Generative Occupancy Mapping for Enhanced Robotic Exploration

by

Alec Reed

B.S., Gonzaga University, 2017

M.S., University of Washington, 2019

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2024

Committee Members:

Christoffer Heckman, Chair

Bradley Hayes

Eric Frew

Timothy Chung

Xinke Deng

Reed, Alec (Ph.D., Computer Science)

Generative Occupancy Mapping for Enhanced Robotic Exploration

Thesis directed by Prof. Christoffer Heckman

Autonomous systems have long held the promise of replacing or assisting humans in a wide range of applications. Today autonomous systems operate effectively in challenging domains such as self-diving cars and trucks, search and rescue, and sidewalk navigation and delivery. However the classic perceive, plan, execute control loop limits the utility and intelligence of systems by requiring robots to make complete observations of the area they are planning over. As opposed to humans, who can make common-sense inference of geometry without direct observation, robots employ a suite of sensors or pre-loaded maps to generate complete observations of their operational environments. Without access to full map information operation can be slow and unintuitive. In this paper we aim to supplement existing robot sensing capabilities by leveraging recent advances diffusion models and generative AI to produce realistic predictions of unobserved space in running occupancy maps.

In this paper we present a transformer-based model for generative occupancy prediction and some of the restrictions of similar models for occupancy mapping. Then we adopt diffusion models for a means of realistic occupancy prediction and show that our model *SceneSense* generates better representations of local occupancy than just the running occupancy map built from sensor measurements. After making key modifications to the original model, we deploy SceneSense onboard a real-world robotic platform and show that SceneSense can be a "drop-in" improvement for existing planning and exploration stacks. We show that the SceneSense enhanced map increased both the rate of exploration and the consistency of exploration when compared to the same planning and explore stack informed by just sensor measurements.

Dedication

I dedicate my dissertation to my loving wife Madison, whose love and support made this entire experience possible. This Ph.D has been an incredible exploration of my interests and without her this dream would still be just that.

Acknowledgements

I would like to first and foremost thank my advisor Christoffer Heckman for his guidance throughout my studies. As I began my Ph.D with very little research experience, Chris maintained an environment where I could learn, grow and, most importantly, take risks. I would also like to acknowledge Bradley Hayes for his invaluable advice and expertise, which was happily provided whenever requested. Finally I would like to thank all my labmates in the Autonomous Perceptions and Robotics Group (ARPG), as well as my colleagues and friends in our neighboring labs, in particular the Collaborative AI and Robotics (CAIRO) Laboratory, Human Interaction and Robotics (HIRO) Group, and the Programming Languages and Verification (CUPLV) Group.

I would additionally like to acknowledge the funding provided by the following sources: NSF Award #1932189, NSF Award #2339328, and the Belgian American Education Foundation (BAEF).

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

In general, autonomous systems are designed to operate over space that has been directly observed. In this paradigm, robots are dependent on sensors to act in previously unobserved space. This approach encourage system to maximize sensor FOV, often with a suite of multi-modal sensors such as lidar, camera, stereo, and radar. These large-scale sensor rigs introduce additional maintenance and cost over the life of the system, and processing of the received data can be computationally expensive. Further when sensors fail to perceive the environment due to obstructions or visual degradation, systems can behave unintuitively, or in some cases fail and require user intervention.

One way to alleviate some of these problems in semi-static environments by operating over previously build maps of the target area while localizing the system in the existing map. This approach has been adopted in fields such as self-driving cars [6, 24], mining [80] a Extraterrestrial exploration and mission planning [76] and has shown to enhance systems abilities to act effectively. While this approach does indeed reduce the system's reliance on sensed information, spacial pre-mapping introduces other issues into the navigation problem. spacial mapping is expensive, both in requiring multiple expert operated runs over the same target space as well as in data storage requirements, as maps are often data rich and can overrun allowed onboard storage [114]. For these reasons, collected and storage of maps for autonomous operations does not scale well to general adoption of autonomous systems [6].

While in some cases the argument against using existing maps as a basis for decision mak-

ing is based in cost and maintenance there are a number of operations where generating maps prior to operation is impossible. One such task where pre-existing maps cannot be collected is search and rescue operations. These operations require true autonomous perception and navigation in challenging environments. This challenge was the inspiration for the DARPA subterranean (SUBT) challenge [38] which hosted teams of academics and industry professionals from around the world to explore challenging unmapped environments and search for "artifacts" to score points. Teams achieved various levels of success in there autonomous operations, but traversal was challenging, with the winning team detecting around 50% of the target artifacts. The results achieved from this challenge reinforce the challenge of intelligent robot operation in previously unmapped environments.

Unlike modern robotic systems that rely on direct measurement of geometry to build plans and make decisions, humans rely extensively on 'common sense' inferences to engage successfully with the world. Humans' natural capability to logically extend geometry or terrain in familiar environments such as homes or offices allows for planning beyond direct observation. Enabling this common sense geometry extension onboard will only enhance existing planning and decision making methods, as well as open new research avenues for planning and control over predicted space. It is based on these observations and assertions that this work seeks to validate this thesis statement:

> *By generating unseen geometry from partial observation we can enhance autonomous system's ability to navigate in unknown environments, while maintaining high reliability.*

1.1 Contributions

The primary contributions of this work are the design and validation of the generative occupancy mapping model *SceneSense*. Our initial attempt at a terrain extension model is outline in Chapter 2. After adopting diffusion models for occupancy prediction we present the full SceneSense design and validation metrics in Chapter 3. SceneSense is diffusion model that generate realistic

occupancy prediction given partial observation, and was tested in a simulated environment generating local occupancy predictions within $2m$ of the robotic platform. In Chapter 4 we present key modifications to enhance the usefulness of the SceneSense model, as well as methods for merging the predicted occupancy maps with observed occupancy maps. Additionally in this chapter we present our implementation of SceneSense on a real robotic platform, and evaluations of the results in the real world in various test scenes. Finally in Chapter 5 we evaluate the performance of existing planners with SceneSense predictions augmenting the map in real time. We present key scenarios that are solved with a successful SceneSense implementation as well as metrics for the effectiveness of overall exploration and navigation over predicted occupancy maps.

1.2 Publications

The following publications were a result of the work conducted during the course of this doctoral degree:

[136] **Alec Reed**, Lorin Achey, Brendan Crowe, Bradley Hayes, and Chris Heckman. Online diffusion-based 3d occupancy prediction at the frontier with probabilistic map reconciliation. Submitted and Under Review, 2024.

[138] **Reed A**, Crowe B, Albin D, Achey L, Hayes B, Heckman C (2024). SceneSense: Diffusion Models for 3D Occupancy Synthesis from Partial Observation . 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2024.

[139] **Reed, A.**, Heckman, CR. Looking Around Corners: Generative Methods in Terrain Extension. Robotics Science and System (RSS) Workshop on Inference and Decision Making for Autonomous Vehicles 2023.

[137] **Reed, A.**, Berger, G. O., Sankaranarayanan, S., Heckman, C. (2023, March). Verified path following using neural control lyapunov functions. *In Conference on Robot Learning* (pp. 1949-1958). PMLR.

Chapter 2

Looking Around Corners: Generative Methods in Terrain Extension

2.1 Introduction

Autonomous exploration of previously unmapped space is a challenging task in field robotics due to the nature of unknown environments. Effective exploration algorithms are essential to successful autonomous search and rescue (SAR) deployments. In SAR scenarios, systems must be both reliable and fast to save as many lives as possible. As shown in the DARPA subterranean (SubT) challenge, robotic systems can be developed to reliably explore unknown spaces [14]. However these current implementations can be seen as slow, jittery, or unintuitive to a human supervisor.

While humans have the ability to use previous experience to infer terrain that may be occluded from view, robots generally rely directly on data received from onboard sensors such as lidar or cameras to develop exploration plans. This reliance on direct measurement can become a problem when approaching common terrain features such as turning hallways or T-intersections. In these cases, a frontier finding robotic system will only plan as far as it can see. For example, in the case of a T-intersection, the system generally will advancing to intersection before pausing to process and plan over the new information gained from its sensors. To alleviate this flood of new information that results in system pauses, we propose a method for predicting the terrain that may be occluded from view. By providing accurate estimates of the terrain geometry the speed at which autonomous systems explore unseen environments can be increased while maintaining high reliability.

