Online Diffusion-Based 3D Occupancy Prediction at the Frontier with Probabilistic Map Reconciliation

Alec Reed, Lorin Achey, Brendan Crowe, Bradley Hayes, Christoffer Heckman

***Abstract*— Autonomous navigation and exploration in unmapped environments remains a significant challenge in robotics due to the difficulty robots face in making commonsense inference of unobserved geometries. Recent advancements have demonstrated that generative modeling techniques, particularly diffusion models, can enable systems to infer these geometries from partial observation. In this work, we present implementation details and results for real-time, online occupancy prediction using a modified diffusion model. By removing attention-based visual conditioning and visual feature extraction components, we achieve a 73% reduction in runtime with minimal accuracy reduction. These modifications enable occupancy prediction across the entire map, rather than limiting it to the area around the robot where sensor data can be collected. We introduce a probabilistic update method for merging predicted occupancy data into running occupancy maps, resulting in a 71% improvement in predicting occupancy at map frontiers compared to previous methods. Finally, our code and a ROS node for on-robot operation can be found on our website: https://arpg.github.io/scenesense/.**

I. INTRODUCTION

Robots typically make decisions based only on the space they have directly observed, either during the current deployment or a previous one. In environments where no prior information is available, an autonomous system must rely solely on real-time observations. This limitation poses significant challenges for navigation, as perception sensors have restricted fields of view and can be obstructed by obstacles. As a result, data products such as 2D or 3D geometric maps often contain gaps in areas the sensors could not reach — especially during runtime while the system is actively exploring.

While there are existing methods for filling these gaps, most focus on filling LIDAR shadows [1] or gaps in the map [2, 3] where the geometry around the target area has been observed. To further enhance robotic decision making, we not only need to fill holes and gaps in the map, but also extend map geometries beyond what can be directly measured.

Occupancy prediction enables robots to infer and extend maps beyond sensor-observed areas, addressing gaps in real-time perception. While [4] demonstrates that such models generate plausible predictions, their range-limited applicability motivates prioritizing frontiers, critical regions where observed voxels transition to unobserved space, as targets for extending map coverage beyond immediate sensor reach.

This work was partially supported by the National Science Foundation award #2339328.

All authors are affiliated with Department of Computer Science, University of Colorado Boulder, USA. E-mail addresses for authors are: FirstName.LastName@colorado.edu

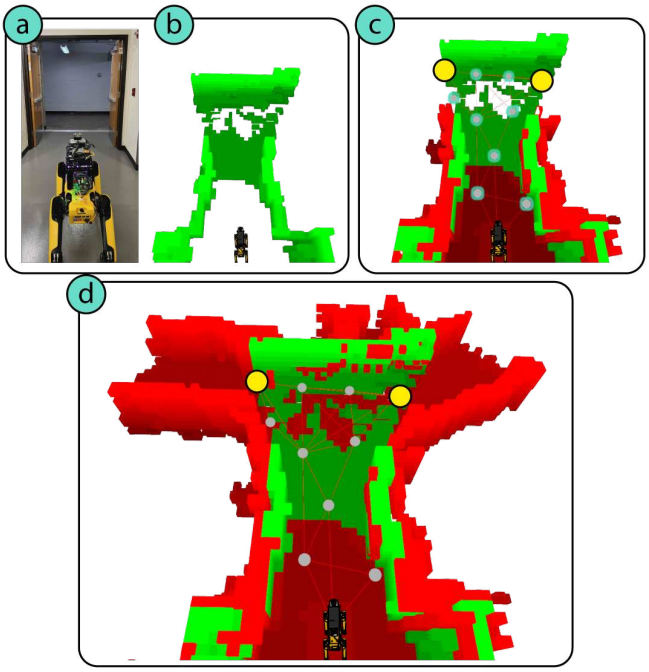


Fig. 1: **Onboard Occupancy Prediction and Map Merging**: Green voxels represent observed occupancy and red voxels represent predicted occupancy. Gray graph points represent vertices and yellow graph points represent vertices identified as frontier points. **(a)** Spot platform is positioned in front of a t-intersection at startup as shown in the photo of the scene. **(b)** The map is populated with the observed 3D occupancy data from the LIDAR sensor. **(c)** Robot-centric (RC) occupancy prediction runs to predict occupancy data around the robot. Then a graph is built over the space to identify frontiers of interest for frontier-centric (FC) occupancy prediction. **(d)** Finally the diffusion model predicts the occupancy around the frontier points. These predicted maps are merged into the running map using our probabilistic map update rule.

