2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Rapid Autonomous Semantic Mapping

Anup Parikh, Mark W. Koch, Timothy J. Blada, Stephen P. Buerger[1]

Abstract— **A semantic understanding of the environment is needed to enable high level autonomy in robotic systems. Recent results have demonstrated rapid progress in underlying technology areas, but few results have been reported on end-to-end systems that enable effective autonomous perception in complex environments. In this paper, we describe an approach for rapidly and autonomously mapping unknown environments with integrated semantic and geometric information. We use surfel-based RGB-D SLAM techniques, with incremental object segmentation and classification methods to update the map in realtime. Information theoretic and heuristic measures are used to quickly plan sensor motion and drive down map uncertainty. Preliminary experimental results in simple and cluttered environments are reported.**

I. Introduction

Many anticipated applications, including disaster response, hazardous materials mitigation, and search-and-rescue, may require unmanned systems (UMS) to operate in unknown interior environments to keep human operators out of harm's way. Such environments present significant challenges, due to limited GPS, communications for remote control, and a lack of complete *a priori* maps. Even basic operations in these conditions requires a high level of autonomy, with more realistic scenarios requiring the ability to rapidly comprehend the environment and react to changes within it. While technology to generate real-time 3D models of environments by mapping occupancy (simultaneous localization and mapping or SLAM) is relatively mature, to realize the full potential of autonomous systems, it is also necessary to classify objects. Even after obtaining a complete (geometric) SLAM map of a space, a robot would be unable to perform simple object-centric tasks such as "swab the gas cylinders" or "check the seating areas for people." The generation of 3D models in which individual objects are partitioned and annotated with abstract representations is a vital need for advancing autonomous operations beyond navigation, and is much more challenging than SLAM. In this paper, we describe our approach for using multi-sensor fusion / classification algorithms with intentional variation of sensor perspective to demonstrate a capability to rapidly move through a space and produce abstract representations of the objects within it.

Most prior work in "semantic labeling" has focused on "personal robotics" applications in home, office, retail or medical environments. Demonstrated techniques include machine learning [1]–[4], hierarchical feature-based approaches [5], and combinations of the two [6]. Many approaches rely on graph methods [7, 8] to iteratively converge on a semantic solution. Results reported to date are often too slow for rapid actions, or operate on reduced resolution imagery, and therefore cannot detect small or far field objects. Our method overcomes these computation and resolution challenges by combining object segmentation algorithms and image based classifiers that are amenable to SIMD calculation on a GPU.

Image based classifiers are often used in the loop to provide semantic evidence. Recent machine learning approaches for image-based detection and classification have shown impressive results [9, 10]. One useful axis of differentiation between these approaches is in the precision with which they localize objects. While image classifiers label an entire image, segmentation algorithms produce a pixel-wise mask of labeled regions in an image, at the expense of additional computation.With most research focusing on building larger and more complex model architectures to maximize detection performance, these approaches are often not applicable for fast-paced autonomy since they are often too slow at inference. However, recent image-based object detectors have demonstrated real-time detection results with near state-of-the-art accuracy and precision [11, 12]. Although not as precise as segmentation models, the detection boxes can be combined with geometric segmentation to produce precise semantic labels in the SLAM map.

Most of the state-of-the-art detection methods heavily rely on large datasets to train models. However, in some applications, the necessary datasets don't exist, or are very expensive to generate. Low shot learning methods are one approach to circumvent this problem [13]. These methods learn general purpose features that can be used to find a new object given a few examples or templates. In the extreme case (zero shot learning), the examples/templates are provided with a different phenomenology (e.g., a verbal description of an object is used to detect the object in an image) [14]. In our approach, we use both deep object detectors, to find common objects, and zero-shot methods, to detect specific objects of interest for which training data is unavailable.

To autonomously map an unknown environment at a fast pace, a system must intelligently decide where next to move to gain the most information and reduce map uncertainty; exhaustive search, such as lawn mower patterns, are too slow.

*This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories. Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

[1]All authors are with the Robotics and Counter Robotics Research and Development group at Sandia National Laboratories, Albuquerque, NM 87185 USA. Email: {aparikh, mwkoch, tjblada, sbuerge}@sandia.gov

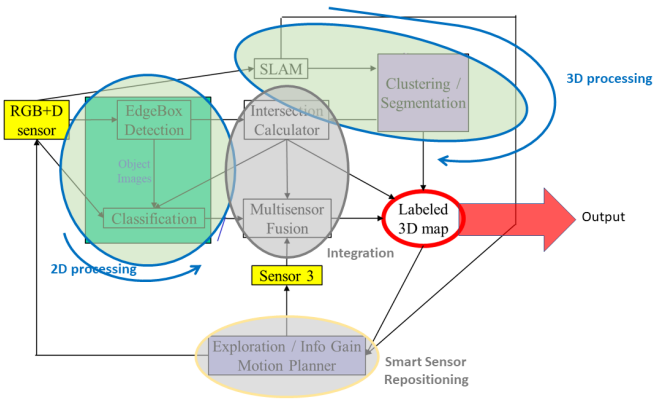

Fig. 1. Overview of our technical approach.

Numerous methods have been developed for active SLAM [15]–[20], typically using information theoretic measures to estimate the reduction in uncertainty in the map and pose estimate of the robot. These are almost exclusively applied to 2D environments, since they rely on occupancy grid probabilistic map representations to calculate information gain, which have large memory and compute requirements when extended to large 3D spaces. Next best view methods [21]–[23] often take a similar approach, but overcome the curse of dimensionality by focusing on a single object. Conversely, active object recognition systems usually focus on differences in viewpoint and appearance to gauge the most informative new views [19, 24, 25]. In our approach, we combine both concepts to determine new viewpoints that help complete the map, as well as drive down uncertainty in the semantic labels within the map.

In this paper, we develop an approach for rapid, autonomous semantic mapping, and report preliminary experimental results in implementing this approach for mapping simple and cluttered environments. The primary contribution of this work is a novel algorithmic framework, as outlined in Figure 1, that enables geometric mapping, segmentation, classification, and exploration to be efficiently inter-operated in real-time.

