Real-time Semantic 3D Reconstruction for High-Touch Surface Recognition for Robotic Disinfection

Ri-Zhao Qiu[†1], Yixiao Sun[†2], João Marcos Correia Marques[1] and Kris Hauser[1]

***Abstract*— Disinfection robots have applications in promoting public health and reducing hospital acquired infections and have drawn considerable interest due to the COVID-19 pandemic. To disinfect a room quickly, motion planning can be used to plan robot disinfection trajectories on a reconstructed 3D map of the room's surfaces. However, existing approaches discard semantic information of the room and, thus, take a long time to perform thorough disinfection. Human cleaners, on the other hand, disinfect rooms more efficiently by prioritizing the cleaning of high-touch surfaces. To address this gap, we present a novel GPU-based volumetric semantic TSDF (Truncated Signed Distance Function) integration system for semantic 3D reconstruction. Our system produces 3D reconstructions that distinguish high-touch surfaces from non-high-touch surfaces at approximately 50 frames per second on a consumer-grade GPU, which is approximately 5 times faster than existing CPU-based TSDF semantic reconstruction methods. In addition, we extend a UV disinfection motion planning algorithm to incorporate semantic awareness for optimizing coverage of disinfection trajectories. Experiments show that our semantic-aware planning outperforms geometry-only planning by disinfecting up to 20% more high-touch surfaces under the same time budget. Further, the real-time nature of our semantic reconstruction pipeline enables future work on simultaneous disinfection and mapping.**

Code is available at: https://github.com/uiuc-iml/RA-SLAM

I. Introduction

Scene understanding plays a crucial role in autonomous robots, as many tasks require the robot to have geometric and semantic knowledge of its environment. Semantic reconstruction has become a topic of intense recent interest with recent developments in deep neural networks [14, 16, 22, 31]. It extends 3D geometric reconstruction by estimating both the 3D representation of a scene and labeled object classes or instances using computer vision methods [14, 16]. Applications of 3D reconstruction with semantic awareness include camera pose estimation [18], Augmented Reality (AR) [16], and robot navigation [22].

The goal of this paper is to reconstruct and recognize *high-touch surfaces* for the purposes of robotic disinfection. Humans are able to disinfect rooms efficiently by prioritizing the cleaning of high-touch surfaces [9, 10], which are pathogen-transmitting surfaces (fomites) that are much more likely than others to have been touched and therefore contaminated. For example, doorknobs, handrails, and switches are commonly handled whereas walls are not. We hypothesize that robots that use semantic information to identify such surfaces can disinfect an environment more efficiently than robots that rely only on geometric 3D mapping [7].

Fig. 1: Comparison of 3D semantic reconstructions of ScanNet [5] stream scene0665_00 with high-touch surface segmentation. High-touch objects such as tables and chairs are colored in the red. Non-high-touch surfaces are colored in green. The estimated semantic reconstruction (right) produced by our system captures the majority of high-touch surfaces in the ground-truth semantic reconstruction (left). (Best seen in color)

To this end, we present a novel volumetric semantic TSDF reconstruction system and a semantic-aware disinfection planner that allows a robot to leverage both semantic and geometric information to optimize its disinfection paths. We develop a GPU-based TSDF reconstruction system that performs attribute segmentation and semantic large-scale TSDF integration purely on GPU. Exploiting the parallel computational power of the GPU and voxel hashing, our system is capable of running at 50 fps on a single consumer-grade NVIDIA GeForce GTX 1060 GPU. In comparison, PanopticFusion [16], a notable semantic TSDF reconstruction system that uses GPU for segmentation and CPU for semantic TSDF reconstruction, operates at a throughput of 4.3 Hz on two NVIDIA GeForce GTX 1080Ti GPUs. We evaluate the performance of our system on the ScanNet dataset [5], demonstrating an Area-Under-the-Curve (AUC) of 0.73 for high-touch surface prediction. Furthermore, when semantic-labeling is incorporated as prioritization scores into the UV disinfection planner of Marques *et al.* [13], a robot is able to disinfect up to 20% more of the high-touch surfaces in the environment given the same time budget.

II. Related Works

A. 3D Geometric Reconstruction

3D reconstruction generates a representation of an environment's geometry given RGBD frames as input. Common representations include surfels, point clouds, and volumetric mapping with Truncated Signed Distance Function (TSDF). TSDF methods reconstruct a scene by averaging weighted TSDF observations from individual frames into a global

[†] Equal contribution

[1]R. Qiu, J. M. C. Marques and K. Hauser are with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA. {rizhaoq2,jmc12,kkhauser}@illinois.edu.

[2]Y. Sun is with the Department of Mechanical Engineering, Stanford University, Stanford, CA, USA. alvinsun@stanford.edu.

*This work is partially supported by NSF Grant #2025782.

frame [17], making it more robust to noise in the camera's depth measurements [19]. In addition, visibility determination and collision detection are more efficient when TSDF-based representations are used [21].

