In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS 2018), Madrid, Spain October 2018

PRISM: Pose Registration for Integrated Semantic Mapping

Justin W. Hart, Rishi Shah, Sean Kirmani, Nick Walker, Kathryn Baldauf, Nathan John, and Peter Stone
Department of Computer Science, University of Texas at Austin, Austin, Texas, USA

***Abstract*— Many robotics applications involve navigating to positions specified in terms of their semantic significance. A robot operating in a hotel may need to deliver room service to a named room. In a hospital, it may need to deliver medication to a patient's room. The Building-Wide Intelligence Project at UT Austin has been developing a fleet of autonomous mobile robots, called BWIBots, which perform tasks in the computer science department. Tasks include guiding a person, delivering a message, or bringing an object to a location such as an office, lecture hall, or classroom. The process of constructing a map that a robot can use for navigation has been simplified by modern SLAM algorithms. The attachment of semantics to map data, however, remains a tedious manual process of labeling locations in otherwise automatically generated maps. This paper introduces a system called PRISM to automate a step in this process by enabling a robot to localize door signs – a semantic markup intended to aid the human occupants of a building – and to annotate these locations in its map.**

I. Introduction

Emerging applications in robotics involve systems in which robots navigate to static, known locations in order to perform their tasking. In environments like hotels, hospitals, and university buildings these locations are frequently understood through a semantic label such as the room number of a classroom or office. Whereas robots generally use Simultaneous Localization and Mapping (SLAM) algorithms for navigation, the poses associated with such semantics are generally labeled through manual processes after the SLAM map has been constructed. Humans, on the other hand, leverage semantics provided to them by the building itself in the form of signage such as room number placards.

A planar homography is a 2D projective transformation which can be thought of as the image of a rigidly transformed 3D plane that has been projected into 2D space. A familiar application of planar homographies is Zhang's Method [1]. Zhang's method is the most common camera calibration algorithm, which utilizes homographies computed from images of chessboard calibration targets. In the case of a camera with known calibration, a homography can be used to determine the pose of a planar target with respect to the camera.

The rectangular placards commonly placed in office buildings provide a convenient geometry for homography computation, from which it is possible to determine their position for labeling on a map.[1] The computed homography can also be used for image rectification, enabling software to compute an image that looks as if it were taken directly facing the sign. This rectified image is simpler for text extraction systems to read than one that may be taken at a non-perpendicular angle, and thus can be used to improve text extraction performance. The system presented in this paper allows a robot that is performing SLAM to automatically detect room placards and to compute the position and rectification of these placards using homographies.

Many organizations, including hospitals, offices, and universities, adopt uniform signage standards. These standards ensure that the geometry, placement, design and typography of placards is consistent and accessible to people with vision impairments. In the United States, these standards are enshrined in law [2]. The prevalence of such signage, reinforced by these standards and regulations, makes the approach presented in this paper broadly applicable to a wide variety of public buildings and businesses.

The unique contribution of this paper is the use of standard computer vision techniques for the purpose of automating the annotation of room numbers. A scanner locates room number placards by finding quadrilaterals in images sampled by the robot's vision system. A homography is then computed to estimate the poses of these signs. These estimates are also used for image rectification to improve text extraction, and the real sign's pose is optimized based on the transformation of a model sign. The system also employs a speech interface to interactively inform a human operator that it has found a potential room placard, allowing the operator to focus the robot's attention on the sign. We call the developed system PRISM: Pose Registration for Integrated Semantic Mapping. We plan to further extend this system to enable our robots to autonomously maintain semantic map annotations for the buildings in which they operate, as well as annotate the locations of objects found in the environment. This system starts with office placards partly due to their relevance to functionality that is currently in place in the Building-Wide Intelligence (BWI) project. Annotations for room numbers are maintained in the robot's knowledge base for many of the tasks that the system performs [3].

II. Related Work

Robotic navigation can be broadly characterized into mapless and map-based methods. Mapless navigation does not use an explicit representation of space, relying instead on visual techniques, such as the recognition or tracking of objects in the environment, to generate a motion plan [4], [5]. Map-based approaches are popular and effective in practice. They often represent free-space and obstacles in a 2D or 3D occupancy grid [6]. PRISM augments a map-based representation with semantic annotations.

