Physical Words for Place Recognition in Dense RGB-D Maps

Ross Finman[1], Thomas Whelan[2], Liam Paull[1], and John J. Leonard[1]

***Abstract*— Appearance-based place recognition systems have been shown to be effective for large-scale mapping but have notable shortcomings. Visual bag-of-words dictionaries require offline training, have tens of thousands of words, and are susceptible to changing environments, either due to lighting or physical changes, between training and deployment. Recent advances allow for online 3D mapping and segmentation using dense RGB-D data. Here we propose the natural extension of previous visual dictionaries to the 3D world through the use of *physical words* that are used to perform place recognition. The main advantages of this approach is generating and detecting physical words is invariant to aspect and lighting changes, and require less words in our *physical dictionary* to recognize scenes. We demonstrate this concept on multiple real world datasets under extreme lighting variations and camera trajectories that typical appearance-based approaches have difficulty with.**

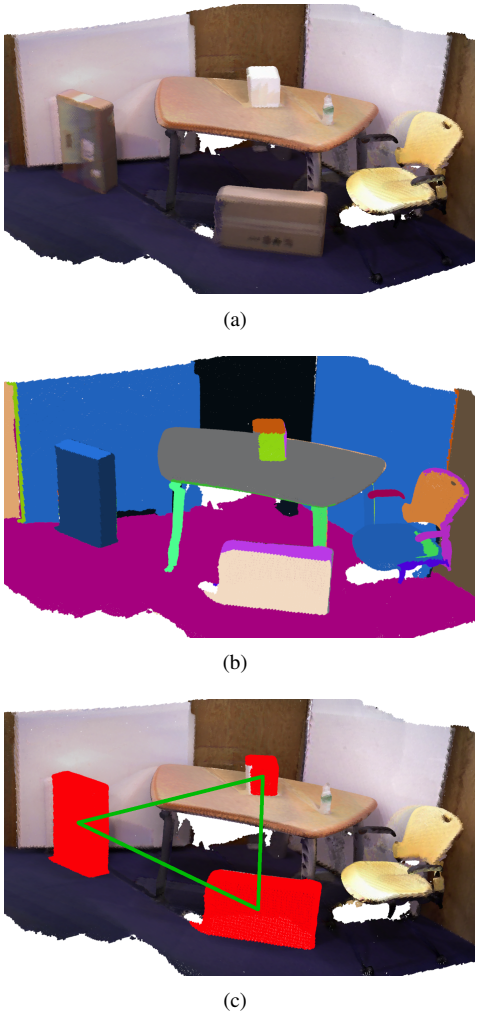
I. Introduction

Place recognition is a pre-requisite to building consistent maps. Most previous methods recognize places by visual appearance, or by looking for similar features in images. These feature matches are subsequently used to find the relative transformation between the images, up to a scale factor. Appearance-based methods have had great success at performing large-scale topological mapping [1], however they suffer some notable shortcomings. For example, they require a training phase to learn the image features, or visual words, in the environment and are commonly not robust to changes in illumination or other dynamics in the world.

In this work we propose to extend place recognition into the dense 3D world through the use of *physical words*. A *place* is now defined by a constellation of physical words that can be robustly extracted from the dense 3D models, as shown in Figure 1. This builds on our previous work [2], where we demonstrate the ability to generate a segmentation from a dense 3D point cloud in real-time. This follows previous results that demonstrate real-time dense mapping from an RGB-D sensor using a system called Kintinuous [3]. When loop closures are detected, the 3D point cloud is deformed to make a consistent map [4].

Our attempts to perform place recognition on the segmentation directly have failed, largely due to the high variability of the segments over multiple views of a scene. However, although the segmentation is variable, the underlying physical objects are robustly detectable by examining the relationships between the segments. We propose to use this invariant detection of physical objects, or words, from dense 3D maps to perform place recognition. Such an approach to place recognition has significant benefits:

- Lighting invariance
- Robustness to physical changes in any parts of the environment not containing the physical words used for matching
- Invariance to aspect or trajectory of observer
- Speed since the *physical dictionary* can be much more compact than a visual dictionary that usually contains



(a) (b) (c)

Fig. 1. A map of a simple office scene with three boxes. (a) The dense RGB-D map. (b) A segmented version of the same map with randomly colored segments. (c) The same map but with the boxes highlighted and connected into a constellation which is used to match against other maps.

[1]R. Finman, Liam Paull, and J. J. Leonard are with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA. {rfinman, lpaull, jleonard}@mit.edu

[2]T. Whelan is with the Department of Computer Science, National University of Ireland Maynooth, Co. Kildare, Ireland. thomas.j.whelan@nuim.ie

tens of thousands words. In this preliminary work we show the ability to recognize a scene using a single physical word.