In this paper, we introduce key modifications to the diffusion-based SceneSense occupancy prediction model [4] to enable predictions at any location in the running map while significantly reducing inference time for online occupancy prediction. We propose a probabilistic map update rule to integrate occupancy predictions with the "observation-only" map generated from LIDAR data. Finally, we implement a graph-based frontier evaluation method to identify optimal regions for occupancy prediction and validate our approach on a real-world robotic system. Our key contributions include:

1) 73% reduction in end-to-end runtime for the Scene-

Sense occupancy prediction model.
2) Expanded prediction capability, allowing occupancy predictions at any location in the map rather than being limited to robot-centric predictions.
3) 75% improvement in frontier occupancy map evaluation compared to the observation-only map.
4) 71% improvement in frontier occupancy map evaluation compared to the one-shot map merging method from the original SceneSense work [4].

II. RELATED WORK

A. Occupancy Prediction

One solution to the challenge of autonomous navigation in occluded environments is to predict occupancy distributions. Deep Learning (DL) approaches have shown promise, but existing methods struggle with scalability, generalization, and handling occluded areas. Wang et al. [5] propose a DL approach to predict occupancy distribution which involves selectively removing data from the Matterport3D [6] dataset during training for the model to learn occluded geometries. This biases the model to predict occupancy for these specific types of occlusions and does not scale well to large unseen sections of an environment. In [7], the authors present a self-supervised method for 3D occupancy prediction using video sequences, which transforms 2D images into 3D representations with deformable attention layers. While effective with nearby cameras, it struggles to predict occupancy beyond the camera's view due to signed distance field's (SDF) limitations in managing occluded geometry.

More recently, diffusion models were shown to successfully generate occupancy predictions behind occluded geometries in indoor environments using a single RGBD sensor mounted on a mobile robot platform [4]. Our research advances this method by introducing a novel, more efficient approach to occupancy prediction, demonstrated on real hardware.

B. Scene Synthesis

Diffusion models [8, 9], are a popular tool that has demonstrated impressive generative results across image [10], video [11], and natural language [12]. Building on these successes, diffusion models are being extended to 3D scene and shape generation. Recent work [13] demonstrates the use of diffusion models for 3D point cloud generation for simple shapes and objects (e.g. tables, chairs). Kim et al. [14] show successful 3D shape generation from 2D content such as images, and Vahdat et al. [15] demonstrate similar 3D shape generation but using point cloud datasets rather than images. In LegoNet [16], diffusion models are used to propose object rearrangements in a 3D scene. In DiffuScene [17], a denoising diffusion model is used with text conditioning to generate 3D indoor scenes from sets of unordered object attributes. Unlike these previous works which primarily focus on generating simple shapes, rearranging objects, or creating indoor scenes, our approach leverages diffusion models to fuse generated terrain with measurements from the local robot field of view, thereby bridging the gap between 3D scene generation and practical, situated robotics applications.

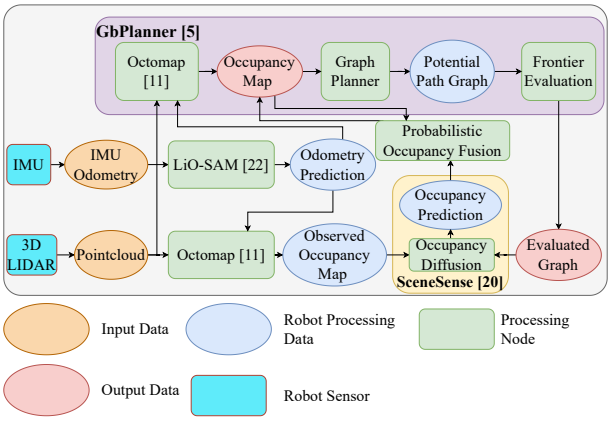


Fig. 2: **System Block Diagram**: Block diagram showing the system design for onboard SceneSense occupancy prediction. The system is comprised of an IMU and LIDAR sensor to generate odometry and occupancy maps. Once the occupancy map is built, a graph is constructed to evaluate frontier points for occupancy prediction. Local occupancy is then sub-selected around these points and sent to the SceneSense framework that provides occupancy predictions. These predictions are then merged with the running occupancy map using the probabilistic update rule.

III. METHODS

A. Problem Definition

Frontier Identification and Evaluation. Let $\mathbb{M}$ be the current occupancy map, built via measurements from an onboard sensor $\mathbb{S}$ and odometer measurements $\mathbb{O}$. The map consists of voxels m that are categorized as $m \in \mathbb{M}_{free}$, $m \in \mathbb{M}_{occupied}$, or $m \in \mathbb{M}_{unknown}$, which represent free, occupied and unknown space respectively. We identify frontiers as regions where $\mathbb{M}_{free}$ or $\mathbb{M}_{occupied}$ transitions to $\mathbb{M}_{unknown}$. In particular, "interesting" frontiers will maximize the number of unknown voxels available for occupancy prediction while considering common exploration metrics such a directionality, distance from target, and reachability [18, 19].