[1]The inversion of the rigid transformation computed to determine this pose could also be used to determine the position of the robot on the map. This is left to future work.

Significant research has been dedicated to the extraction and annotation of various forms of semantic information for inclusion in spatial maps. Bormann et al. [7] and Armeni et al. [8] have extracted room segmentations (e.g. kitchen, living room, bedroom) from 2D and 3D representations, respectively. Pillai [9] and Blodow [10] both propose systems that detect sets of objects and extract their pose with respect to world maps. Case et al. [11] present a system that bears a similarity to PRISM in that it extracts text by automatically reading signs such as door placards. PRISM differs significantly from theirs in its implementation. Their system focuses on a capability of the robot to autonomously wander hallways to find signs, the problem of text detection in images, and on the post-processing of text extracted from the image. PRISM exploits the planar geometry of signs explicitly in its implementation. It approximates the pose of each placard through the use of homographies. Homographies are also utilized for image rectification, improving text extraction performance. The system then refines estimated poses through an optimization based on a 3D model placard. Whereas the system presented in Case et al. [11] autonomously wanders the hallways of their building in order to find signs, it requires a preexisting map to extract these hallways from. PRISM samples data interactively with a human user, during the initial construction of its map; combining the human operation step for mapping with the data collection process for room placards. It is able to simultaneously construct a map using SLAM techniques [12] and localize and annotate the positions and contents of room signs onto this map.

III. Background

Homogeneous coordinates describe a projective coordinate system in which a point in 2D space is expressed as a vector with three elements, $< x, y, w >$ and a point in 3D space is described as a vector with four elements $< x, y, z, w >$. The equivalent Cartesian representations are $(x : w, y : w)$ and $(x : w, y : w, z : w)$, respectively. As such, the interpretation of homogeneous coordinates is unchanged by scalar multiplication. Homogeneous coordinates are a useful representation for computer vision problems because they unify translation and rotation into a single matrix multiplication called a *rigid transformation*. They enable a convenient representation of perspective, the effect whereby parallel lines appear to meet in the distance.

The standard pinhole camera model, Equation 1, is used to represent the calibrations of cameras in this system, where α and β denote focal length, γ is a skew factor in the 2D image, and (u_0, v_0) is the center of the 2D image sampled by the camera, known as the *principal point*. This representation is referred to as the *camera intrinsic matrix*.

$$A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

A planar homography, H, is a 2D projective transformation, which can be thought of as the image of a plane transforming through space. It can be inferred from paired images of planar targets, but is generally computed as the transformation between a model target stored in memory and the image of a similar target found in the real world. For a homography computed between an image sampled by a calibrated camera and a model target, the rigid transformation between the camera and the target can be inferred [1].

Fig. 1: Image of one of the placards used to test PRISM.

IV. Methods

PRISM automatically constructs and annotates maps with semantic data extracted from room placards, Figure 1, in images sampled by the robot's vision system based on the robot detecting, localizing, and extracting text from signs in its environment during map construction. It is assumed that the robot is running a SLAM algorithm for the construction and interpretation of its navigational map as well as for the estimation of its pose in its environment. We have chosen Hector SLAM [12] for this purpose. As configured, the system creates a cost map based on input from a front-facing Hokuyo UST-20LX laser rangefinder mounted in the robot's base. The inputs to PRISM are images from the robot's vision system paired to estimates of its pose in the map; though the map can be constructed on-the-fly and then paired to the image data by recording a history of the robot's inputs and computing this pairing once loop closure has been established.

