

Room Layout Estimation from Rapid Omnidirectional Exploration

Robert Lukierski¹, Stefan Leutenegger¹ and Andrew J. Davison¹

Abstract—A new generation of practical, low-cost indoor robots is now using wide-angle cameras to aid navigation, but usually this is limited to position estimation via sparse feature-based SLAM. Such robots usually have little global sense of the dimensions, demarcation or identities of the rooms they are in, information which would be very useful to enable behaviour with much more high level intelligence.

In this paper we show that we can augment an omnidirectional SLAM pipeline with straightforward dense stereo estimation and simple and robust room model fitting to obtain rapid and reliable estimation of the global shape of typical rooms from short robot motions. We have tested our method extensively in real homes, offices and on synthetic data. We also give examples of how our method can extend to making composite maps of larger rooms, and detecting room transitions.

I. INTRODUCTION

Cameras and high performance processors are now reaching the low cost, power consumption and size which enables computer vision to become a standard part of mass market products. Consumer robotics is an important emerging sector, and the only true success to date is in autonomous floor cleaning devices. Several of the latest models on the market now have a vision capability (notably the Dyson 360 Eye and the iRobot Roomba 980) which enables positioning via the creation of a sparse feature-based SLAM map. However this seems only to be the beginning of what vision will enable, even in small floor cleaning robots.

Taking as an example the newly released Dyson 360 Eye, this robot performs feature-based SLAM using an omnidirectional camera, enabling repeatable positioning. But its sense of free space and obstacles comes only from additional short-range infra-red sensors at the front. The robot only discovers the dimensions of a whole room by laboriously exploring it while cleaning and avoiding obstacles.

In this paper we show that a straightforward additional vision pipeline can be applied to the omnidirectional images captured by such a robot as it makes a short manoeuvre in order to perform model-based estimation of the global shape of typical rooms. This ability opens up the possibility of much more intelligent high level behaviour from small cleaning robots without the need for additional sensors or infrastructure. A robot could always be aware of the global size, shape, demarcation and identity of the room that it is in.

Our method is based on motion stereo as the robot completes a small circular motion, to estimate an omnidirectional

depth map. This depth map will usually lack precision due to the limited resolution of omnidirectional images and the typical abundance of textureless areas in indoor scenes, but its omnidirectional completeness allows the fitting of a low dimensional room model in a manner which we have found to be very forgiving of a range of imaging conditions in real scenes.

We have tested our method in a variety of real rooms in homes and offices, with additional validation using synthetic data. We also present examples of scaling the method to composite estimation of large rooms, and detecting room to room transitions.

II. RELATED WORK

There are various previous approaches to room shape estimation from images, often from a single image which usually requires a machine learning approach to overcome the large amount of ambiguity present. In our method we show that the multiple omnidirectional images, which can easily be gathered by a controlled robot, can be used in a straightforward model fitting which does not require training data.

One single image approach was published by Hedau *et al.* [1]. This was a rather handcrafted approach to single image room estimation based on vanishing points and extracting line segments. This information was used to produce a set of possible layouts. These layouts were then scored with an evaluation function obtained from learned models. The best scoring box layout was then refined by estimating surface labels, using superpixel segmentation.

The work of Zhang *et al.* [2] extended this to panoramic images and added more context to the room estimation (estimating other objects, like furniture). As in [1] the method employs line segment extraction, vanishing point estimation, and hypothesis generation, both from line segments and image segmentation. A random forest classifier assigns classes to extracted object cuboids. Equipped with both whole room hypotheses and classified object cuboids, a trained SVM classifier is employed to choose the best hypothesis.

Probably the conceptually closest approach to ours is the one by Cabral and Furukawa [3]. It also employs wide field of view cameras and 3D reconstruction. The sparse SfM reconstruction is then converted into a type of occupancy grid. Then a graph cut is used to extract the floorplan from occupancy grid evidence. Additionally, the input image undergoes superpixel segmentation and dynamic programming labelling to classify pixels into floor, wall and ceiling classes. The overall results are impressive, going beyond cuboid estimation, but the method is aimed at high quality

¹Robert Lukierski, Stefan Leutenegger and Andrew J. Davison are with the Dyson Robotics Laboratory, Department of Computing, Imperial College London, London, UK {r.lukierski12, s.leutenegger, a.davison}@imperial.ac.uk

visualizations, the input comes from high resolution DSLR cameras and the method runtime is in tens of minutes thus making it rather unsuitable for a low cost mobile robot.

The model fitting in our approach was inspired by the differentiable renderer idea presented in [4], though in our approach the forward process operates on depth maps and not image intensities. We use an omnidirectional camera, thus making it incompatible with OpenGL that OpenDR is coupled with (OpenGL supports only perspective camera model). Therefore our approach, to be precise, could be called a differentiable raytracer. Another approach derived from OpenDR, applied to classical cameras and finer model (height map) was presented in [5].