Dense Occupancy Prediction. Dense occupancy prediction predicts the occupancy from $[0, 1]$ where 0 is unoccupied and 1 is occupied for every voxel m in a target region $x \subseteq \mathbb{M}$.

B. Robotic System Architecture

Our robotic system consists of a quadruped (Spot), as shown in Figure 1, along with an off-board high performance computer to handle computationally expensive requests. A block diagram of the system is shown in Figure 2.

Sensor Suite The equipped sensor suite was designed with the purpose of providing 3D point cloud information and sufficient data for accurate localization. The primary sensor in the spot sensor suite is the Ouster OS1-64 LIDAR which provides 3D point clouds for mapping and localization. In addition a LORD Microstrain 3DM-GX5-15 IMU is used to measure the linear and angular acceleration of the Ouster,

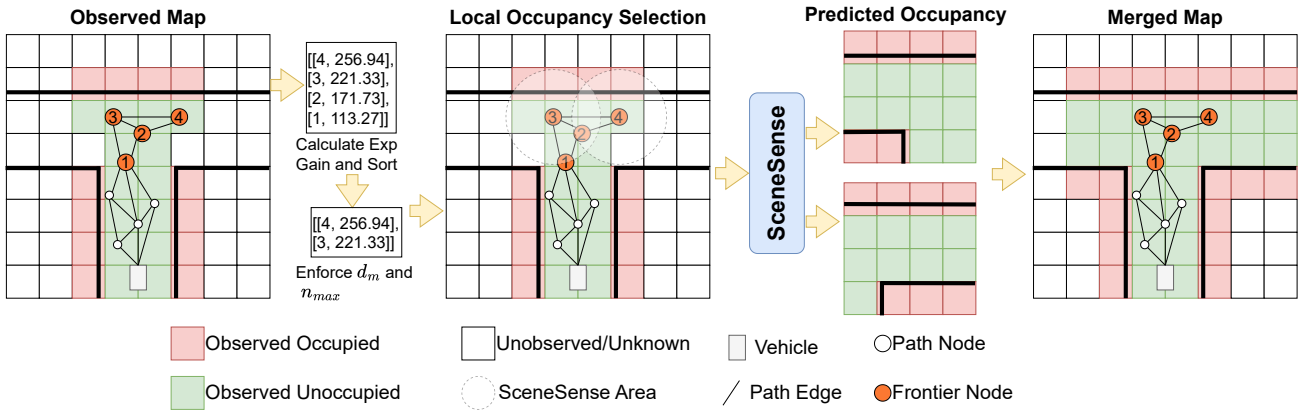


Fig. 3: **Map Merging Example**: Process for generating and merging occupancy predictions with an observed map. A graph is generated and evaluated to identify frontier nodes. Then, the frontier nodes are sorted by exploration gain as defined in Eq. 1, and d_m (min node spacing) and n_{max} (max frontier prediction nodes) are enforced on the frontier points set. For each frontier point identified for occupancy prediction, local occupancy is selected from the observed map and sent to SceneSense for occupancy prediction. Finally, the predicted maps are merged into the running occupancy map using Eq. 2.

for use in the localization system.

Localization. Localization is required for Spot to perform volumetric mapping. We have implemented the popular LIDAR-based localization method LiO-SAM [20] to provide localization at run-time.

Occupancy Prediction. We adopt the SceneSense occupancy prediction diffusion model [4] with modifications to enable novel functionality and improve performance at runtime. Originally, SceneSense was designed as a conditional diffusion model where the conditioning was RGBD data from a camera/depth sensor on the robot. However, ablation studies of the model show that including this RGBD conditioning has very little performance impact when *occupancy inpainting* is enabled [4]. By removing the RGBD conditioning we enable two key characteristics for the model; anywhere occupancy prediction and increased inference speed.

Removing the RGBD conditioning data obviates the need to center occupancy predictions at the robot. By removing the need for image conditioning, occupancy can be predicted anywhere in the observed map, allowing for occupancy predictions at range. Secondly, we can replace the cross-attention based noise prediction model with the equivalent unconditional model. This reduces the number of trainable parameters for similar unconditioned performance. Further, this change also removes the need for a feature extraction backbone, saving additional computation time.