KinectFusion [17] introduced TSDF-based reconstruction on a GPU to carry out real-time dense surface mapping over a dense voxel grid. InfiniTAM [8, 21] overcame limitations of a fixed voxel grid by utilizing the voxel hashing technique [19] to create a sparse voxel representation of the environment. These methods are highly efficient, but produce geometric reconstruction without any semantic information that can benefit downstream tasks.

B. 3D Semantic Reconstruction

A pioneering work in semantic reconstruction is SemanticFusion [14], which was extended from ElasticFusion [30], a surfel-based dense SLAM algorithm. SemanticFusion uses CNN to predict semantic label maps on RGB images, and then propagates labels to the surfel integrator, which uses a Bayesian update scheme to fuse newly predicted probabilities with existing ones. Finally, a post-processing step is applied to regularize the map. SemanticFusion leverages the GPU to update the surfel map, yielding a frame rate of 25.3 Hz. However, surfel-based representations are less convenient for visibility determination and collision detection than TSDF-based representations.

PanopticFusion, by Narita *et al.* [16], produces a TSDF volumetric semantic mapping for larger scenes. Besides semantic labeling, PanopticFusion performs panoptic label tracking as well, giving countable objects in the scene that belong to the same class different instance labels. However, PanopticFusion builds on voxblox [20], a CPU-based TSDF mapping framework, so it does not fully leverage GPU parallelism and is reported to run at a throughput of 4.3 Hz with an Intel Core i7-7800X CPU running at 3.50 GHz and two NVIDIA GeForce GTX 1080Ti GPUs. In comparison, our GPU-based semantic-enabled TSDF integrator operates at approximately 50 Hz for efficient semantic reconstruction. However, our method does not perform multi-class segmentation or panoptic labeling, but this is suitable for our application because only a binary label is needed for high-touch surface prediction.

The recently released Kimera [22] package provides a semantic SLAM solution. Kimera is a CPU-based implementation that provides a per-frame 3D mesh with a latency of 5 ms, which supports real-time applications such as obstacle avoidance. However, for semantic labeling, Kimera is reported to operate at 10 Hz using provided segmentation labels, which is not as efficient as our implementation.

C. Disinfection Trajectory Planning

Researchers have recently been inspired to address the COVID-19 pandemic by studying the problem of trajectory planning for UV disinfection. Sanchez and Smart [25] execute boustrophedon-style coverage paths [2] over human-segmented planes. Conte *et al.* [4] propose a teleoperated robotic platform for disinfection whose operator is guided by a real-time 3D disinfection estimated map. Both Ruan *et al.* [23] and Vyshnavi *et al.* [29] propose an autonomous system that traces a user-specified set of waypoints while avoiding collisions with dynamic obstacles in the environment. These methods rely on human expertise, which limits their convenience and performance.

Others perform optimal coverage path planning on 2D floor plans. Conroy *et al.* [3] propose selecting waypoints on a 2D floor plan via solving a linear program over a regular grid and formulating the connection of the selected waypoints as a Travelling Salesman Problem. Similarly, Tiseni *et al.* [28] uses artificial random-fields to guide a robot's motion according to the estimated disinfection status of the floor plan, using Genetic Algorithms to optimize the potential functions used in path generation. 3D geometry is important to address for disinfecting real environments, and Marques *et al.* [13] present a dosage planning and waypoint selection algorithm that uses a 3D mesh of the environment. Most related to this work, Hu *et al.* [7] propose a dense semantic reconstruction pipeline for predicting high-touch surfaces on a scene, then using handcrafted disinfection motion primitives to irradiate predicted high-touch areas. We extend the work of Marques *et al.* [13] by using predictions of high-touch probability to prioritize high-touch surfaces.

III. Perception System

A. System Overview

Fig. 2 presents a flowchart of the proposed semantic reconstruction pipeline. Our system takes in timestamped RGBD frames as inputs. During the pre-processing phase, we feed the RGBD frame into a 2D affordance segmentation model and a camera pose estimation module run in parallel. We follow Narita *et al.* [16] and use an external visual SLAM algorithm (such as OpenVSLAM [27] or ORB-SLAM [15]) for camera pose estimation. The resulting estimated poses and the predicted affordance map are then sent to a semantic TSDF integrator where the estimated geometry and semantic probabilities are incrementally integrated. As illustrated in Fig. 2, our system supports various real-time scene extraction methods such as direct generation of 2D projections of TSDF via ray casting, rasterization, and meshing with the Marching Cubes [12] algorithm.

B. 2D Affordance Segmentation

For every RGB image in the RGBD frame, we use a convolutional neural network (CNN) to predict a pixel-wise high-touch affordance map. The high-touch affordance map estimates the likelihood of every observed pixel being a part of a high-touch surface or a low-touch surface. The input to the model are the RGB images from the camera, and the model outputs high-touch/low-touch probability maps of the same spatial resolution as the input images. For efficiency, we select lightweight RefineNet [11] with ResNet-18 [6] backbone as our model.