Here we present a proof-of-concept method using physical words such as rectangular prisms. This is a stepping-stone towards the next step of matching more complex objects as combinations of physical words that are reliably detectable. Our results show that for small experiments we are able to find a correct association where current state-of-the-art appearance-based approaches would fail. It should be noted that the proposed method could work in conjunction with a standard appearance-based method for increased performance.

In Sec. II we review some related work and provide a more detailed review of our previous work on online segmentation. In Sec. III we describe the process of building the constellations from the 3D point cloud. In overview the following progression is followed: 3D point cloud $\rightarrow$ segmentation $\rightarrow$ physical letters $\rightarrow$ physical words $\rightarrow$ constellations. These constellations are used for the actual place recognition, as described in Sec. IV. We present some results for a small example in Sec. V and finally conclude and describe future directions for the work in Sec. VI.

II. Background and Related Work

This work is built on prior research in the area of place recognition. Our work differs from the majority of past work in the field by operating over the 3D point cloud rather than the space of images. However, since the point cloud data is so dense, we require a method of reducing the search space and exploiting the fundamental geometric properties of the underlying world. In this section we will describe related literature in place recognition, and review our previous work on online segmentation.

A. Appearance-Based Place Recognition

The ability to recognize a place is a fundamental capability that crucially enables the calculation of a transformation between the different views of the place.

The most common approach to associating views is through matching features in imagery. Without real-time requirements, this can be done with an exhaustive search for scene reconstruction from a collection of non-temporally sequential images [5]. This approach is generally infeasible for robotics applications since the search space scales quadratically with the number of images in the database. A common dimensionality reduction approach is to build a dictionary of *visual words* and an inverted index for quick lookup of images. Online place recognition is often built into an appearance-based, topological, or hybrid metric/topological SLAM system. One of the most popular is fast appearance-based mapping (FAB-MAP [6] and FAB-MAP 2.0 [1]) which uses a hierarchical information-based tree (Chow Liu tree) to speed up performance. This approach is purely topological (maintains no metric information) and defines a place as a location from which an image is recorded. An issue is perceptual aliasing, which arises when certain places have similar appearances, for example brick walls or trees. This is handled probabilistically in [7] using a particle filter approach to track multiple hypotheses. In hybrid approaches, such as [8], robust place recognition is used to define the origins of metric submaps.

Perhaps the most related work to our own is FABMAP 3D [9] which uses the metric spatial relationships between features in images to improve the loop closing performance over FABMAP in terms of the precision/recall curve. Our approach is similar but has a few notable differences. First, we are matching scenes using the entire constellation generated by spatial relationships of the physical words rather than the pairwise ranges between features. Secondly, and more importantly, our approach here extends the dictionary itself into the 3D space.

A common problem with bag-of-words type appearance matching is that it is highly dependent on the quality of the dictionary of visual words. Generating these dictionaries requires an onerous training phases. In our proposed approach the dictionary size can be greatly reduced and requires no offline training, however some care must be taken to properly define the physical words in the dictionary. In this initial work we use a single physical word: a rectangular prism.

In one other related work, place recognition using laser data has been demonstrated [10] and has the advantage that it is lighting and illumination invariant. However, it requires an expensive and large 3D laser-scanning sensor.

B. Graph-Based Segmentation

Our approach to dense 3D map segmentation is based on the widely used Felzenszwalb segmenter [11]. This method builds a graph connecting 3D points within a map and then segments the graph using dynamic thresholding. Specifically This segmenter builds a graph with all the 3D points in a map being nodes and edges between them. An edge is only connected if the points are within a threshold distance of each other, set to twice the volumetric resolution of the map. Of the set of edges E, a specific edge is

$$e_{ij} = ((d_i, d_j), w_{ij}) \tag{1}$$

where we define w_{ij} as

$$w_{ij}(n_i, n_j) = \begin{cases} (1 - n_i \cdot n_j)^2, & \text{if } (d_j - d_i) \cdot n_j > 0 \\ (1 - n_i \cdot n_j), & \text{otherwise} \end{cases} \tag{2}$$

for normals n_i and n_j of points d_i and d_j respectively. The edges are then compared to a dynamic threshold which decides whether to join the two points into a segment. For more detailed information, please refer to the paper. This algorithm provides the basis for our work so we can extract higher-level physical words from the noisy segments.