PRISM can be divided into four stages. In the first, quadrilateral regions are extracted from images sampled by the robot's vision system as candidates to contain a placard. In the second stage, the pose of each placard is estimated using a two-step process where in the first step an estimated pose is computed from its planar homography and in the second step an optimization is computed with respect to a simple 3D model of the placard. In the third stage, the initially-inferred homography is used to compute an image rectification, and text is extracted from this rectified image. Finally, in the fourth stage, multiple samples hypothesized to represent the same sign are aggregated to compute an improved estimate of each placard's pose and text contents. PRISM is run during the construction of the system's cost map, a manual process

on our robot in which an operator drives the robot around the area to be mapped. We supplement this process by asking the operator to turn the robot to face room placards head-on in order to be classified. To help guide the human operator, the robot provides speech feedback indicating when it has located a placard. For this purpose, we use the eSpeak text-to-speech engine. Additionally, we provide visual feedback in RViz [13] when the system has located a placard. An outline of the PRISM algorithm is provided in Algorithm 1.

Algorithm 1 PRISM

procedure DETECT SIGN
    Detect placard corners and approximate polygons as in Section IV-A.1.
    Filter sign detection results as in Section IV-A.2.
end procedure
procedure ESTIMATE PLACARD POSE
    Compute homography between model placard and detected points using DLT method as in Section IV-B.1.
    Estimate placard pose by Equation 4.
    Refine rotation matrix by Equation 8.
    Optimize placard pose by Equation 9.
end procedure
procedure EXTRACT TEXT
    Rectify image based on homography, using *cv::warpPerspective* [14].
    Extract text from placard using Tesseract OCR [15].
end procedure
procedure AGGREGATE DATA
    Estimate sign pose from aggregated samples as in Section IV-D.
end procedure

A. Sign Detection

PRISM first detects signage to be annotated and extracts point correspondences for computations against a model placard. In theory, detectors for a variety of signs could be plugged into the system by replacing this part. The detector presented in this section should be able to detect a variety of rectangular office placards, but has only been tested against the office placards in the UT Austin Computer Science Department.

1) Detect Potential Signs from Features Bounding Quadrilaterals: This method begins with the extraction of four corners defining a warped quadrilateral from images from the robot's computer vision system during navigation. To do so, it uses the approach from Suzuki and Abe [16] then attempts a polygonal approximation with the Ramer–Douglas–Peucker algorithm [17]. If the approximation has more than four vertices, the approach relaxes the approximation epsilon up to a constant ratio of the contour length. If this process produces an approximation with four vertices the image region is considered to be a candidate for containing a placard. This method is selected over other approaches such as Harris corner detection [18], as such methods do not associate sets of four points as belonging to the same quadrilateral.

Fig. 2: Crosshairs mark candidate quadrilaterals detected by the placard detector. False positives are visible in this image, which are filtered out prior to final annotation. Two sets of crosshairs for the true placard are visible, illustrating the importance of the sample aggregation procedure.

2) Filter to Remove False Positives: Preliminary testing revealed this approach to have very high recall. Most images contain many closed contours that are well-approximated as quadrilaterals; which will be found by this approach. Quadrilaterals with very large area often correspond with non-placard scene elements, such as ceiling panels or walls. Those with low area are unlikely to provide useful text extraction output. If such quadrilaterals do contain placards which are simply far away, better 3D pose information can be extracted from images sampled at a closer range. The relationship between 2D pixels and 3D positions can be stated more succinctly in terms of degrees of visual angle. As a 3D position moves farther away from the camera, a degree of visual angle subtends greater 3D space, thus providing only coarser estimates of true 3D pose. As such, candidate contours with area less than .1% and greater than 10% of the total image area are removed. Figure 2 shows an example of the output of this stage of the pipeline.

B. Estimation of Placard Pose

The process of computing the pose of each placard proceeds in four steps. In the first, a homography is computed between a model placard and the image of a placard as detected by the method described in Section IV-A. In the second, a candidate pose is estimated from the homography. In the third, this estimate is refined over a nonlinear optimization between the corners of a model 3D placard as projected down into 2D space using the model projection of the calibrated camera and the detected 2D corners. In the fourth, samples of placard data are clustered based on position and then aggregated into a global pose estimate for each placard.