II. Mapping

The mapping component forms the foundation for accumulating, fusing, and representing the geometric and semantic information collected from the sensors. Our SLAM approach builds on the point-based fusion technique [26], with additional modifications to incorporate object classes and view metrics. For completeness, the mapping approach is briefly reviewed here.

A. Preliminaries

Our approach relies on an RGB-D camera to perform SLAM. Let $\mathcal{C} : \Omega \rightarrow \mathbb{N}^3$ represent the color image and $\mathcal{D} : \Omega \rightarrow \mathbb{R}$ the depth image of an RGB-D frame, where $\Omega \subset \mathbb{N}^2$ is the image space domain (i.e., the set of valid pixels in an image). Although the perspective points and imaging sensors of the RGB and depth imagers are not collocated, the sensor hardware typically provides registered depth and color images, where the depth of a particular pixel in the color image is provided by the data at the same pixel coordinates in the depth image. Using a pinhole camera model, a 3D point, $\mathbf{p} = \begin{bmatrix} x & y & z \end{bmatrix}^T \in \mathbb{R}^3$ projects onto the imaging sensor at 2D pixel coordinates $\mathbf{u} = \begin{bmatrix} u & v \end{bmatrix}^T \in \Omega$ via the perspective projection function (including dehomogenisation) $\mathbf{u} = \pi(\mathbf{p})$, using the intrinsic camera calibration matrix, $K \in \mathbb{R}^{3\times3}$ [27]. Conversely, the 3D position of a pixel can be determined using the depth image via back-projection, $\mathbf{p} = \mathcal{D}(\mathbf{u}) K^{-1} \mathbf{u}^H$, where $(\cdot)^H$ notation is used to denote homogeneous representations $\mathbf{u}^H = \begin{bmatrix} \mathbf{u}^T & 1 \end{bmatrix}^T$. For an RGB-D frame at time t, consisting of a color and depth image, $\mathcal{C}_t$ and $\mathcal{D}_t$, the vertex map, $\mathcal{V}_t : \Omega \rightarrow \mathbb{R}^3$, and normal map, $\mathcal{N}_t : \Omega \rightarrow \mathbb{R}^3$, are images of the pixel vertex location and local unit normal vectors, respectively. The vertex map is calculated using back-projection: $\mathcal{V}_t(\mathbf{u}) = \mathcal{D}_t(\mathbf{u}) K^{-1} \mathbf{u}^H$. The normal map is calculated from neighboring pixels in the vertex map as $\mathcal{N}_t(\mathbf{u}) = (\mathcal{V}_t(u+1, v) - \mathcal{V}_t(u-1, v)) \times (\mathcal{V}_t(u, v+1) - \mathcal{V}_t(u, v-1))$, with border pixels marked as invalid. A segment map, $\mathcal{L}_t : \Omega \rightarrow \mathbb{N}$, and propagated segment map, $\mathcal{L}_t^p : \Omega \rightarrow \mathbb{N}$, assign a segment number to each pixel in the image, using the method described in Section III.

The global semantic and geometric map, $\mathcal{M}$, consists of a list of surfels (surface elements), $\mathbf{s}$, each made up of the following attributes: instantiation time, $t \in \mathbb{R}$, global position, $\mathbf{p} \in \mathbb{R}^3$, global normal direction $\mathbf{n} \in \mathbf{R}^3$, stability flag $s \in \mathbb{B}$, global basis direction $\mathbf{b} \in \mathbb{R}^3$, color $\mathbf{c} \in \mathbb{N}^3$, radius $r \in \mathbb{R}$, confidence weight $w \in \mathbb{R}$, segment $l \in \mathbf{N}$, segment weight $w_l \in \mathbb{R}$, independent (multilabel) classification probabilities $\mathbf{p}_c \in \mathbb{R}^{n_c}$, and information bins $\mathbf{b}_v \in \mathbb{R}^{n_v}$. The pose of the sensor is represented by its translation vector, $\mathbf{t} \in \mathbb{R}^3$, and rotation matrix, $R \in \mathrm{SO}(3)$. Given a global map, $\mathcal{M}$, and a camera pose, $T \in \mathrm{SE}(3)$, a prediction of the color $\mathcal{C}_t^r$, depth $\mathcal{D}_t^r$, vertex $\mathcal{V}_t^r$, normal $\mathcal{N}_t^r$ and segment $\mathcal{L}_t^r$ maps can be rendered using simple surfel splatting (forward projection and depth culling). This method is also used to render map metadata, such as the index map, $I_t^r : \Omega \rightarrow \mathbb{N}$, containing the index of the surfel in $\mathcal{M}$ that projects onto each pixel.

B. Map Initialization

The global map is initialized from the data in the first RGB-D frame. A new surfel is added to $\mathcal{M}$ for each valid pixel in $\mathcal{V}_t$ and $\mathcal{N}_t$. For each new surfel, the position is sampled from $\mathcal{V}_t$, the normal from $\mathcal{N}_t$, the color from $\mathcal{C}_t$ and the segment number from $\mathcal{L}_t$. A confidence value is calculated based on empirical sensor performance characteristics [28, 29] as $w = \exp\left(-\gamma^2 / \left(2\sigma^2\right)\right)$, where $\sigma = 0.6$ and γ is the normalized radial distance from the image center to the pixel. This confidence value is used to initialize the confidence weight, w, and segment weight, w_l. The surfel radius is initialized by back-projection of the circle circumscribing the pixel, $r = \mathcal{V}_t(\mathbf{u})_z / \left(\sqrt{2} f \mathcal{N}_t(\mathbf{u})_z\right)$, where f is the camera focal length in pixels. The class probabilities, $\mathbf{p}_c$, are all initialized to 0.5, and the information bins are initialized using the method described in Section V-B.

C. Localization

Localization of the sensor uses an iterative closest point (ICP) algorithm with projective data association [30]. ICP is performed between the current frame and a rendering of the global map from the camera pose at the previous time step, $\mathcal{V}^r_{t-1}$ and $\mathcal{N}^r_{t-1}$, to reduce the drift rate caused by random errors. Only stable points (those with high confidence weights) in the map are rendered, reducing noise in the pose estimate.

