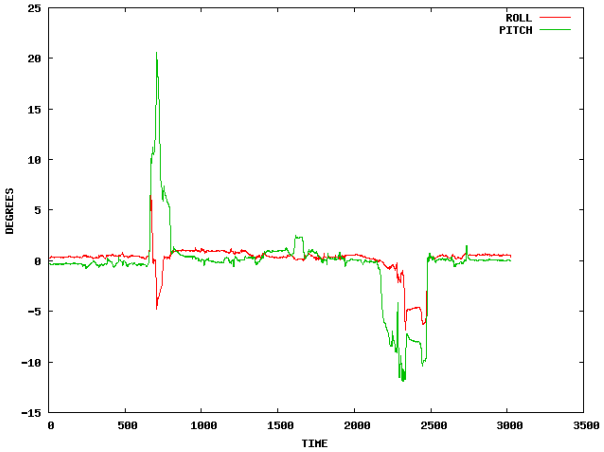

Figure 1: The *ROLL* and *PITCH* data coming from the IMU sensor. The robot navigates in a planar environment except a single small step (5cm) that the robot has to climb down and then up. In correspondence with these events it is evident a variation in the data coming from the IMU sensor.

4.1 PLANE TRANSITION DETECTION

To detect a possible change in the plane we analyze data from the IMU sensor. The IMU is collocated on the mobile robot, and we use it to retrieve the ρ,σ Euler angles, respectively the *roll* and *pitch* angles of the robot (in fact, other information would be available, but they are not of interest in this application). An example of the ρ and σ values provided by the IMU is reported in figure 1. The data refer to a path in which the robot has first to climb down and then climb up a small step (less than 5 centimeters). Even if the depth of the step is small, the data show how clearly the sensor can be used to detect such a change. In fact, it is enough to apply a threshold to the pitch angle σ (the pitch is enough because our robot cannot move sideways) to detect a change. It must be noted that this procedure is affected by false positives. Indeed, when a robot need to overcome an obstacle on its path, there might be a significant change of the σ value. This will be taken into account in the 1D SLAM process by merging planes at the same height (see below).

4.2 VISUAL ODOMETRY

In order to determine the movement of the robot while it is going through a transition phase, we use a visual odometry technique that processes the input images from the stereo camera mounted on the robot. While the robot is in a transition phase, the data coming from the scanner are not taken into account (thus also preventing the 2D mapping algorithm to take into account the non valid laser data when the robot is climbing over an obstacle). Instead the data coming from the camera are used to determine the movement of the robot.
The visual odometry technique that we have used is based on a standard approach (Hartley and Zisserman, 2000). This process is implemented by using a feature detector to identify feature matches between two consecutive stereo images. Then triplets of features are used to determine the 3D displacement of the camera (and thus of the robot) with a RANSAC approach.

Feature detection is based on the well known KLT feature tracker (Shi and Tomasi, 1994). For our visual odometry implementation, we consider 200 features tracked over consecutive frames.

To compute the rigid displacement between two consecutive frames f_t and f_{t+1}, in a noise-less case, it would be enough to have three feature associations over the two frames. Having to drop the noise-less assumption, one might consider using a least square method (Lorusso et al., 1995), possibly with more than three associations. Nevertheless, the presence of outliers would still represent a major problem that should be taken into account. In order to do this, a RANSAC algorithm (Fischler and Bolles, 1981) is first used to remove outliers. In particular, a set of candidate transformations $\mathbf{T}_i$ from f_t to f_{t+1} are calculated by randomly sampling triplets of feature associations over the two frames. Each transformation $\mathbf{T}_i$ is evaluated by calculating the residual distance

$$d(\mathbf{T}_i) = \sum_{\langle \alpha, \beta \rangle} (\mathbf{T}_i \alpha - \beta)^2$$

where α is a generic feature of frame f_t and β is the associated feature in frame f_{t+1}. The triplets with smallest residual distance are chosen and optimized together to yield the final estimated rigid transformation between the two frames.

Visual odometry process explained above is iterated for a small number of frames (10 to 20 depending on the situation) that are selected in such a way they cover the passage from one plane to another. More specifically, by analyzing IMU data, we can select two time steps: t_S is the starting time of the change of level, i.e., the robot at time t_S is on the first plane, t_E is the ending time of this process, i.e., the robot at time t_E is on the second plane. Then, we consider a set of intermediate frames within this time interval.