1) Homography Computation: Computing a homography between a model placard and a real placard requires measurements. The placards used in the computer science department at UT Austin measure $151\text{mm} \times 51\text{mm}$. A homography relates model corners to image corners through Equation 2, where x is a model corner, x' is its corresponding corner found in image data, and H is the homography. Though a homography is a 3×3 matrix, it has only 8 degrees of freedom. This is because homogeneous coordinates are unaffected by scalar multiplication. As such, a set of constraints can be derived whereby the terms of the homography can be recovered through a set of 4 or more image correspondences. This is accomplished by rearranging the terms of Equation 2 such that the terms of the homography can be found as the right null space of the system. This method is referred to as the Direct Linear Transformation (DLT) method. Hartley and Zisserman [19] provide a thorough treatment of this subject.

$$x' \times Hx = 0 \tag{2}$$

2) Recovering placard pose from its homography: The image coordinates of a rigidly transformed 3D point can be computed by multiplying their ideal projection by the camera intrinsic matrix, as in Equation 3, where R is the object's orientation and t is its position. The extension of the 2D target used in homography computation to 3 dimensions can be accomplished by laying the target in the plane perpendicular to the z axis of the camera with the model z terms set to 0. A homography describing the transformation of this target can be computed as in Equation 4 [1].

$$x' = A[R|t]x = A[r_1 r_2 r_3 t]x \tag{3}$$

$$x' = A[r_1 r_2 t]x \tag{4}$$

While homographies are invariant to scalar multiplication, the transformations R and t are applied over Euclidean space, and thus their factors in Equation 4 must be normalized. This can be accomplished by dividing r_1, r_2, and t by the norm of r_1. The rotation matrix R can then be computed as in Equation 5.

$$R = [r_1 r_2 (r_1 \times r_2)] \tag{5}$$

The computation of the pose of the placard in the image relies on having the camera intrinsic matrix. A calibration for the cameras in the present system was computed using in-house software which first approximates camera calibrations using Zhang's method [1] then refines those calibrations through bundle adjustment [20].

The matrix R obtained in (5) may not be a proper rotation matrix due to empirical errors. This can be corrected by finding the rotation that is "closest" to the estimate of R. For mathematical convenience, the Hilbert-Schmidt operator norm is used as our notion of distance, as in Equation 6. $(\cdot)^*$ denotes the Hilbert adjoint. As such, R can be corrected by dropping the diagonal matrix from its singular value decomposition, as in Equation 8, giving us the closest rotation matrix, S. This can be thought of simply as rectifying the transformation such that it is orthonormal.

$$\begin{aligned} \arg\min_S \|R - S\|^2 = \operatorname{trace}((R-S)^*(R-S)) \\ \text{s.t. } S^*S = I \end{aligned} \tag{6}$$

$$R = UDV \tag{7}$$

$$S = UV \tag{8}$$

3) Optimization of Placard Pose: PRISM computes an initial estimate of the pose of each placard using a homography, but goes on to refine this estimate using nonlinear optimization. This approach echoes that of Zhang's method [1] for camera calibration, wherein rigid transformations to 3D planar targets are computed from 2D homographies, then refined in the same fashion. Nonlinearities such as noise in the localization of the corners of each placard can contribute to error in determining the pose of the placard. It is straightforward to formulate an optimization between the model placard and extracted image corners by extending Equation 3 as in Equation 9, where X is the set of model placard corners, and x' is the point in the image corresponding to each x. The variable R is stated either as a rotation vector by Rodrigues' formula or as a quaternion. In PRISM, it is represented as a quaternion. Care must be taken to maintain R as a valid rotation.

$$\arg\min_{(R,t)} \sum_{x \in X} \|A[r_1 r_2 r_3 t]x - x'\| \tag{9}$$

C. Placard Reading and Annotation

Once a placard's pose has been determined, PRISM attempts to read the text off of it in order semantically annotate the map. The OpenCV [14] function *cv::warpPerspective* is used to rectify the placard's image prior to text extraction. The Tesseract OCR engine is then used to extract text [15]. The engine is constrained to detecting uppercase English letters, numbers, and periods. If no text can be extracted, it is determined that the scanner detected a false-positive placard, which is then discarded.