Given the associated points in the current and rendered frames, the point-to-plane error for each valid pixel is used to calculate the ICP cost, which is minimized with respect to the relative transformation between the last and current frame, $T_{t-1,t}$. To improve the basin of convergence, ICP is first performed with quarter-resolution versions of the vertex and normal maps, producing a coarser, but more robust, estimate of the relative pose. This coarse estimate is used as an initial guess for ICP with half resolution frames, producing an improved estimate which in turn is used to initialize ICP with the full resolution frames.

D. Map fusion

Every new RGB-D frame provides an opportunity to expand the coverage of the SLAM map and update the data of the points already in the map. The first fusion step is associating points in the map with points in the new RGB-D frame, again using projective data association. Given the pose of the camera determined via the localization procedure, the map can be rendered onto the current camera frame. For map fusion, the map is rendered as point samples rather than surfel splats and rendered at 4x horizontal and vertical resolution. This enables checking against multiple correspondence candidates for the most similar point (based on radius, distance and surface normal) to be updated. This also ensures that incorrectly occluding points in the map (e.g., from moving objects previously added to the map) can be detected and cleared. If no points in the map match with a point in the current frame, a new point is added and initialized as described above. If a match is found, the position, normal, color and radius are updated using a weighted average with the current and new confidence weight. The new confidence weight is also added to the current weight and stored in the surfel.

Surfel segment labels are updated using $\mathcal{L}^p_t$, given by the procedure described in Section III. If the segment number in $\mathcal{L}^p_t$ matches that in the corresponding surfel, the segment weight is also accumulated, else it's subtracted from the stored value until 0, at which point the surfel segment number is changed to the new segment number. Once the confidence weight reaches a threshold, it is marked as a stable point in the map. Surfel information bins and basis vectors are also updated as described in Section V-B.

After fusing new data into the map, incorrect surfels are pruned from the map. Old surfels (as determined by a time threshold) that have not been flagged as stable are removed from the map. Points in the map that project onto valid pixels in the current frame, but are closer to the camera, are also removed (akin to space carving).

III. Segmentation

The SLAM system described in the previous section only provides geometric and color information about the scene, which is not sufficient for higher-level autonomous reasoning. One approach for imbuing semantic information in the map is to directly project object detection bounding boxes from image-based classifiers onto the map. However, this approach falsely classifies foreground and background objects. Instead, we use a geometric approach to cluster surfels into segments, and then fuse segments with the classification output, as described in Section IV. This also enables us to capture some higher-level information for a wider set of objects (i.e., those not considered in the classifier).

We base our segmentation approach on the method described in [31]. Using the vertex map $\mathcal{V}_t$ and normal map $\mathcal{N}_t$, an edge map, $\mathcal{E}_t$, is constructed indicating borders between adjacent objects. Edges are determined using two features: depth discontinuities and concavity (under the assumption that most objects are convex). Specifically, $\Gamma(\mathbf{u}) = \max_{N(\mathbf{u})} \{|(\mathcal{V}_t(\mathbf{u}_i) - \mathcal{V}_t(\mathbf{u})) \cdot \mathcal{N}_t(\mathbf{u})|\}$ represents the maximum local point-to-plane distance and $\Phi(\mathbf{u}) = \min_{N(\mathbf{u})} \{\Phi_i(\mathbf{u})\}$ represents a measure of concavity, where $N(\mathbf{u})$ is the neighborhood of pixels around $\mathbf{u}$ and

$$\Phi_i(\mathbf{u}) = \begin{cases} 1 & (\mathcal{V}_t(\mathbf{u}_i) - \mathcal{V}_t(\mathbf{u})) \cdot \mathcal{N}_t(\mathbf{u}) > 0 \\ \mathcal{N}_t(\mathbf{u}_i) \cdot \mathcal{N}_t(\mathbf{u}) & \text{otherwise.} \end{cases}$$

Then, $\mathcal{E}_t(\mathbf{u}) = \Phi(\mathbf{u}) < \delta_\Phi \,||\, \Gamma(\mathbf{u}) > \delta_\Gamma$.

In comparison to [31], we found the edge maps were often dominated by noise, especially when observing dark and textured materials. To compensate, we apply additional bilateral filtering to the input normal map. Furthermore, the pixel neighborhood, $N(\mathbf{u})$, is extended to beyond the surrounding 8 pixels, to include the first valid pixel in each of the cardinal directions. Finally, morphological opening and closing operations are applied to the edge map to reduce speckle and to thin edges.

After constructing the edge map, a connected component analysis is applied to cluster connected segments, yielding the label map $\mathcal{L}_t$. Since the label numbers for a segment may change across consecutive frames, a correspondence with existing labels in the global map is constructed in $\Pi \in \mathbb{R}^{n_{rl} \times n_l}$, where n_{rl} is the number of labels in the rendered label map, $\mathcal{L}^r_t$, and n_l is the number of labels in the current label map $\mathcal{L}_t$. $\Pi(\mathcal{L}^r_t(\mathbf{u}), \mathcal{L}_t(\mathbf{u}))$ is incremented for every pixel, and then normalized by the label cardinality. The propagated label map, $\mathcal{L}^p_t(\mathbf{u})$, is then constructed from the maximum of the correspondence matrix if the overlap percentage exceeds a threshold, or set to a new label otherwise. The propagated label map is used in the map fusion to update the segment weight of surfels. If the label of the matched surfel is equal to the corresponding point in $\mathcal{L}^p_t$, the segment weight is increased by the confidence value, w. Otherwise, the segment

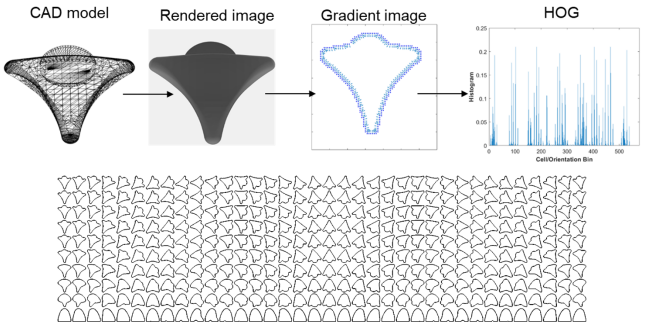

Fig. 2. Zero-Shot classifier training procedure.