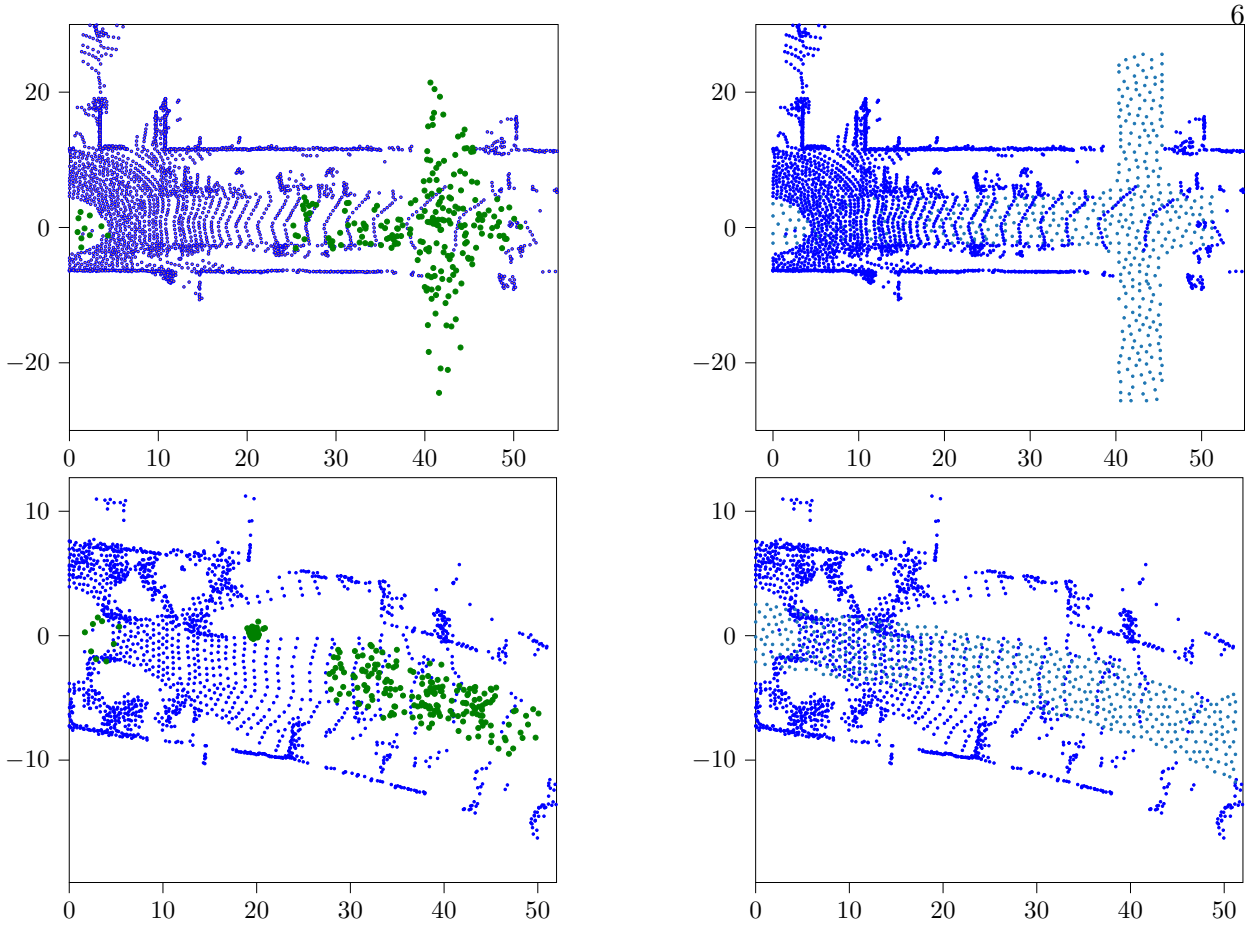


Figure 2.1: High level view of the terrain extension framework. The full lidar inputs are down sampled using the FPS down sampling method [100]. The point cloud is then further downsampled using a DGCNN [180] in conjunction with the FPS downsampling method. The remaining points called **point proxies** [202] retain geometric information from the DGCNN [180] and are passed to a geometry-aware transformer encoder. The one-dimensional encoder outputs are then passed through an MLP to raise the dimensionality to 3 which provides the predicted missing output points. Finally, the missing points are concatenated with the input to generate the final terrain enhanced image

2.2 Related Works

While an dedicated terrain extension framework has not been created to our knowledge, similar works in the fields of point cloud completion (PCC) and semantic scene completion (SSC) seek to

generate complete point clouds or maps given partial input data.

At first glance, SSC seems to be more applicable to our tasks, since it is generating full maps from input lidar scans. When looking at models such as S3CNet [34] which is a SSC implementation that scores well on the outdoor SemanticKitti dataset [10], it is found that the point cloud generation is very similar to PCC methods. In both methods, the features are first from the input point cloud. Then, the input is passed through an encoder/decoder architecture to generate output voxels/points where the model predicts the missing data to be. While most PCC methods end here, SSC methods generally attemt to fill in gaps in the 3D prediction as well as refine the semantic label predictions, however this step is not yet applicable to our problem. SSC is a relatively new field with few models to look for inspiration. However PCC is a well established field affording many models to build upon.

As discussed above, there are a wide variety of PCC implementations but all seek to complete a point cloud of an objects given some partial point cloud input [56]. The most interesting approach for our purposes is the transformer [172] based approach PoinTr by [202]. This method drastically improves the performance compared to other PCC models by making 2 primary adjustments:

(1) Unlike PCN [204] and many other implementations where a single feature vector is extracted for the full point cloud. PoinTr generates **point proxies** using a DGCNN [180] to generate feature vectors per point for a down-sampled point cloud. This alleviates both the compute requirements associated with transformer encoders and provides point features for better decoder interpretation.

(2) PoinTr [202] uses a geometry-aware transformer encoder decoder architecture rather than an MLP. As seen recently in many applications [63, 133, 52], the change to a transformer architecture results in a substantial performance boost due to the multi-head attention mechanisms. Providing geometry context to the transformer further increases the performance of the model.

Given the recent success of transformers in other fields it is desirable to explore transformers as

the backbone of our terrain extension framework. Additionally, transformers have been shown to be powerful tools for generative AI [125, 49]. By designing this terrain extension framework using transformers it allows for the flexibility of a future, fully generative terrain extension implementation.

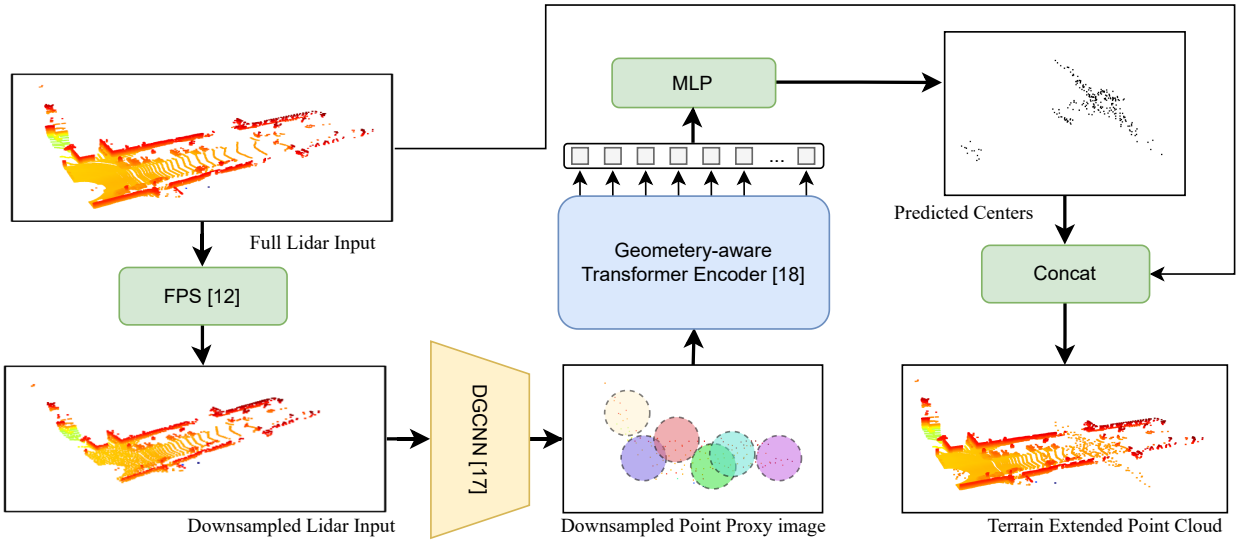


Figure 2.2: High level view of the terrain extension framework. The full lidar inputs are down sampled using the FPS down sampling method [100]. The point cloud is then further downsampled using a DGCNN [180] in conjunction with the FPS downsampling method. The remaining points called **point proxies** [202] retain geometric information from the DGCNN [180] and are passed to a geometry-aware transformer encoder. The one-dimensional encoder outputs are then passed through an MLP to raise the dimensionality to 3 which provides the predicted missing output points. Finally, the missing points are concatenated with the input to generate the final terrain enhanced image.

2.3 Method

Our framework is based on the PoinTr architecture developed by [202]. As shown in Figure 2.2 our terrain extension framework treats the problem of generating terrain as a point cloud completion

(PCC) problem. It is well known that the computational complexity of a transformer encoder grows quadratically with the size of the inputs. Therefore, we must take steps to reduce the input size before utilizing our transformer-based framework. Upon reception of a point cloud scan, the scan is first downsampled using furthest point sampling [100]. Then the input point cloud is passed through a DGCNN [180] to further downsample the input, while maintaining a feature vector in the neighborhood of the final downsampled center points. As defined in [202] we will refer to these DGCNN output points and associated feature vectors as **point proxies**. After the point proxies are generated they are passed to the geometry-aware transformer encoder which generates a $M \times 1$ array of outputs that will become the predicted output point cloud. The encoder outputs are passed through a linear projection layer similar to [204] to generate $M \times 3$ dimensional features that are reshaped to be the final output coordinates. Since these outputs are intended to fill in missing points in the lidar image, they are then concatenated with the input point cloud to produce a final terrain extended point cloud.

2.3.1 Training Data

The training data is developed to achieve the target output of "outpainting" in the space that is occluded or out of range of the input lidar scan. As shown in Figure 2.3 the SemanticKitti dataset [10] contains both labeled input point clouds and labeled voxel groundtruth data. To find the target terrain for prediction, the traversable terrain (road) of both the input point cloud and the ground truth voxel grid is isolated from the rest of the scene. Then the centroids of each ground truth voxel are used to generate the complete road ground truth point cloud. To utilize the limited output size of the transformer network, it is desirable to generate only output points where they do not already exist in the input. Given the input road points X and the full ground truth road points G we simply find $Y \leftarrow G \backslash X$. To generate Y, some distance d_y is defined to create a boundary around the input point cloud where the target output cannot exist. An algorithm such as KNN [128] can be used to generate ground truth data to ensure that this buffer region is not violated. The result of this target output generation is shown in Figure 2.3.

In addition to generating the target point cloud output, we also need to generate target output masks to check if outputs are contained within the clusters of target output data. While there are many machine learning methods for unsupervised data clustering such as mean shift [20] or DBScan [53], we find these methods to be fairly inconsistent in identifying and clustering ground truth points. The most successful method we find to generate target masks is the vision-based method of Segment Anything (SA) by [92]. Given that Segment Anything is a vision-based approach, it does not suffer from the same issues as those seen in traditional clustering methods as the shape of the data changes. Additionally, since the SA masks are calculated in pixel space, it is straightforward to check if a point in real space is contained within the target output mask in pixel space.

2.3.2 Loss Function

The loss function for this framework prioritizes the following:

(1) Distance from each predicted points to the target points.

(2) Distance from each target point to the predicted points.

(3) predicted points being contained within the ground truth cluster.