Frontier Identification and Evaluation. With these modifications to the occupancy prediction framework we can generate occupancy predictions at any point in the map. To identify interesting areas for prediction we adopt the graph-based exploration planner GBPlanner [19]. GBPlanner builds a graph where nodes are potential exploration points and edges are feasible paths to navigate from node to node. A ray casting algorithm is run at each node in the graph to quantify the number of observed, free, and unknown voxels in that node's field of view. After finding the shortest path to each node the *Exploration Gain* can be calculated for each node in the graph as follows:

$$\textbf{ExplorationGain}(\sigma_i) = e^{-\gamma_{\mathcal{S}}\mathcal{S}(\sigma_i,\sigma_{exp})} \cdot \sum_{j=1}^{m_i} \textbf{VolumetricGain}(v_j^i) e^{-\gamma_{\mathcal{D}}\mathcal{D}(v_1^i,v_j^i)}, \quad (1)$$

where $\mathcal{S}(\sigma_i, \sigma_{exp})$, $\mathcal{D}(v_1^i, v_j^i)$ are weight functions with tunable factors $\gamma_{\mathcal{S}}, \gamma_{\mathcal{D}} > 0$ respectively, and m_i is the number of vertices in the path. $\mathcal{D}(v_1^i, v_j^i)$ is the cumulative Euclidean distance from a vertex v_j^i to the root v_1^i along a path γ_i.

To improve exploration efficiency, exploration gain is used to rank nodes for occupancy prediction, prioritizing locations that help complete unmapped areas where occupancy cannot be directly observed. The highest-ranked nodes are often positioned far from the robot, potentially beyond its sensor field of view, providing occupancy predictions at a considerable distance from its current location. Given a minimum node spacing d_m and a maximum number of frontier prediction nodes n_{max}, SceneSense generates occupancy predictions at this extended range, centered around the identified frontiers as shown in Figure 3.

Mapping. The probabilistic volumetric mapping method OctoMap [21] is selected as the mapping framework. OctoMap is adopted for its log-odds update method for predicting occupied and unoccupied cells. This approach allows for computationally efficient fusion of observed occupancy and predicted occupancy maps. Further discussion on map fusion is provided in Section III-C.

C. Probabilistic Map Merging

In previous work, predicted occupancy was merged into the running occupancy map in a "fire and forget" approach [4]. A given occupancy prediction was temporarily merged into the running occupancy map by setting the predicted

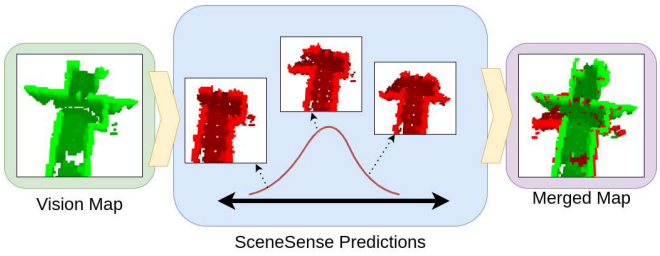


Fig. 4: **Multi-Prediction Occupancy Merging**: SceneSense predicts various occupancy maps based on equivalent input data that form a distribution. This distribution forms a curve where more likely predictions occur more often, and less likely predictions occur infrequently. These predictions are fused into the merged map using Eq. 2. The resulting merged map naturally filters out the unlikely voxel predictions, forming an extended occupancy map.

occupied cells to 1. Then, when a new occupancy prediction was generated, the previous prediction would be removed from the running map and the new prediction would be merged in the same way. While this approach is effective in some applications, it limits the ability to accurately maintain a coherent and continuous understanding of the environment. To address these issues, we modify the probabilistic occupancy update formula presented for the OctoMap mapping framework [21].

We define the probability that a voxel m is occupied given an occupancy prediction d_t or sensor reading z_t as $P(m|d_t)$ or $P(m|z_t)$ respectively. The set of sensor estimates $z_{1:t}$ and diffusion estimates $d_{1:t}$ populate the joint set $\{z_{1:t}, d_{1:t}\}$ which we denote as $j_{1:t}$. As discussed in [4], SceneSense only operates on voxels m that are not contained in the observed set $\mathcal{O}$, where $z_{t:t-1} = \varnothing$, and therefore $P(m|j_{1:t})$ will never require an update given $P(m|d_t)$ and $P(m|z_t)$ at the same time. As such we generate the piece-wise update rule for merging diffusion into the running occupancy map.