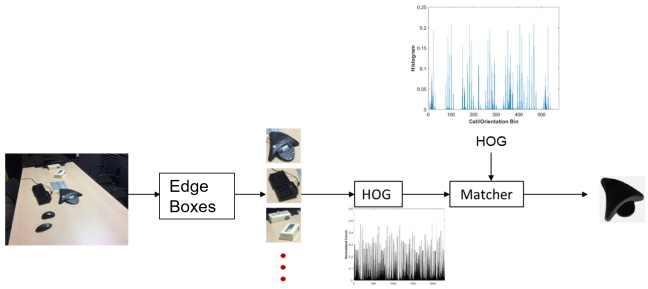

Fig. 3. Zero-Shot inference.

weight is decreased by w until 0, at which point the surfel label is updated to the new value in $\mathcal{L}_t^p$.

Single objects may be segmented into separate clusters if there are foreground occlusions. To track potential opportunities for merging, a persistent merge confidence matrix, $\Psi \in \mathbb{R}^{n_{ml} \times n_{ml}}$ is updated on every new RGB-D frame. For each label pair in $\mathcal{L}_t^r$ that is associated with the same label in $\mathcal{L}_t$, the corresponding entry in Ψ is incremented, with all other entries decremented with a clamp at 0. If a pair of labels have a merge confidence exceeding a threshold, the segments are merged, and the new label is set to the minimum of the pair. Following merging, all labels in the global map are remapped to fill any unused labels.

IV. Classification

Image-based object detectors are used to classify objects in the scene. We use YOLOv3 [11] to detect common objects and a zero-shot classifier for detecting novel objects. Both detectors operate on RGB images, and output a list of bounding boxes with associated scores, that we treat as approximate independent (multilabel) probabilities, $\mathbf{p}_{c,d} \in \mathbb{R}^{n_c}$.

After map fusion, segments in the global map are associated with detection bounding boxes. Matrices $I \in \mathbb{R}^{n_{ml} \times n_d}$ and $U \in \mathbb{R}^{n_{ml} \times n_d}$ capture the cardinality of the intersection and union, respectively, between the sets of pixels in $\mathcal{L}_t^p$ and detection bounding boxes, for each pair of segment and detection. The corresponding segment for each detection is calculated with the intersection-over-union (IoU) metric $s_d = \arg\min_l \{I(l,d)/U(l,d)\}$. For each corresponding segment, s_d, the class probabilities for all surfels in that segment are updated with Bayes rule, assuming the false positive and false negative likelihoods are approximately equal, using $\mathbf{p}_c(i) \leftarrow (\mathbf{p}_{c,d}(i)\,\mathbf{p}_c(i)) / (\mathbf{p}_{c,d}(i)\,\mathbf{p}_c(i) + (1 - \mathbf{p}_{c,d}(i))(1 - \mathbf{p}_c(i)))$.

A. Zero-Shot Classifier

In some cases, a large dataset of imagery of objects of interest may not be available, but approximate geometric models may be constructed. For example, a generic model of a cup may be constructed, without knowledge of exact dimensions. To detect never-before-seen objects, we developed a zero-shot classifier (ZSC) that trains using geometric models and can detect objects using RGB imagery. By detecting using RGB imagery rather than directly matching segments, the classification performance is independent of clustering, and can potentially inform the segmentation boundaries. In our approach, we assume that generic models of objects of interest are available, optionally parameterized by unknown dimensions. Renderings of such models can then be matched with RGB frames at runtime to detect objects of interest.

As part of training, orthographic views of the geometric model are rendered from different viewpoints, at 5° increments in azimuth and elevation. If model dimensions are parameterized, multiple images can be rendered from each viewpoint for different combination of parameter values. Images are scaled down to 16x16 resolution templates. Since objects in the scene may have different colors than the geometric model, Histogram of Oriented Gradients (HOG) [32] feature descriptors are used for template matching. To increase runtime performance, HOG features are precomputed for each of the rendered templates.

During runtime, Edge Boxes [33] are calculated on the RGB image to generate a list of object proposals. After scaling down to 16x16 images, HOG descriptors are calculated for each object proposal, and matched with the templates using the cosine similarity distance to generate a match score. If the minimum score is below a threshold, the object proposal is forwarded as a detection, with the associated template class.

With the limited dataset, simplistic features, and reliance on object silhouettes, our approach is likely to produce many false positives, as shown in Section VII-A. In some applications, high recall and sensitivity is favorable, even at the expense of additional false positives. An added benefit of this rendered template based approach is that the matching pose is also detected, which can be used in a "predict-verify" loop to further reduce uncertainty.

V. Exploration

The final major component of the toolchain is a method to rapidly and autonomously move the robot and sensors to drive down uncertainty in the map. There are two major sources of uncertainty in the map: (a) unexplored regions of the working space (e.g., unseen rooms in a building) and (b) uncertain classification of objects. To decide where to move/look to next, we quantify the predicted information gain (uncertainty reduction) from a number of potential actions and pick the best one. Specifically, we calculate the information gain as $IG(a) = w_1 \Delta_a H + w_2 \Delta_a IC$ for each

action, a, where $\Delta_a H$ is the change in entropy for exploring unseen areas of the map, and $\Delta_a IC$ is the change in the information content attribute in each surfel in the map. A list of potential actions is generated by (a) randomly sampling the working space and (b) picking views around objects of interest. For (b), we perform principal component analysis (PCA) on the positions of surfels in each segment. This provides a rough size of each object and can be used to filter out segments that are too small/large with respect to objects of interest. The directions of each component provide the dominant directions of the object features. These dominant directions should provide orthogonal, but "interesting", views of objects in the scene. This list of potential viewpoints may include views that are unreachable by the robotic platform, and are pruned by the path planner (see Section VI). For the remaining viewpoints, the information gain is evaluated, and the viewpoint with the maximum IG is selected.