To address the first two loss targets for our model, we can use the symmetrical chamfer distance (CD) metric [55]:

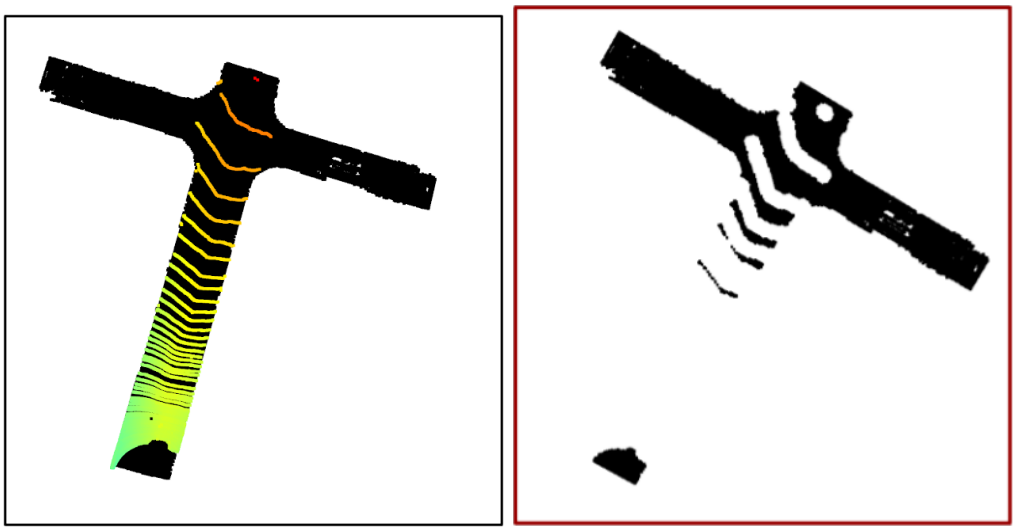
$$d_{cd}(\mathcal{P}, \mathcal{G}) = \frac{1}{\mathcal{P}} \sum_{p \in \mathcal{P}} \min_{g \in \mathcal{G}} ||p - g|| + \frac{1}{\mathcal{G}} \sum_{g \in \mathcal{G}} \min_{p \in \mathcal{P}} ||g - p||, \tag{2.1}$$

where $\mathcal{P}$ is the set of predicted points and $\mathcal{G}$ is the set of ground truth points. This metric is popular in point cloud completion methods as it balances the output points being located near the ground truth data and the ground truth data having an output point near each data point.
However, since this metric is strictly distance-based, points can be predicted outside of the target areas without additional penalty. To penalize the model against generating predictions outside of the target area, we implement a cost multiplier using the generated masks discussed in Section

2.3.1. Points predicted outside the target area will incur a penalty that multiplies the associated d_{cd} term by δ. We generate an output matrix Δ which is multiplied by the calculated matrix d_{cd}. This is formalized as:

$$C(\mathcal{P}, \mathcal{G}) = \Delta d_{cd}(\mathcal{P}, \mathcal{G}). \tag{2.2}$$

With this cost function $\mathcal{C}$ the goals discussed previously are all addressed, however tuning constants may be necessary to prioritize some goals of the function. Additional terms may be required to encourage some behaviors, such as area coverage, or distance between output points.



(a) (b)

Figure 2.3: (a) Shows the road ground truth (black) overlayed with the measured lidar point cloud (color). 9b) Shows the final ground truth after running the input removal algorithm discussed in Section 2.3.1 where $d_y = 1m$.

2.4 Results and Discussion

The initial results presented in Figure 2.1 are promising as the terrain extension framework is able to learn areas of high value to place the predicted traversable terrain points. Currently, the model has only been successful during overfit testing, where the predicted results shown in Figure 2.1

are results generated on the same dataset from which the model was trained. Initial attempts to generalize the model fail, generally resulting in predicted points that are tightly clustered along the axis of target output data. In the following Section 2.5 we will discuss potential methods for increasing the generalizability of the model.

In general, the chosen metrics should evaluate the following characteristics of model:

(1) **Prediction Accuracy**: How close are the predicted terrain points to the target terrain? Additionally, is the predicted terrain contained within the shape that defines the target terrain?

(2) **Coverage**: How well do the predicted terrain points cover the total target terrain area?

The choice of performance metric in this problem is particularly subtle, owing in part to the fact that scene extension is not an equivalent operation to point-cloud completion. In terms of prediction accuracy, CD described in Eq. (2.1) is a common metric to measure the performance of point cloud completion algorithms. In our work, it is a large part of the loss function for training the scene extension framework. While CD is a valuable metric of general performance, the results can be misleading or difficult to interpret when predicting traversable terrain. First, CD is most useful when comparing different methods. Since to our knowledge no other scene extension method exists, the CD of our method will be difficult to interpret. Additionally, since CD is purely a distance metric, there is no notion of staying within the target clusters. Therefore, this metric will not penalize the framework for generating traversable terrain outside the target areas, as long as it stays close to the edge of the target terrain. In our loss function, we address this with the masking penalty described in Eq. (2.2). However, by including this term in the evaluation metric will make the metric even more difficult to interpret.

A potential metric to measure the accuracy of the prediction framework could be the number of predicted terrain points contained in the target terrain mask and the number of points predicted outside the masks. However, this metric is flawed as well as it does not measure the overall coverage of the predicted points of the target space.

Finally, IoU and mIoU are popular metrics for point-cloud completion and semantic scene completion models. However both do not directly apply to scene extension. IoU requires complete objects for comparison. As our scene extension framework completely predicts the target terrain, some modifications would be needed for this metric to be applicable, such as only looking at the IoU of the predicted points. In addition, the framework currently predicts points, not complete maps. For IoU to work well, we would need to voxelize the output and compare a voxelized output terrain map to a target terrain map. This would also require some method of algorithm filling or many more predicted points to be generated.

Based on the discussion above, we propose a number of metrics to capture the performance of the model. While each metric may not independently capture the performance of the model, using the metrics together should provide a good idea of how well the model extends the map terrain. The proposed metrics are as follows:

(1) **Prediction Accuracy**: Of the predicted points, how many correctly predict traversable terrain.

$$acc(\mathcal{P}) = \frac{1}{\mathcal{P}} \sum_{p \in \mathcal{P}} f(p), \tag{2.3}$$

where f is a function that returns 1 if p is located in the ground truth traversable terrain and 0 if not.

(2) **Average CD - Prediction to ground truth**: Provides an idea of how close to the ground-truth data the predicted points are.

$$cd_{pt}(\mathcal{P}, \mathcal{G}) = \frac{1}{\mathcal{P}} \sum_{p \in \mathcal{P}} \min_{g \in \mathcal{G}} ||p - g||. \tag{2.4}$$

(3) **Histogram of CD - ground truth to prediction points**: provides an idea of the overall area coverage of the predicted points.

$$g \in \mathcal{G}, \min_{p \in \mathcal{P}} ||g - p||. \tag{2.5}$$

Tables 2.1, 2.2 provide the results of the terrain predictions shown in Figure 2.1 (top: Scene 0, bottom: Scene 1).

Table 2.1: Accuracy and cd_{pt} of predicted terrain shown in Figure 2.1.

	acc	cd_{pt}
Scene 0	88.14	0.224
Scene 1	85.71	0.072

Table 2.2: Histogram of CD - ground truth to prediction point of predicted terrain shown in Figure 2.1 shown as a percentage.

	0.4	0.8	1.2	1.6	2
Scene 0	28.45	50.93	18.40	2.20	0
Scene 1	57.53	37.36	5.12	0	0

Analyzing the results we can see that scene 0 is more difficult for the model than scene 1. While the model has a better accuracy score on scene 0 the average distance from each predicted point to the nearest ground truth point is much smaller in scene 1. Furthermore, a better coverage of the overall scene is shown in Table 2.2 for Scene 1, where more than 50% of ground truth points are within 0.4m of a prediction.

2.5 Conclusions and Future Work

In this paper, we have provided the initial framework for a terrain extension model. The model has been shown to overfit well, accurately predicting points in targeted areas that are occluded or out of range of the input data. However, the model has been unsuccessful in generalizing these points to data outside the training set. Given the nature of these failures, where the predicted points are a single tight cluster, it is likely that adding a tuning multiplier to each term of the symmetrical CD Eq. (2.1) can be used to encourage the model to spread the predictions farther apart. Additionally an explicit cost term could be included to the cost function $\mathcal{C}$ that calculated

the KNN of each prediction and penalizes predictions being too close together. The future work of the project expands beyond the ability of the framework to generalize. Real time terrain extension on real robotic platforms would provide interesting data both of the real time inference capabilities as well as the sim-2-real performance. Finally, exploring frontiers using the maps generated from the terrain extend point clouds is an interesting problem as to how to treat the hybrid maps. The proposed terrain extension framework discussed in this paper is the first step towards faster scene exploration.

Chapter 3

SceneSense: : Diffusion Models for 3D Occupancy Synthesis from Partial Observation

3.1 Introduction

Humans rely extensively on 'common sense' inferences to engage successfully with the world, while robots are limited to making decisions over directly measured data, such as those captured by cameras or lidar. Humans' natural capability to logically extend geometry or terrain in familiar environments such as homes or offices allows for planning beyond direct observation. In this work, we propose a solution addressing this important technical gap, to expand the scope of situations where autonomy can succeed.

Recent advances in AI systems that generate open-ended representations, known as generative AI, give us the building blocks to develop a generative model for predicting out of view or occluded geometry. Previous attempts at generating "extended terrain" borrowing from point cloud completion (PCC) methods [139] struggle to generalize to new environments. Existing methods for generating out-of-view geometry such as semantic scene completion (SSC) [35, 160, 195] and more recently scene synthesis based approaches [59, 165] are promising. However, SSC is limited as it is a completion or hole-filling method applied only to the frustum of the sensor, rather than a truly generative approach that allows for full, 360° occupancy prediction. For their part, synthesis-based approaches require many views for scene synthesis and are not usable as an online method due to slow inference speed.

