3D Modeling of Indoor Environments for a Robotic Security Guard

Peter Biber
University of Tübingen
WSI/GRIS
Tübingen, Germany
biber@gris.uni-tuebingen.de

Sven Fleck
University of Tübingen
WSI/GRIS
Tübingen, Germany
fleck@gris.uni-tuebingen.de

Tom Duckett
Örebro University
Dept. of Technology/AASS
Örebro, Sweden
tom.duckett@tech.oru.se

***Abstract*— Autonomous mobile robots will play a major role in future security and surveillance tasks for large scale environments such as shopping malls, airports, hospitals and museums. Robotic security guards will autonomously survey such environments, unless a remote human operator takes over control. In this context a 3D model can convey much more useful information than the typical 2D maps used in many robotic applications today, both for visualisation of information and as human machine interface for remote control.**

This paper addresses the challenge of building such a model of a large environment ($50 \times 60m^2$) using data from the robot's own sensors: a 2D laser scanner and a panoramic camera. The data are processed in a pipeline that comprises automatic, semiautomatic and manual stages. The user can interact with the reconstruction process where necessary to ensure robustness and completeness of the model. A hybrid representation, tailored to the application, has been chosen: floors and walls are represented efficiently by textured planes. Non-planar structures like stairs and tables, which are represented by point clouds, can be added if desired. Our methods to extract these structures include: simultaneous localization and mapping in 2D and wall extraction based on laser scanner range data, building textures from multiple omnidirectional images using multiresolution blending, and calculation of 3D geometry by a graph cut stereo technique. Various renderings illustrate the usability of the model for visualising the security guard's position and environment.

I. Introduction

Robotic research is now in a mature state and ready to focus on complete mobile robotics applications. The research in the AASS Learning Systems Lab, for example, is aimed at building a Robotic Security Guard for remote surveillance of indoor environments. This robot will learn how to patrol a given environment, acquire and update maps, keep watch over valuable objects, recognise known persons, discriminate intruders from known persons, and provide remote human operators with a detailed sensory analysis. The system should enable automation of many security operations and reduce the risk of injury to human workers. The design philosophy is based on augmenting remote human perception with super-human sensory capabilities, including (see also fig. 1):

- omni-directional vision
- hi-resolution pan-tilt-zoom camera
- laser and ultrasonic range-finder sensors
- thermal infrared camera for human detection and tracking
- metal-oxide gas sensors for chemical monitoring.

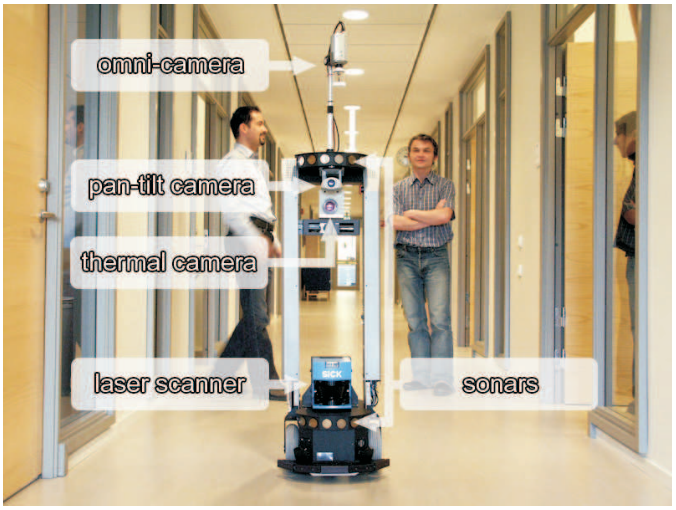


Fig. 1. Robotic platform for security guard with sensors marked. The laser range scanner and the omni-directional camera is used to build a 3D model of the robot's operation environment.

Part of this large and complex application will be a visualisation and remote control module based upon a 3D model of the robot's operation environment. The robot acquires this model using its own sensors in a training phase (in which, e.g., humans can also be presented so that the robot can update its database of known persons [16]). After deployment, the robot performs autonomous surveillance tasks unless the remote human operator takes over control, either by full teleoperation or by activating behaviours such as person following or point-to-point navigation. In this context a 3D model can convey much more useful information than the typical 2D maps used in many robotic applications today, both for visualisation of information and as human machine interface for remote control. By combining vision and 2D laser range-finder data in a single representation, a textured 3D model can provide the remote human observer with a rapid overview of the scene, enabling visualisation of structures such as windows and stairs that cannot be seen in a 2D model.