III. METHOD

A. Requirements

The omnidirectional SfM and motion stereo estimation elements of our method are similar to those presented by Lukierski *et al.* [6], whose work was aimed at building occupancy grid free space maps from omnidirectional vision.

First, the omnidirectional camera was calibrated, assuming the Geyer & Barreto ([7], [8]) catadioptric camera model. The raw sensor images are then unwrapped into spherical panoramas that are uniform in sampling.

The next step is to perform structure from motion to obtain consistent camera poses for each frame. Our system uses the BRISK [9] feature detector and descriptor and the Ceres-Solver library ([10]) to perform non-linear optimization of a sparse feature and keyframe map.

As the camera is carried by a robot, this enables us to use the wheel odometry for pose constraints to recover the correct scale. Each new frame is tracked against the map, or the last 20 keyframes. If defined criteria are met (low feature matches count, distance from the previous keyframe above threshold, etc.) the current, tracked, live frame becomes a keyframe and is added to the map. The whole map then undergoes bundle adjustment optimization.

With a geometrically calibrated camera and known SE(3) poses for each frame we are able to perform dense depth map estimation. This is performed using a cost volume/plane sweep approach, as described in [11], [6]. Usually we control our robot to perform a small circular motion, as this ensures good disparity in all directions which is important in the omnidirectional camera case. The resulting depth map (Figure 1, estimated for a selected reference frame) is of low quality, due to the sensor’s limited resolution and dynamic range, and also to the environment in which the robot operates (plain textureless areas, specular surfaces, light sources). We employ minimal filtering to the depth map, bilateral filter over the cost volume and rejecting measurements with excessive standard deviation (following the approach of [6]). Additionally, as our method is aimed at room shape estimation, we remove points below and above predefined floor and ceiling planes, as these are assumed known. These measurements do not contribute to our estimation process and might contain noise (Figure 1).

B. Model

Our model of a room is a cuboid box parametrized as follows: $\mathbf{p} = [x_-, x_+, y_-, y_+, \theta]$. The box revolves around the robot’s reference pose (for which the depth map was calculated) with the angle θ and the four other parameters are distances to the walls. Therefore the box has width $x_- + x_+$ and length $y_- + y_+$. The height of the box is not estimated and is to an arbitrary large value that ensures full coverage of the camera viewing angles. From the model coefficients described above we generate a triangular 3D model, composed of 8 triangles and therefore 24 vertices. A visual representation of the model is presented in the Figure 2.

C. Automatic Differentiation

Our method employs automatic differentiation with forward accumulation to compute Jacobians. Each step of the calculation (triangular mesh generation, ray-triangle intersection, camera projection, residual and loss function computation), both on the CPU and the GPU, carries partial derivatives with respect to the model coefficients. After computing each per-pixel residual, the residual value and the partial derivatives are summed with a GPU reduce operation.

D. Estimation

With the triangular 3D model of the current estimate, we perform per pixel ray-tracing. For each pixel we iterate through all the triangles of the model, check ray-triangle intersection ([12]) and calculate the resulting depth $d_b(\mathbf{p}, x, y)$ at which the intersection occurs (with Z-buffer like logic in place).

As described in the Section III-A, for each pixel we also have our measured depth value $d_m(x, y)$. These two depths create the residual of a cost function:

$$E(\mathbf{p}, x, y) = M(d_m(\mathbf{p}, x, y) - d_b(x, y), c_n, c_p), \quad (1)$$

where $M(e, c_n, c_p)$ is an asymmetric Cauchy loss function defined as follows:

$$M(e, c_n, c_p) = \begin{cases} \frac{c_n^2}{2} \log \left(1 + \left(\frac{e}{c_n} \right)^2 \right), & \text{if } e < 0 \\ \frac{c_p^2}{2} \log \left(1 + \left(\frac{e}{c_p} \right)^2 \right), & \text{otherwise} . \end{cases} \quad (2)$$

An example plot of this function is presented in the Figure 2. The reason to use an asymmetric cost function is that when we fit a box to an omnidirectional depth map with the aim of finding the outer shape of a room, we should pay less attention to depth data which is closer to the camera than predicted by the model (since this will often be caused by furniture or other clutter) than to depth data which is farther away than predicted by the model.

The final energy function, after the reduction over the entire image described in the Section III-C, and therefore the optimization problem, is:

$$\arg \min_{\mathbf{p}} E(\mathbf{p}) = \arg \min_{\mathbf{p}} \sum_{x,y} M(d_m(\mathbf{p}, x, y) - d_b(x, y), c_n, c_p) . \quad (3)$$

The resulting cost function value, together with the Jacobian, is used in a Levenberg-Marquardt optimization scheme



Fig. 1: Raw depth map and the resulting point cloud after filtering.