We use the open-source multi-class dataset ADE20K [32] and the ScanNet dataset [5] to train our segmentation model, and then we create a surjective mapping ϕ to output binary

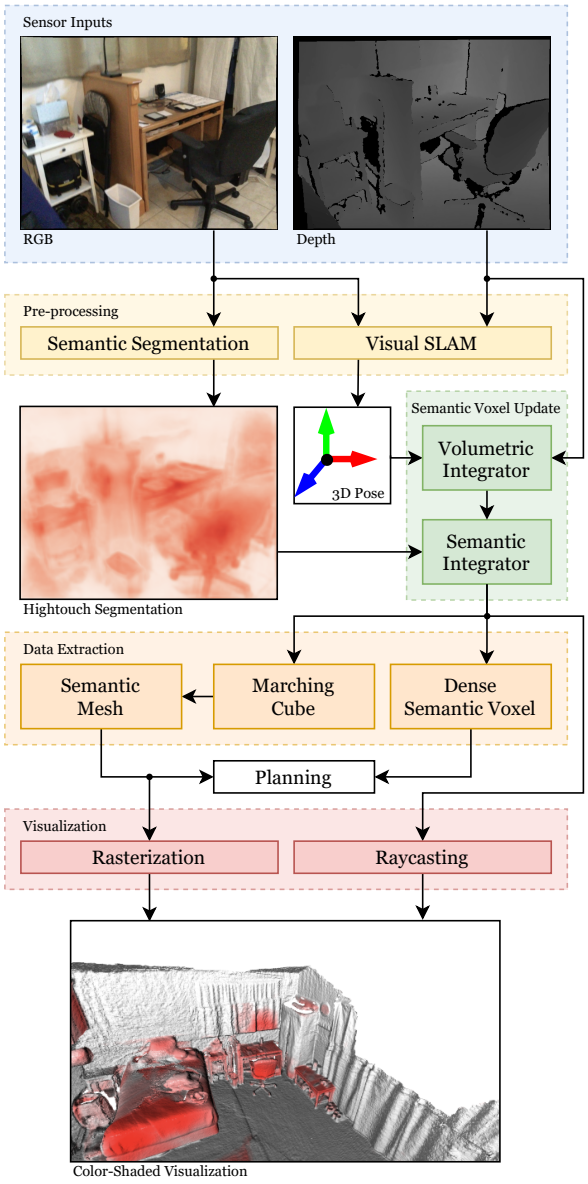

Fig. 2: High-level overview of our GPU-based volumetric semantic reconstruction system. All modules run on GPU except for the external visual SLAM module in the pre-processing stage. (Best seen in color)

high-touch prediction labels. We divide the set of all classes in the open-source dataset L into the classes L_{ht} that are high-touch and L_{other} which contains the rest (such as wall and floor). The mapping function ϕ is given by $\phi(x) = I[x \in L_{ht}]$. The subset L_{ht} of high-touch surfaces from the ADE20K dataset [32] and the ScanNet dataset [5] are given in the Appendix. The segmentation model is trained using multi-class prediction using cross entropy loss. That is, for an input RGB image of size (H, W), the segmentation model predicts a probability tensor P of shape $(|L|, H, W)$. Let $P_c(i,j)$ be the predicted probability of pixel (i,j) belonging to the c-th class. Then we define $\hat{Y}_h(i,j)$ and $\hat{Y}_l(i,j)$ as the estimated high-touch and non-high-touch probability of pixel (i,j), respectively. During inference, we max-aggregate the probability maps using

$$\hat{Y}_h(i,j) = \mathbf{max}_{c \in L_{ht}} P_c(i,j) \quad (1)$$

$$\hat{Y}_l(i,j) = \mathbf{max}_{c \in L_{other}} P_c(i,j). \quad (2)$$

C. Dense Semantic Reconstruction - Voxel Hashing

Unlike many previous works in GPU-based semantic reconstruction [14], our system adopts a TSDF representation of 3D space implemented with sparsely hashed voxels, leveraging both the parallel computational power from the GPU and the memory benefits of hashing techniques.

We build a voxel hashing scheme that supports incremental semantic probability integration based on the work of Nießner *et al.* [19]. As in [19], the voxel hashing table manages an intermediate unit called *voxel blocks*, where each voxel block manages 8^3 voxels. To resolve hashing collision, we followed [19] and use the linked list technique. That is, we traverse the array representation and find the next available entry if two voxels are hashed into the same entry.

Different from Nießner *et al.* [19], in our design, the size of the voxel hashing table is initialized to support up to 2^{21} voxel blocks. The work of Nießner *et al.* [19] stores 8 channels in every voxel. To support high-touch surfaces predictions, we extend the voxel to store 9 channels with an addition high-touch probability channel. The computation and integration process of this additional channel is described in Section III-D.4.