$$P(m \mid j_{1:t}) = \begin{cases} \left[1 + \frac{1-P(m|d_t)}{P(m|d_t)} \frac{1-P(m|j_{1:t-1})}{P(m|j_{1:t-1})} \frac{P(m)}{1-P(m)}\right]^{-1} & \text{if } m \notin \mathcal{O} \\ \left[1 + \frac{1-P(m|z_t)}{P(m|z_t)} \frac{1-P(m|j_{1:t-1})}{P(m|j_{1:t-1})} \frac{P(m)}{1-P(m)}\right]^{-1} & \text{if } m \in \mathcal{O}. \end{cases} \tag{2}$$

In this framework $P(m|z_t)$ and $P(m|d_t)$ can be configured to different values prior to runtime. Generally $P(m|d_t)$ given a predicted occupied cell is set lower than $P(m|z_t)$ given a sensed occupied cell, as we trust the sensor more than our generative model. By using this probabilistic approach to map merging, the final merged map benefits from prediction persistence as the system explores as well as increased map fidelity due to multi-prediction occupancy refinement.

Multi-Prediction Occupancy Refinement. As shown in Figure 4, the occupancy predictions generated by SceneSense can be unique even given the same conditioning information. Similar to image generation tasks it is desirable for SceneSense to generate various realistic results given the same input data [22, 10]. Given this behavior, we can collect various predictions from the same location forming a distribution. Then we can aggregate the predictions using the probabilistic update rule defined in Eq. 2 and generate a map constructed from the distribution. The resulting merged map will naturally filter out outlier occupancy predictions and result in only the most probable voxels maintaining occupancy in the final merged map.

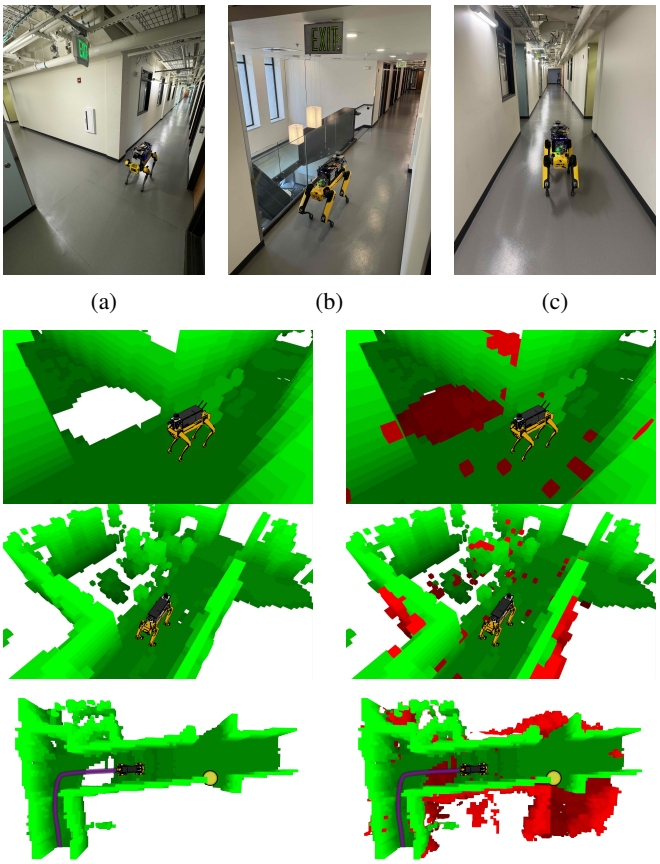


Fig. 5: **Example Occupancy Predictions**: Scene images at the top of the figure correspond to the 3 pairs of occupancy maps, where (a) corresponds to the top pair of occupancy maps. The left column of occupancy maps shows the observation-only map, while the right column shows the merged observation and prediction maps. **(a)** Spot approaches a hallway corner and given the LIDAR mounting position cannot observe the floor after entering the hall junction. SceneSense is able to fill the floor and the missing wall information that was not observed. **(b)** Spot navigates down a hallway and enters an area with a glass railing above the stairs. SceneSense does not fill the open space, where algorithms like hole filling or normal ground expansion may fail. **(c)**. Spot navigates down a hallway generating predictions along the way. Spot's trajectory is shown in purple, and the identified frontier point is shown in yellow. Beyond providing predictions for the areas that have already been observed, SceneSense generates a frontier prediction at the 4-way intersection. This prediction shows the left side to be a dead-end, while a hallway or entryway is predicted on the right. In reality, these halls are really classrooms, where doors may be open or closed to allow for robot traversal.