In this paper we present our easy to use method to acquire such a model. The laser range scanner and the panoramic camera collect the data needed to generate a realistic, visually convincing 3D model of large indoor environments. Our geometric 3D model consists of planes that model the floor and walls (there is no ceiling yet, as the model is constructed from a set of bird's eye views). The geometry of the planes is extracted from the 2D laser range scanner data. Textures for the floor and the walls are generated from the images captured by the panoramic camera. Multi-resolution blending is used to hide seams in the generated textures stemming, e.g., from intensity differences in the input images. Then, the scene is further enriched by 3D-geometry calculated from a graph cut stereo technique to include non-wall structures such as stairs, tables, etc. An interactive editor allows fast postprocessing of the automatically generated stereo data to remove outliers or moving objects.

So our approach builds a hybrid model of the environment by extracting geometry and using image based approaches (texture mapping). A similar approach was applied by Früh and Zakhor [8] for generating a 3D model of downtown Berkley. A complete review of hybrid techniques is beyond the scope here and we refer to references in [8] and to the pioneering work of Debevec [6]. We believe that such hybrid techniques will outperform pure image based techniques like Aliaga's work [1] that needs advanced compression and caching techniques and still provides only a limited set of viewpoints. Free choice of viewpoints and possibilities for flexible addition of additional content (e.g., for visualising the robot's position) are more important in the context considered here than photo-realistic renderings like in [1].

II. Overview

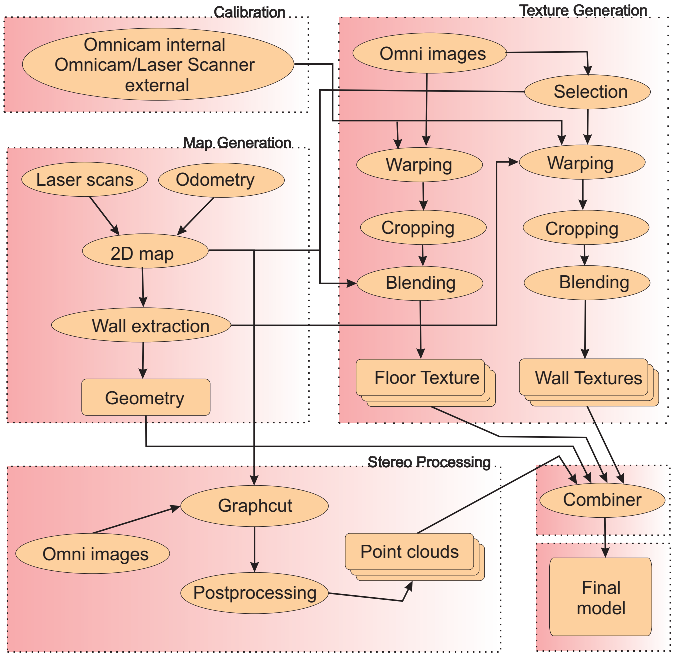


Fig. 2. An overview of our method to build a 3D model of an indoor environment. Shown is the data flow between the different modules.

This section gives an overview of our method to build a 3D model of an office environment after remotely steering the mobile robot through it.

At regular intervals, the robot records a laser scan, an odometry reading and an image from the panoramic camera. The robot platform is described in section III. From this data, the 3D model is constructed. Fig. 2 gives an overview of the method and shows the data flow between the different modules. Five major steps can be identified as follows (the second step, data collection, is omitted from Fig. 2 for clarity).

1) Calibration of the robot's sensors.
2) Data collection.
3) Map generation
4) Texture generation
5) Stereo processing

Our method consists of manual, semi-automatic and automatic parts. Recording the data and calibration is done manually by teleoperation, and extraction of the walls is done semi-automatically with an user interface. Stereo matching is automatic, but selection of extracted 3D geometry and postprocessing includes semi-automatic and manual parts. Thus the user can interact with the reconstruction process where it is necessary to ensure robustness (which plays a key role for large real world environments) and completeness of the model (there should be no holes, etc.).

After describing the hardware platform of our security guard, the remaining sections cover the mentioned steps. The paper ends with concluding remarks and of course various renderings of the resulting model.

III. Hardware platform

The robot platform is an ActivMedia Peoplebot (see Fig. 4). It is equipped with a SICK LMS 200 laser scanner and a panoramic camera consisting of an ordinary CCD camera (interlaced and TV resolution) with an omni-directional lens attachment (NetVision360 from Remote Reality). The panoramic camera has a viewing angle of almost 360 degrees (a small part of the image is occluded by the camera support) and is mounted on top of the robot looking downwards, at a height of approximately 1.6 meters above the ground plane. It has been calibrated before recording data using a calibration pattern mounted on the wall of the robotics laboratory.