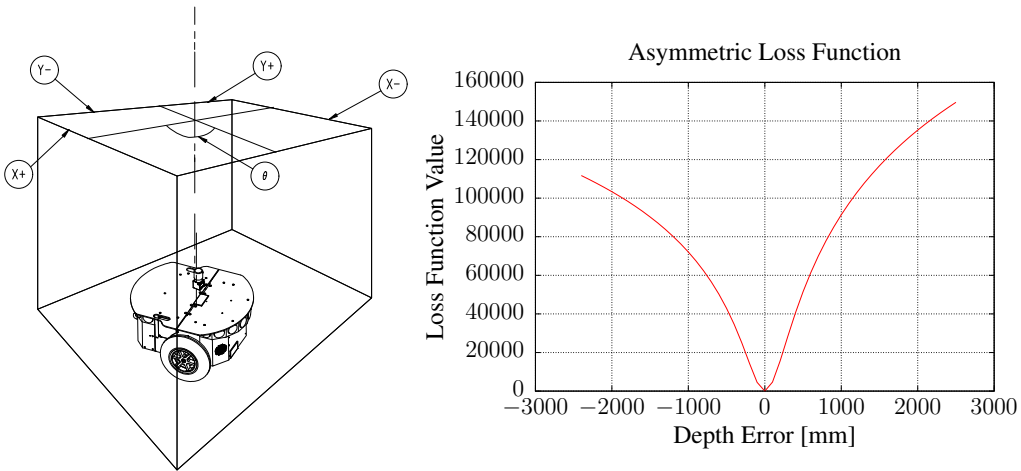


Fig. 2: Box model and asymmetric loss function.



Fig. 3: Model raytraced depth map and successfully estimated box.

[10] to estimate model parameters. We would like to mention that performing a coordinate descent approach (first estimating just box dimensions, then just box orientation θ) yields faster convergence and better final results. The output of the method is the fitted box model and, optionally, a raytraced depth map, as seen in Figure 3. It is important to note, that while the result is 2D, as we do not estimate the height of the box, the optimization process runs in 3D — over the entire depth map.

IV. RESULTS

This section describes the results of multiple experiments where our approach was put to the test using synthetic data and real data from office rooms and the household rooms which are our real target case. In Section IV-D we discuss additional capabilities of our system that might be useful for some applications. In Figures 4 and 5 we mark the estimated box with a green rectangle and the ground truth with a red one. Vertical stripes on the reference frame mark the corners

of the respective box.

A. Synthetic Data

To first characterize our method, we present purely synthetic datasets, generated a modified PovRay raytracer (published by Handa *et al.* [13]), but employing an omnidirectional camera model. We generate 30 input frames from a circular trajectory. This synthetic dataset provides us with perfect $SE(3)$ poses and ground truth depth maps.

It is worth noting that our algorithm has the favourable property of graceful degradation of the estimate in adverse circumstances (see the textureless dataset in Figure 5), instead of catastrophic failure.

The synthetic datasets above are rather extreme examples as the total area is 40m^2 , which is unusually large for a typical household room.

B. Household Data

To prove the value of our approach for low cost mobile robots equipped with an omnidirectional camera, we have

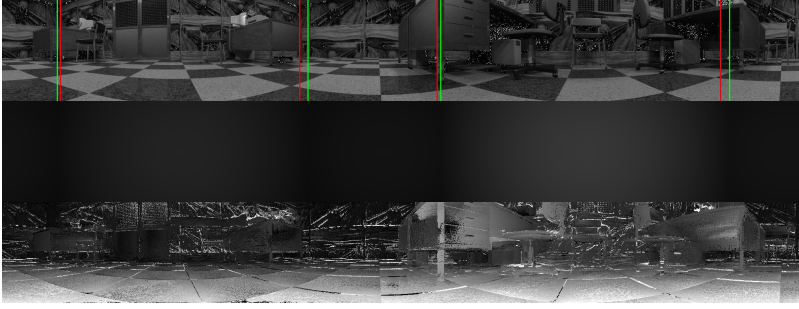


Fig. 4: Dataset “Synthetic Textured”; from top to bottom: reference frame, raytraced depth map, raw depth map. On the right: top down view of the point cloud with the box estimate. Ground truth dimensions: $8.00 \times 5.00\text{m}$. Estimated: $8.17 \times 4.72\text{m}$.

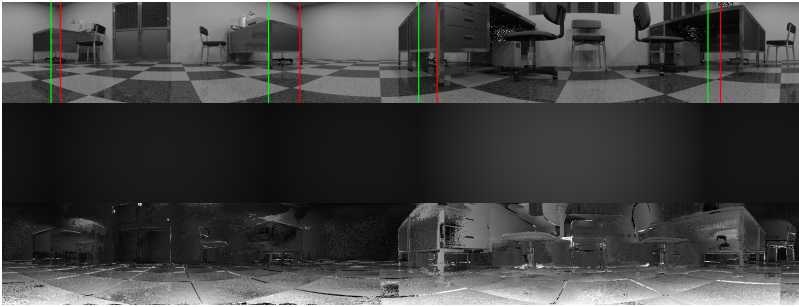


Fig. 5: Dataset “Synthetic Textureless”, from top to bottom: reference frame, raytraced depth map, raw depth map. On the right: top down view of the point cloud with the box estimate. Ground truth dimensions: $8.00 \times 5.00\text{m}$. Estimated: $7.28 \times 4.95\text{m}$.

performed extensive testing in real world environments, spanning multiple houses and apartments with multiple rooms, from the tiniest bedrooms, through bathrooms (textureless and specular, so difficult) to large scale living rooms.