A. Geometric Exploration

A geometric exploration metric is used to encourage exploration into new and unseen areas in the working space. Our approach extends the information gain-based technique in [34] to 3D. In addition to the global surfel map, we construct a coarse occupancy grid while mapping. A direct 3D representation of an occupancy grid would have enormous memory requirements and would be infeasible to use for map exploration. Instead, we use an octree data structure [35] to efficiently represent large-scale, 3D occupancy. With the probabilistic representation of space occupancy, we can apply information gain measures directly on the octree. Specifically, assuming each cell's occupancy to be independent, we can calculate the entropy in the octree before, and predicted entropy after, a new measurement, and use the entropy reduction as a measure of information gain. For an occupancy grid, entropy is calculated by $H(m) = -\sum_{c \in m} \left[p(c) \log p(c) + (1 - p(c)) \log (1 - p(c)) \right]$ where m is the occupancy map, c is a cell in m, and $p(c)$ is the probability the cell is occupied. To estimate the predicted occupancy map after a potential measurement, we render a predicted depth map from the global surfel map, and raycast through the octree using the predicted depth map.

B. Object-Centric Exploration

Calculating a similar metric for expected entropy reduction in the class probabilities would require a very large number of samples, making it infeasible in real-time for a large map with a large number of possible classes, and would likely require a calibrating each classifier with a pose-dependent confusion matrix. Instead, we use a heuristic similar to a pixels-on-target metric to quantify information gained about specific surfels from a relative perspective. Similar to [36], we construct a series of relative view bins for each surfel, $\mathbf{b}_v$. However, instead of storing only a binary histogram, each cell stores an information value, calculated as $IC(\mathbf{s}) = \exp\left(-\lambda_1 \left(\max\left\{\|\mathbf{p} - \mathbf{t}\|, d_{\text{min}}\right\} - d_{\text{min}}\right)\right)$. During map fusion, the surfels in view (as determined by the rendered index map I_t^r, ensuring occluded and back facing surfels are excluded) are updated with new IC values by $\mathbf{b}_v(i) \leftarrow \max\left\{\mathbf{b}_v(i), IC(\mathbf{s})\right\}$, where the bin i is selected based on the relative azimuth and elevation angles of the camera with respect to the surfel. To calculate these angles in a consistent manner, a basis unit vector, $\mathbf{b}$, in the plane of the surfel is stored and updated during fusion by $\mathbf{b} \leftarrow -\mathbf{n} \times (\mathbf{n} \times \mathbf{b})$. This update ensures the basis remains in the plane of the surfel, and rotates with the surfel rotation corresponding to the update in the normal vector.

Fig. 4. Robotic platform used for experiments.

During information gain calculations, predicted information maps are then rendered using the IC values from proposed new views to estimate information gain from better views of objects in the map. Pixel-wise changes are calculated and then summed to form the final scalar IC. This approach favors moving closer to an object (increasing IC within a bin), as well as seen objects from new views (thereby selecting new bins which currently have low IC values).

VI. Implementation Details

A key goal of this work is to autonomously generate semantic maps at a rapid pace. Many of the underlying algorithms in our approach are highly parallelizable, and are similar to a typical rendering pipeline. As such, we exploit GPU computing resources in most of our software implementation. The localization, map fusion, segmentation (excepting connected component analysis), YOLO-based classification, and object-centric information fusion and gain prediction are all performed using CUDA kernels. To maximally exploit computing resources, the zero-shot classifier operates on the CPU while YOLO occupies the GPU, both of which are started and complete at approximately the same time. While the semantic mapping components (Sections II-IV) run synchronously within a thread, at up to the camera frame rate, the exploration components run in a separate thread, and continuously provide new viewpoints to which to move the sensor. Experiments were performed using a laptop with an Intel Core i9-8950HK CPU, Nvidia GTX 1080 and 32 GB RAM, with all components running in realtime onboard the platform. The output maps were generated "on the fly," and continually updated and expanded during exploration, without any post-processing.

For our experiments, we used a Microsoft Kinect v2

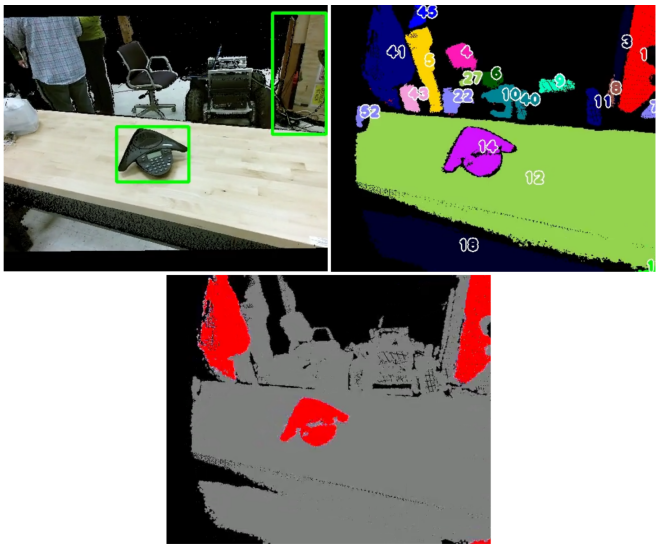

Fig. 5. Registered RGB image (top left), label map $\mathcal{L}_t^p$ (top right), and output class map (bottom) renderings during ZSC test.

RGB-D sensor mounted on a Segway RMP 440 SE [37] platform, as shown in Figure 4. The mobile platform uses ROS for telemetry and control. In particular, we used the ROS move_base interface to filter unreachable waypoints considered in the exploration waypoint generator. Furthermore, since the ground robot only has 3 degrees of freedom, we limit our action space to $a \in \mathrm{SE}(2)$, suitably corrected for the mounting orientation of the sensor on the robot.