There are numerous alternatives to [11] for segmenting maps in a variety of data domains. Wolf *et al.* [12] use map segmentation of 3D terrain maps to assist with classification of traversable regions. Brunskill *et al.* [13] describe a 2D map segmentation method that builds a topological map of the environment by incrementally segmenting the world using

spectral clustering. These methods are more specific in their application and are not as computationally efficient as the Felzenszwalb segmenter.

In dense RGB-D sourced data, Karpathy *et al.* [14] uses segmentation of maps to perform object discovery by analyzing shape features of extracted segments. Izadi *et al.* [15] describe an impressive live segmentation within a dense volumetric reconstruction using geometric tracking. These methods are limited in the combined density and scale that they can map. In larger size maps, Finman *et al.* [16] detail a method for learning segmentations of objects in maps using object change cues to optimize the segmentation.

III. Segmentation-based Physical Words

With the goal of doing place recognition in maps, we start with a segmented map and combine the segments into physical words to create a constellation. Specifically, We build segment features and compare those features and the metric relationships between segments against models of *physical letters*, which are the building blocks of the physical words. From there we merge and refine the physical letters into physical words. Lastly, we build constellations between the words, which we can match against other constellations.

A. Segments to Physical Letters

We begin with a segmented map $S = \{s_1, s_2, ..., s_{|S|}\}$ from which we want to extract physical letters P'.

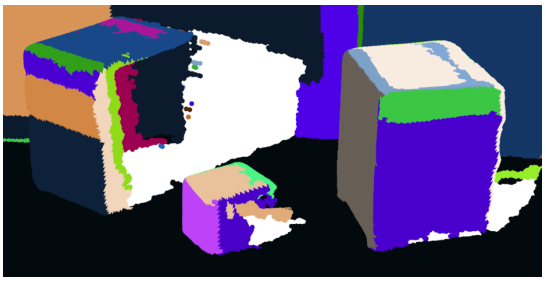
A naive approach to place recognition on these dense maps would be to match directly on the segments. However, if there are significant changes in camera orientations when building a map, then the different sources of noise can cause the segmentation algorithms to inconsistently segment the same scene. An example of such inconsistencies is shown Fig. 2. As an alternative, we look for the hidden variables in the world, physical words, their physical letters, and the segments that suggest them.

To find the physical letter, we first build a feature vector of every segment to compare against a model. The feature vector is populated by first running principle component analysis on the points d_{s_i} of each segment $s_i \in S$. The length, width, and height of the segment are calculated by taking the magnitudes of the vectors connecting the two extremum points are projected onto their respective axes. The neighbors of a segment are defined in two ways, shown in (3). First, the segments that share an edge are included. As the segmentation algorithm can be messy along the edges of large segments, direct segment adjacency is insufficient. Therefore, segments that are within a distance τ between

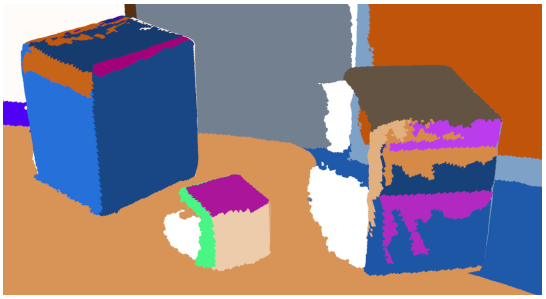
TABLE I

Segment Features

Notation	Description
$L,\ W,\ H$	First three principle axis unit vectors
$\lambda_L,\ \lambda_W,\ \lambda_H$	First three eigenvalues
$l,\ w,\ h$	Max distance along first principle axis
c	Mean XYZ location of all the points
S^N	Segment neighbors



(a)



(b)

Fig. 2. (a) A segmentation of two file cabinets and a box with the camera trajectory from left to right. (b) A segmentation of the same scene as (a), but with the camera trajectory being from right to left. Note the large variance in segment shapes and sizes for the same objects. Surfaces that are viewed perpendicularly by the RGB-D camera are stable, while other surfaces are not. Colors in each segmentation are randomly assigned and do not correspond to segments between maps.