It is important to observe here that using visual odometry for a short time allows for ignoring the incremental error that is generated with this method. Moreover, we can further reduce such an error, by using a bundle-adjustment approach, like the one proposed in (Konolige and Agrawal, 2007), that considers not only consecutive frames but also frames that are distant in time to improve the quality of the solution.

4.3 1D SLAM

The multiplanar mapping could be handled as a series of 2D planar mappings if one could separate the data coming from each of the planes and could know the relative position of one floor level with respect to the others. The problem of calculating this displacement can be termed 1D SLAM, since the unknown value is principally the vertical position z_t of the robot (and, as a consequence, of the different planes). We can model the probelm as

$$z_{t'} = z_t + \Delta z_{[t:t']} \tag{1}$$

where $\Delta z_{[t:t']}$ is the displacement between z_t and $z_{t'}$ calculated from the observations. The problem then becomes that of evaluating $\Delta z_{[t:t']}$. Exploiting again the assumption, it is easy to realize that for most of the times the value of $\Delta z_{[t:t']}$ for two frames close in time f_t and $f_{t'}$ will be zero, since most of the time the robot will be navigating a planar environment. Therefore, it is sufficient to evaluate $\Delta z_{[t:t']}$ while a transition between two planes is occurring. Transitions are detected by using IMU data and measured through visual odometry, as explained before. Therefore $\Delta z_{[t:t']}$ is modeled as

$$\Delta z_{[t:t']} = \begin{cases} 0 & \text{if } |\sigma| < threshold; \\ \Delta z^{\mathtt{VO}}_{[t:t']} & \text{otherwise.} \end{cases} \tag{2}$$

where $\Delta z^{\mathtt{VO}}_{[t:t']}$ is the vertical component of the displacement of the robot position between time t and t' measured with visual odometry and σ is the pitch of the robot measured with the IMU. However, this modeling does not consider the *loop closure* problem, that arises when visiting for a second time a place. In the 1D SLAM problem, this means that the robot can visit the same floor level twice. For example, a robot might explore a room, leave the room by climbing down the stairs, explore another floor level and then, possibly through another entrance, enter again the already visited room by climbing up the stairs or a ramp. Being the visual odometry, and as a consequence the $\Delta z^{\mathtt{VO}}_{[t:t']}$, affected by noise, the z_t will not be the same both times the robot visit the same floor. A procedure to recognize if the floor level has already been visited must be considered. In our case, not having to deal with many different floors, we used a simple nearest neighbor approach. In particular, a new floor g_i is initialized after a change in the level has been detected at time t (and at the beginning of the exploration, of course) and inserted in a set $\mathcal{G}$. The floor is assigned with the measured z_t. Then each floor g_i is checked against every g_j in $\mathcal{G}$ and if the distance is less than a threshold, the two planes are merged and one of them is removed from $\mathcal{G}$. Though the simplicity of the approach, the procedure has been found to successfully merge the same plane when explored twice.

4.4 2D SLAM

For each plane g_i in $\mathcal{G}$, a 2D map is computed. In order to do this, a SLAM algorithm (Grisetti et al., 2006) is applied on all the laser data collected in each plane. Since the different planes have been separated and opportunely merged, there is no need to further develop the 2D SLAM method, that indeed can be applied in its original formulation. The only thing that is necessary to do is to opportunely reinitialize the robot position every time a transition between two planes occurs. This can be done by simply spreading the variance around the position estimated with visual odometry. The spreading must be taken into account accordingly to the extent of the stereo measurement noise.

4.5 MAPS ALIGNMENT

The final process is to algin the 2D maps by determining, for each pair of adjacent maps, the displacement of two points in them. Notice that, our assumption is to navigate in a multi-level planar environment, with all parallel planes, thus only 4 parameters are needed to register different 2D maps. Consequently, for each pair of adjacent and consecutive 2D maps, the values Δx,Δy,Δz,$\Delta \theta$ computed by visual odometry are used to align the two maps and a single multi-level map of the environment is thus computed.