Observed vs. Predicted Voxels. SceneSense uses observed data (observed occupied and observed unoccupied)

for occupancy inpainting during diffusion. In this paradigm SceneSense will never modify observed cells, and only make occupancy prediction in unobserved space. In Eq. 2 if a voxel has not been observed, SceneSense will generate occupancy predictions and update the voxel given the update rule. If that voxel is later observed, the previous $P(m \mid j_{1:t})$ is used to calculate the probability of occupancy given $P(n \mid z_t)$. In practice it is often the case that the user trusts the LIDAR sensor more than the SceneSense predictions and therefore would configure $P(m \mid d_t) < P(m \mid z_t)$. This means that when a voxel that has been previously populated by SceneSense is directly observed $P(m \mid j_{1:t})$ it will more quickly be updated to reflect the occupancy observed by sensor z_t.

Furthermore this approach ensures that SceneSense will never overwrite direct observations. Observed occupancy information (occupied information from LIDAR hits, and unoccupied information calculated from ray casting) is mapped into the diffusion process at every timestep t to perform occupancy inpainting. Therefore any observations, whether those observations be occupied or unoccupied are maintained and guaranteed to persist through the diffusion process.

IV. EXPERIMENTS AND RESULTS

Training and Implementation. To train SceneSense we collected real-world occupancy maps from various buildings. We gathered approximately 1 hour worth of occupancy data, resulting in $11,296$ unique poses with associated complete local occupancy maps. Any areas that were used to train the model are omitted from the results presented here.

We implement the same denoising network structure presented in [4]. It is a U-net constructed from the HuggingFace Diffusers library of blocks [23] and consists of Resnet [24] downsampling/upsampling blocks. The diffusion model is trained using randomly shuffled pairs of ground truth local occupancy maps x. We use Chameleon cloud computing resources [25] to train our model on one A100 with a batch size of 32 for 250 epochs or $88,208$ training steps. We use a cosine learning rate scheduler with a 500 step warm up from 10^{-6} to 10^{-4}. The noise scheduler for diffusion is set to $1,000$ noise steps.

At inference time we evaluate our dataset using an RTX 4070 TI Super GPU for acceleration. The number of diffusion steps is configured to 30 steps.

A. Inference time

We evaluate the inference time of the unconditional diffusion model against the inference time of the conditional model presented in the original SceneSense paper [4]. The cross-attention enabling trainable parameters are removed for the unconditional model, but the number of output channels for the constructed U-net are held constant between both models. As the ablation results of the original paper show minor, or no performance gain between the conditional and unconditional model in this configuration we do not evaluate the results of the model predictions in these experiments.

TABLE I: Inference time and model size results. "Full inference" and "end-to-end" evaluations are computed using 30 diffusion steps.

	Cond. Model [4]	Uncond. Model
Trainable Params	141,125,261	**101,144,845**
Diffusion Step (s)	0.03707	**0.0147**
Full Inference (s)	1.11	**0.4437**
Backbone (s)	.55099	N.A.
End-to-end (s)	1.66	**0.4437**

Discussion. As shown in Table I, removing the conditioning from the diffusion model reduces the computation requirements substantially. The unconditioned model reduces the number of trainable parameters by 28%, the model inference time by 60% and the end to end computation time by 73%. These improvements enable SceneSense to operate in real-time more effectively, allowing for more flexible implementations for onboard robotic applications.

B. SceneSense Generative Occupancy Evaluation

For the following experiments we evaluated the occupancy generation capabilities of SceneSense onboard a real world robot in 2 unique test environments. In particular, we examine the fidelity of predictions around the robot with predictions at the frontiers of the map, ablating the map update methods and the running sensor only map.

Experimental Setup. SceneSense predictions are evaluated in 2 environments. Environment 1 was a long hallway with cutouts for classrooms and 1 right turn. Select frames shown in figure 5a and 5c are from Environment 1. Environment 2 consists of similar carpeted area with 4 hallways and 4 turns, forming a square shape. We evaluate the occupancy prediction framework using the following test configurations.

1) **Baseline or SceneSense**: A comparison between OctoMap sensor only local occupancy (BL) with the SceneSense occupancy prediction included (SS).
2) **Robot-centric or Frontier-centric**: Robot-centric diffusion (RC) predicts occupancy at a radius of $3.3m$ about the robot while frontier-centric diffusion (FC) predicts occupancy at a radius of $3.3m$ at an identified location in the map, which has a maximum range of $7m$ from the robot.
3) **One Shot Map Merging or Probabilistic Map Merging**: One shot map merging (OSMM) simply takes the current local occupancy map and a SceneSense occupancy prediction and populates the predicted occupancy information in the running map. Probabilistic map merging (PMM) keeps a running local merged occupancy map that uses update equation 2 to update the occupancy map for every new occupancy prediction. In practice, each pose will receive 3-5 SceneSense predictions to merge into the running map.