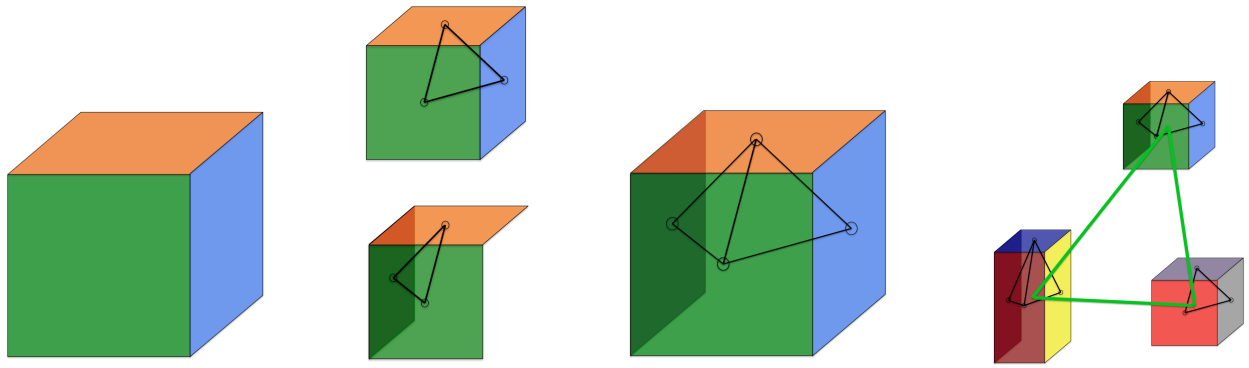
segment centers are also added to the neighbor set. In this work τ was empirically chosen to be 0.5 meters. The complete feature vector is shown in Table I.

$$S^N_{s_i} \leftarrow \{s_j \in S \mid \{\exists e_{ij} \in E\}, \{s_i \neq s_j\}\} \cup \{s_j \in S \mid \|c_{s_i} - c_{s_j}\| < \tau\} \quad (3)$$

From these segments and corresponding feature vectors, we want to connect segments that make up our final physical word model. Using the neighbors of the segments as edges, we construct an unweighted graph of all segments. Within this graph of locally connected segments we can begin to find our physical words. In the scope of this paper, we restrict the words to only rectangular prisms. To find these shapes, we look for segments that form a corner - a physical letter of the rectangular prism physical word. Intuitively, a corner is formed by a clique of three perpendicular, planar segments that are spatially connected and convex. Cliques are found by iterating over a segment's neighbors and its neighbor's neighbors and storing any clique that has three and only three segments. This is shown below in (4).

$$Q \leftarrow \{\{s_i, s_j, s_k\} \mid s_i \in \{S^N_{s_j} \cap S^N_{s_k}\}, s_j \in \{S^N_{s_i} \cap S^N_{s_k}\}, s_k \in \{S^N_{s_i} \cap S^N_{s_j}\}\} \quad (4)$$

With the three-connected cliques, we can filter Q for only planar segments. We define a segment to be planar if the



(a) Segments (b) Physical Letters (c) Physical Words (d) Constellation of Words

Fig. 3. Overview of the process from segments to constellations

second eigenvalue of the point cloud is greater than a factor of the third eigenvector. That is, $\lambda_{W_{s_j}} >> \lambda_{H_{s_j}}$. This enforces that the length and width of a segment are significantly larger than the depth of the segment.

With the three-connected cliques of planar segments, Q^p, we filter based on the perpendicularity of the segments. With the planar segments, $\lambda_{H_{s_i}}$ is the approximate surface normal of an entire segment s_i. We compare the volume of the parallelepiped formed by the unit normal vectors of all three segments against the unit cube. Intuitively, if all segments are perfectly perpendicular, then the normals will make a perfect unit cube.

$$Q^c \leftarrow \{q \in Q^p \mid ((\lambda_{H_{s_i}} \times \lambda_{H_{s_j}}) \cdot \lambda_{H_{s_k}}) > \gamma\} \quad (5)$$
$$s_i, s_j, s_k \in q \;\&\; s_i \neq s_j \neq s_k$$

with γ set to 0.95.

There is no guarantee that the segments in Q^c make a fully convex corner. We want to find the corner made by three sides of a box and not two sides of a box and the table the box is on. Our last step in making physical letters is to find the convex corners. We expand the simple convexity measure from Finman *et al.* [16] from points to segments.

$$Q' \leftarrow \{q \in Q^c \mid (c_{s_j} - c_{s_i}) \cdot \lambda_{H_{s_j}} > 0, \forall s_i, s_j \in q\} \quad (6)$$

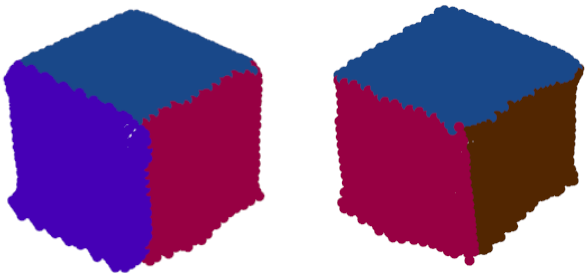
where Q' containing all segments that form corners of rectangular prisms.