We chose to implement a binary attribute prediction system instead of a multi-class prediction system so that our system can run on consumer-grade GPUs or other edge-computing platforms. In our current implementation, the size of the map is fixed and is pre-allocated during initialization of the system. When the voxel size of 1 cm is used, the system is measured to occupy 4,367 megabytes of GPU memory. Every additional float feature for every voxel would incur an additional 512 megabytes of GPU memory usage, given the memory overhead of semantic TSDF representation and all intermediate buffers. To accommodate the NYU40 [26] labels used in the ScanNet [5] dataset, we would need to introduce 40 semantic classes, which is infeasible for consumer-grade GPUs.

D. Volumetric Integration

1) Voxel Selection: We follow Nießner *et al.* [19] and use a variant of the Digital Differential Analyzer (DDA) algorithm proposed in [1] to select a subset of voxel blocks to access. The DDA algorithm sends a ray from the focal point of the camera and selects voxel blocks that are within the specified truncation distance from the measured point as candidates. If voxel blocks selected by the DDA algorithm do not exist, new voxel blocks will be initialized at that spatial location with TSDF weight of 0. Existing voxel blocks are updated as described in the following subsection.

2) TSDF Update: Consider an arbitrary voxel v that is selected by the DDA algorithm. Let d_v be the depth of the voxel if it were to be observed by the camera. Let d be the observed depth value of the projected pixel of this voxel on the camera plane along the casted ray. To fuse the newly

observed TSDF value with the existing value stored in the voxel. We follow the design of the weighted average scheme in InfiniTAM [8]. In particular, we use the heuristic that the values observed by the depth camera become noisier the further it is from the focal point of the camera. We compute the TSDF weight w_{new} for every measurement according to

$$w_{new} = \left(1 - \frac{d}{d_{\max}}\right) \cdot w_{inc} \tag{3}$$

where d_{max} is the maximum specified range of the depth camera used and $w_{inc} = 4$ represents the maximum weight of a single observation. Let TSDF_{old} be the existing TSDF value and w_{old} be the associated TSDF weight stored in the voxel. We update the TSDF value by

$$\text{TSDF} = \frac{(d - d_v) \cdot w_{new} + \text{TSDF}_{old} \cdot w_{old}}{w_{new} + w_{old}}. \tag{4}$$

After the TSDF update, we update the weight stored in the voxel by

$$w = \textsc{Min}\{w_{old} + w_{new}, w_{max}\} \tag{5}$$

where we use $w_{max} = 25 \cdot w_{inc}$ to clip the accumulated TSDF weight. This is done so that if there are multiple new measurements of the voxel that are different from the existing one, the TSDF value can be shifted towards the new observations.

The TSDF weight w is initialized to 1 when their corresponding voxel blocks are first initialized in the GPU memory.

3) Color (RGB value) Update: In the color update, we use a similar scheme as the TSDF update in equation 4,

$$\text{RGB} = \frac{\text{RGB}_{new} \cdot w_{new} + \text{RGB}_{old} \cdot w_{old}}{w_{new} + w_{old}} \tag{6}$$

where w_{new} and w_{old} are the TSDF weights in equation 4.

4) Semantic Probability Update: For semantic probability integration, we use a different weighting scheme. For any observed pixel, its semantic probability weight is given by

$$v_{new} = -\frac{16d(d - d_{max})}{d_{max}^2} \tag{7}$$

which is a quadratic function with a negative quadratic coefficient. Empirically, CNN-based segmentation models perform best on mid-range objects. When the object is too far away or too close to the camera, the prediction of the model becomes less reliable. We introduce this heuristic weighting to offset the prediction confidence at different ranges.

Let p_{old} be the existing probability and v_{old} be existing weights. The semantic probability is computed by

$$p_{ht} = \exp \frac{v_{old} \cdot \log p_{old} + v_{new} \cdot \log \hat{Y}_h(i,j)}{v_{new} + v_{old}} \tag{8}$$

$$p_{lt} = \exp \frac{v_{old} \cdot \log(1 - p_{old}) + v_{new} \cdot \log \hat{Y}_l(i,j)}{v_{new} + v_{old}} \tag{9}$$

$$p_{new} = \frac{p_{ht}}{p_{ht} + p_{lt}}. \tag{10}$$

Note that when $v_{old} = v_{new} = 1$, the equations reduce to recursive Bayesian update.

Similarly, after semantic probability integration, we update the weight stored in the voxel by

$$v = \textsc{Min}\{v_{old} + v_{new}, v_{max}\} \tag{11}$$

where $v_{max} = 100$.