The real world data is difficult to analyse quantitatively due to lack of precise ground truth. For our real-world household datasets we have generated pseudo ground truth using the ElasticFusion system [14]. The rooms were scanned and reconstructed using a hand-held depth camera. Since these scans were done using the built-in tracking of ElasticFusion and focusing on global consistency, they often contained holes so the resulting models were remeshed using Poisson Surface Reconstruction to create denser, watertight models. Additionally, without a VICON tracker for example (difficult to install in the field and could disturb the structure of the room), there is no common coordinate frame origin, so the resulting 3D models had to be manually aligned to the frame of reference of the bundle adjusted robot trajectories. With such 3D models we can render the scene (both RGB and depth) from a particular viewpoint (where the robot captured the frame) onto a cube map and then transform the textures onto the cube, using our camera model to obtain the ground truth depth map as it would look on a hypothetical omnidirectional RGBD camera.

In our experiments we present various uses of these pseudo ground truth reconstructions. In most rooms, especially small

ones, the ground truth shape of the room was estimated by applying our method to the ground truth depth map. In larger rooms, especially those with multiple incremental steps, we have decided to set ground truth to the actual outer wall-to-wall shape of the room, in the hope that incremental operation would estimate the major part of the ground truth. As metrics we present ground truth floor area, the area of our estimate (the union if multiple boxes were estimated), the overlap with the ground truth and the percentage of this overlap.

The dataset presented in Figure 6 perfectly illustrates the limitations imposed by the purely visual nature of our method. In this room one of the walls is behind a bunk bed, thus for the robot the bunk bed becomes a wall and therefore the results cannot be treated as an architectural blueprint. Top down floor plans therefore require more than visual reasoning and do not reflect the robot’s world view. The ground truth was estimated using a 3D model based depth map and is actually worse than our estimate, due to the poor quality of the 3D model.

Figure 7 is another example, where large furniture (cupboard units) that is featured prominently in the field of view becomes the outer boundary for the shape of the room. The ground truth was established from the 3D model based depth map and both estimates fit closely, with minor errors due to some changes in the scene between the time the robot

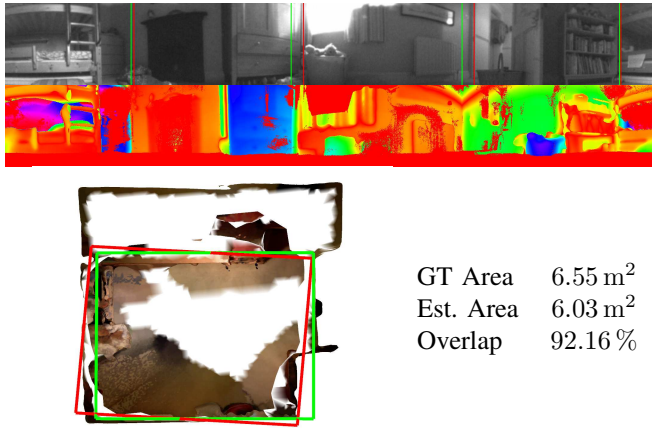


Fig. 6: Dataset “Ealing Bedroom 1”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

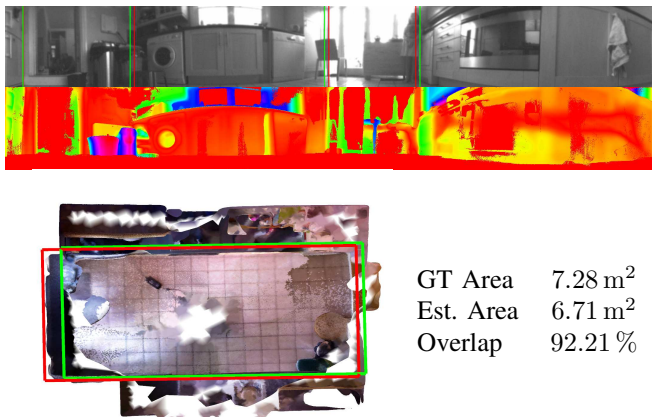


Fig. 7: Dataset “Ealing Kitchen”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

captured the dataset and the time the 3D model was created.