Motivation for development of these generative models can be found in the results of the DARPA

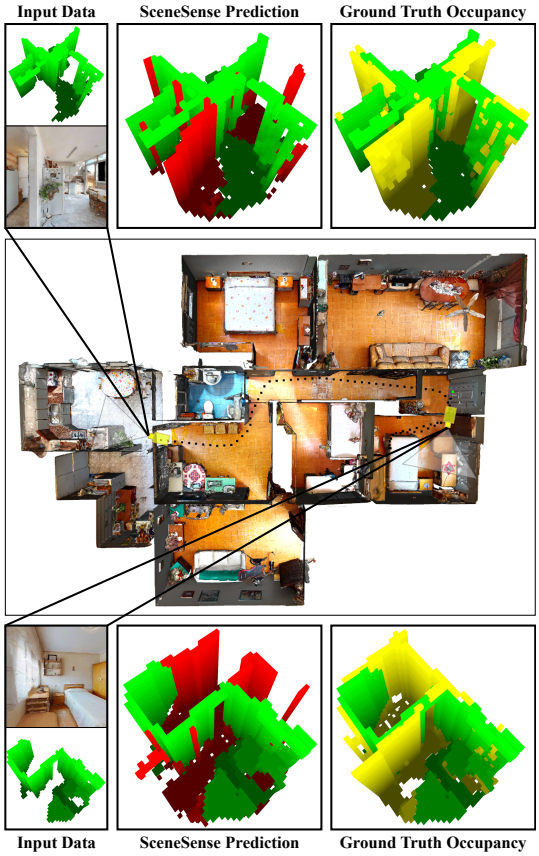


Figure 3.1: Test house 2 where the robot exploration trajectory is shown via the black points, and the starting point is shown as green. Two SceneSense generations are shown. From left to right (1) Inputs are on the left where green voxels are the local occupancy information as well as the current camera view from the robot. (2) SceneSense occupancy prediction is shown where occupancy information is shown in green and new predicted occupancy is red. (3) The running occupancy information is again shown in green and the ground truth full local occupancy data is shown in yellow.

subterranean (SubT) challenge [38]. The SubT challenge tasked robot teams with the goal of locating human artifacts after being released into various unknown environments. These search and rescue missions provide a challenging and impactful venue for the deployment of autonomous systems. While teams were fairly successful in locating artifacts, the search took place with a long tail of artifact discovery over an hour. Systems over-searched areas to ensure maximum volumetric frontier gain and frequently paused when there was an influx of new information (such as turning corners into hallways or entering new rooms) [14]. It is our hypothesis that platform exploration speeds can be accelerated using generative models for occupancy prediction.

In this work we leverage recent advances in generative AI as well as practically available robotics data to generate a predicted occupancy grid around a robotic platform. We show that even with a single RGB-D camera and limited training data we can develop an occupancy prediction model that effectively infers the existence of geometry that is out-of-view or occluded. During training, Gaussian noise controlled by a noise scheduler [72] is added to ground truth occupancy data to generate noisy occupancy grids. We simultaneously train a U-net and perception backbone to reduce noise in the occupancy grid, conditioned by an RGB-D image. At inference time, features are extracted from the input images using the trained PointNet++ model [131] and used as conditioning during the reverse diffusion process. Additionally, we take advantage of the occupancy map constructed during exploration to perform **occupancy inpainting**, increasing the fidelity of our results. Critically, using an inpainting approach ensures that the predicted occupancy grid around the robotic platform will never be modified in areas of the scene that have been directly observed to be occupied or free. Finally, we evaluate our framework in home environments from the HM3D dataset [134] against a running occupancy map. Our results show that our proposed approach (**SceneSense**) enhances the local occupancy predictions around the platform. The primary contributions of this work are as follows:

(1) A generative framework for estimating out of view or occluded occupancy around the robotic platform.

(2) A diffusion inference method we call **occupancy inpainting** that both enhances the predictions of the generative framework and ensures predictions will never overwrite observed free or occupied space.

(3) An extensive ablation study outlining the performance trade offs of various tunable parameters that are configurable at runtime.

3.2 Related Works

3.2.1 Semantic Scene Completion (SSC)

SSC seeks to generate a dense semantically labeled scene in a target area given some sparse scene representation in that area. Generally the provided information is quite sparse and requires the SSC models to fill in large gaps due to occlusions from the viewpoint. SSC methods are often designed specifically for outdoors [35] or indoor applications [31]. While outdoor SSC implementations focus on SSC using a 3D lidar, indoor methods use aimed sensors such as a RGB-D camera. Due to the shape of this input data outdoor models focus on prediction and labeling all voxels in a grid around the platform, while indoor models generally focus on performing SSC in the frustum of the sensor. Predicting correct geometry and semantic labels is a challenging task. SSC methods have been noted to suffer from poor performance given the challenging nature of the problem [143]. Indoor methods perform around 40% mIoU [143] when "filling in" data in the frustum of the sensor, but are not intended to be generative models that can expand a partially observed scene. Our method expands the role of these models to not only fill in occluded information in the sensor's field of view but to also generate a prediction of what geometry may look like around the platform.

3.2.2 Generative 3D Scene Synthesis

View synthesis is a field of study that seeks to construct a 3D scene from multiple camera views. This is primarily accomplished using a deep learning framework called Neural Radiance Fields (NeRFs) [59]. Original works in this space synthesized 3D views of objects from multiple camera views [118] , while recent works have synthesized full indoor scenes [142]. Current NeRF implementations are slow at inference time, often taking on the order of minutes to render a scene [59]. This characteristic makes them unsuited to real-time scene rendering. In addition, NeRFs require multiple views of the environment to generate reliable volumetric scenes, which may not be available when operating in real-time.

Recently, diffusion models have been explored as a means for synthesising scenes [165]. Initial

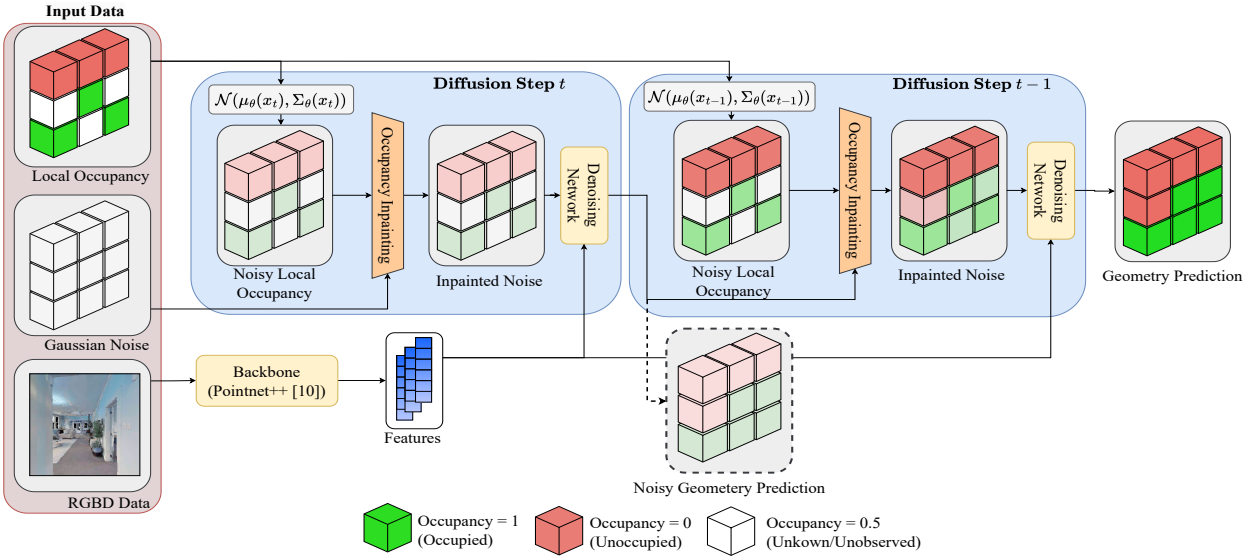


Figure 3.2: **Reverse Diffusion Process**: The reverse diffusion process takes the local occupancy information, the current sensor measurements (RGB-D image in this case) and the Gaussian noise of the area to be diffused over. Noise commensurate with the current diffusion step is added to the local occupancy information, which includes observed occupied (green) and observed unoccupied (red) data. The result is inpainted into the noisy local occupancy prediction as discussed in section 4.2.2. The inpainted noise data and the feature vectors generated by the perception backbone are provided to the denoising network which generates a new noisy geometry prediction at $t-1$. This process is repeated as the starting noise x_T is iteratively denoised to x_0 which is the final geometry prediction from the framework.

studies show these models outperform traditional models in scene synthesis and introduce popular generative metrics to the scene synthesis research space. While these implementations are not directly usable in a robotics context, the ideas and evaluation metrics for 3D scene synthesis are applicable to our problem space.

3.2.3 3D Diffusion and Diffusion in Robotics

Diffusion models [72, 157] have had great success as generative models, generating impressive results across diverse modalities, such as image [146] video [69], audio [78] and natural language [4]. A number of surveys have been published in recent years providing further details on various implementations [195, 41]. Research on diffusion in 3D is limited and generally applied to single object generation [206]. However recent works have begun to explore application of diffusion models

for scene generation. Notable work in this space include LegoNet [201] which applies diffusion models to rearrange objects in a 3D scene and DiffuScene [165] which supports unconditional or prompted diffusion of 3D scenes. While these diffusion methods are effective at their task they do not directly translate to robotic applications due to inference time requirements and the required conditioning data.

Diffusion for Robotic Applications. The success of diffusion models have inspired researchers to begin to apply them in the robotics domain. While the Markovian nature of diffusion models can be a bottleneck for systems requiring real-time inference diffusion models have been successfully applied to real-time robotics problems such as planning problems [82, 81] and perception [83, 26]. These generative models are both increasing the effectiveness of current methods in traditional robotics problems as well as providing new research areas to tackle.

3.3 Preliminaries and Problem Definition

3.3.1 Problem Definition: Dense Occupancy Prediction

The objective of dense occupancy prediction is to predict the occupancy from $[0, 1]$ where 0 is unoccupied and 1 is occupied for every voxel v in a target region x where $v \in \mathbb{R}^{\mathbf{z}\times\mathbf{x}\times\mathbf{y}}$.

3.3.2 Forward Diffusion

x_0 is defined as a clean occupancy grid where the distribution of x_0 can be defined as $q(x_0)$. By sampling from the data distribution $x_0 \sim q(x_0)$ the forward diffusion process is defined as a Markov chain of variables $x_1, ..., x_T$ that iteratively adds Gaussian noise to the sample. A diffusion step at time t in this chain is defined as:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1},\ \beta_t I), \tag{3.1}$$

where t is the time step $t \in [1, T]$, β_t is the variance schedule $0 \leq \beta_t \leq 1$ and I is the identity matrix. The joint distribution of the full diffusion process is then the product of the diffusion step

defined in eq. (3.1):

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}). \tag{3.2}$$

Conveniently we can apply the reparameterization trick to directly sample x_t given x_0 using the conditional distribution:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\overline{\alpha}_t}x_0, (1-\overline{\alpha}_t)\mathcal{I}), \tag{3.3}$$

where $x_t = \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon$ where $\alpha_t := 1-\beta_t$, $\overline{\alpha}_t := \prod_{r=1}^{t} \alpha_s$, and ϵ is the noise used to corrupt x_t.