IV. Calibration of external sensor parameters

All methods in the rest of the paper assume that the laser scanner and the panoramic camera are mounted parallel to the ground plane. It is difficult to achieve this in practice with sufficient precision. While a small slant of the laser scanner has less effect on the measured range values in indoor environments, a slant of the panoramic camera has considerably more effect. Fig. 3(a) shows one panoramic image along with the corresponding laser scan mapped onto the ground plane under the above assumption. Especially for distant walls, the alignment error is considerable. As a mapping like this is used to extract textures for walls, we have to correct this error.

A model for the joint relation between panoramic camera, laser scanner and ground plane using three parameters for the

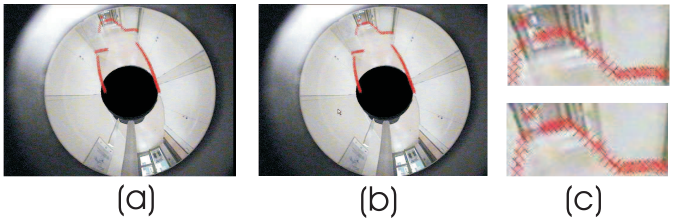


Fig. 3. Joint external calibration of laser, panoramic camera and ground plane tries to accurately map a laser scan to the edge between floor and wall on the panoramic image. (a) without calibration (b) with calibration (c) zoom

rotation of the panoramic camera turned out to be accurate enough. The parameters can be recovered automatically using full search (as the parameters' value range is small). To get a measure for the calibration, an edge image is calculated from the panoramic image. It is assumed that the edge between floor and wall produces also an edge on the edge image and therefore we count the number of laser scan samples that are mapped to edges according to the calibration parameter. Fig 3(b) shows the result of the calibration: the laser scan is mapped correctly onto the edges of the floor.

V. Building the 2D map by scan matching

An accurate 2D map is the basis of our algorithm. This map is not only used to extract walls later, it is also important to get the pose of the robot at each time step. This pose is used to generate textures of the walls and floor and provides the external camera parameters for the stereo processing.

Our approach belongs to a family of techniques where the environment is represented by a graph of spatial relations obtained by scan matching [14], [10], [7]. The nodes of the graph represent the poses where the laser scans were recorded. The edges represent pairwise registrations of two scans. Such a registration is calculated by a scan matching algorithm, using the odometry as initial estimate. The scan matcher calculates a relative pose estimate where the scan match score is maximal, along with a quadratic function approximating this score around the optimal pose. The quadratic approximations are used to build an error function over the graph, which is optimized over all poses simultaneously (i.e., we have $3 \times$ `nrScans` free parameters). Details of our method can be found in [3]. Fig. 4 shows a part of the map's graph and the final map used in this paper.

VI. Generation of geometry

The geometry of our 3D model consists of two parts: the floor and the walls. The floor is modeled by a single plane. Together with the texture generated in the next section, this is sufficient: the floor's texture is only generated where the laser scans indicate free space.

The walls form the central part of the model. Their generation is a semi-automatic step, for reasons described here. The automatic part of this process assumes that walls can be identified by finding lines formed by the samples of the laser scans. So in a first step, lines are detected in each single