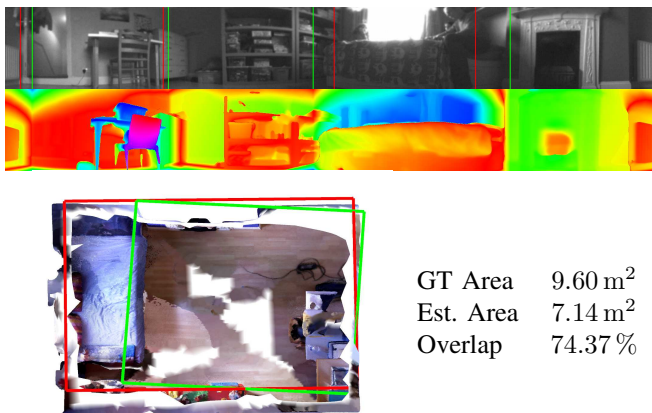


Fig. 8: Dataset “Ealing Bedroom 2”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

In Figure 8 the ground truth was manually set to the real outer shape of the room to show the discrepancy between true room dimensions and vision based ones. It is worth noting that the raytraced depth map error (created from our box estimate) clearly shows the nature of the asymmetric cost function, as small furniture (here a table and chair) stands out in the error color map, showing how the cost function tries to approach the “wall”.

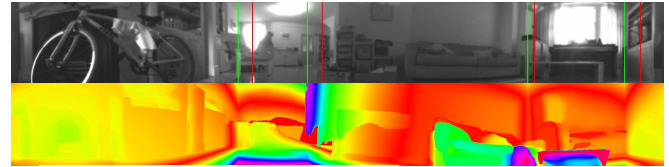


Fig. 9: Dataset “Ealing Livingroom”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

Incremental operation of the method on a difficult large scale dataset is shown in Figure 9. Due to the nature of our model (box/rectangle), neither our estimates nor the manual ground truth fit the unusual shape of the room. This can be seen in the second estimate, where one dimension was following the width of the room (like the first estimate), while the other the length (as in the third estimate). Combining these two causes noticeable error.

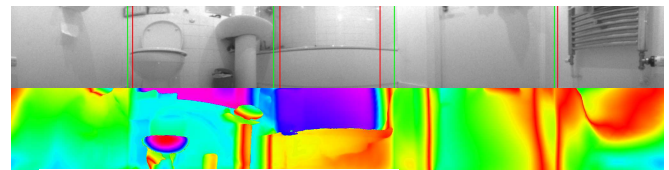


Fig. 10: Dataset “Elephant & Castle Bathroom”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

Bathrooms (Figure 10) are extremely challenging datasets for any kind of computer vision method, often even for active sensors (e.g. RGB-D), due to the lack of texture and

specularity of virtually all surfaces. The only saving grace is that such rooms are frequently small. In this particular example the side of the bathtub became a wall for the robot, which was to be expected.

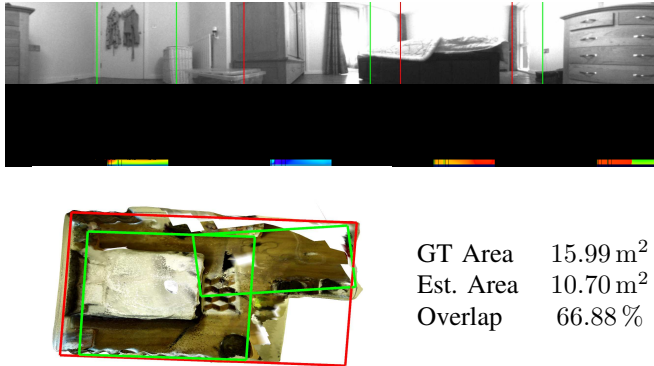


Fig. 11: Dataset “Elephant & Castle Bedroom 1”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

In Figure 11, as mentioned before, our box model is not always a good fit for real rooms. Here we see that the box-shaped ground truth is actually poor and our incremental estimation represents the L-shaped room faithfully.

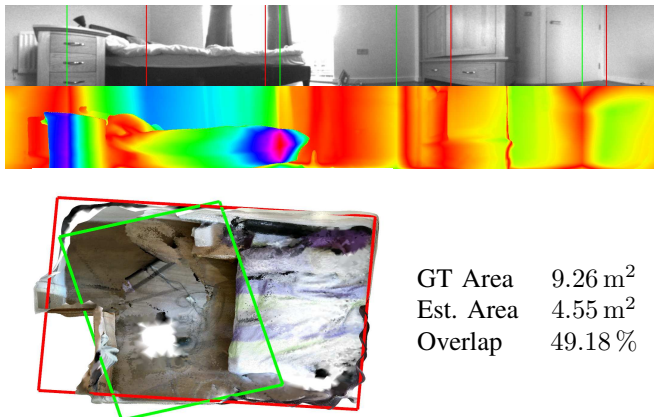


Fig. 12: Dataset “Elephant & Castle Bedroom 2”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

Our method has some drawbacks, as seen in Figure 12. Due to the purely geometric nature of our optimization sometimes the estimated cube doesn’t align with the vanishing lines and fits the depth map instead. In this experiment the complex structure of the room combined with the robot’s location (viewpoint) produced an estimate that is roughly correct in size, but erroneously rotated to encompass maximum area. The vanishing lines of interest in our case are primarily horizontal, and are not well estimated due to the planar motion of the robot.