B. Physical Letters to Words

The set of physical letters P', may have several parts of a larger word P. For example, one can observe two corners

TABLE II
PHYSICAL WORD FEATURES FOR RECTANGULAR PRISM

Notation	Description
$l^p,\ w^p,\ h^p$	Lengths of all sides
$\lambda_L^p,\ \lambda_W^p, \lambda_H^p$	Unit vectors of word coordinates
c^p	Mean XYZ location
S^p	Set of segment members



(a) (b)

Fig. 4. Two *physical letters* used to build a *physical word.* (a) A set of three segments detected as a physical letter of a rectangular prism. (b) A different physical letter of the same rectangular prism. Note the two overlapping segments in red and blue showing that multiple segments can be part of the same physical letter.

on a rectangular box, but they all represent parts of the same box (Figure 3(b) and 4). We combine these physical letters into one physical word as in Figure 3(c). The detailed feature description of the rectangular prism word is given in in Table II. We begin the word building process by finding the center of the cube. Simple computing the mean of all the points or the segment centers would bias the calculated center towards the detected corner and is thus incorrect. Instead we calculate the point intersection, x, between all three planes. This is guaranteed to be a point if no planes are coplanar, parallel, or share the same intersecting line between them. This is a linear set of equations that can be solved as follows.

$$A \cdot x = b \quad (7)$$
$$x = A^{-1} \cdot b \quad (8)$$
$$A \leftarrow [\lambda_{H_{s_i}}, \lambda_{H_{s_j}}, \lambda_{H_{s_k}}]^T$$
$$b \leftarrow [\lambda_{H_{s_i}} \cdot c_{s_i}, \lambda_{H_{s_j}} \cdot c_{s_j}, \lambda_{H_{s_k}} \cdot c_{s_k}]^T$$
$$s_i, s_j, s_k \in q, q \in Q'$$

From the corner point x, the center can be simply found by subtracting half the lengths of each segment in the opposite

direction of the normals.

$$c_q^{p'} \leftarrow \left[x - \left| \begin{matrix} \frac{l_{s_i}}{2} \cdot (-\lambda_{H_{s_i}}) \\ \frac{l_{s_j}}{2} \cdot (-\lambda_{H_{s_j}}) \\ \frac{l_{s_k}}{2} \cdot (-\lambda_{H_{s_k}}) \end{matrix} \right| \right] \quad (9)$$
$$s_i, s_j, s_k \in q, q \in Q'$$

The dimensions of the words are the lengths of the segments that make up the physical word. For example, if there are three segments of length 1, 2, and 3 that make up a word. Then the length of the word is 3, the width is 2, and the height 1. The set of segment members for the physical letter and the corresponding dimensions are

$$l^{P'} \leftarrow max(l_{s_i}, l_{s_j}, l_{s_k}) \quad (10)$$
$$w^{P'} \leftarrow median(l_{s_i}, l_{s_j}, l_{s_k}) \quad (11)$$
$$h^{P'} \leftarrow min(l_{s_i}, l_{s_j}, l_{s_k}) \quad (12)$$
$$S^{P'} \leftarrow q \quad (13)$$
$$s_i, s_j, s_k \in q; q \in Q'; s_i \neq s_j \neq s_k$$

1) Merging Physical Letters: With the physical letter features defined, we merge the physical letters by finding the overlap between them. An overlap occurs when the vector between the centers is fully contained within the other letter. Formally, this is

$$v_{ij} \leftarrow c_i^{P'} - c_j^{P'} \quad (14)$$
$$Overlap(P_i', P_j') \leftarrow \begin{cases} True, \text{if } v_{ij} - \left| \begin{matrix} l_i^{P'}/2 \\ w_i^{P'}/2 \\ h_i^{P'}/2 \end{matrix} \right| < \left| \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \right| \\ False, \text{otherwise.} \end{cases} \quad (15)$$

If there is an overlap, then the larger word is chosen to be the root word and the segment members set is updated with any new segments from the new physical letters. If there is a single physical letter and no other physical letters to merge with, then that is a final physical word.

C. Word Constellations

Using the physical words P built up in the previous sections, we make another graph of the full physical words. Examples of this can be seen in Figures 1(c) and 3(d). We make a constellation $\mathcal{C}$ between all metrically local word centers c^P where each edge corresponds to the distance $\mathcal{D}$ between each center c^P of the word. We only look at local constellations for two reasons. First, any mapping system has drift associated with it and drift increases with the size of the map so for future matching, using metrically local words is beneficial. Second, the environment in which we are mapping is on the scale of an office building and finding a place based on observations dozens or hundreds of meters apart is less descriptive. We use the term local to say that no edge between two words is longer than a threshold, empirically chosen to be 2 meters

IV. Constellation Matching

In Section III, we detailed how to find physical words P in a map and build them into a constellation. In this section we will use the constellation $\mathcal{C}$ to match against other map constellations.