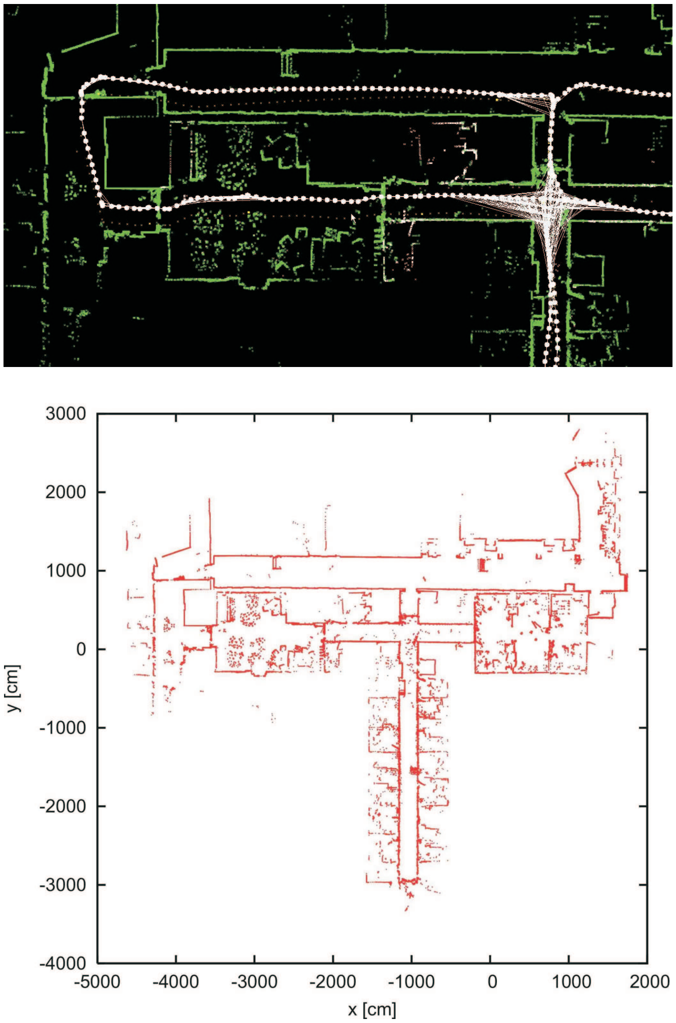


Fig. 4. Part of the graph that the map consists of (top) and final map (bottom)

laser scan using standard techniques. The detected lines are projected into the global coordinate frame. There, lines that seem to correspond are fused to form longer lines. Also, the endpoints of two lines that seem to form a corner are adjusted to have the same position. In this way, we try to prevent holes in the generated walls.

This automatic process gives a good initial set of possible walls. However, the results of the automatic process are not satisfying in some situations. These include temporarily changing objects and linear features, which do not correspond to walls. Doors might open and close while recording data, and especially for doors separating corridors, it is more desirable not to classify them as walls. Otherwise, the way would be blocked for walk throughs. Also, several detected lines were caused by sofas or tables. Such objects may not only cause the generation of false walls, they also occlude real walls, which are then not detected. So we added a manual postprocessing step, which allows the user to delete, edit and add new lines. Nearby endpoints of walls are again adjusted to have the same position. In a final step, the orientation of each wall is determined. This is done by checking the laser scan points that correspond to a wall. The wall is determined to be facing in the direction of the robot poses where the majority of the points were measured.

VII. Generation of textures

The generation of textures for walls and for the floor are similar. First, the input images are warped onto the planes assigned to walls and floor. A floor image is then cropped according to the laser scan data. Finally, corresponding generated textures from single images are fused using multi-resolution blending.

The calibration of the panoramic camera, the joint calibration of robot sensors and ground plane, and the pose at each time step allows for a simple basic acquisition of textures for floor and for walls from a single image. Both floor and walls are given by known planes in 3D: the floor is simply the ground plane, and a wall's plane is given by assigning the respective wall of the 2D map a height, following the assumption that walls rise orthogonally from the ground plane. Then textures can be generated from a single image by backward mapping (*warping*) with bilinear interpolation.

The construction of the final texture for a single wall requires the following steps. First, the input images used to extract the textures are selected. Candidate images must be taken from a position such that the wall is facing towards this position. Otherwise, the image would be taken from the other side of the wall and would supply an incorrect texture. A score is calculated for each remaining image that measures the maximum resolution of the wall in this image. The resolution is given by the size in pixels that corresponds to a real world distance on the wall, measured at the closest point on the wall. This closest point additionally must not be occluded according to the laser scan taken at that position. A maximum of ten images is selected for each wall; these are selected in a greedy manner, such that the minimum score along the wall is at a maximum. If some position along the wall is occluded on all images, the nonocclusion constraint is ignored. This constraint entails also that image information is only extracted from the half of the image where laser scan data are available (the SICK laser scanner covers only 180°). Finally, a wall texture is created from each selected image, then these are fused using the blending method described as follows.

The generation of a floor texture from a single image is demonstrated in Fig. 5. The image is warped onto the ground plane. Then it is cropped according to the laser scanner range readings at this position, yielding a single floor image. This entails again that one half of the image is not used. Such a floor image is generated from each input image. Then, these images are mapped onto the global 2D coordinate frame.

Both floor and wall textures are fused from multiple input images (Fig. 6 shows an example). The fusion is faced with several challenges, among them

- image brightness is not constant,
- calibration and registration may be not accurate enough,
- parts of the input image may be occluded by the robot or support of the panoramic camera, and
- walls may be occluded by objects in front of them and thus effects of parallax play a role.