The result presented in Figure 13 is almost a perfect fit, despite the complex nature of the dataset (overexposed

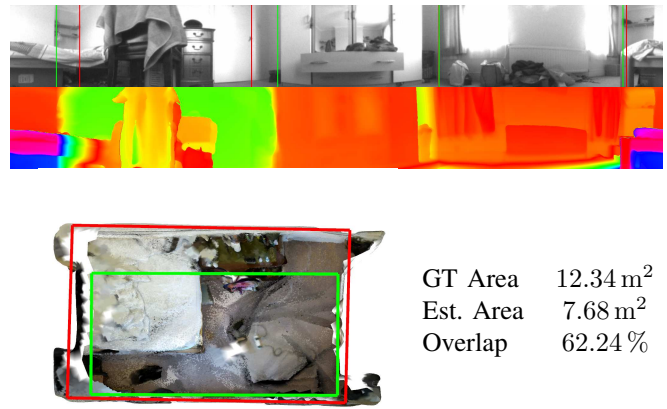


Fig. 13: Dataset “Wandsworth Bedroom 3”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

windows, mirrors). The only difference when compared to ground truth is caused by the chair/desk which are prominent in the field of view and highly textured, while the wall behind is completely textureless and thus devoid of depth estimates that our method could rely on.

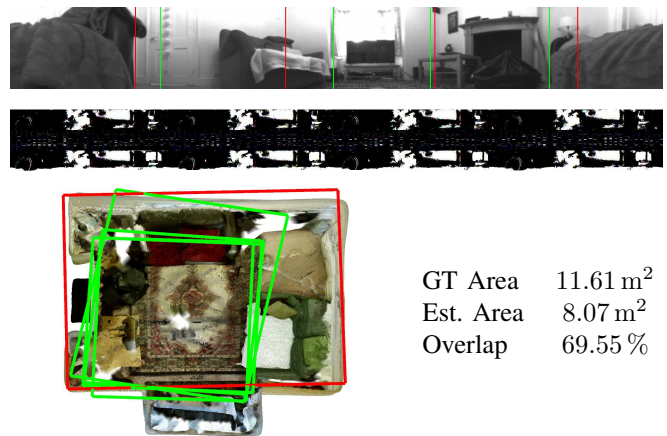


Fig. 14: Dataset “Wandsworth Livingroom”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

We also wanted to test how our estimates overlap with multiple viewpoints in the same room, as shown in the Figure 14. As in Figure 12 the misalignment of the boxes is an inherent drawback of our method in some types of scenes, but the overall overlap of the boxes is very good. It is important to note that one cannot guarantee perfectly overlapping boxes from varying viewpoints, as different parts of the scene would exhibit themselves differently from each viewpoint. This can be seen in the first, larger, box, when the robot was far away from the sofa (close to the TV), thus the sofa on the opposite side was not an overwhelming large part of our depth map. That is why the estimate reaches the back wall correctly. However, in subsequent viewpoints, in the centre of the room and on the opposite side this scenario

no longer holds and the estimates are guided more by the large pieces of furniture around the robot.

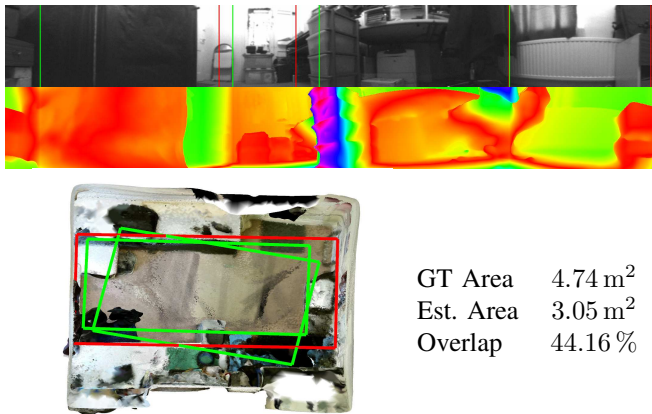


Fig. 15: Dataset “Wandsworth Lab”; from top to bottom: reference frame, raytraced depth map error, top down view with the box estimate.

While we are aiming to estimate the shape of a room, this is often difficult, as the robot is low on the ground, and close to furniture, and especially some furniture exhibits overhangs. This was the case in dataset 15. Here a top-down view sometimes doesn’t do justice to the method. While the first square is roughly correct, the second one is reduced in size and misaligned due to two factors. The bedside on the left hand side is the furthest away object that still has discernible texture, thus acceptable depth estimates. The wall behind it is even further away and textureless, and the same is true of the back of the upright bed that is entirely black. On the other side, the robot was very close to a workbench, which has an empty cavity below the surface, therefore these under the desk depth estimates allowed the final box to rotate.

C. Office Data

In addition to typical household environments, the system was evaluated on office rooms. These are, in our experience, more challenging as often the amount of texture is substantially lower, the overall scale is larger and there are more specular objects (e.g. big windows, glass walls, panel lighting). Ground truth is marked as in the household datasets.