3.3.3 Reverse Diffusion

Reverse diffusion is a Markov chain of learned Gaussian transitions $p_\theta(x_{t-1}|x_t)$ which is parameterized by a learnable network θ:

$$p_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)), \tag{3.4}$$

where $\mu_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$ are the predicted mean and covariance respectively of the Gaussian x_{t-1}. Given the initial state of a noisy occupancy map from a standard multivariate Gaussian distribution $x_t \sim \mathcal{N}(0, I)$ the reverse diffusion process iteratively predicts x_{t-1} at each time step t until reaching the final state x_0 which is the goal occupancy map. Similar to the Markov chain defined forward diffusion process the joint distribution on of the reverse diffusion process is simply the product of the applied learned Gaussian transitions $p_\theta(x_{t-1}|x_t)$:

$$p_\theta(x_{0:T}) := p_\theta(X_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t). \tag{3.5}$$

3.3.4 Conditional Diffusion

The conditional diffusion model extends the diffusion process to guide the diffusion by some conditioning y. In particular we use a class of conditional diffusion models called classifier-free diffusion [73]. During training the diffusion model $f_\theta(x_t, y, t)$ is trained to predict x_0 from x_t under the

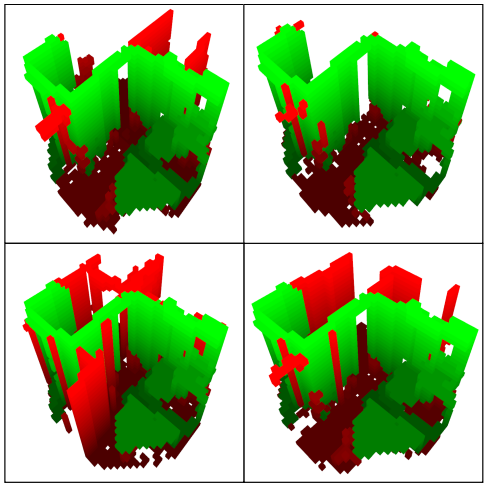


Figure 3.3: Various SceneSense predictions from equivalent input data where green is the running occupancy map and red is the SceneSense predicted occupancy. Given the limited input information the diffusion framework can generate multiple reasonable predictions from the same input conditioning.

guidance of condition y. During training conditioning y is replaced with a null label $\emptyset$ with a fixed probability. At inference time x_0 is reconstructed from x_T with guidance from the conditioning y. During sampling at inference time the output of the model is "pushed" toward the conditional model result $f_\theta(x_t|y)$ and away from the unconditioned result $f_\theta(x_t|\emptyset)$ as follows:

$$\hat{f}_\theta(x_t|y) = f_\theta(x_t|\emptyset) + s \cdot (f_\theta(x_t|y) - f_\theta(x_t|\emptyset)), \tag{3.6}$$

where s is the guidance scale defined as $s \in \mathbb{R}_{\geq 0}$. s is a configurable parameter at runtime that allows for the user to configure how closely the model should adhere to the provided conditioning.

3.4 Method

3.4.1 Architecture

Denoising Network. The denoising network in our method is inspired by the popular image generation diffusion network Stable Diffusion [146]. It is a U-net constructed from the HuggingFace Diffusers library of blocks [174] and consists of Resnet [70] downsampling/upsampling blocks with cross-attention as well as regular ResNet downsampling/upsampling blocks. The conditioning

features generated by Pointnet++ are mapped to the intermediate layers of the U-net via the cross attention layers of the transformer blocks as discussed by Stable Diffusion [146].

Feature Extraction and Conditioning. As discussed in section 3.3 classifier-free diffusion models are conditioned on a set of guidance data y. Our model uses the Pointnet++ [131] backbone to generate a $N \times F$ feature matrix from a given RGB-D image, where F is the number of desired Pointnet++ features per point $n \in N$. As is common with other vision based pipelines [106] the feature generation backbone can be replaced with other models to increase performance or add/remove different types of perception modalities. Further, the conditioning of the diffusion model can be any modality encoding into feature vectors. Conditioning could include standard robotic sensors such as camera, lidar or radar, but could also be extended to include informational modalities such as human text input or sketches of the scene [146].

Occupancy Mapping. Occupancy mapping allows for platforms to build a running map of areas that have been measured to contain matter using onboard sensors like lidar or RGB-D cameras. For our framework we use the popular occupancy mapping framework Octomap [77] to generate an occupancy map as the platform explores the environment. Importantly, Octomap provides a probability of occupancy $o \in [0, 1]$ for every voxel in the map that has been observed using pose ray casting. This means that as we explore we will be maintaining not only a map of occupied areas M_o, but also a map of areas that have been measured to not contain any data M_u. These maps will later be used to inform SceneSense where occupancy predictions should be made.

3.4.2 Training

During training, we generate a noisy local occupancy map x_t where $t \in [1, T]$ from a ground truth local occupancy map x. We train the diffusion model f_θ to predict the noise applied to x_t given the associated RGB-D conditioning y.

Occupancy Corruption. To corrupt each ground truth local occupancy map x to train the network we add Gaussian noise to x to generate x_t. This corruption process is defined in eq. (3.3)

where the intensity of the noise is controlled by α_t which is configured by a linear noise scheduler [72].

Loss Function. The network f_θ is trained using the calculated l_2 loss between the denoised x_t prediction and the associated ground truth data x. l_2 loss is a popular diffusion loss function, however other loss functions such as cross-entropy loss or mean squared error can be applied and have had some success in similar diffusion frameworks [124, 146, 26, 83].

3.4.3 Inference

Sampling Process The trained noise prediction network f_θ takes isotropic Gaussian noise $\mathcal{N}(0, \mathcal{I})$ as the starting point x_T to begin the reverse diffusion process. The noise is iteratively removed by using f_θ and the associated RGB-D features y to compute x_{t-1} The RGB-D features y are applied as conditioning during the process with the cross-attention mechanism [171].

Occupancy Inpainting. Our method of occupancy inpainting ensures observed space is never overwritten with SceneSense predictions. Additionally occupancy inpainting enhances the predictions from the models seen in fig. 3.6 (c). Inspired by image inpainting methods seen in image diffusion [146] and guided image synthesis methods [116], occupancy inpainting continuously applies the known occupancy information to the diffusion target during inference. To perform occupancy inpainting we select our target region x as a sub map of the full occupancy map. From x we build a occupied map M_o and unoccupied map M_u composed of observed occupied voxels and observed unoccupied voxels respectively. From M_u and M_o we generate noisy representations commensurate with the current diffusion step t using Eq. 3.3, M_{un} and M_{on} respectively. Finally the data from M_{un} and M_{on} replaces the data in the diffusion target at the associated coordinates. This process is repeated for each inference step and can be seen in fig. 3.2. This method both increases the fidelity of the scene predictions and ensures the diffusion model does not predict or modify geometry in space that has already been observed.

Multiple Prediction. Diffusion is a noisy process that can generate different results given the

same context. In image generation this is a desirable characteristic as the framework can generate different results given the same prompt, increasing the diversity of the generated data. As shown in fig. 3.3 SceneSense has the same behavior as these networks and can generate different reasonable predictions based on the same input information. Further there is no compute time increase (assuming enough compute is available) as multiple predictions can be done in parallel. For simplicity we simply generate one prediction at each pose, but additional heuristics or voting schemes could be added to the system to score multiple outputs and select preferable predictions.

3.5 Experiments

We use the Habitat lab simulation platform [130] and the Habitat-Matterport 3D research dataset (HM3D) Dataset [134] to generate training and test data. We operate a simulated platform with a 256×256 RGB-D camera through 12 different house environment to generate full occupancy grids with voxel resolution of $0.1m$ of the homes as well as ≈9000 poses and associated RGB-D camera views to be used as conditioning. We split the dataset into a training and test set by house number. Houses 3–12 are used in the training set and houses 1 and 2 as shown in fig. 3.4 and fig. 3.1 respectively are used as the test set. For testing arbitrary trajectories are taken to navigate through each home, and SceneSense is used to predict the local occupancy information at each timestep.

Implementation. The diffusion model is trained using randomly shuffled pairs of conditioning y and ground truth occupancy grids x, where various houses may be mixed in a batch. We use Chameleon cloud computing resources [88] to train our model on one A100 with a batch size of 16 for 250 epochs or 119,250 training steps. We use a cosine learning rate scheduler with a 500 step warmup from 10^{-6} to 10^{-4}. We set dropout to 0.2 where the conditioning y is set to $\emptyset$. The noise scheduler for diffusion is set to 1000 noise steps. At inference time we evaluate our dataset using an RTX 4090 GPU for acceleration. On average we measure a diffusion step for our model to be 0.0633 seconds.