IV. Semantic-Aware UV Dosage Planner

We modify the optimal disinfection trajectory planner by Marques *et al.* [13] to consider semantic information in dosage optimization. The original planning pipeline is as follows: Given a mesh with N triangles s_j and a robot model, K candidate vantage points x_i are proposed within the workspace of the robot from a 3D grid. For each reachable candidate point, an irradiance vector I_i is calculated, where I_{ij} is the irradiance produced by the light source at x_i over the mesh triangle s_j. We consider that the robot will stay on each vantage point for a time t_i and those times can be optimized by the following LP, where T_{max} is the total time budget for disinfection, μ_{min} is the target disinfection fluence, σ_j is a slack variable that assures the feasibility of the LP under tight time limits and m_j is a penalty for not disinfecting triangle s_j.

$$\underset{t_i, \sigma_j \geq 0}{\textbf{argmin}} \sum_{i=1}^{K} t_i + \sum_{j=1}^{N} m_j \sigma_j \tag{12}$$

$$\textbf{s.t.} \sum_{i=1}^{K} I_{ij} t_i + \sigma_j \geq \mu_{min} \quad \forall j = 1, \cdots, N \tag{13}$$

$$\sum_{i=1}^{K} t_i \leq T_{max}, \tag{14}$$

In Marques *et al.* [13], since no semantic information was available, m_j was set to be simply proportional to the area of the surface patch s_j. In order to incorporate the high-touch surface probabilities from semantic reconstruction, we consider 2 different paradigms: soft thresholding (**SoT**) and hard thresholding (**HaT**). In ST, failure to disinfect a given surface s_i, represented by the slack variable σ_i, is penalized linearly according to their surface area and high-touch probability. More concretely, when ST is used, the penalization terms m_j is defined as in Equation 15, where $A(s)$ is the area of triangle s, p_{ht}^j is the estimated probability that s_j is a high-touch surface, M_a and M_s are the area and semantic penalty scale factors, respectively. The area penalty is normalized by the area of the smallest triangle in the scene to ensure this penalty is always greater than one and that it is invariant with respect to the units of the mesh.

$$m_j = \left(M_s \cdot p_{ht}^j\right) \left(M_a \frac{A(s_j)}{min_l A(s_l)}\right) \tag{15}$$

The intuition behind the role of M_s is that, for triangles with the same area - and in a room where all surfaces have identical irradiances, a given vantage point would have to disinfect at least $M_s p_{ht}^j$ low touch surfaces before it would be considered for the optimal solution instead of another point that disinfects the high-touch triangle s_j. A similar argument is valid for the area penalty M_a for triangles with the same high-touch probability, p_{ht}. The semantic and area penalties are multiplied together to preserve this *ceteris paribus* logic in this extension of the big M method.

In HaT, the penalty for not disinfecting is set to zero for all surfaces with p_{ht} below a specified cutoff probability α, while surfaces above that threshold get penalized according to their relative areas. The penalization terms m_j are then

given by,

$$m_j = \begin{cases} 0 & p_{ht}^j < \alpha \\ M_a \frac{A(s_j)}{min_l A(s_l)} & p_{ht}^j \geq \alpha \end{cases} \quad (16)$$

After optimizing the dwell times, the tour through all points with non-zero time is calculated following the pipeline described in Marques *et al.* [13].

V. Experiments

A. Evaluation Setup

We evaluate our system on the ScanNet V2 dataset [5], which is a dataset consisting of streams of different scenes for 3D indoor scene understanding. It contains 1201 sensor streams for training, 312 streams for validation, and 100 streams for testing purpose. Each stream provides RGBD frames, ground-truth camera trajectories, and reconstructed 3D representation of the underlying scene with $3D$ semantic annotations. Similar to how we trained the high-touch segmentation model, to adapt the dataset for evaluating semantic disinfection performance, we define a partition which maps NYU40 [26] classes in ScanNet to either high-touch or low-touch. The high-touch segmentation model is trained using the ADE20K [32] dataset, which contains 150 classes from $20,210$ images in the training split and $2,000$ images in the validation split. As in previous works such as PanopticFusion [16], this evaluation uses reference camera poses provided by the ScanNet dataset to isolate errors introduced by external visual SLAM algorithms.

B. Segmentation Model Implementation Details

The backbone ResNet-18 was initialized using ImageNet [24] pre-trained weights. Images are downscaled to $(240, 320)$ and a batch size of 16 was used for training. The training process uses an SGD optimizer to train weights and runs for 100 epochs. The learning rate is initialized to 0.01 and is reduced by 0.1 every 40 epochs.

We use various augmentation techniques during training to make the model more robust against robot motion. Besides standard horizontal flipping, random resizing and random scaling augmentations, Gaussian blur is also applied probabilistically during training to simulate camera motion.

C. Evaluation of Semantic Reconstruction System

1) Efficiency: Table I illustrates the throughput of our system compared to existing TSDF-based semantic reconstruction works. SemanticFusion [14] works real-time on a GPU and scales to large scenes thanks to its surfel-based representations. However, it does not support TSDF representations, which is an efficient type of 3D representation for many downstream robotics tasks such as robot navigation. DA-RNN [31] is based on the KinectFusion [17] work and fails to scale to large scenes. Moreover, the usage of RNNs poses heavy computational cost. Finally, CPU-based methods [16, 22] rely on VoxBlox [20] and their throughput is far from camera frame rate.