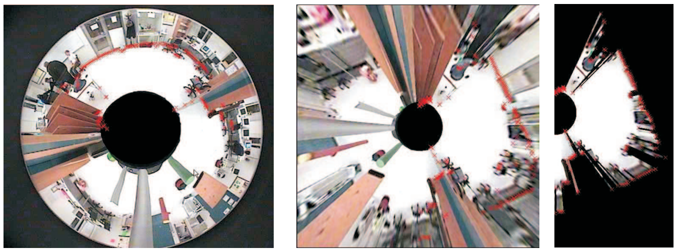


Fig. 5. Generation of floor texture from a single image.

Additionally, the quality along a wall texture degrades with the distance from the closest point to the robot position (this effect is due to scaling and can be seen clearly in Fig. 6). Similar effects can be observed for floor textures. These problems also exist in other contexts, e.g. [2], [15].

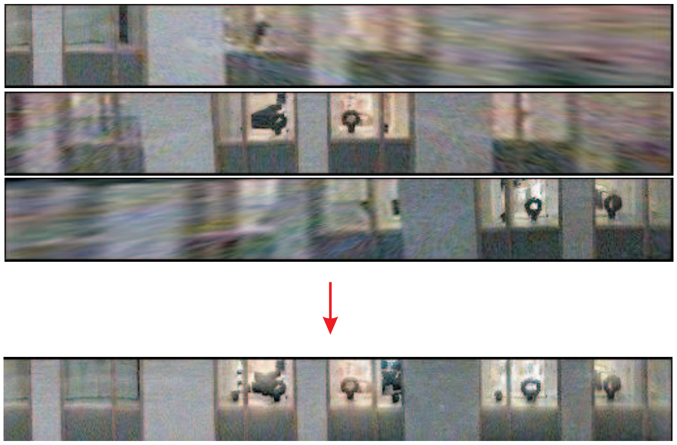
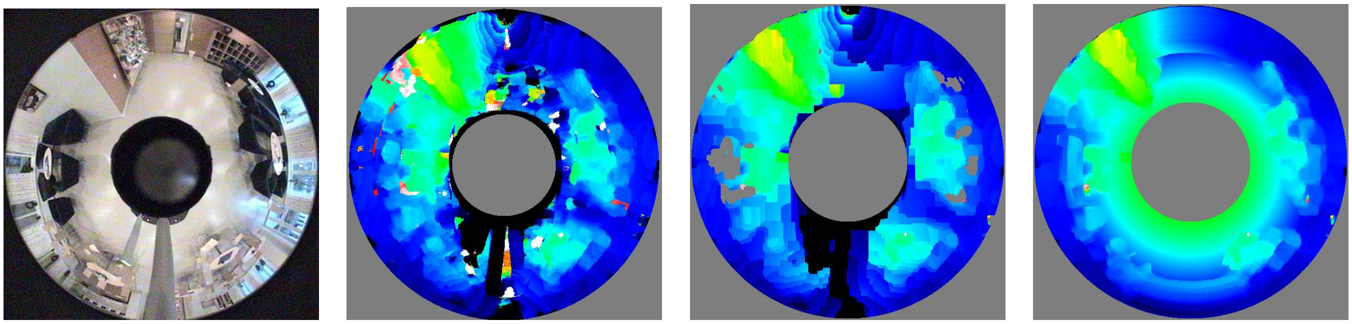


Fig. 6. Final textures of walls are generated by blending multiple textures generated from single panoramic images. Shown here are three of ten textures which are fused into a single texture.

We use an adaption of Burt and Adelson multiresolution blending [5]. The goal of the algorithm is that visible seams between the images should be avoided by blending different frequency bands using different transition zones.

The outline is as follows: a Laplacian pyramid is calculated for each image to be blended. Each layer of this pyramid is blended separately with a constant transition zone. The result is obtained by reversing the actions that are needed to build the pyramid on the single blended layers. Typically, the distance from an image center is used to determine where the transition zones between different images should be placed. The motivation for this is that the image quality should be best in the center (consider, e.g., radial distortion) and that the transition zones can get large (needed to blend low frequencies). To adapt to the situation here, we calculate a distance field for each texture to be blended, which simulates this "distance to the image center". For the walls, this image center is placed at



(a) One example source image. (b) Winner takes all solution of stereo matching. (c) Result of graph cut algorithm. (d) Final disparity map after post-processing the graph cut results.