5 DETECTING SEMANTIC STRUCTURES IN THE ENVIRONMENT

Once a 2D map has been retrieved for each plane of the environment, the next step is that of extracting significant structures. In our case we concentrated mostly on two kind of structures, namely *walls* and *stairs*, but others can be added with some ad-hoc procedures. For the extraction of walls, a first approximation of wall positions and sizes can be achieved looking for lines in the 2D map of each floor. However, the 2D map coming from the SLAM algorithm is a metric representation, and is not provided with information about segment or line positions in it. These information need to be extracted using some post-processing. Basically, our approach is divided in three steps. Since in the metric representation coming from the SLAM algorithm the low intensity pixels represent walls, in the first step we apply a threshold to

distinguish the wall pixels from the other map pixels that refer to an empty or unknown area. In the second step we used a probabilistic Hough transform (Kiryati et al., 1991) that computes segments in the map. Finally, the third step deletes those segments that are *too close* each other. In particular, the segments extracted are first ordered by length, and those that are too short are not taken into account. This is done to avoid to consider small obstacles, such as chairs or tables, as walls. From the longest to the shortest, each segment is enclosed in a rectangle that is as long as the segment and with a width related to the sensor (in this case the laser range finder) measurement error. When a new segment has to be checked for inclusion, the overlapping area between its enclosing rectangle and the rectangles enclosing the other segments previously included is checked. If the relative overlapping area is greater than a certain threshold, the segment is not included.

After the procedure has completed, there will be many holes among the walls. This arises as a side effect of the first step of the procedure just described, when we decided to filter out the shortest segments. To recover from this situation, the walls are connected by their extremities. In particular, for each wall we determine its two extremities w_{e_1} and w_{e_2}. Then, for each extremity w_{e_i}, we find the nearest extremity w_{e_j} that does not belong to the same wall and whose distance is smaller than the threshold that we used before to filter out the short segments. An example of the procedure just described is reported in Figure 2.

The second kind of semantic element extraction is performed by training a neural network on 3D stereo data, for classifying five kinds of structure in front of the robot: planar ground, stairs (up and down), ramps (up and down). The neural net classifier is very simple and has an accuracy of about 80%. Although this is not very high, the possibility in our approach to interact with the system in the model generation phase (see Section 7) allows for discarding false positives and adding features not detected by this module. Moreover, IMU data can be used to refine this result, for example filtering out some of the false positives.

Finally, it must be noted that not all the human recognizable structures present in the environment are attempted for a reconstruction. That is to say that those that are not in our knowledge base will not be taken into account. So far, we only reconstruct them in the map as obstacles, by simply introducing in the model a small 3D column for each point in the 2D laser map that has not been classified otherwise. In the future, we aim both at increasing the number of objects that might be recognized by our application and at reconstructing in the model with higher fidelity also those objects that are not recognized.

6 TEXTURE EXTRACTION

In the current implementation of our system, the texture extraction phase includes only one automatic process for detecting the floor texture; for the other elements user interaction is needed.

Texture of the ground floor is automatically extracted by analyzing stereo data and using the registered pose of the robot in the map. For each frame, given the robot pose at that time (computed by the SLAM module) and the pose of the camera with respect to the robot (that is fixed, and thus pre-computed), we extract from the stereo data the 3D point cloud of the scene relative to the map. Selecting from this point cloud those pixels that are on the ground does not return very precise result, due to noise in computing stereo data, specially if the floor is poorly textured. Therefore, we use 3D information to segment the image in two parts: the floor and the remaining of the scene. To do this, we consider that this process is performed only when the robot is in a planar part of the environment (we know this from the 1D SLAM procedure), and therefore it is possible to assume that the closer part of the image is the floor. Consequently, we proceed to examine vertical scan-lines from the bottom of the image stopping the process at *contact points*. These *contact points* are either pixels that has a value of the height from the ground higher than a given threshold, or points corresponding to a 3D position in the environment that corresponds to a wall or an obstacle in the map. By connecting these *contact points*, we obtain a curve in the image that delimits floor ground from the rest of the scene. The pixels below this curve are then rectified (i.e., transformed in the ground plan coordinate system) and shown on the map. The texture of the floor extracted with our technique is shown in the Figures 3 and 4.

The texture of other elements (walls, stairs, etc.) is extracted with human interaction. The user interface we have developed allows for selecting a position in the map and the system can display the image taken by the robot near that position. In this way, extracting textures of the walls is very simple, since it only requires the user to point a pose on the map and to select a region in the corresponding image to be used as texture of that part of the wall.