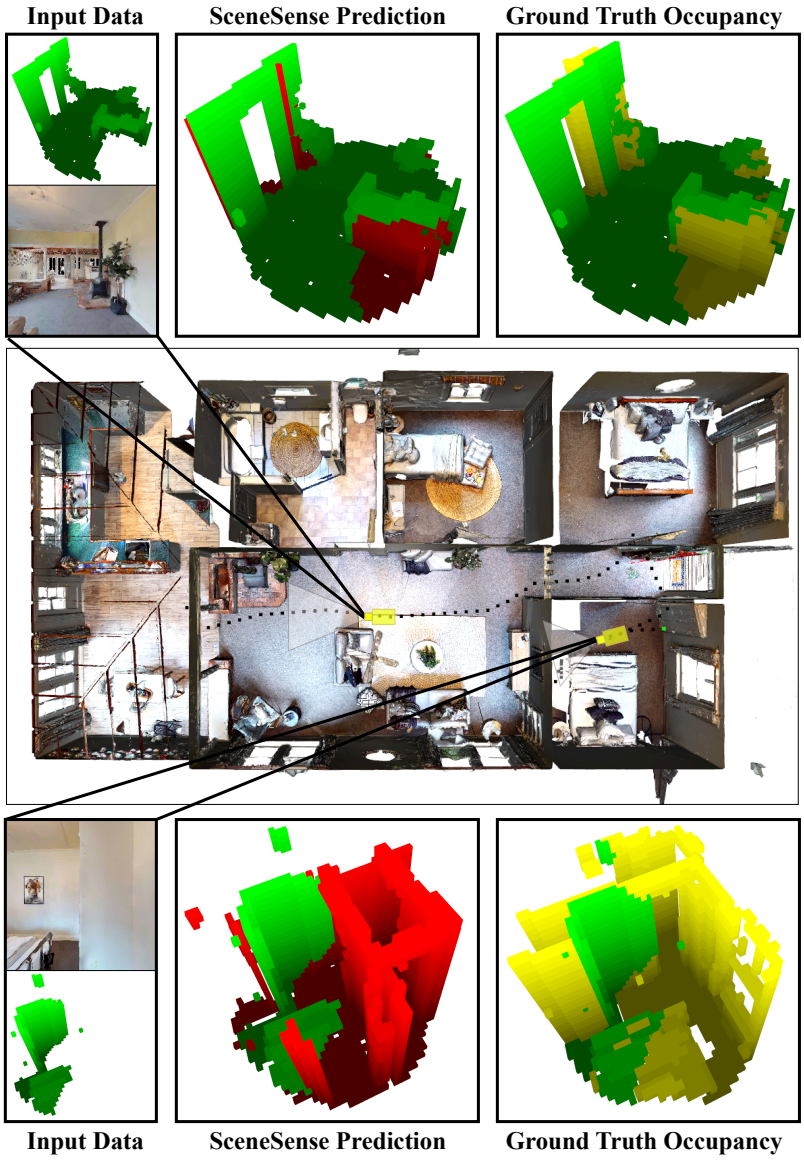
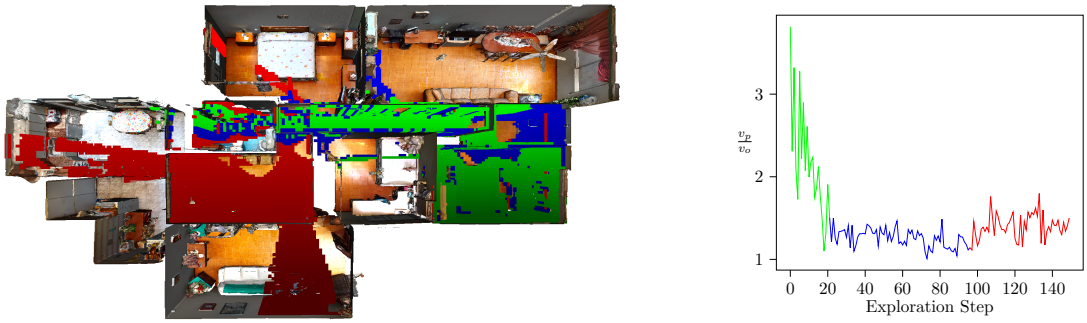


Figure 3.4: Test house 1 where the robot explore trajectory is shown via the black points, and the starting point is shown as green. Two SceneSense generations are shown. From left to right (1) Inputs are on the left where green voxels are the local occupancy information as well as the current camera view from the robot. (2) SceneSense occupancy prediction is shown where occupancy information is shown in green and new predicted occupancy is red. (3) The running occupancy information is again shown in green and the ground truth full local occupancy data is shown in yellow.



(a) (b) **Conditioning**

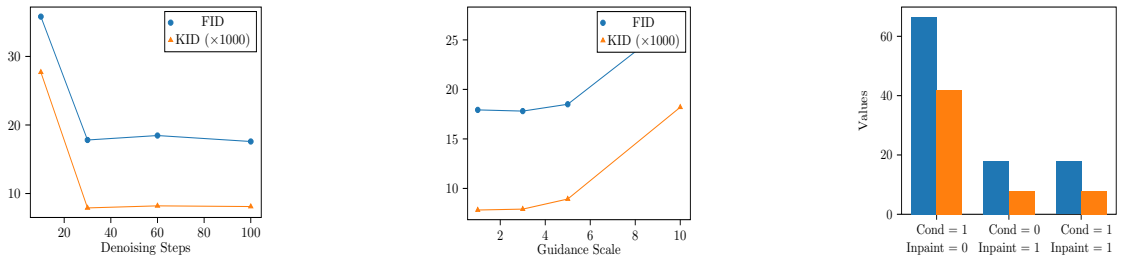
Figure 3.5: Calculated predicted voxels v_p over occupied voxels v_o ($\frac{v_p}{v_o}$) over the house 2 exploration. (a) Superimposes the running occupancy map over the house mesh where the colors of the occupancy map show how many steps have ran to that point. Green voxels are the running occupancy map from step 0 to step 20, blue are step 0 to step 95, and red are step 0 to 150. These colors correspond with the plot line colors in (b). (b) Shows the $\frac{v_p}{v_o}$ as the robot explores the space. $\frac{v_p}{v_o}$ starts high at time step 0, when the occupancy map is sparse, and quickly drops over the green exploration where more of the local scene is observed. $\frac{v_p}{v_o}$ stays relatively low as the vehicle completes the exploration of the green room, navigates back to the start point and traverses the hallway. $\frac{v_p}{v_o}$ increases slightly as the robot traverses previously unobserved space (red), which requests more predicted voxels as less of the scene has been observed.

Baselines. To our knowledge this is the first architecture to apply a generative method to predict 3D occupancy around a platform from a single aimed sensor. Notably, OPNet [176] uses a neural network to predict the geometry of partially visible objects but cannot generate complete scenes, making it unsuitable for comparison with SceneSense, which generates full scenes beyond occluded views. As such, the best evaluation of our method is an evaluation against the running octomap (BL). Improvement upon this baseline shows that occupancy predictions from SceneSense better represents the ground truth occupancy information at a given pose than the running occupancy grid.

Evaluation. Following similar generative scene synthesis approaches [165, 178] we employ the Fréchet inception distance (FID)] [71] and the Kernel inception distance [15] (KID $\times$1000) to evaluate the generated local occupancy grids using the clean-fid library [127]. Generating good metrics to evaluate generative frameworks is a difficult task [122]. FID and KID have become the standard metric for many generative methods due to their ability to score both accuracy of predicted results, as well as diversity or coverage of the results when compared to a set of ground truth data. Traditional metrics such as accuracy or IoU penalize generative methods for making reasonable predictions as they have no means of evaluating if a prediction beyond the alignment of the predictions to a ground-truth. While these metrics are fairly new to robotics we show that they are an effective measure of the success of a generative framework like SceneSense.

Table 3.1: Quantitative comparisons of local occupancy synthesis from two test home environments from the HM3D [134] dataset.

	House 1		House 2	
Method	FID ↓	KID ↓	FID ↓	KID ↓
BL	26.18	18.91	22.55	14.06
SS	17.81	7.93	20.94	6.93



(a) **Denoising steps** (b) **Guidance scale** s (c) **Conditioning**

Figure 3.6: **SceneSense ablation experiments**: All ablation experiments were run on test house 1 using the same trained diffusion network. For all experiments conditioning and inpainting are enabled, s is set to 3 and 30 denoising steps are used unless these values are being ablated. (a) Figure (a) ablates various denoising step values. (b) Figure (b) ablates various guidance scale s values as defined in eq. (3.6). (c) Figure (c) ablates enabling conditioning and inpainting for the network. A 1 indicates the value is enabled and 0 indicates it is disabled.

3.6 Results

Quantitative results for the test set are shown in table 4.2 while qualitative results showing example diffused occupancy grids can be seen in fig. 3.4 and fig. 3.1. SceneSense achieved substantial reductions in both KID and FID when compared to the baseline running occupancy method. Additionally, the qualitative results show reasonable estimates of potential geometry around the platform given limited input information.

Quantitative Discussion. The results presented in table 4.2 favor SceneSense when compared to the local running occupancy information in both scenes tested. In house 1 substantial reduction in FID and KID, 31% and 58% respectively, were reported. While the reduction in FID was lower in house 2 (7%) the reduction in KID was similar to that measured in house 1, 50%. KID is known to be less sensitive to outliers and is considered by some to be an advantageous metric for evaluating generative frameworks when compared to the FID [15]. As shown in fig. 3.1 house 2 is a much different layout than house 1, with many small hallways, rooms and corners. It is likely that house 1 is more similar to those captured in the training set than house 2, leading to an increase in erroneous predictions in the house 2 scene. These predictions result in a larger FID score while the KID remains low due to the native robustness to these skewed distributions.

Further we can look at example predictions for qualitative evaluation and to examine FID and KID as evaluation metrics compared to traditional metrics such as IoU. Take for example the upper prediction in fig. 3.1 where the platform is entering the kitchen. SceneSense adds useful information to the problem, predicting the existence of floor as well as a wall that would obstruct motion to the right of the robot. However the wall is 0.1 - 0.2m off the actual existing wall. The result of this incorrect locating results in a worse IoU (0.52) than the local occupancy information (0.61). These problems in quantitative evaluation are only exacerbated when there is more geometry to estimate such as in the bottom observation of fig. 3.4. Generative models incur a large penalty in the IoU metric for guessing at geometry, even when given very little context for the prediction. Predictions increase the total union of the space, however a missed prediction does not increase the intersection of the space. In the case of the top prediction in 3.1 even if the prediction would be useful in planning, since the wall is mislocated the metric reports a much worse IoU than the running occupancy. Additionally in cases where little information is known, such as in the bottom prediction of 3.4, an occupancy prediction of what the geometry may look like is heavily penalized since the metric does not evaluate if a prediction distribution is reasonable. However the FID and KID metrics evaluate both the accuracy of predictions as well as the distribution of the predictions, allowing for generative frameworks that attempt to generate challenging results to be fairly scored. FID and KID are already widely adopted metrics in generative fields such as image generation [135, 146] and scene synthesis [165, 178] for these reason and our results support their use in this context. **Qualitative Discussion**. Predicted frame can be qualitatively evaluated against the following criteria:

(1) **Reasonability**: Is the generated occupancy grid reasonable? Does the platform have enough information to make a better guess?

(2) **Usefulness**: Does the generated occupancy grid provide useful information? Would the addition of this information be useful in planning or control execution?

(3) **Accuracy**: Does the generated occupancy grid accuracy describe the geometry of the

scene?

Examining the frames from fig. 3.4, the top SceneSense prediction is reasonable and accurate. SceneSense does a good job filling in the occluded geometry on the back of the couch and floor, and matches well with the ground truth data. While this information is useful, in particular providing the additional ground for potential planning, it likely would not have a large impact on most planning frameworks. The bottom SceneSense prediction shown in fig. 3.4 is quite different from the previously examined prediction. The accuracy of this scene is quite low, predicting additional walls and doorways where they do not exist. The usefulness of this scene is debatable, there is a doorway shown in the constructed wall behind the platform, but it likely due to the total volumetric gain past the bed a planner would chose to explore beyond the bed rather than behind it. The reasonability however of this scene is arguably high. As this is a very early pose in the occupancy mapping process the model has very little information to go off of as shown in the input data. It is entirely reasonable to assume that the doorway is behind the robot which exits to the hallway structure that is generated. Given such little information, one cannot expect perfect prediction of previously unexplored space. However due to the inpainting method implemented in our framework, once more information is made available, predictions become more grounded as less area is available to predict over. This behavior is shown in fig. 3.5 The same quantitative analysis applies to the bottom prediction for test house 2 fig. 3.1.