VII. Preliminary Results

A. Zero-Shot classifier

We performed a brief test using the ZSC in the semantic mapping loop operating in a cluttered lab environment, as shown in Figure 5. In this test, we generated a dataset using CAD models of a teleconferencing phone and consumer headphones, and tested in an environment only containing the phone. Figure 5 shows the semantic mapping output from the test. The instantaneous detections shown in the upper left are associated with the segments shown in the upper right, and ultimately fused, as shown in the rendered class map on the bottom. Although the ZSC consistently detected the objects of interest, it also produced many false positives from objects that had similar HOG profiles to our templates, as shown in the misclassified objects in the bottom of Figure 5. Since the ZSC also produces object pose estimates, future work will investigate the use of pose estimate consistency as the sensor observes objects from multiple views to reduce the affect of false positives.

B. Controlled Environment

To test the full toolchain running at operational speeds, we began testing in a controlled environment. A sparse set of objects was included in the environment, only one of which (wall clock) was in the known class set of the YOLOv3 classifier. After an initial "run" signal from the operator, all sensor collection and platform motion decision making were performed autonomously by the system. Visualizations of the output of the system are shown in Figure 6, similar to what a potential user of the system may have access to. The second column shows the evolution of the RGB surfel point cloud as the robot collects data, similar to what would be generated from a SLAM system. Blue arrows indicate potential new sensor views considered by the autonomous exploration subsystem. The third column shows the object segmentation map, where distinct objects (boxes, walls, clocks, etc.) are clustered separately. The fourth column shows the generated semantic map, where objects that are classified are highlighted with bright colors. In this experiment, the clock was correctly identified and localized, though one of the boxes was misclassified. These results demonstrate that the RACE system can autonomously and semantically map an area in the span of only a few minutes.

C. Cluttered Environment

A similar experiment was conducted using the same robotic hardware and software toolchain, except this time in a more realistic cluttered lab environment. Figure 7 shows visualization output from this test. In this test, most of the objects in the lab were not part of the known class set, and therefore only a few objects were correctly identified, mostly chairs and computer peripherals. However, most objects were still correctly clustered into distinct, though unknown, objects.

VIII. Conclusion

In this work, we have developed a system for rapidly and autonomously generating geometric and semantic maps of unknown, indoor environments. Our method capitalizes on state-of-the-art results in various research areas, such as SLAM, object segmentation, image-based classification, and sensor planning. Our framework enables these capabilities to be efficiently operated together to form a novel end-to-end exploration and semantic mapping capability. We have presented preliminary results using a prototype implementation, and demonstrated how an end-to-end system may operate in realistic environments. These preliminary results indicate that it is plausible to segment (and ultimately classify) most objects in even a large, cluttered room in just a few minutes.

A key limitation in performing quantitative and comparative analyses of our approach is the fact that we control the sensor in the loop; existing SLAM and segmentation datasets have fixed sensor motions, and most robotic simulators do not produce high fidelity sensor simulations. Furthermore, optimal sensor planning in complex environments with high degrees of freedom continues to be a challenging problem, without clear ground truth solutions to compare against. With the recent advances in photorealistic robotic simulation environments [38, 39], our future work includes evaluating our method with at least the additional mapping and localization ground truth provided by these simulations.

References

[1] F. Husain *et al.*, "Combining Semantic and Geometric Features for Object Class Segmentation of Indoor Scenes," *IEEE Robot. Autom. Let.*, vol. 2, no. 1, pp. 49–55, Jan. 2017.