Finally, synthetic textures can be used as well for all the elements of the map. An example of maps produced by using only synthetic textures is described in Section 8.

7 3D MODEL GENERATION

The 3D model generation is achieved in two steps: in the first step the procedures described in the previous sections automatically extract the model of the environment. Both because at the end of this step there are still unmodelled structures and because we want to allow the user to introduce objects that were not present in the explored scenario, the model is subsequently modified with some user interventions.
The purpose of our reconstruction is that of focusing on the semantic of the environment structures. In other words, while we do not want to disregard the metric aspects, our goal is that of reconstructing the environment using as many as possible known structures. This fact implies that the description of the environment can be easily parametrized. Roughly speaking, the environment can be reconstructed by only specifying where to add a particular structure, together with some specific parameters of that structure. In order to do this, a language with a simple syntax has been developed. The language offers the possibility of building, interacting and navigating the reconstructed world. This is done including three sets of instructions:

- *add/remove* instructions: allow to insert or remove an element in the environment. The instruction takes as arguments the name chosen for the structure together with some parameters, such as those specifying the desired pose or some specific parameters of the structure.
- *set/change* instructions: used to change the properties of some structures already present in the environment, such as moving an object or opening a door.
- *navigate* instructions: allow the user to navigate the environment, changing the point of view or specifying a path to follow in the reconstructed environment; these instructions have effect only if used with a map viewer (see below), since they do not modify the model

The first set of instructions is used by the automatic process to generate the model, while the user can interact with it using all

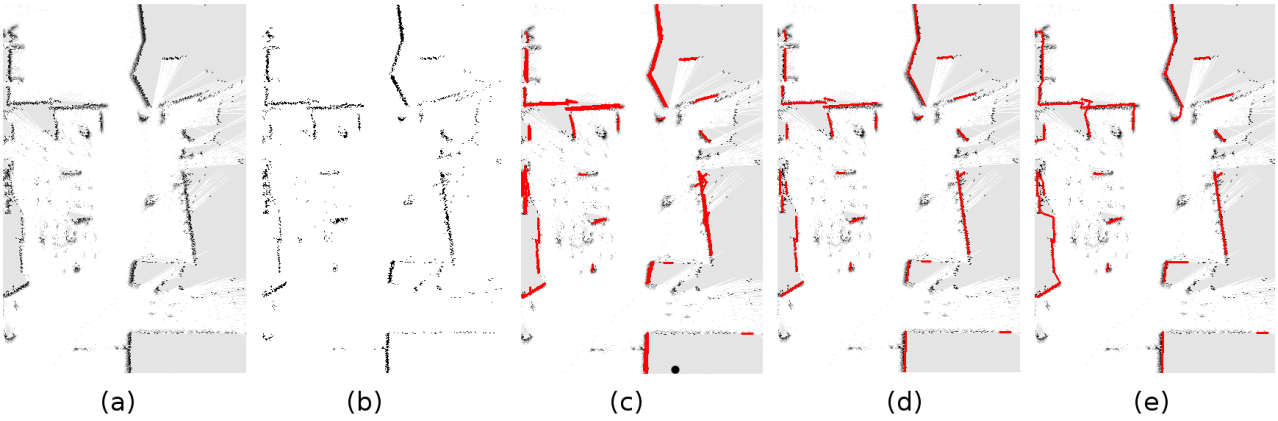

Figure 2: The five steps of the procedure used to extract walls. The input to the procedure is the 2D map (a) coming from the SLAM algorithm. In the next step a threshold is applied to distinguish wall pixels from those coming from empty or unknown areas. The resulting image (b) is processed with a probabilistic Hough filter(c). In the forth step, overlapping segments are removed (d). Finally, only close wall extremities are connected each others (e).

the instructions. In particular, our application is provided with a user utility that shows the real images collected by the robot in the exploration phase simultaneously with the views of the reconstructed environment from the same point of observation. This allows the user to easily modify the environment appropriately.This utility is a component of the map viewer that has been built using the openGL library to visualize the reconstructed environment. In particular, the viewer is provided with an interpreter that reads the instructions (automatically generated or provided by the user) and draws a 3D scene accordingly. In section 8 examples of images and a link to a video of the reconstructed environment and its views through the interpreter will be reported.