D. Sample Aggregation

The final stage of PRISM aggregates samples to improve the estimate of each placard's pose. Since multiple images are sampled for each placard, this data can be aggregated to improve PRISM's precision. Samples are first clustered geometrically. As there is only minimal noise present by this stage of the pipeline, it is sufficient to group all samples within radius 50cm of each other. Each placard's position is determined as the mean of the estimated position of the clustered samples.

The rotation estimate is determined as the average of the estimated sample quaternions; found according to [21]. This entails solving an optimization problem whose solution follows.

TABLE I: Sign Detection Metrics

	Rectified	Unrectified
Num Signs	49	49
Total Detections	51	42
False Positives	4 (2 duplicates)	0
False Negatives	2	7
Precision	92%	100%
Recall	96%	86%

Define M as the outer products of the quaternions, Equation 10.

$$M = \sum_{i=1}^{n} q_i q_i^\top \tag{10}$$

The desired average quaternion is the unit eigenvector of M corresponding to the largest eigenvalue.

V. Experiments and Results

PRISM was deployed on a Toyota Human Support Robot (HSR), shown in Figure 3. Development and tuning of the algorithm were performed in the north corridor of the AI floor of the Computer Science Department at UT Austin. Results are presented for evaluations on the south corridor of the same floor. In these tests, the robot is driven through a set of student cubicles, with performance evaluated on its ability to extract the pose and text of signs marking these locations. The robot uses an omnidirectional drive base for locomotion and has a Point Grey Chameleon global shutter camera mounted in a pan tilt unit representing the robot's head. Each eye camera outputs a 1.25 megapixel image. These tests use only the robot's left eye camera. They evaluate the performance of the system on multiple fronts, including placard detection, localization, and text extraction. The robot's operator was instructed to drive the robot to directly face the placard labeling the cubicle's room number. An interface implemented in RViz [13] enables the operator to see if a placard is localized, as does speech feedback from the robot.

No ground-truth positional data regarding the precise poses of the placards placed in the environment exists since the robot is operating in the real world where it would be difficult to make such measurements. Therefore, no quantitative results are presented regarding PRISMS's ability to register the pose of each placard. Results are presented for precision and recall of the robot's ability to detect each of the placards from the set in the corridor. The robot performs perfectly in localizing these placards in the corridor in which it was tuned. On the test corridor, the robot finds each placard with 92% precision and a 96% recall rate, as can be seen in Table I.

A. Sign Localization Performance

To provide a more intuitive impression of how well the robot determines the pose of room placards, Figure 4 compares the automatic annotations made by the robot against human performance. Automatic annotations are marked as orange arrows overlaid with human annotations marked with blue arrows. The mean distance between automatically-annotated and human-annotated placards is 0.936m (SD = 0.412m). However, we are significantly more prone to trust the robot's annotations than those of the human. The human annotation process involves marking up the processed cost map with no direct visual reference to scene geometry. On the other hand, sources of error in the robot's placard localization are simply cumulative errors in the robot's odometry combined with errors in visual localization. These errors should be relatively small. This agrees with a simple qualitative analysis. For example, see Figure 6, which is based on the robot's annotation, and Figure 5, which is based on human-annotations. The robot is also able to perform this annotation while constructing its navigational map using SLAM, whereas the human annotations require additional manual effort. It took roughly 28:21 (mm:ss) for the human operator to collect this data in terms of time driving the robot using PRISM.

Fig. 3: The Toyota HSR used for these experiments.

The data collected to construct this map was used for both the annotation provided by the PRISM system and that provided by the human annotator, but a second cost map was constructed so as to provide an estimate of the overall time and effort required for map construction and manual annotation. One would imagine that the additional effort of directing the robot towards placards in the PRISM case would cause cost map construction to be slower. We had an

Fig. 4: SLAM map with placard location annotations overlaid. Human annotations are in blue (dotted line). PRISM annotations are in orange (solid line).