We carried out detailed run-time analysis of our method using ScanNet stream scene0000_00. We observe that the main bottleneck of our method is the semantic segmentation module, which is measured to have a latency of 16.91±2.65 ms. In stark contrast to the >100 ms volumetric integration time of CPU-based semantic TSDF methods [16, 22], our semantic TSDF integration implementation has a latency of 1.91±0.43 ms on a consumer-grade GPU. In sum, the throughput of our system is 53.1 Hz.

2) Accuracy: We evaluate accuracy of high-touch area segmentation on the validation split of the ScanNet dataset [5]. To create correspondence between vertices in the ground-truth mesh and the estimated mesh, we use nearest neighbor (NN) search. Every vertex in the estimated mesh is considered as the center of the NN search. If the nearest vertex in the ground truth mesh is within 2 cm of the center, then the correspondence is established and the labels of these two vertices are compared. If no vertex in the ground-truth mesh is within 2 cm of a vertex in the estimated mesh, the prediction of this estimated vertex is marked as incorrect regardless of its predicted label.

Figure 3 shows the Precision-Recall curve of voxel high-touch prediction in the reconstructed meshes. The Area-Under-Curve (AUC) of the 3D semantic reconstruction system is 0.73 with an Intersection-over-Union (IoU) of 54.0 when a cutoff of 0.4 is used. The AUC of the 2D high-touch segmentation model is 0.45, which is significantly worse than the AUC of the 3D segmentation. By integrating and merging 2D semantic predictions from multiple frames, our semantic reconstruction system produces a 3D segmentation with higher segmentation quality.

Table II compares the semantic accuracy of our system with SemanticFusion [14], where ground truth poses and the same 2D affordance segmentation model are used for both methods. Our method achieves best performance due its improved probability integration scheme and different scene representation than SemanticFusion. Noticeably, our method largely outperforms the native implementation of SemanticFusion, which computes segmentation only once every 10 frames for efficiency. The modified SemanticFusion runs at a reduced throughput to compute segmentation for every frame and, as a result, considerably outperforms its native counterpart but is still worse than our method.

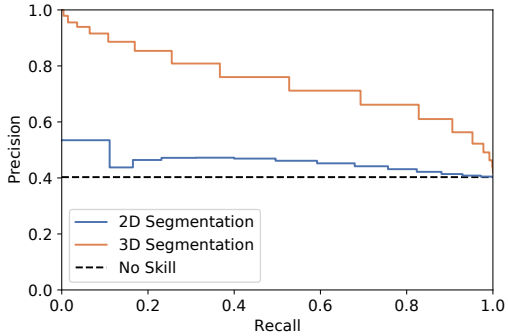

Fig. 3: Precision-Recall (PR) curve of high-touch surface prediction task. The orange PR curve is 3D reconstruction from our semantic reconstruction system. The blue PR curve is generated by the 2D segmentation model in our semantic reconstruction system. The black dashed line illustrates the PR curve of a no-skill model that predicts everything in a mesh as high-touch. (Best seen in color)

Method	Multi-Class	TSDF	Large-scale	GPU Integration	Throughput
SemanticFusion [14]	✓	✗	✓	✓	25.3 Hz
DA-RNN [31]	✓	✓	✗	✓	5 Hz
PanopticFusion [16]	✓	✓	✓	✗	4.3 Hz
Kimera [22]	✓	✓	✓	✗	<10 Hz[1]
Ours	✗	✓	✓	✓	**53.1** Hz

TABLE I: Comparisons of semantic 3D reconstruction systems. Results from other works are reported in corresponding publications. [1] Estimate; reported latency is 0.1 s not accounting for segmentation latency.

Method	IoU
SemanticFusion [14]	43.4
SemanticFusion [14][1]	46.9
Ours	**51.2**

TABLE II: Comparisons of 3D IoU between SemanticFusion [14] and our method on a subset of ScanNet [5] sequences used to evaluate disinfection efficiency. [1]: native SemanticFusion implementation performs segmentation every 10 frames for efficiency; here we perform segmentation every frame for fair comparison with our method.

D. Semantic Impact on Disinfection Efficiency

We evaluate the improvements of incorporating semantic information in UV disinfection by planning disinfection trajectories for 10 rooms on ScanNet[5] with the largest floor plans. We consider 3 different planning paradigms: Surface Agnostic, SoT, and HaT with a cutoff of 0.4. The rooms were disinfected by the "Floatbot" robot model proposed in [13] to obtain a proxy of robot-independent best possible disinfection result and speed up collision checks during the experiments, though a 6-DOF arm mounted on a holonomic base, "Armbot", was shown to have comparable disinfection capabilities [13]. The target fluences for the rooms were set to $280mj/cm^2$ and the light source considered was a spherical 10W UV light source. The penalties M_a and M_s were both set to 10.