8 RESULTS AND DISCUSSION

The system described in the previous sections has been tested on a real scenario integrating data from a 2D laser range finder, an IMU sensor and a stereo camera. The scenario used for the experiments is on 3 levels: our laboratory, an outside corridor (slightly below the lab level), and the street level. Two different data sets from the same environment have been acquired and processed. The results are similar, so we will report only the ones from the first data set. The size of the explored environment is 18 x 12 meters and the total acquisition time has been 13 minutes. Storage requirements are mainly due to the stereo vision data. In the current implementation, we did not use a compressed format to store stereo data, thus for each stereo image we store on a local disk the left 640x480 color image and the disparity map at 1 frame per second. The total amount of disk space needed to store stereo data has been 1.12 GB, that is about 86.8 MB/min. Data from the LRF and IMU sensors have been acquired at 10 Hz, but disk space used for their storage is very small compared to the stereo data.

As already mentioned, all the map reconstruction processing was performed off-line. The only processing modules that were active on-board were the autonomous exploration module based on a 2D map generated through a simple scan matching method (Grisetti, 2006) and the stereo correlation algorithm to produce the disparity map.

The following figures show some results of our system. Since the stereo camera was pointing down (about 25 degrees from the horizon line), the reconstruction focusses on the ground level and on the lower parts of the walls and other objects.

Figure 3 shows an example of automatic reconstruction. All the processing in this case was done without the user interaction. Videos showing a parallel navigation in the real and in the simulated environment, as well as additional information, extracted data, and reconstructed environments are available on-line at `http://www.dis.uniroma1.it/~iocchi/3DMG`.

The map shown in Figure 3 is augmented with user interventions by adding three desks, removing some small obstacles and lowering some walls according to the aspect of the real environment. The result is reported in Figure 4. A video analogous to the previous one just described, but this time taking into account the user augmented map, is available on the above mentioned website.

Finally, the map in Figure 5 is generated by using only synthetic textures. Even if this representation does not reproduce exactly the environment from which the data have been acquired, it is still a realistic representation with a nicer appearance.

9 CONCLUSIONS AND FUTURE WORK

We have proposed a strategy to reconstruct a piecewise quasi-planar scenario through the use of a laser range finder, a stereo camera and a IMU. First the localization and mapping problem are decomposed in a 2D SLAM method that makes use of the laser data and a 1D method that makes use of the IMU and the stereo camera. The reconstraction exploits the structuredness of the environment, by searching for walls and stairs with ad-hoc procedures. A refined reconstruction is achieved by allowing the user to interact with the process. All the processing results in a description of the environment specified by a set of simple statements. The description is finally used to generate a model of the environment. Future work will focus on developing other procedures to recognize object in the environment, in order to reduce to a minimum the user interaction. Also, a more advanced optimization technique (namely bundle adjustment) is being implemented in order to extract a better displacement estimation from the visual odometry data. Finally, we would like to integrate the generated models in other applications. For example, USARSim[1] is

[1] `http://usarsim.sourceforge.net/`

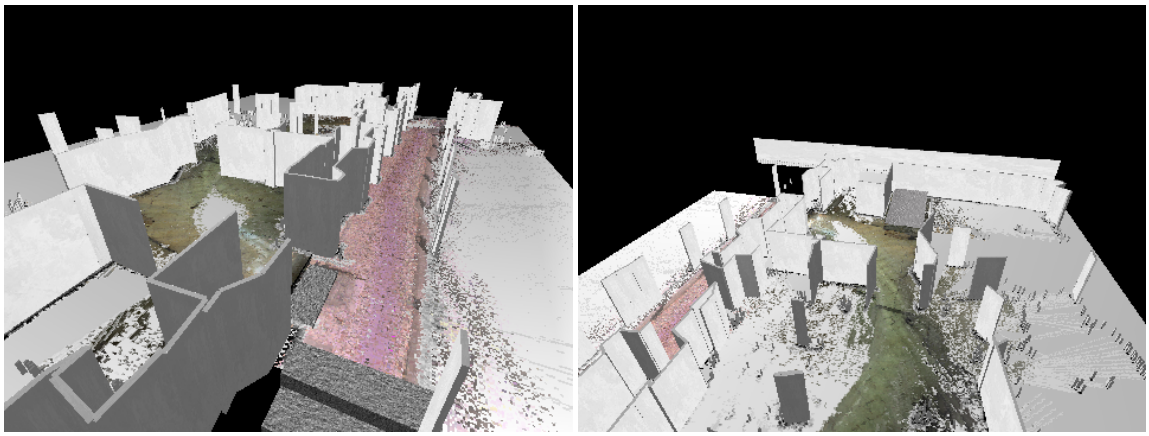

Figure 3: Two views of the map automatically reconstructed.