Fig. 7. Stereo processing using graph cut algorithm and postprocessing steps (subpixel refinement, epipole removal, floor correction and hole filling).

an x-position that corresponds to the closest point to the robot's position (where the scaling factor is smallest). Using such a distance field, we can also mask out image parts (needed on the floor textures as in Fig.5 to mask both the region occluded by the robot and regions not classified as floor according to the laser scanner).

VIII. Acquisition of additional 3D geometry using stereo

Thanks to the available camera positions and the calibrated camera we are in an ideal setting to apply stereo algorithms to the input images. A high quality, state-of-the-art stereo algorithm - namely the *graph cut* algorithm by Kolmogorov and Zabih - is used to calculate a disparity map for each panoramic image. Our implementation is based upon the graph cut implementation of Per-Jonny Käck [11] that extends the publicly available source code of Kolmogorov and Zabih [12].

A. SSD matching

Our stereo matching pipeline starts with the following stage: first, for each pixel in the first image the epipolar curve in the second image is created according to the epipolar geometry of our panoramic camera. This epipolar curve is represented by a set of points in image space where each point denotes a different disparity. These points are used to construct a rectified window taking zoom into account. Then, an SSD error value for each disparity on this epipolar curve is computed and saved. The image that is being processed is compared both to the next and to the previous image. The matching costs are then mixed into one big array containing all matching costs for each pixel, except for those parts of the image where one curve contains more points than the other – here only the matching values of the longer curve are used. These steps provide the data needed by the graph cut algorithm.

B. Graph Cut

The graph cut algorithm used here follows the work of Kolmogorov & Zabih [12] and is adapted for omnidirectional imaging.

The key is formulating the correspondence problem as an energy minimization problem. This is done by an algorithm based on α-expansion moves [4]. The minimization is done iteratively by transforming the energy minimization problem into several minimum cut problems. These lead to a strong local minimum of the energy function by computing the best α-expansion of lowest energy for different values of α, until convergence is reached. To ensure that each α-expansion succeeds, which is key to above correspondence problem, is its implemented via graph cuts. Kolmogorov & Zabih [13] investigated the necessary characteristics for an energy functions of binary values to be optimized by graph cuts. We use an appropriate energy function E of the form (in the notation of [13]):

$$E(f) = E_{data}(f) + E_{occ}(f) + E_{smooth}(f)$$

$E_{data}(f)$ embodies the SSD-based matching cost of corresponding pixels, i.e.

$$E_{data}(f) = \sum_{<p,q>\epsilon A(f)} |I_{k-1}(p) - I_k(q)|^2$$

The occlusion term $E_{occ}(f)$ adds an additional cost C_p for each occluded pixel:

$$E_{occ}(f) = \sum_{p\epsilon P} C_p T(|N_p(f)| = 0)$$

$E_{smooth}(f)$ imposes a penalty $V_{a1,a2}$ for neighboring pixels having different disparity values:

$$E_{smooth}(f) = \sum_{\{a1,a2\}\epsilon N_1} V_{a1,a2} T(f(a1) \neq f(a2))$$

We utilized the graph cut implementation in [11] as the starting point for our work.

The resulting disparity map is converted into a point cloud and postprocessed. Disparity values are refined to subpixel accuracy by finding the optimum of a local quadratic model built using the original matching cost at the integer disparity value and its adjacent disparity values. Regions around the epipoles (there are two epipoles in omnidirectional images, e.g., [9]) are removed because these typically provide too few constraints to extract reliable depth information. In a further step depth values that belong to the floor with high probability are corrected to be exactly on the floor. The epipole removal and the graph cut algorithm both mark some pixels as unknown or occluded. The distance values for these pixels are interpolated from the surrounding, known distances using linear interpolation along concentric circles.

Figure 7 shows one source image, the winner takes all solution based on the SSD score, the result of the graph cut algorithm and the final disparity map after postprocessing. The point cloud from this figure (fused with the walls and floor model) is rendered in Fig. 9.

C. Point cloud postprocessing

Point clouds created by the stereo matcher are combined applying some heuristics to suppress outliers. For example, points are only counted as valid if they receive support also from other point clouds. Points that are already represented by walls or by the floor are omitted. Finally the point clouds are combined with the rest of the model. An interactive point cloud editor and renderer allows the user to select the objects supposed to be part of the final model and to delete outliers. Future versions will also allow point cloud filtering and application of hole filling algorithms.

This tool uses features of modern graphics hardware (vertex and pixel shader) to allow fast rendering and editing of large point clouds (several million points). A screenshot of this tool is shown in figure 9 (while editing a staircase).