To assess the impact of reconstruction error to disinfection planning, two experiments were conducted: one using the estimated mesh produced by our system and the other using the ground truth meshes (labelled gt). In both cases, the disinfection coverage was evaluated by executing the plan on the ground-truth mesh of the environment. Note that for the ground truth meshes, there is no difference between HaT and SoT, so only HaT experiments were performed. The plans are created using time budgets of [1,2,4,6,8,10,15,20,25,30] minutes to understand how the fraction of high-touch surfaces gets sterilized when given different time-budgets. Trajectory differences are not visualized because they are not very informative due to the probabilistic nature of motion planning algorithms.

Since different meshes are of different sizes, to compare results across meshes, we evaluate disinfection progress by fraction of disinfected high-touch area given constant time budgets. The results are presented in Figure 4, where uplift is the difference between the fraction of high-touch areas disinfected by the corresponding strategy and the surface agnostic strategy using the same time budget on the same mesh. The x axis shows the fraction of high-touch surfaces disinfected in that budget. As can be seen by the trend lines, all semantic disinfection planners outperform surface-agnostic planning (uplift > 0), even when using our estimated meshes instead of the ground truths for planning. There seems to be little difference between HaT and SoT planning. The uplift is less pronounced at higher disinfection percentages, since disinfection has a "long tail": large, open areas get easily disinfected early on, while smaller hidden surfaces often have poor reachability and visibility, taking longer to disinfect and enabling the surface-agnostic strategy to "catch-up" to semantically informed ones when given ample time.

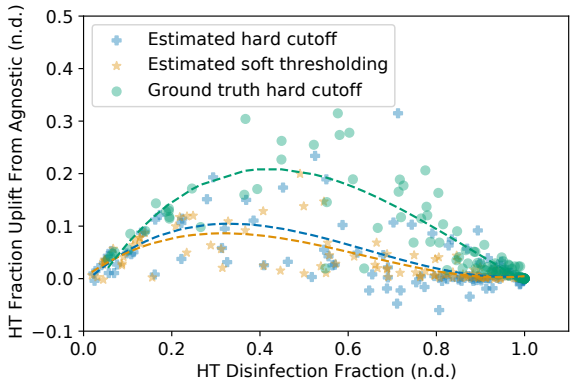

Fig. 4: Increase in disinfected high-touch surface fraction over surface agnostic planning for different surface-aware disinfection strategies over 10 ScanNet Scenes (Best seen in color)

VI. Conclusion

We present a framework to optimize UV disinfection planning problems in real-world environments. Specifically, we design an efficient TSDF-based semantic reconstruction system that is able to run at over 50Hz in consumer-grade hardware and a semantic-aware UV disinfection planner as a downstream module that outperforms semantically uninformed planners by disinfecting up to 20% more relevant surfaces in indoor environments under the same time budget. We, therefore, show that incorporating semantic knowledge in disinfection planning can provide real efficiency gains, especially under tighter time budgets.

As future work, we aim to extend our system to an actively exploring perception system that reconstructs the environment and disinfects its high-touch surfaces simultaneously to further increase the system's real-world applicability, and bypass initial mapping steps in planning. The real-time nature of the proposed semantic reconstruction pipeline is a core enabler of this future development. Further, extending this work to multiple lower powered disinfection robots would also be a fruitful direction to enable disinfection of larger or more crowded areas, while presenting interesting challenges in distributed mapping and exploration.

References

[1] J. Amanatides and A. Woo, "A fast voxel traversal algorithm for ray tracing," *Proceedings of EuroGraphics*, vol. 87, Aug. 1987.

[2] H. Choset, "Coverage of known spaces: The boustrophedon cellular decomposition," *Autonomous Robots*, vol. 9, no. 3, pp. 247–253, 2000.

[3] J. Conroy, C. Thierauf, P. Rule, E. Krause, H. Akitaya, A. Gonczi, M. Korman, and M. Scheutz, *Robot Development and Path Planning for Indoor Ultraviolet Light Disinfection*, 2021.

[4] D. Conte, S. Leamy, and T. Furukawa, "Design and Map-based Teleoperation of a Robot for Disinfection of COVID-19 in Complex Indoor Environments," in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2020, pp. 276–282.

[5] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[7] D. Hu, H. Zhong, S. Li, J. Tan, and Q. He, "Segmenting areas of potential contamination for adaptive robotic disinfection in built environments," *Building and Environment*, vol. 184, p. 107 226, Oct. 2020.

[8] O Kahler, V. A. Prisacariu, C. Y. Ren, P. H. S. S. X., Torr, and D. W. Murray, "Very high frame rate volumetric integration of depth images on mobile device," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, 11 2015.