Examining the top frame of house 2 from fig. 3.1, the SceneSense prediction is useful. When entering a new room the platform does not directly measure the wall to its right, leaving the occupancy map empty in that space. A plan could be formed to explore that empty space, however SceneSense constructs a wall impeding that movement which would encourage exploration further into the room. While this prediction is somewhat low accuracy due to the misplacement of the wall, it is indeed reasonable and useful to the platform.

SceneSense Accuracy Over Time. As discussed in section 4.2.2 SceneSense only predicts occupancy in areas that have not been directly measured to be occupied or unoccupied. This

means that as exploration of the space approaches 100% SceneSense will have no unobserved space to predict over. This can be measured as $\frac{v_p}{v_o}$ where v_p are the predicted voxels from SceneSense and v_o are the local occupied voxels. Given 100% exploration this metric will approach 1 where all predicted voxels are simply the local occupied voxels. This reduction in prediction space is shown in fig. 3.5. Initially $\frac{v_p}{v_o}$ is quite high, since very little of the scene has been observed but quickly drops during exploration. Spikes in $\frac{v_p}{v_o}$ are seen when the platform moves to new areas that have occlusions such as hallways or when entering new rooms. These metrics support the assertion that SceneSense respects measured space and only generates geometry where no measurement has been taken.

3.6.1 Ablations

Denoising Steps Discussion. The number of diffusion steps defines the size of each diffusion step during the reverse diffusion process. Generating reasonable results using the fewest possible denoising steps is desirable behavior to reduce computation time. Additionally too many denoising steps have been shown to introduce sampling drift which results in decreased performance [26, 83]. As shown in fig. 3.6 (a) our method saw the best results when configured to 30 denoising steps. Too few denoising steps results in the network being unable make accurate predictions over the large time step. Increasing the number of denoising steps keeps results relatively stable over time, however you can see the KID is slightly worse using more steps due to sampling drift. Sampling drift is a result of the discrepancy between the distribution of the training and the inference data. During training, the model is trained to reduce a noisy map x_t to a ground truth map x, at inference time the model iteratively removes noise from its already imperfect noise predictions. These predictions will drift away from the initial corruption distribution which becomes more pronounced at smaller time steps due to the compounding error.

Conditioning and Guidance Scale Discussion. The guidance scale s as defined in eq. (3.6) is a constant that multiplies the difference of the conditional diffusion and unconditional diffusion to "push" the diffusion process towards the conditioned answer. Setting s too high results in too large

of pushes away from reasonable predictions and results in poor generalization to new environments. The best FID is measured when $s = 3$ however KID is slightly lower when $s = 1$. Further, when examining chart (c) it is shown that the results when conditioning is removed all together ($s = 0$) are very similar to the best results seen with conditioning enabled (albeit slightly worse). This is likely because most of the useful conditioning information is captured in the local occupancy data, and mapping measured RGB-D points from area in front of the to geometry under or behind the platform is a very difficult task. The performance of the conditioning only data may be seen to have a larger impact on the overall results if the sensor could capture more local information, such as given a wider FOV or different mounting angle.

Chapter 4

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

4.1 Introduction

In general robots are limited to evaluating and making decisions over space that has been directly observed, either during the present deployment or a prior one. For deployments to environments where prior information does not exist, the autonomous system relies only on what it can observe at present. These deployments are particularly challenging for autonomous navigation as perception sensors have limited fields of view, and are often occluded by obstacles in the environment. Data products, such as 2D or 3D geometric maps that are generated for these environments, can have holes where the sensor could not observe, particularly at runtime when the system is exploring. While there are existing methods for filling LIDAR shadows [51] or gaps in the map [189, 33], most focus on hole filling where the geometry around the hole has been observed. To further enhance robotic decision making, we need to not only fill holes in the map, but also be able to extend map geometry beyond what can be directly measured.

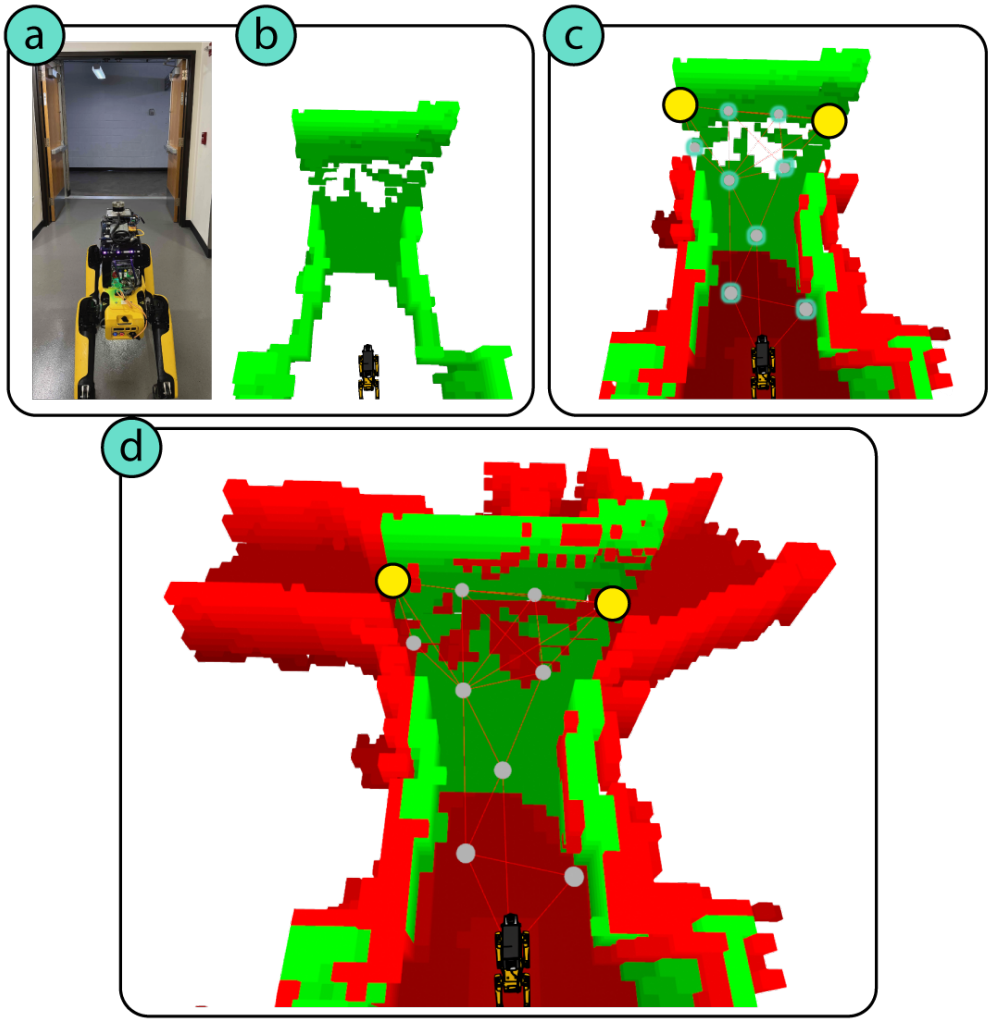


Figure 4.1: **Onboard Occupancy Prediction and Map Merging**: Green voxels represent observed occupancy and red voxels represent predicted occupancy. Gray graph points represent vertices and yellow graph points represent vertices identified as frontier points. **(a)** Spot platform is positioned in front 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.

Occupancy prediction is a method to fill and extend observed maps beyond direct measurements made by sensors. Recent works [138] have shown that occupancy prediction models can create realistic and likely predictions of what complete occupied space could look like around a robot. However it is not obvious how these methods would transfer from simulation to a real-world system. In this paper, we make key modifications to the diffusion-based SceneSense occupancy prediction model [138] to enable occupancy predictions at any point in the running map, as well as achieve decreased inference times for online occupancy prediction. Further, we define a probabilistic map update rule to merge the occupancy predictions with the running observed map. We implement a graph-based frontier evaluation method for identifying ideal areas for occupancy prediction and evaluate it with a real-world robotic system. The primary outcomes of the contributions discussed are as follows:

(1) 73% end-to-end run time reduction for SceneSense [138] occupancy prediction model.

(2) Enabling occupancy predictions at range, anywhere in the map.

(3) 75% improvement in frontier occupancy map evaluation metrics when compared to the vision only map

(4) 71% improvement in frontier occupancy map evaluation metrics when compared to the one shot map merging method presented in the original SceneSense work [138].

4.2 Related Work

4.2.1 Occupancy Prediction

One solution to the challenge of autonomous navigation in occluded environments is to predict occupancy distributions. Though deep learning (DL) approaches have shown promise, existing methods struggle with scalability, generalization, and handling occluded areas. Wang et al. [177] propose a DL approach to predict occupancy distribution which involves selectively removing data from the Matterport3D [22] 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 [79], 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 [138]. Our research advances this method by introducing a novel, more efficient approach to occupancy prediction, demonstrated on real hardware.

4.2.2 Scene Synthesis

Diffusion models [72, 158], are a popular tool that has demonstrated impressive generative results across image [145], video [69], and natural language [78]. Based on these successes, diffusion models are being extended to 3D scene and shape generation. Recent work [108] demonstrates the use of diffusion models for 3D point cloud generation for simple shapes and objects (e.g. tables, chairs). Kim et al. [91] shows successful 3D shape generation from 2D content such as images, and Vahdat et al. [170] demonstrate similar 3D shape generation but using point cloud datasets rather than images. In LegoNet [182], diffusion models are used to propose object rearrangements in a 3D scene. In DiffuScene [165], 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.