IX. Results and conclusion

A data set of 602 images and laser scans was recorded at Örebro university by teleoperation, covering parts of a region of about 60×50 meters. The built 2D-map was shown in Fig. 4. A screen shot of the resulting 3D model without stereo results can be seen in Fig. 8. This model can be exported as a VRML model, so that it can be viewed in a web browser with a VRML plugin. It is also possible to build a VRML model with point clouds (figures 11 and 12), but there are tight limits on the number of points such that the frame rate allows real time walkthroughs. For larger models it is suggested to use a native visualisation environment based upon our point clouds editor (which makes heavily use of modern graphics hardware features like vertex and pixel shaders).

We see our technique as a successful easy to method to acquire a 3D model that is highly useful for the robotic security guard. Considerable work has been done also on other components and with ongoing work to integrate these technologies we are confident to reach a state where autonomous mobile robots leave their labs to do useful work in the real world, based on their own sensor data and in cooperation with humans.

Acknowledgments

Thanks to Henrik Andreasson (AASS) for calibrating the omnidirectional camera and data collection, to Florian Busch (GRIS) his implementation work on the graph cut part and Michael Wand (GRIS) together with his students for the great point cloud tool.

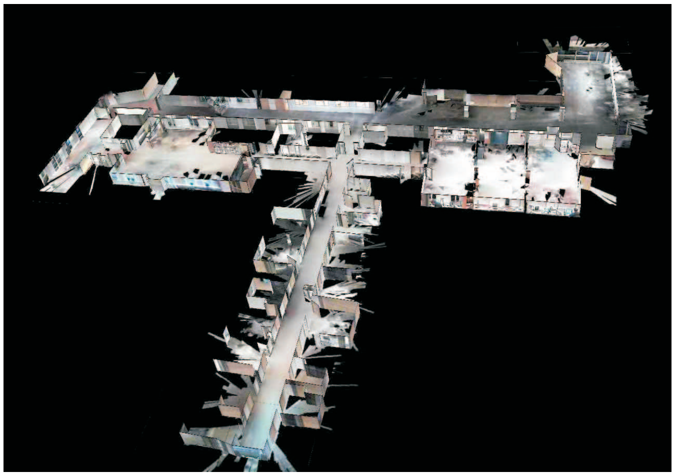


Fig. 8. A view of the VRML model - yet without results from stereo matching.

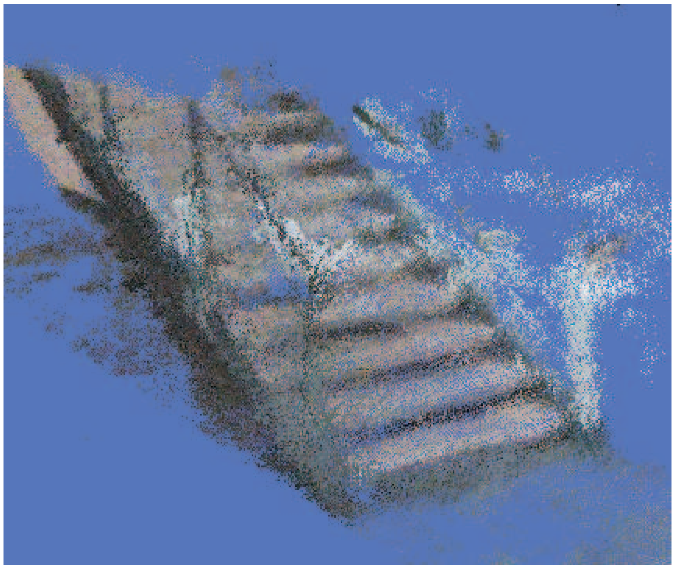


Fig. 9. A staircase: output of graph cut-algorithm after removing walls and floor, but before removing outliers manually.

References

[1] D. Aliaga, D. Yanovsky, and I. Carlbom. Sea of images: A dense sampling approach for rendering large indoor environments. *Computer Graphics & Applications, Special Issue on 3D Reconstruction and Visualization*, pages 22–30, Nov/Dec 2003.

[2] A. Baumberg. Blending images for texturing 3d models. In *Proceedings of the British Machine Vision Conference*, 2002.

[3] P. Biber and W. Straßer. The normal distributions transform: A new approach to laser scan matching. In *International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[4] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1124–1137, 2004.

[5] P. J. Burt and Edward H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, 1983.