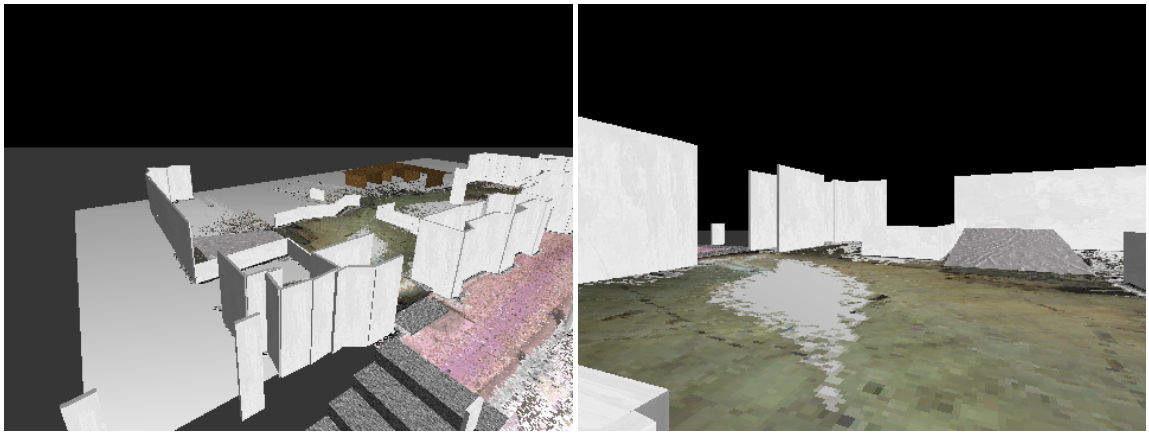

Figure 4: Two views of the map obtained from the map in Figure 3 with some user interventions

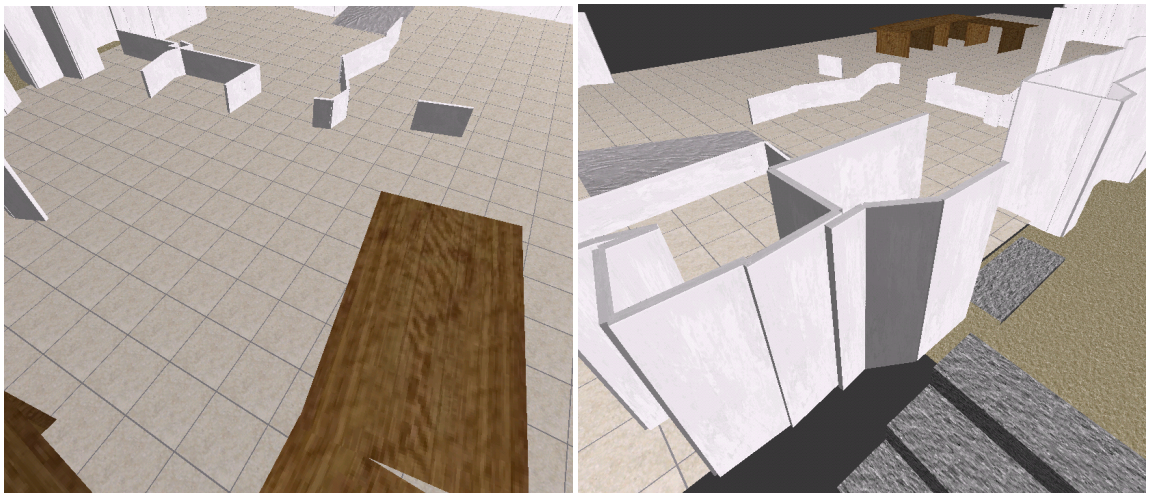

Figure 5: Two views of the map obtained from the map in Figure 3 with some user interventions and using only synthetic textures.

a 3D robotic simulator based on a game engine that allows to run robotic tasks in simulated 3D environments. At the moment, the 3D environments are defined by the user using CAD-like tools. In this context, our objective is to automatically generate 3D maps for the USARSim simulator, that are reproductions of real environments.