Dataset 16: ground truth dimensions: unknown, estimated: $3.54 \times 2.03\text{m}$. In this case the room is of pentagonal shape, with one wall curved and another being made of glass. Here the method fits the room model to the most pronounced furniture, bookshelves and the edge of the desk.

Dataset 17: ground truth dimensions: $3.14 \times 2.87\text{m}$, estimated: $2.90 \times 2.29\text{m}$. This is another dataset aimed to test the method at extreme boundary conditions. Here the room is almost completely without texture. A slight error in one direction is caused by the desk being the only major feature in the room, skewing the estimate.

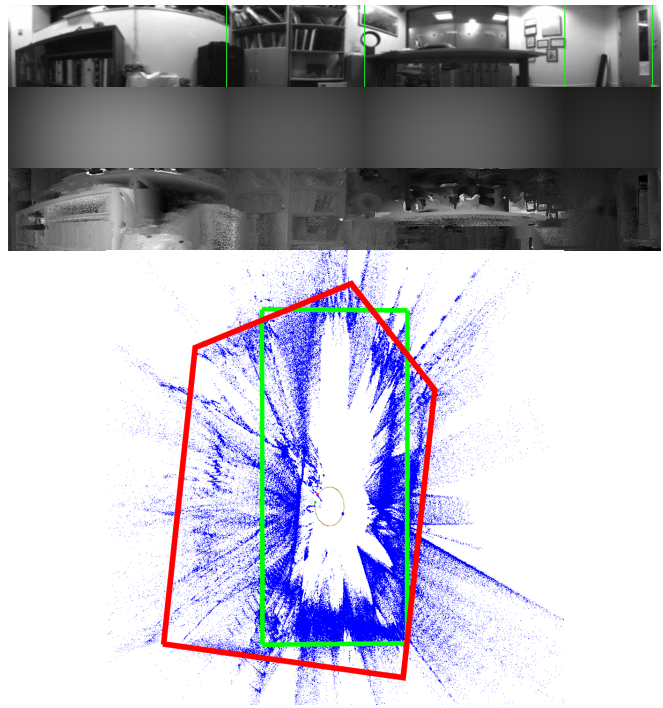


Fig. 16: Dataset “Cluttered Office”; from top to bottom: reference frame, raytraced depth map, raw depth map, top down view of the point cloud with the box estimate.

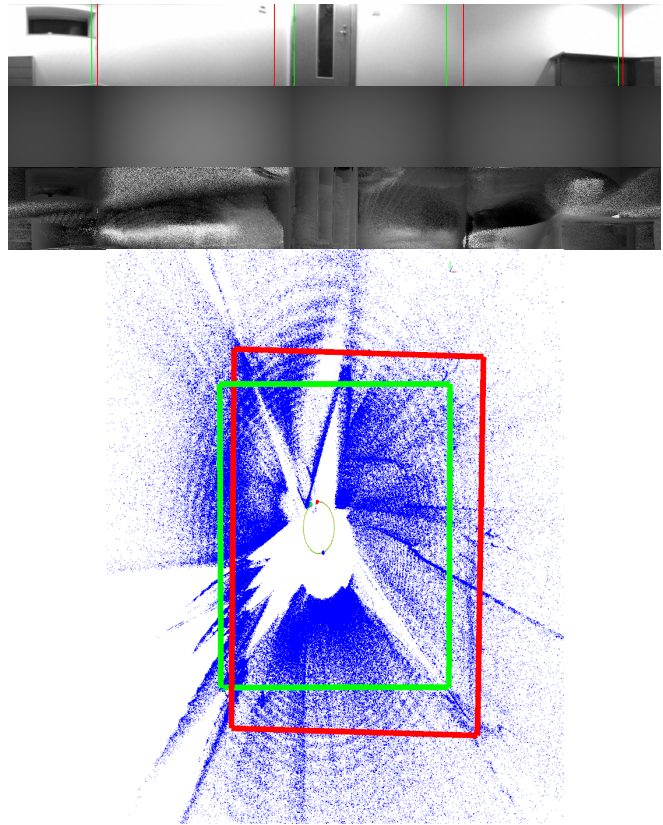


Fig. 17: Dataset “Empty Office”, from top to bottom: reference frame, raytraced depth map, raw depth map, top down view of the point cloud with the box estimate.

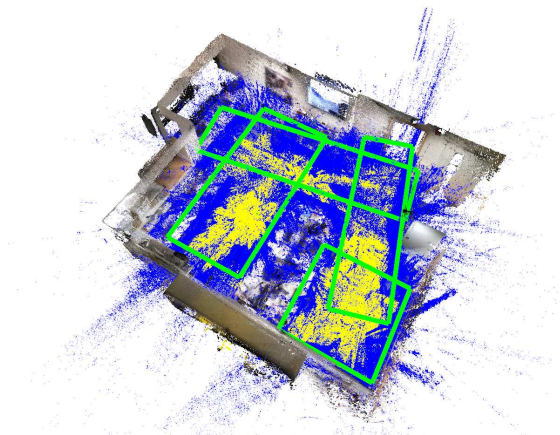


Fig. 18: Dataset “Lab”, showing a union of multiple box estimates.