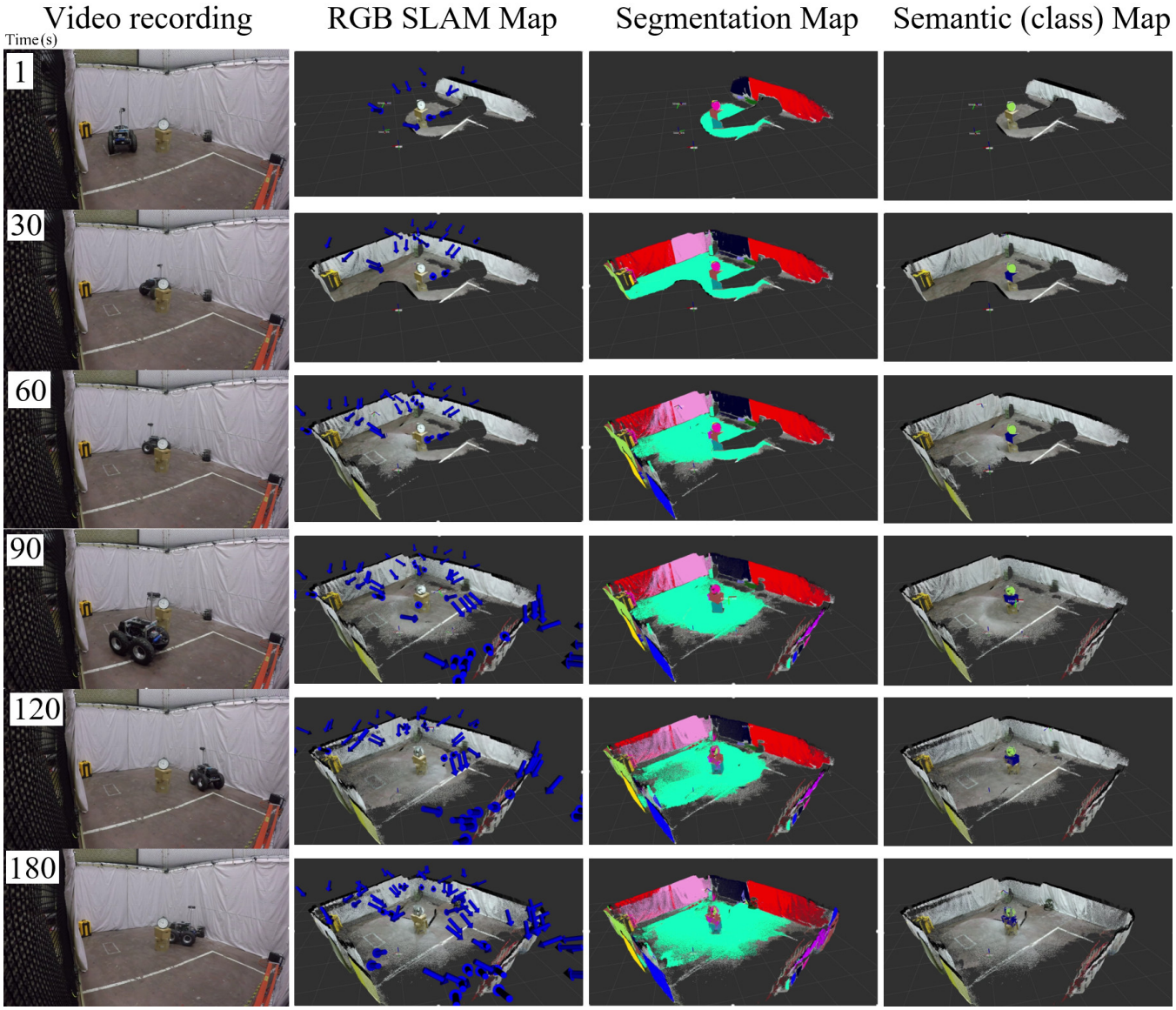

Fig. 6. Experiment in a controlled environment with a single object within the class set.

[2] J. Mccormac *et al.*, "Fusion++: Volumetric Object-Level SLAM," in *Int. Conf. 3D Vision*, Sep. 2018, pp. 32–41.

[3] L. Ma *et al.*, "Multi-view deep learning for consistent semantic mapping with RGB-D cameras," in *IEEE Int. Conf. Intell. Robot. Syst.*, Sep. 2017, pp. 598–605.

[4] J. McCormac *et al.*, "SemanticFusion: Dense 3d semantic mapping with convolutional neural networks," in *Int. Conf. Robot. Autom.*, May 2017, pp. 4628–4635.

[5] C. Wu, I. Lenz, and A. Saxena, "Hierarchical Semantic Labeling for Task-Relevant RGB-D Perception," in *Robot. Sci. Syst.*, vol. 10, Jul. 2014.

[6] R. B. Rusu *et al.*, "Model-based and learned semantic object labeling in 3d point cloud maps of kitchen environments," in *IEEE Int. Conf. Intell. Robot. Syst.*, Oct. 2009, pp. 3601–3608, iSSN: 2153-0858, 2153-0866.

[7] Z. Zeng *et al.*, "Semantic Mapping with Simultaneous Object Detection and Localization," in *IEEE Int. Conf. Intell. Robot. Syst.*, Oct. 2018, pp. 911–918.

[8] A. Hermans, G. Floros, and B. Leibe, "Dense 3d semantic mapping of indoor scenes from RGB-D images," in *Int. Conf. Robot. Autom.*, May 2014, pp. 2631–2638.

[9] L. Liu *et al.*, "Deep Learning for Generic Object Detection: A Survey," *Int J Comput Vis*, vol. 128, no. 2, pp. 261–318, Feb. 2020.

[10] Z.-Q. Zhao *et al.*, "Object Detection With Deep Learning: A Review," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019.

[11] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767 [cs]*, Apr. 2018, arXiv: 1804.02767.

[12] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector," in *Eur. Conf. Comput. Vision*, 2016, pp. 21–37.

[13] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.

[14] B. Romera-Paredes and P. Torr, "An embarrassingly simple approach to zero-shot learning," in *International Conference on Machine Learning*, Jun. 2015, pp. 2152–2161.

[15] L. Carlone *et al.*, "Active SLAM and Exploration with Particle Filters Using Kullback-Leibler Divergence," *J Intell Robot Syst*, vol. 75, no. 2, pp. 291–311, Aug. 2014.

[16] H. Carrillo *et al.*, "On the monotonicity of optimality criteria during exploration in active SLAM," in *Int. Conf. Robot. Autom.*, May 2015, pp. 1476–1483.

[17] N. Atanasov *et al.*, "Decentralized active information acquisition: Theory and application to multi-robot SLAM," in *Int. Conf. Robot. Autom.*, May 2015, pp. 4775–4782.

[18] M. Kontitsis, E. A. Theodorou, and E. Todorov, "Multi-robot active SLAM with relative entropy optimization," in *Am. Control Conf.*, Jun. 2013, pp. 2757–2764.