Occupancy Prediction Metrics. Following similar generative scene synthesis approaches [17, 26] we employ the Fréchet inception distance (FID) [27] and the Kernel inception distance [22] (KID $\times 1000$) to evaluate the generated

local occupancy grids using the clean-fid library [28]. Generating good metrics to evaluate generative frameworks is a difficult task [29]. FID and KID have become the standard metric for many generative methods due to their ability to score both accuracy of predicted results, and diversity or coverage of the results when compared to a set of ground truth data. While these metrics are fairly new to robotics, which traditionally evaluates occupancy data with metrics like accuracy, precision and IoU, these metrics have been shown to be an effective measure for evaluating predicted scenes [4, 17].

TABLE II: Results comparing running occupancy (BL) to occupancy enhanced with SceneSense prediction (SS). Evaluations of each method are provided as robot-centric generations (RC) and frontier-centric generations (FC).

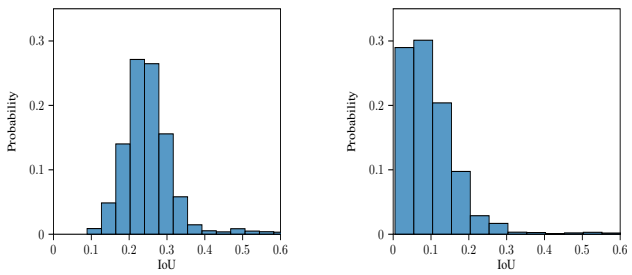
	Env. 1		Env. 2	
Method	FID ↓	KID×1000 ↓	FID ↓	KID×1000 ↓
BL-RC	36.0	16.4	30.3	16.3
SS-RC-OSMM	**26.3**	**7.7**	**20.8**	10.1
SS-RC-PMM	29.2	10.4	21.0	**9.1**
BL-FC	116.9	81.6	150.6	118.8
SS-FC-OSMM	104.2	66.3	133.4	104.4
SS-FC-PMM	**30.1**	**10.3**	**34.5**	**9.0**

Results Discussion. The results in Table II show that RC predictions are quite similar between OSMM and PMM approaches, reducing the FID of the environments by an average of 28.5% and 25% for OSMM and PMM respectively. These results are similar to the simulation-based results presented in [4]. However, the FC results show a much greater improvement for PMM, with an average FID reduction of 75%, compared to OSMM, which only achieves an average FID reduction of 11%.

Interestingly, The KID values are nearly identical between SS-RC-PMM and SS-FC-PMM, indicating that the model occupancy predictions at range are as reasonable as the predictions made around the robot, even though there is less information for the predictions at range. KID is known to be less sensitive to outliers when compared to FID [22]. It is likely that the unreasonable predictions that can occur when performing FC predictions are better filtered out by the KID metric, resulting in similar scores.

The large discrepancy between OSMM and PMM results when evaluated at frontiers is due to the sparsity of the occupancy map at the frontier. We predict that as the number of unknown voxels grow, so too does the distribution of predicted scenes. Intuitively, if there are no observed voxels to guide the prediction SceneSense will predict a wide variety of possible occupancy maps. On the other end, if all voxels in the target space are observed, the same occupancy map will be generated every time.

We can analyze the number of available voxels for occupancy prediction as the number of unknown voxels in the target area x_{rm} as a percentage of the total observed voxels in x_{gt}. Using the results from environment 2 to evaluate this prediction, we calculate that on average 59.18% of target area voxels are unknown when performing RC occupancy prediction. However, when performing FC prediction this number jumps to 70.79%. This result supports the intuitive statement that there are more available (unknown) voxels for prediction around the frontiers of the map than around the robot. To confirm that the increase in unknown voxels widens the distribution of occupancy prediction we generate a distribution of results by calculating the IoU of each prediction against all other predictions made during the run. The results of this approach as provided in Figure 6 show that RC predictions are more likely to be similar, while FC predictions are more likely to be dissimilar with very little overlap. When predictions are all similar, PMM becomes less important for accurate predictions, since OSMM would result in a similar map each time. However PMM is needed at range to achieve reasonable results since it can negotiate the wider distribution of possible occupancy predictions.



(a) **RC IoU PMF** (b) **FC IoU PMF**

Fig. 6: **Env 2 IoU Probability Mass Function (PMF)**. (a) IoU histogram of RC SceneSense predictions. (b) IoU histogram of FC SceneSense predictions. The IoU distributions show that RC occupancy predictions are more likely to be similar than FC predictions.