ACKNOWLEDGMENTS

We would like to thank Gian Diego Tipaldi and Giorgio Grisetti for providing us with the implementation of the 2D SLAM algorithm that has been used in this work.

REFERENCES

Biber, P., Andreasson, H., Duckett, T. and Schilling, A., 2004. 3d modeling of indoor environments by a mobile robot with a laser scanner and panoramic camera. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), IEEE, Sendai, Japan.

Calisi, D., Farinelli, A., Iocchi, L. and Nardi, D., 2005. Autonomous navigation and exploration in a rescue environment. In: Proceedings of the 2nd European Conference on Mobile Robotics (ECMR), Edizioni Simple s.r.l., Macerata, Italy, pp. 110–115. ISBN: 88-89177-187.

Davison, A. J., Reid, I. D., Molton, N. D. and Stasse, O., 2007. MonoSLAM: Real-time single camera SLAM. IEEE Transactions on Pattern Analysis and Machine Iintelligence.

Diebel, J., Reuterswd, K., Thrun, S., Davis, J. and R., G., 2004. Simultaneous localization and mapping with active stereo vision. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS).

Fischler, M. A. and Bolles, R. C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM 24, pp. 381–395.

Früh, C. Zakhor, A., 2004. An automated method for large-scale, ground-based city model acquisition. Int. Journal of Computer Vision 60(1), pp. 5–24.

Grisetti, G., 2006. Scaling Rao-Blackwllized Simultaneous Localization And Mapping to Large Envronments. PhD thesis, University of Rome 'La Sapienza', Dipartimento Di Informatica e Sistemistica.

Grisetti, G., Tipaldi, G. D., Stachniss, C., Burgard, W. and Nardi, D., 2006. Speeding up rao blackwellized slam. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), Orlando, FL, USA, pp. 442–447.

Hartley, R. and Zisserman, A., 2000. Multiple View Geometry in Computer Vision. Cambridge University Press.

Iocchi, L., Konolige, K. and Bajracharya, M., 2001. Visually realistic mapping of a planar environment with stereo. In: Experimental Robotics VII, pp. 521–532. ISSN: 0170-8643.

Kiryati, N., Eldar, Y. and Bruckstein, A. M., 1991. A probabilistic hough transform. Pattern Recogn. 24(4), pp. 303–316.

Konolige, K., Agrawal, M., Bolles, R. C., Cowan, C., Fischler, M. and Gerkey, B., 2006. Outdoor mapping and navigation using stereo vision. In: Proc. of Int. Symposium on Experimental Robotics (ISER).

Konolige, K. and Agrawal, M., 2007. Frame-frame matching for realtime consistent visual mapping. In: Proc. of International Workshop on Robot Vision.

Lorusso, A., Eggert, D. W. and Fisher, R. B., 1995. A comparison of four algorithms for estimating 3-d rigid transformations. In: BMVC '95: Proceedings of the 1995 British conference on Machine vision (Vol. 1), BMVA Press, Surrey, UK, UK, pp. 237–246.

Nüchter, A., Lingemann, K., Hertzberg, J. and Surmann, H., 2005. 6D SLAM with approximate data association. In: Proc. of the 12th International Conference on Advanced Robotics (ICAR), pp. 242–249.

Nüchter, A., Lingemann, K., Hertzberg, J. and Surmann, H., 2006. 6d slam – mapping outdoor environments. In: Proc.s Intl. Workshop on Safety, Security and Rescue Robotics (SSRR), Gaithersburg, MD, USA.

Nüchter, A., Surmann, H. and Hertzberg, J., 2003. Automatic model refinement for 3d reconstruction with mobile robots. 3dim 00, pp. 394.

Shi, J. and Tomasi, C., 1994. Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 593–600.

Thrun, S., 2002. Robotic mapping: A survey. In: G. Lakemeyer and B. Nebel (eds), Exploring Artificial Intelligence in the New Millenium, Morgan Kaufmann.

Thrun, S., Martin, C., Liu, Y., Hähnel, D., Emery-Montemerlo, R., Chakrabarti, D. and Burgard, W., 2004. A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots. IEEE Transactions on Robotics and Automation 20(3), pp. 433–443.