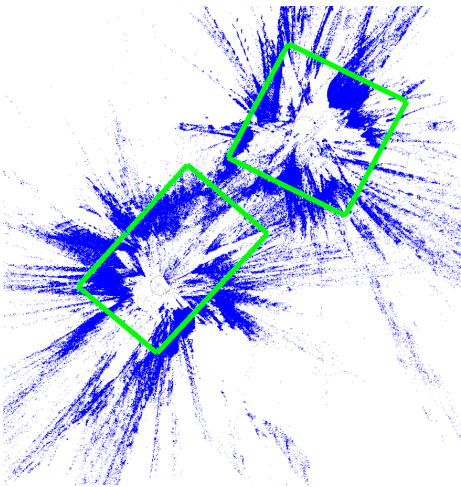


Fig. 19: Dataset “Through Door”, showing two non-overlapping boxes, enabling inference that the robot passed between rooms.

D. Additional applications

Here we present additional possible applications of our method, beyond single room shape estimation:

- Large scale, more complex, room mapping can be performed in an incremental manner during active exploration by computing the boolean union of single box estimates; Figure 18 is a typical example of such a result,
- The method, employed in an active exploration loop, could allow for room demarcation and other higher level reasoning. Figure 19 shows two box estimates, each taken in a separate room. The fact that boxes don’t overlap indicates that the robot has driven through a portal (door).
- As the method is robust, the result of the method (room dimension estimates) could be fed into a Machine Learning classifier to enable room recognition,
- Swapping asymmetric cost function coefficients estimates the innermost box instead of the outermost boundary. This can be thought of as free space estimation. This representation may be beneficial when

compared to occupancy grids as presented in [6].

V. CONCLUSIONS

In this paper we have presented a novel room size estimation approach for a small mobile robot equipped with an omnidirectional camera. The method underwent extensive testing in a series of experiments, ranging from purely synthetic data to real household environments, often pushing the conditions to the limits of passive monocular vision. The algorithm, implemented in C++ with GPU acceleration, is running on a PC in a matter of seconds and we see few major obstacles for the algorithm to be ported to the embedded processors of current household robots — the method is not intended to be run at frame rate. In future work, we thus would like to explore in more depth the ideas presented in Section IV-D. We also believe that a fisheye camera would exhibit better performance than our current setup, as it has wider upward viewing angles that would allow our method to use depth measurements on the wall-ceiling boundary, therefore improving the final estimates, especially of rotation.

ACKNOWLEDGMENT

Research presented in this paper has been supported by Dyson Technology Ltd.

REFERENCES

- [1] V. Hedau, D. Hoiem, and D. A. Forsyth, “Recovering the spatial layout of cluttered rooms.” in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009.
- [2] Y. Zhang, S. Song, P. Tan, and J. Xiao, “Panocontext: A whole-room 3d context model for panoramic scene understanding,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.
- [3] R. Cabral and Y. Furukawa, “Piecewise planar and compact floorplan reconstruction from images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [4] M. Loper and M. J. Black, “OpenDR: An Approximate Differentiable Renderer,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.
- [5] J. Zienkiewicz, A. J. Davison, and S. Leutenegger, “Real-Time Height-Map Fusion using Differentiable Rendering,” in *Proceedings of the IEEE/RISJ Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [6] R. Lukierski, S. Leutenegger, and A. J. Davison, “Rapid Free-Space Mapping From a Single Omnidirectional Camera,” in *Proceedings of the European Conference on Mobile Robotics (ECMR)*, 2015.
- [7] C. Geyer, “Catadioptric projective geometry: theory and applications,” Ph.D. dissertation, University of Pennsylvania, 2003.
- [8] J. P. d. A. Barreto, “General central projection systems: Modeling, calibration and visual servoing,” Ph.D. dissertation, University of Coimbra, 2004.
- [9] S. Leutenegger, M. Chli, and R. Siegwart, “BRISK: Binary robust invariance scalable keypoints,” in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.
- [10] S. Agarwal, M. K., and Others, “Ceres solver,” <http://ceres-solver.org>.
- [11] R. A. Newcombe, S. Lovegrove, and A. J. Davison, “DTAM: Dense Tracking and Mapping in Real-Time,” in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.
- [12] T. Möller and B. Trumbore, “Fast , Minimum Storage Ray / Triangle Intersection,” *Journal of Graphics Tools*, vol. 2, no. 1, pp. 21–28, 1997.
- [13] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison, “Real-Time Camera Tracking: When is High Frame-Rate Best?” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.
- [14] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison, “ElasticFusion: Dense SLAM without a pose graph,” in *Proceedings of Robotics: Science and Systems (RSS)*, 2015.