[9] S. Kundrapu, V. Sunkesula, L. A. Jury, B. M. Sitzlar, and C. J. Donskey, "Daily Disinfection of High-Touch Surfaces in Isolation Rooms to Reduce Contamination of Healthcare Workers' Hands," *Infection Control & Hospital Epidemiology*, vol. 33, no. 10, pp. 1039–1042, Oct. 2012.

[10] E. K. Kurgat, J. D. Sexton, F. Garavito, A. Reynolds, R. D. Contreras, C. P. Gerba, R. A. Leslie, S. L. Edmonds-Wilson, and K. A. Reynolds, "Impact of a hygiene intervention on virus spread in an office building," *International Journal of Hygiene and Environmental Health*, vol. 222, no. 3, pp. 479–485, Apr. 2019.

[11] G Lin, A Milan, C Shen, and I Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[12] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *SIGGRAPH*, pp. 163–169, 1987.

[13] J. Marques, R. Ramalingam, Z. Pan, and K. Hauser, "Optimized coverage planning for uv surface disinfection," in *International Conference on Robotics and Automation (ICRA)*, 2021.

[14] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "Semanticfusion: Dense 3d semantic mapping with convolutional neural networks," in *International Conference on Robotics and Automation (ICRA)*, 2017.

[15] R. Mur-Artal and J. D. Tardós, "Orb–slam2: An open-source slam system for monocular, stereo and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, pp. 1255–1262, 5 2017.

[16] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji, "Panopticfusion: Online volumetric semantic mapping at the level of stuff and things," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4205–4212, 2019.

[17] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," 2011, pp. 127–136.

[18] L. Nicholson, M. Milford, and N. Sünderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2018.

[19] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3d reconstruction at scale using voxel hashing," *ACM Transactions on Graphics (TOG)*, 2013.

[20] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[21] V. A. Prisacariu, O. Kähler, S. Golodetz, M. Sapienza, T. Cavallari, P. H. Torr, and D. W Murray, "InfiniTAM v3: A Framework for Large-Scale 3D Reconstruction with Loop Closure," *arXiv preprint arXiv:1708.00783*, Aug. 2017.

[22] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: An open-source library for real-time metric-semantic localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[23] K. Ruan, Z. Wu, and Q. Xu, "Smart Cleaner: A New Autonomous Indoor Disinfection Robot for Combating the COVID-19 Pandemic," *Robotics*, vol. 10, no. 3, 2021.

[24] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, 3 Dec. 2015.

[25] A. G. Sanchez and W. D. Smart, *Surface Disinfection using Ultraviolet Lightwith a Mobile Manipulation Robot*, 2021.

[26] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," *European Conference on Computer Vision (ECCV)*, pp. 746–760, 2012.

[27] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: A versatile visual slam framework," *Proceedings of the 27th ACM International Conference on Multimedia (ACM MM)*, pp. 2292–2295, 2019.

[28] L Tiseni, D Chiaradia, M Gabardi, M Solazzi, D Leonardis, and A Frisoli, "UV-C Mobile Robots with Optimized Path Planning: Algorithm Design and On-Field Measurements to Improve Surface Disinfection Against SARS-CoV-2," *IEEE Robotics Automation Magazine*, vol. 28, no. 1, pp. 59–70, Mar. 2021.

[29] A Vyshnavi, A Manasa, C. Hamsika, S. Sai, and P Shalini, "UV Disinfection Robot with Automatic Switching on Human Detection," *EAI Endorsed Transactions on Internet of Things*, vol. 6, no. 23, 2020.

[30] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *The International Journal of Robotics Research*, vol. 35, pp. 1697–1716, 14 2016.

[31] Y. Xiang and D. Fox, "Da-rnn: Semantic mapping with data associated recurrent neural networks," in *Robotics: Science and Systems (RSS)*, Jul. 2017.

[32] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," *arXiv preprint arXiv:1608.05442*, 2016.

Appendix

A. High-touch Mapping from Classes in the ADE20K dataset and the ScanNet dataset

As discussed in the perception system section, we create a high-touch mapping from the 150 classes in the ADE20K dataset and the 40 NYU40 classes in the ScanNet dataset.

Categories in ADE20K that are mapped to be 'high-touch': bed; cabinet; door; table; chair; sofa; shelf; armchair; seat; desk; wardrobe; lamp; railing; cushion; chest; counter; refrigerator; pillow; bookcase; countertop; kitchen island; swivel chair; towel; pole; bannister; stool; cradle; blanket; tray; plate.

Categories in ScanNet that are mapped to be 'high-touch': bookshelf; counter; desk; shelves; dresser; pillow; refrigerator; towel; cabinet; night stand; lamp; otherfurniture; bed; otherprop; chair; sofa; table; door;

All other categories and unannotated background are mapped to 'low-touch'. We acknowledge that our partition is not perfect and there are controversial categories (e.g., whether A.C. outlets are high-touch), but we believe such problem is out of the scope of this paper and does not affect the novelty of our technique.