TABLE II: Text Extraction Accuracy

Edit Distance	Rectified (N=47)	Unrectified (N=42)
0	60%	50%
1	92%	88%
2	96%	97%

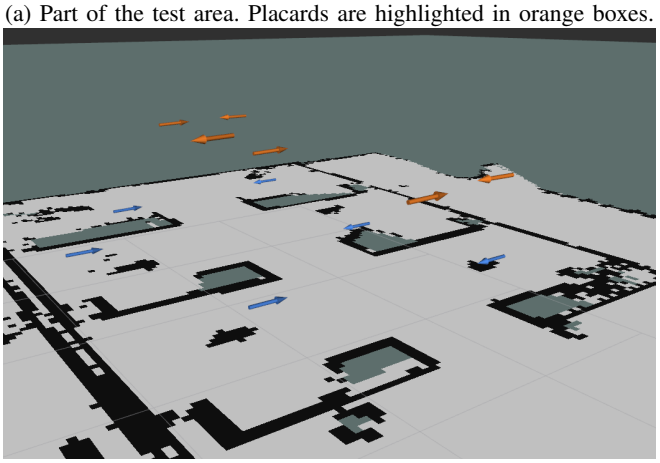
operator drive the robot to construct a cost map in the manner which we have classically used alongside human annotation, without running PRISM. It took a similar amount of time, at 29:26 (mm:ss). The additional effort of human annotation took 18:51 (mm:ss). As the system matures, we expect this entire process to happen autonomously and to run in the background while the robot attends to other tasking.

In addition to typical 2D annotations, PRISM is able to mark the 3D location of each placard in its map, creating more-detailed annotations and opening up the possibility of applications where the robot must indicate the placard (or potentially other items) to a user. Figure 6 shows a side-by-side example of an area of the test corridor and the annotated placards rendered in RViz. Figure 7 provides a 3D view of the annotated map with placards rendered as orange boxes clearly lined up in rows matching the building's arrangement and floor geometry.

B. Text Extraction Performance

Text extraction performance results are presented as mean Levenshtein distance [22] from ground truth for the system in two modes. The first uses unrectified images passed directly into Tesseract OCR. The second rectifies the image using a homography before text extraction. Results can be found in Table II. Here, we see that while rectification improves results, there is room for improvement regarding reliable text transcriptions from the room placards.

(a) Part of the test area. Placards are highlighted in orange boxes.



(b) Perspective view of the same area with human (blue) versus PRISM (orange) annotations.

Fig. 5: Subfigure (a) shows the true locations of placards imaged in the test environment. In Subfigure (b) it can be seen that the human annotations differ from these locations.

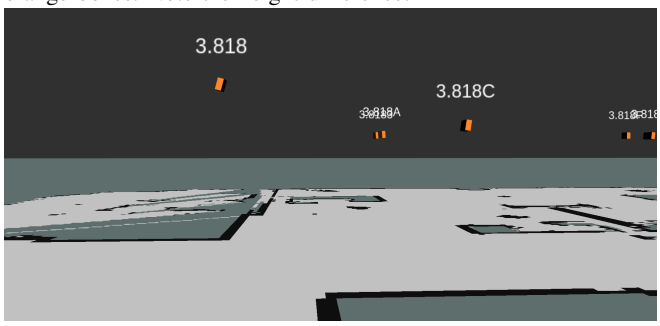
VI. Conclusion

PRISM is a system which enables a robot to localize room placards, transcribe their contents, and automatically annotate them on a map. It utilizes existing SLAM techniques for the navigational component of this task, focusing on adding markup to the map denoting the semantics of the locations it contains. PRISM extends the existing systems in the Building-Wide Intelligence project to enable the robots to automatically extract semantics which otherwise require hand-annotation. The construction of such semantic maps will be a significant enabling technology for emerging applications where robots must navigate to known locations on a map based on their semantic significance.

Acknowledgement

This work has taken place in the Learning Agents Research Group (LARG) at the Artificial Intelligence Laboratory, The University of Texas at Austin. LARG research is supported in part by grants from the National Science Foundation (CNS-1305287, IIS-1637736, IIS-1651089, IIS-1724157), The Texas Department of Transportation, Intel,

(a) Image of a portion of the test area. Placards are highlighted in orange boxes. Note the height difference.



(b) 3D poses of the same area rendered in RViz using annotations automatically captured by PRISM. Notably, the placards are visibly in the correct positions.