V. CONCLUSIONS

In this work we present key architectural changes to the SceneSense [4] occupancy prediction model to enable real-time occupancy inference at any location of interest in the map. Further we present our integration of SceneSense to a real robotic system, a method for probabilistically merging occupancy predictions into a running occupancy map, as well as evaluations of these occupancy predictions. Future work will explore how these predictions can be utilized to improve existing planning and exploration architectures.

REFERENCES

[1] David Doria and Richard J. Radke. "Filling large holes in LiDAR data by inpainting depth gradients". In: *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. 2012, pp. 65–72. DOI: 10.1109/CVPRW.2012.6238916.

[2] Yan Xu et al. *Depth Completion from Sparse LiDAR Data with Depth-Normal Constraints*. 2019. arXiv: 1910.06727 [cs.CV]. URL: https://arxiv.org/abs/1910.06727.

[3] Ran Cheng et al. *S3CNet: A Sparse Semantic Scene Completion Network for LiDAR Point Clouds*. 2020. arXiv: 2012.09242 [cs.CV]. URL: https://arxiv.org/abs/2012.09242.

[4] Alec Reed et al. *SceneSense: Diffusion Models for 3D Occupancy Synthesis from Partial Observation*. 2024. arXiv: 2403.11985 [cs.RO]. URL: https://arxiv.org/abs/2403.11985.

[5] Lizi Wang et al. "Learning-based 3d occupancy prediction for autonomous navigation in occluded environments". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 4509–4516.

[6] Angel Chang et al. "Matterport3d: Learning from rgb-d data in indoor environments". In: *arXiv preprint arXiv:1709.06158* (2017).

[7] Yuanhui Huang et al. "SelfOcc: Self-Supervised Vision-Based 3D Occupancy Prediction". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2024, pp. 19946–19956.

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[9] Jascha Sohl-Dickstein et al. "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International conference on machine learning*. PMLR. 2015, pp. 2256–2265.

[10] Robin Rombach et al. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[11] William Harvey et al. "Flexible diffusion modeling of long videos". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 27953–27965.

[12] Rongjie Huang et al. "Prodiff: Progressive fast diffusion model for high-quality text-to-speech". In: *Proceedings of the 30th ACM International Conference on Multimedia*. 2022, pp. 2595–2605.

[13] Shitong Luo and Wei Hu. "Diffusion probabilistic models for 3d point cloud generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 2837–2845.

[14] Seung Wook Kim et al. "Neuralfield-ldm: Scene generation with hierarchical latent diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 8496–8506.

[15] Arash Vahdat et al. "Lion: Latent point diffusion models for 3d shape generation". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 10021–10039.

[16] Qiuhong Anna Wei et al. "Lego-net: Learning regular rearrangements of objects in rooms". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 19037–19047.

[17] Jiapeng Tang et al. "Diffuscene: Scene graph denoising diffusion probabilistic model for generative indoor scene synthesis". In: *arXiv preprint arXiv:2303.14207* (2023).

[18] Harel Biggie et al. "Flexible supervised autonomy for exploration in subterranean environments". In: *arXiv preprint arXiv:2301.00771* (2023).

[19] Tung Dang et al. "Graph-based subterranean exploration path planning using aerial and legged robots". In: *Journal of Field Robotics* 37.8 (2020), pp. 1363–1388.

[20] Tixiao Shan et al. *LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping*. 2020. arXiv: 2007.00258 [cs.RO].

[21] Armin Hornung et al. "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees". In: *Autonomous Robots* (2013). Software available at https://octomap.github.io. DOI: 10.1007/s10514-012-9321-0. URL: https://octomap.github.io.

[22] Mikołaj Bińkowski et al. "Demystifying MMD GANs". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=r1lUOzWCW.

[23] Patrick von Platen et al. *Diffusers: State-of-the-art diffusion models*. https://github.com/huggingface/diffusers. 2022.

[24] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[25] Kate Keahey et al. "Lessons Learned from the Chameleon Testbed". In: *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20)*. USENIX Association, July 2020.

[26] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. "Sceneformer: Indoor scene generation with transformers". In: *2021 International Conference on 3D Vision (3DV)*. IEEE. 2021, pp. 106–115.

[27] Martin Heusel et al. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf.

[28] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. "On Aliased Resizing and Surprising Subtleties in GAN Evaluation". In: *CVPR*. 2022.

[29] Muhammad Ferjad Naeem et al. *Reliable Fidelity and Diversity Metrics for Generative Models*. 2020. arXiv: 2002.09797 [cs.CV].