Specifically, we compare two constellations of the same size $\mathcal{C}$ and $\mathcal{C}'$. For each constellation, we define a set $\mathcal{C}^e$ of all the distances between nodes within the constellation.

$$\mathcal{C}^e = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_{|\mathcal{C}^e|}\} \quad (16)$$

where $\mathcal{D}_i$ is the distance between two word centers in the constellation. Three non-colinear points can define a transformation between two surfaces with only an ambiguity of direction.

We begin by defining the association between edges in the two constellations being compared:

$$M : \mathcal{C}^e \mapsto \mathcal{C}'^e \quad (17)$$

To find the best match between the constellations, we exhaustively search for the association between nodes that has the minimum Mahalanobis distance between the associated edges. If we model distances in the constellations as Gaussian and uncorrelated with constant covariance [1] Σ:

$$\mathcal{D}_i \sim \mathcal{N}(\mu_{\mathcal{D}_i}, \Sigma), \quad (18)$$

then the difference between the edge values is also a Gaussian distribution. For example if edge $\mathcal{D}_i$ is associated with edge $\mathcal{D}_j^{'}$ ($M(\mathcal{D}_i) = \mathcal{D}_j^{'}$) then the difference is:

$$\mathcal{D}_i - \mathcal{D}_j^{'} \sim \mathcal{N}(\mu_{\mathcal{D}_i} - \mu_{\mathcal{D}_j^{'}}, 2\Sigma). \quad (19)$$

The best match M^* can then be found as:

$$\begin{aligned} M^* &= \underset{M}{\text{argmax}} \prod_{i=1}^{|\mathcal{C}^e|} \exp -\frac{1}{2} ||\mu_{\mathcal{D}_i} - \mu_{M(\mathcal{D}_i)}||_{2\Sigma} \\ &= \underset{M}{\text{argmin}} \sum_{i=1}^{|\mathcal{C}^e|} ||\mu_{\mathcal{D}_i} - \mu_{M(\mathcal{D}_i)}||_{2\Sigma} \end{aligned} \quad (20)$$

using the property that $\text{argmax}\{\cdot\} = \text{argmin}\{-\log\{\cdot\}\}$, and where $||e||_\Sigma$ is the Mahalanobis distance.

We accept a match between two constellations if the sum of the Mahalanobis distances for all edge associations is below a threshold:

$$Match(\mathcal{C}^e, \mathcal{C}'^e) = \begin{cases} 1, \text{if } \sum_{i=1}^{|\mathcal{C}^e|} ||\mathcal{D}_i - M^*(\mathcal{D}_i)||_{2\Sigma} > \tau'(|\mathcal{C}|) \\ 0, \text{otherwise} \end{cases} \quad (21)$$

where $\tau'(x) = t^x$ is the threshold for accepting the match which is a function of the size of the constellation.

[1] A more rigorous treatment of this covariance will be the subject of future work

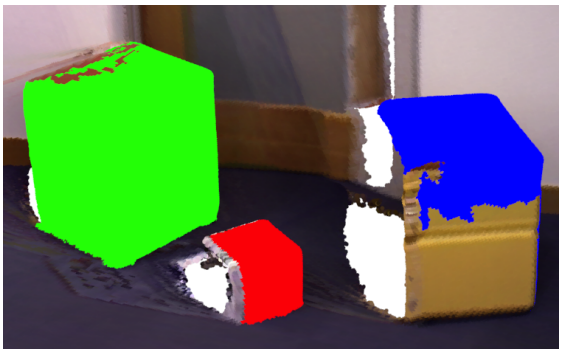


Fig. 6. The same map in Fig. 2(b) with the detected cubes showing. Our method does not have full coverage of the object, but the bounding cube's dimensions are qualitatively similar to the true object.

V. Results

We evaluate this method in two ways on real world datasets: qualitatively and timing. We compare our approach against the popular DBoW method in conditions with varying lighting and camera trajectories when creating the map.