Fig. 6: Demonstrating the 3D annotation capability, Subfigure (a) shows a section of the test area with two placards at different heights. Subfigure (b) shows the same placards as extracted, annotated, and rendered in RViz.

Raytheon, and Lockheed Martin. Peter Stone serves on the Board of Directors of Cogitai, Inc. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

References

[1] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[2] Americans with disabilities act: Accessibility guidelines for buildings and facilities. https://www.access-board.gov/attachments/article/1350/adaag.pdf.

[3] Piyush Khandelwal, Shiqi Zhang, Jivko Sinapov, Matteo Leonetti, Jesse Thomason, Fangkai Yang, Ilaria Gori, Maxwell Svetlik, Priyanka Khante, Vladimir Lifschitz, J. K. Aggarwal, Raymond Mooney, and Peter Stone. BWIBots: A platform for bridging the gap between AI and human–robot interaction research. *The International Journal of Robotics Research*, 36(5–7):635–659, 2017.

[4] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2938–2946. IEEE, 2015.

[5] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7272–7281. IEEE, July 2017.

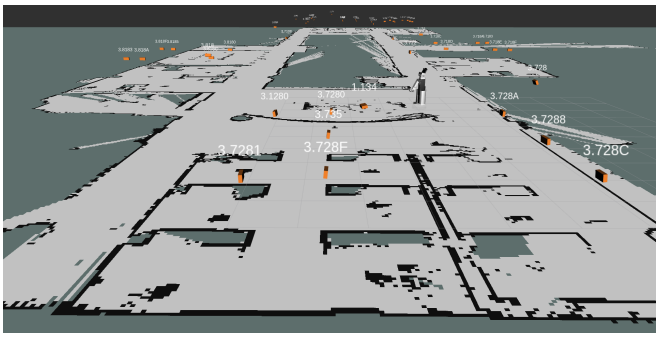


Fig. 7: Perspective view of map with localized placards.

[6] David Filliat and Jean-Arcady Meyer. Map-based navigation in mobile robots:: I. a review of localization strategies. *Cognitive Systems Research*, 4(4):243 – 282, 2003.

[7] Richard Bormann, Florian Jordan, Wenzhe Li, Joshua Hampp, and Martin Hägele. Room segmentation: Survey, implementation, and analysis. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1019–1026. IEEE, May 2016.

[8] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1534–1543, June 2016.

[9] Sudeep Pillai and John J. Leonard. Monocular SLAM supported object recognition. *Computing Research Repository*, abs/1506.01732, June 2015.

[10] Nico Blodow, Lucian Cosmin Goron, Zoltan-Csaba Marton, Dejan Pangercic, Thomas Rühr, Moritz Tenorth, and Michael Beetz. Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4263–4270. IEEE, 2011.

[11] Carl Case, Bipin Suresh, Adam Coates, and Andrew Y. Ng. Autonomous sign reading for semantic mapping. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3297–3303, 2011.

[12] Stefan Kohlbrecher, Johannes Meyer, Thorsten Graber, Karen Petersen, Uwe Klingauf, and Oskar von Stryk. Hector open source modules for autonomous mapping and navigation with rescue robots. In Sven Behnke, Manuela Veloso, Arnoud Visser, and Rong Xiong, editors, *RoboCup 2013: Robot World Cup XVII*, pages 624–631, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[13] Hyeong Ryeol Kam, Sung-Ho Lee, Taejung Park, and Chang-Hun Kim. Rviz: a toolkit for real domain data visualization. *Telecommunication Systems*, 60(2):337–345, October 2015.

[14] Gary Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[15] Ray Smith. An overview of the tesseract OCR engine. In *Proceedings of the International Conference on Document Analysis and Recognition*, volume 2, pages 629–633. IEEE, 2007.

[16] Satoshi Suzuki and Keiichi Abe. Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, 30(1):32–46, 1985.

[17] Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing*, 1(3):244–256, 1972.

[18] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Vision Conference*, pages 147–151, 1988.

[19] Richard I Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[20] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment - a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999.

[21] Landis Markley, Yang Cheng, John Crassidis, and Yaakov Oshman. Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, July.

[22] VI Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707, 1966.