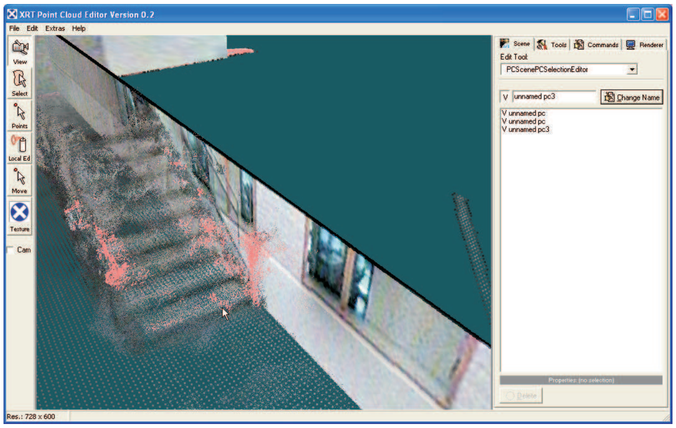


Fig. 10. Screen shot of the tool that can be used to edit point clouds comfortably.

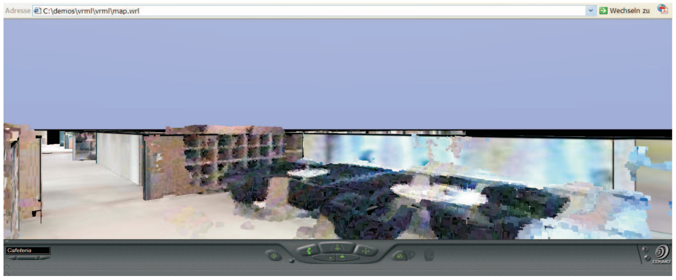


Fig. 11. A view of the cafeteria with results from stereo matching included.

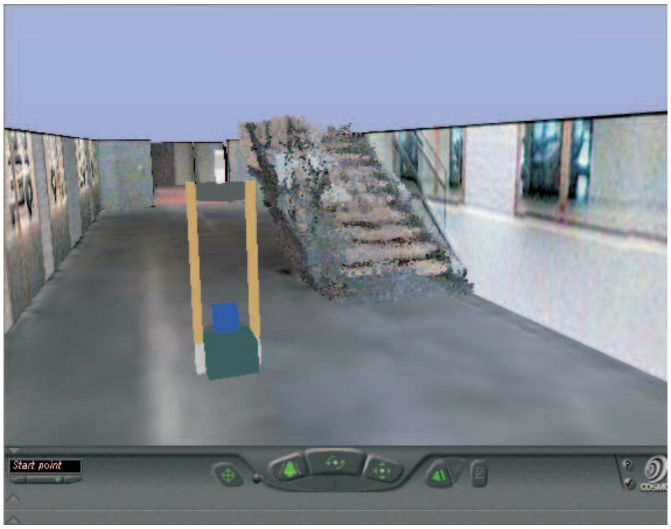


Fig. 12. The map can be used to visualise information in 3D by mixing in virtual content, here for example the position of the robot.

[6] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *SIGGRAPH 96*, 1996.

[7] Udo Frese and Tom Duckett. A multigrid approach for accelerating relaxation-based SLAM. In *Proc. IJCAI Workshop on Reasoning with Uncertainty in Robotics (RUR 2003)*, 2003.

[8] C. Früh and A. Zakhor. Constructing 3d city models by merging ground-based and airborne views. *Computer Graphics and Applications*, November/December 2003.

[9] Christopher Geyer and Kostas Daniilidis. Conformal rectification of omnidirectional stereo pairs. In *Omnivis 2003: Workshop on Omnidirectional Vision and Camera Networks*, 2003.

[10] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Computational Intelligence in Robotics and Automation, 1999*.

[11] P.J. Käck. Robust stereo correspondence using graph cuts (master thesis), *www.nada.kth.se/utbildning/grukth/exjobb/rapportlistor/-2004/rapporter04/kack_per-jonny_04019.pdf*, 2004.

[12] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions using graph cuts. In *International Conference on Computer Vision (ICCV'01)*, 2001.

[13] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[14] F. Lu and E.E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[15] Claudio Rocchini, Paolo Cignomi, Claudio Montani, and Roberto Scopigno. Multiple textures stitching and blending on 3D objects. In *Eurographics Rendering Workshop 1999*, pages 119–130.

[16] André Treptow, Grzegorz Cielniak, and Tom Duckett. Active people recognition using thermal and grey images on a mobile security robot. In *submitted to IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005.