We present a number of datasets collected with a handheld RGB-D camera of varying length. In total, 12 datasets were recorded, the statistics of which can be seen in Table III. The datasets were captured with a map volumetric resolution between 8.7 and 11.7 mm. We made a point of collecting varying datasets. Figures 5(b), (c), (d) show three different maps of the same scene, with variations in lighting, chairs, while the camera perspective being the exact opposite in the two dark datasets. Figures 5(f), (g), (h) show the three maps segmented, from which the boxes are detected. Finally, Figures 5(j), (k), (l) show the three matching boxes highlighted according to their match across maps. Running DBoW on these three datasets of the same scene did not lead to any matching frames since DBoW is sensitive to variations detailed in the datasets. Figures 5(a), 5(e), and 5(i) show a similar progression but in a larger and more cluttered environment. The box detections there are arbitrarily colored and do not correspond to the other figures. Lastly, Figure 6 shows how some rectangular prisms are not fully found due to the noisy segmentation shown in Figure 2(b).

TABLE III
Dataset Statistics and Timing

	Datasets		
	Small	Medium	Large
Number of Datatsets	7	4	1
Avg Vertices	213,892	773,611	2,155,284
Avg Map Build Time (s)	15.08	36.70	92.57
Avg Map Length (m)	3.50	7.37	26.21
Function	Avg Timing (s)		
Segmentation	0.22	0.79	1.83
Feature creation	0.47	1.32	2.70
Letter creation	0.08	0.10	0.15
Word creation	¡0.01	¡0.01	0.01
Constellation creation	¡0.01	¡0.01	¡0.01
Total timing	0.80	2.51	4.88

VI. Discussion and Future Work

In this week we presented a novel method for place recognition using physical words to form constellations. We demonstrated how we segment a map, combine those segments into physical letters and words, and finally build a constellation of words. These constellations were then used to match places in other maps. We tested our algorithm on multiple real-world datasets under extreme lighting and camera viewing angles.

In future work, we will extend this concept by creating words from the data rather than predefining them. Given noisy measures of some latent physical word, we will infer what words there are in the world. Once those are defined, we can break the words into physical letters as we did in this work. With more physical words, we will add these words into our constellation matching method. Furthermore, we will tie the method described in this paper into our incremental segmentation algorithm [2] that can efficiently segment a map as it is being built in real-time. This capability provides the groundwork for performing scene matching based on these incrementally generated segments in real time over large graphs.

VII. Acknowledgements

This work was partially supported by ONR grants N00014-10-1-0936, N00014-11-1-0688, and N00014-12-10020, NSF grant IIS-1318392, and by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the Irish National Development Plan, the Embark Initiative of the Irish Research Council.

References

[1] M. Cummins and P. Newman, "Appearance-only SLAM at large scale with FAB-MAP 2.0," *The International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2011.

[2] R. Finman, T. Whelan, M. Kaess, and J. Leonard, "Efficient incremental map segmentation in dense rgb-d maps," in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, 2014.

[3] T. Whelan, J. McDonald, M. Kaess, M. Fallon, H. Johannsson, and J. Leonard, "Kintinuous: Spatially Extended KinectFusion," in *3rd RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, (Sydney, Australia), July 2012.

[4] T. Whelan, M. Kaess, J. Leonard, and J. McDonald, "Deformation-based loop closure for large scale dense RGB-D SLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS*, (Tokyo, Japan), November 2013.

[5] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski, "Building rome in a day," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 72–79, Sept 2009.

[6] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[7] A. Ranganathan and F. Dellaert, "Online probabilistic topological mapping," *The International Journal of Robotics Research*, vol. 30, no. 6, pp. 755–771, 2011.

[8] S. Tully, G. Kantor, and H. Choset, "A unified bayesian framework for global localization and slam in hybrid metric/topological maps," *The International Journal of Robotics Research*, vol. 31, no. 3, pp. 271–288, 2012.

[9] R. Paul and P. Newman, "FAB-MAP 3D: Topological mapping with spatial and visual appearance: Topological mapping with spatial and visual appearance," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2649–2656, IEEE, 2010.