[19] D. Stampfer, M. Lutz, and C. Schlegel, "Information driven sensor

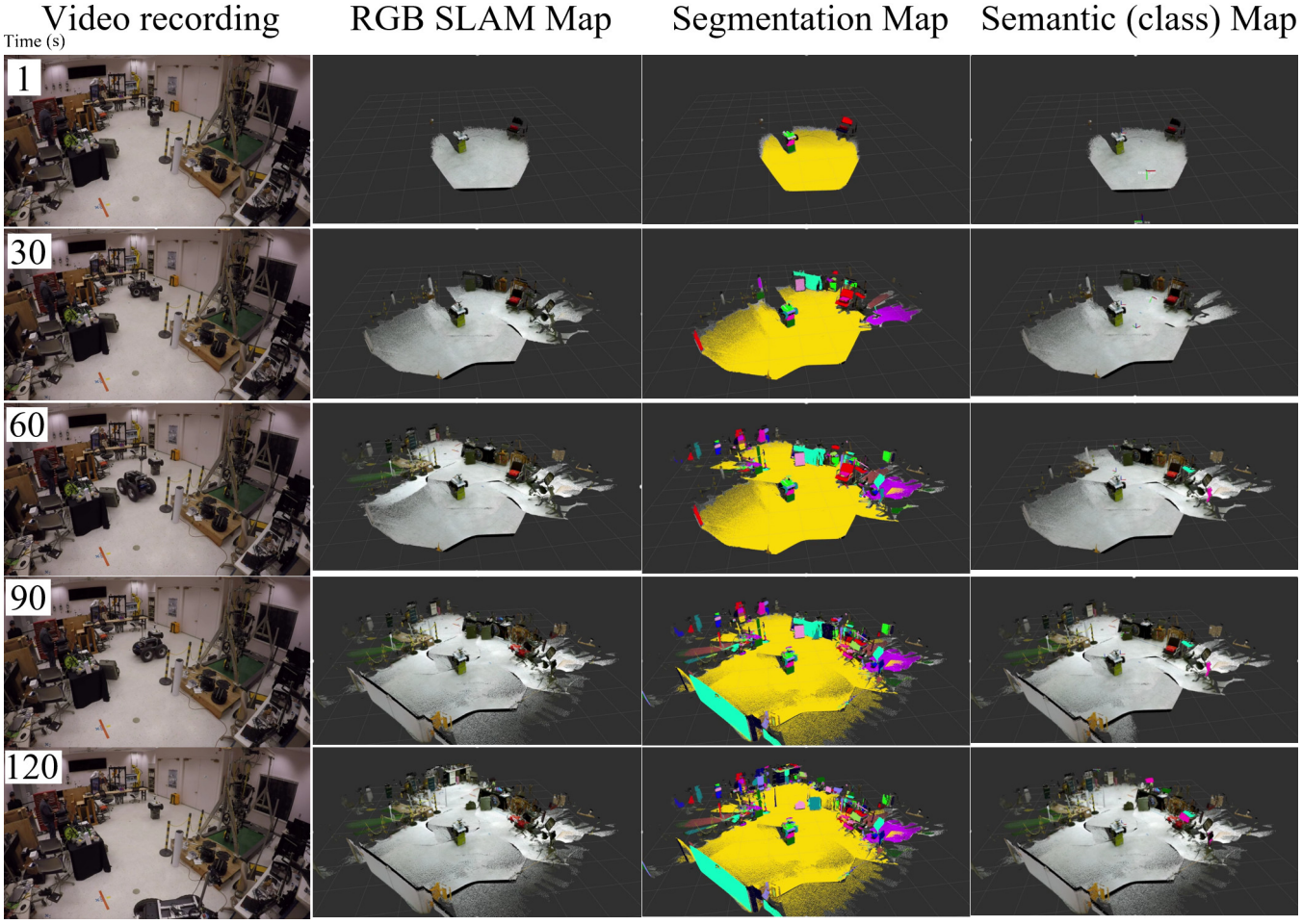

Fig. 7. Experiment in a cluttered lab environment.

placement for robust active object recognition based on multiple views," in *IEEE Int. Conf. Technol. Pract. Robot Appl.*, Apr. 2012, pp. 133–138.

[20] R. Valencia and J. Andrade-Cetto, "Active Pose SLAM," in *Mapping, Planning and Exploration with Pose SLAM*, 2018, pp. 89–108.

[21] J. E. Banta *et al.*, "A next-best-view system for autonomous 3-D object reconstruction," *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans*, vol. 30, no. 5, pp. 589–598, Sep. 2000.

[22] S. Haner and A. Heyden, "Covariance Propagation and Next Best View Planning for 3d Reconstruction," in *Eur. Conf. Comput. Vision*, 2012, pp. 545–556.

[23] C. Potthast and G. S. Sukhatme, "A probabilistic framework for next best view estimation in a cluttered environment," *J. Visual Commun. Image Represent.*, vol. 25, no. 1, pp. 148–164, Jan. 2014.

[24] B. Browatzki *et al.*, "Active object recognition on a humanoid robot," in *Int. Conf. Robot. Autom.*, May 2012, pp. 2021–2028.

[25] I. González-Díaz, V. Buso, and J. Benois-Pineau, "Perceptual modeling in the problem of active object recognition in visual scenes," *Pattern Recognit.*, vol. 56, pp. 129–141, Aug. 2016.

[26] M. Keller *et al.*, "Real-Time 3d Reconstruction in Dynamic Scenes Using Point-Based Fusion," in *Int. Conf. 3D Vision*, Jun. 2013, pp. 1–8.

[27] Y. Ma *et al.*, *An Invitation to 3-D Vision: From Images to Geometric Models*, ser. Interdisciplinary Applied Mathematics. New York: Springer-Verlag, 2004.

[28] T. Mallick, P. P. Das, and A. K. Majumdar, "Characterizations of Noise in Kinect Depth Images: A Review," *IEEE Sens. J.*, vol. 14, no. 6, pp. 1731–1740, Jun. 2014.

[29] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling Kinect Sensor Noise for Improved 3d Reconstruction and Tracking," in *Int. Conf. 3D Imaging Model. Process.*, Oct. 2012, pp. 524–530, iSSN: 1550-6185.

[30] R. A. Newcombe *et al.*, "KinectFusion: Real-time dense surface mapping and tracking," in *IEEE Int. Symp. Mix. Augm. Reality*, Oct. 2011, pp. 127–136.

[31] K. Tateno, F. Tombari, and N. Navab, "Real-time and scalable incremental segmentation on dense SLAM," in *IEEE Int. Conf. Intell. Robot. Syst.*, Sep. 2015, pp. 4465–4472.

[32] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Comput. Vision Pattern Recognit.*, vol. 1, Jun. 2005, pp. 886–893 vol. 1.

[33] C. L. Zitnick and P. Dollár, "Edge Boxes: Locating Object Proposals from Edges," in *Eur. Conf. Comput. Vision*, 2014, pp. 391–405.

[34] C. Stachniss, G. Grisetti, and W. Burgard, "Information Gain-based Exploration Using Rao-Blackwellized Particle Filters," in *Robot. Sci. Syst.*, vol. 01, Jun. 2005.

[35] A. Hornung *et al.*, "OctoMap: an efficient probabilistic 3d mapping framework based on octrees," *Auton Robot*, vol. 34, no. 3, pp. 189–206, Apr. 2013.

[36] T. Weise *et al.*, "In-hand scanning with online loop closure," in *IEEE Int. Conf. Comput. Vision*, Sep. 2009, pp. 1630–1637.

[37] "Segway RMP 440 SE."

[38] P. Martinez-Gonzalez *et al.*, "UnrealROX: an extremely photorealistic virtual reality environment for robotics simulations and synthetic data generation," *Virtual Reality*, Aug. 2019.

[39] M. Savva *et al.*, "Habitat: A Platform for Embodied AI Research," in *IEEE Int. Conf. Comput. Vision*, 2019, pp. 9339–9347.