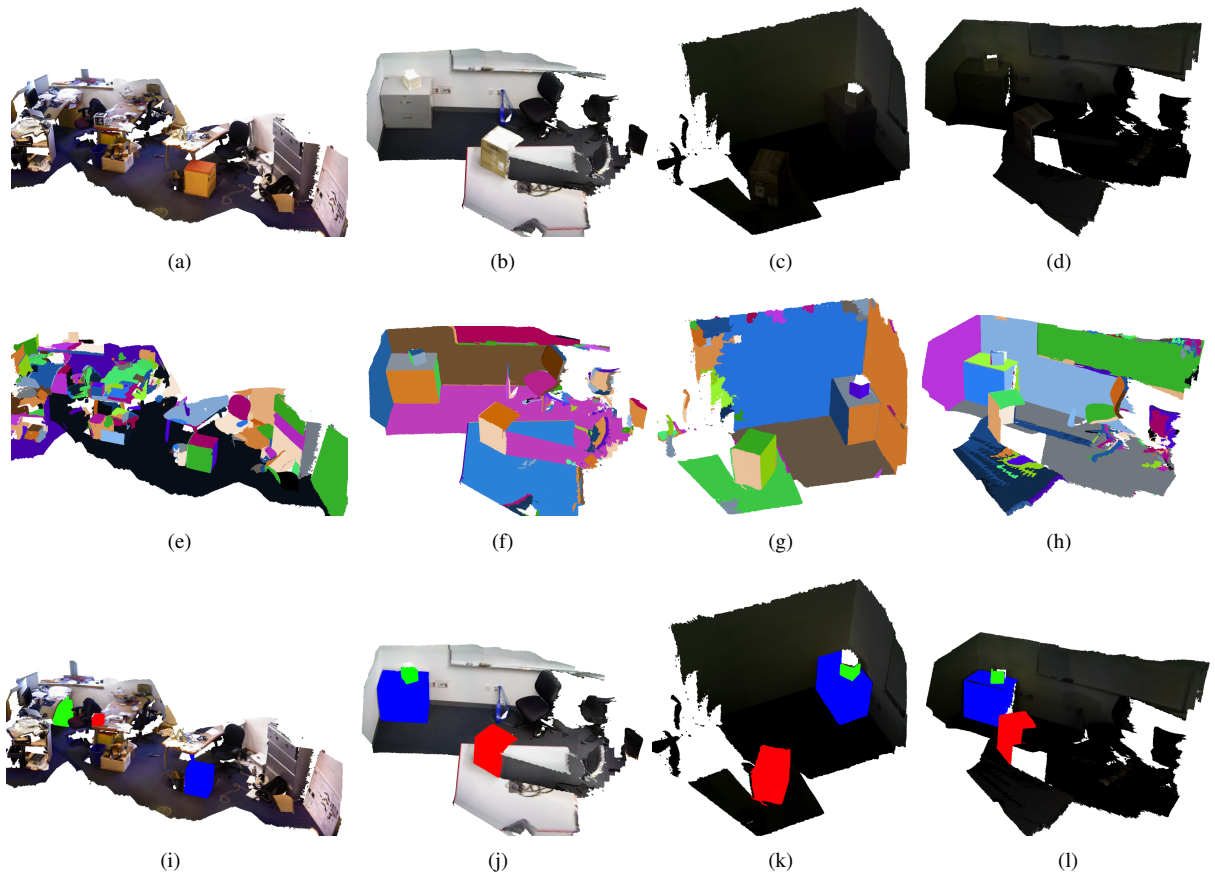


Fig. 5. (a) A real-world dataset of a cluttered office scene. (b) (c) (d) Multiple maps of the same scene from vastly different camera trajectories and lighting conditions. (e) The segmented map of the cluttered office scene. (f) (g) (h) The corresponding segmentations of the same maps in (b). (i) All boxes found in the scene highlighted in arbitrary colors. (j) (k) (l) All highlighted boxes found in the scene highlighted according to there corresponding segments in the other maps. The color assignment in (j) (k) (l) shows matching between maps.

[10] C. McManus, P. Furgale, B. Stenning, and T. D. Barfoot, "Lighting-invariant visual teach and repeat using appearance-based lidar," *Journal of Field Robotics*, vol. 30, no. 2, pp. 254–287, 2013.

[11] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[12] D. F. Wolf, G. S. Sukhatme, D. Fox, and W. Burgard, "Autonomous terrain mapping and classification using hidden markov models," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 2026–2031, IEEE, 2005.

[13] E. Brunskill, T. Kollar, and N. Roy, "Topological mapping using spectral clustering and classification," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pp. 3491–3496, IEEE, 2007.

[14] A. Karpathy, S. Miller, and L. Fei-Fei, "Object discovery in 3D scenes via shape analysis," in *International Conference on Robotics and Automation (ICRA)*, 2013.

[15] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "KinectFusion: Real-Time 3D reconstruction and interaction using a moving depth camera," in *Proc. of the 24th annual ACM symposium on User interface software and technology*, UIST '11, (New York, NY, USA), pp. 559–568, ACM, 2011.

[16] R. Finman, T. Whelan, M. Kaess, and J. Leonard, "Toward lifelong object segmentation from change detection in dense RGB-D maps," in *European Conference on Mobile Robots (ECMR)*, (Barcelona, Spain), Sep 2013.