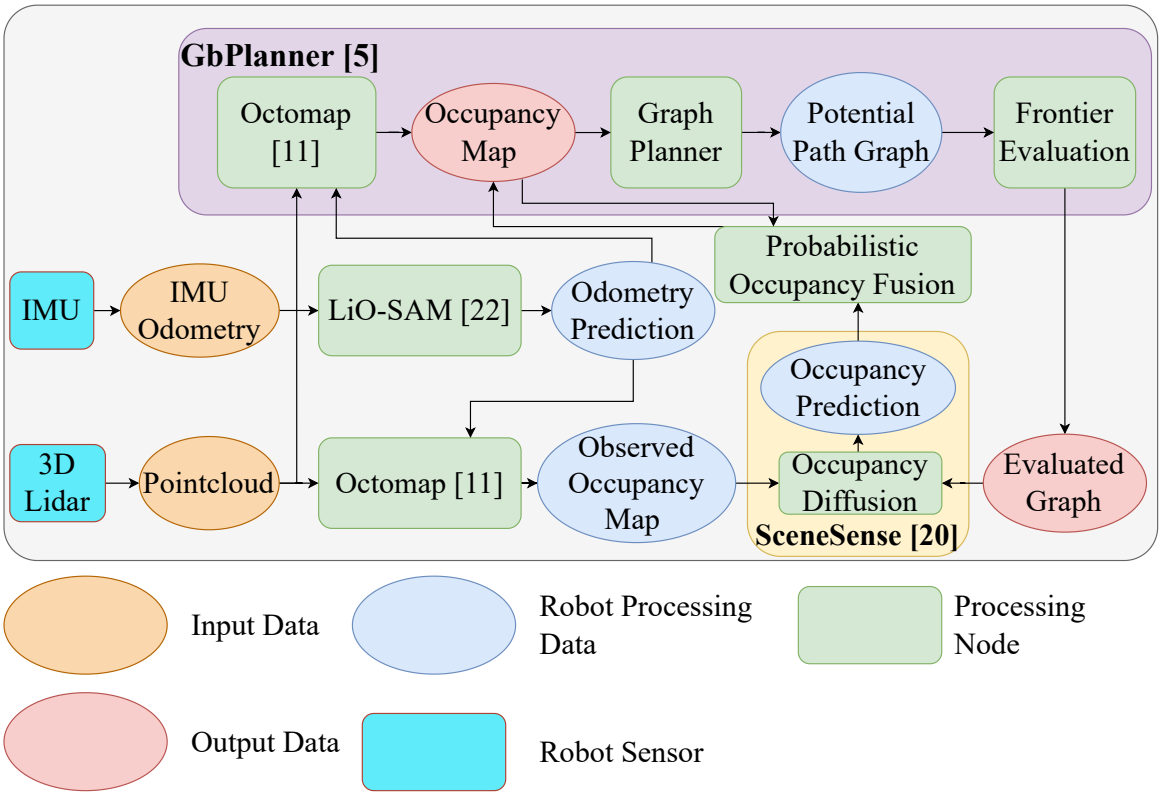


Figure 4.2: **System Block Diagram**: Block diagram showing the system design for onboard SceneSense occupancy prediction. The system is comprised of an IMU and Lidar sensor to generate odometry and occupancy maps. Once the occupancy map is built, a graph is constructed to evaluate frontier points for 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.

4.3 Methods

4.3.1 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 $m \in \mathbb{M}_{free}$, $m \in \mathbb{M}_{occupied}$, and $m \in \mathbb{M}_{unknown}$, representing free, occupied and unknown space respectively. We seek to identify and evaluate frontiers in $\mathbb{M}$ that can enhance the robot's decision making for potential exploration. In general, "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 [13, 44].

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 $x \subset \mathbb{M}$.

4.3.2 Robotic System Architecture

Our robotic system is constructed as a quadruped (Spot) as shown in Figure 4.1 as well as an off-board high performance computer to handle computationally expensive requests. A block diagram of the system is shown in Figure 4.2.

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, 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 [152] to provide localization at run-time.

Occupancy Prediction. We adopt the SceneSense occupancy prediction diffusion model [138]

with modifications to enable novel functionality and performance improvements. Originally, SceneSense was designed as a conditional diffusion model where the conditioning is 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 [138]. 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 the same unconditioned performance. Further, this change also removes the need for a feature extraction backbone, saving additional computation time.

Frontier Identification and Evaluation. With the 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 [44]. 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: where $\mathcal{S}(\sigma_i, \sigma_{exp})$, $\mathcal{D}(v_1^i, v_j^i)$ are weight function with tunable factors $\gamma_{\mathcal{S}}, \gamma_{\mathcal{D}} > 0$ respectively. Furthermore $\mathcal{D}(v_1^i, v_j^i)$ is the cumulative Euclidean distance from a vertex v_j^i to the root v_1^i along a path γ_i.

Exploration gain is used to rank nodes at which occupancy prediction should run. Given some minimum node spacing d_m and some maximum number of frontier prediction nodes n_{max} SceneSense is run centered at the identified frontiers to generate occupancy predictions at range in ideal areas.

Mapping. The probabilistic volumetric mapping method Octomap [77] was selected as the map-

ping framework. Octomap was adopted for its log-odds update method for predicting occupied and unoccupied cells. This approach allows for elegant fusion of observed occupancy and predicted occupancy maps. Further discussion on map fusion is provided in Section section 5.3.4.

4.3.3 Probabilistic Map Merging

In previous work predicted occupancy was merged into the running occupancy map in a "fire and forget" approach [138]. A given occupancy prediction was temporarily merged into the running occupancy map by setting the predicted 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 [77].

We define the probability that a voxel m is occupied given a 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 [138], 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.

$$P(m \mid j_{1:t}) = \begin{cases} \left[1 + \frac{1-P(m|d_t)}{P(m|d_t)} \frac{1-P(m|j_{1:t-1})}{P(m|j_{1:t-1})} \frac{P(m)}{1-P(m)}\right]^{-1} & \text{if } m \notin \mathcal{O} \\ \left[1 + \frac{1-P(m|z_t)}{P(m|z_t)} \frac{1-P(m|j_{1:t-1})}{P(m|j_{1:t-1})} \frac{P(m)}{1-P(m)}\right]^{-1} & \text{if } m \in \mathcal{O}. \end{cases} \tag{4.1}$$

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 voting.

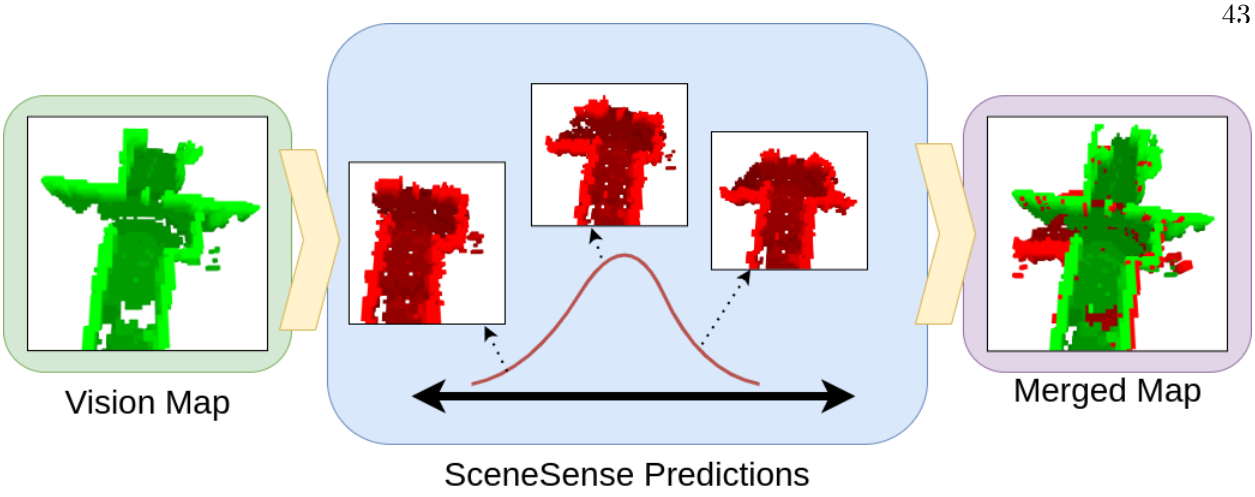


Figure 4.4: **Multi-Prediction Occupancy Merging**: SceneSense predicts various occupancy maps based on equivalent input data that form a distribution. This distribution forms a curve where more likely predictions occur more often, and less likely predictions occur infrequently. These predictions are fused into the merged map using Eq. 5.2. The resulting merged map naturally filters out the unlikely voxel predictions, forming an extended occupancy map.

Multi Prediction Voting. As shown in Figure 5.3, 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 conditioning data [15, 145]. Given this behavior, we can collect various predictions of the same scene and think of these predictions as a distribution, where the more likely scenes occur more frequently and less likely scenes occur at the tails of the distribution. Then we can aggregate the scene predictions using the probabilistic update rule defined in Eq. 5.2 and generate a scene 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.

Observed vs. Predicted Voxels. The architecture of 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. 5.2 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 $P(m \mid d_t) < P(m \mid z_t)$. 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.

Further 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.

4.4 Experiments and Results

In this section, we provide results and evaluations of the modified SceneSense occupancy prediction framework onboard a real-world system.

Training and Implementation. To train SceneSense we collected real-world occupancy maps from various buildings. Approximately 1 hour worth of occupancy data, which resulted 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 [138]. It is a U-net constructed from the HuggingFace Diffusers library of blocks [174] and consists of Resnet [70] 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 [88] 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 1000 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.

4.4.1 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 [138]. 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 4.1: Inference time and model size results. "Full inference" and "end-to-end" evaluations are computed using 30 diffusion steps.

	Cond. Model [138]	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 4.1, removing the conditioning from the diffusion model reduces the computation requirements substantially. The unconditional 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.

4.4.2 SceneSense Generative Occupancy Evaluation

For the following experiments we evaluated the occupancy generation capabilities of the 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 potential map update methods and the running sensor only map.

Experimental Setup. The SceneSense occupancy prediction is evaluated in 2 test environments. Test environment 1 was a long hallway with hallway cutouts for classrooms and 1 right turn. Select frames shown in figure 4.5a and 4.5c are from Environment 1. Test environment 2 consists of enclosed carpeted area where 4 hallways for a closed box. 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-centeric**: robot-centeric 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 some identified location in the map, which is 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 5.2 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 [165, 178] we employ the Fréchet inception distance (FID) [71] and the Kernel inception distance [15] (KID $\times 1000$) to evaluate the generated local occupancy grids using the clean-fid library [127]. Generating good metrics to evaluate generative frameworks is a difficult task [122]. 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, we show that they are an effective measure of the success of a generative framework like SceneSense.

Table 4.2: 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**

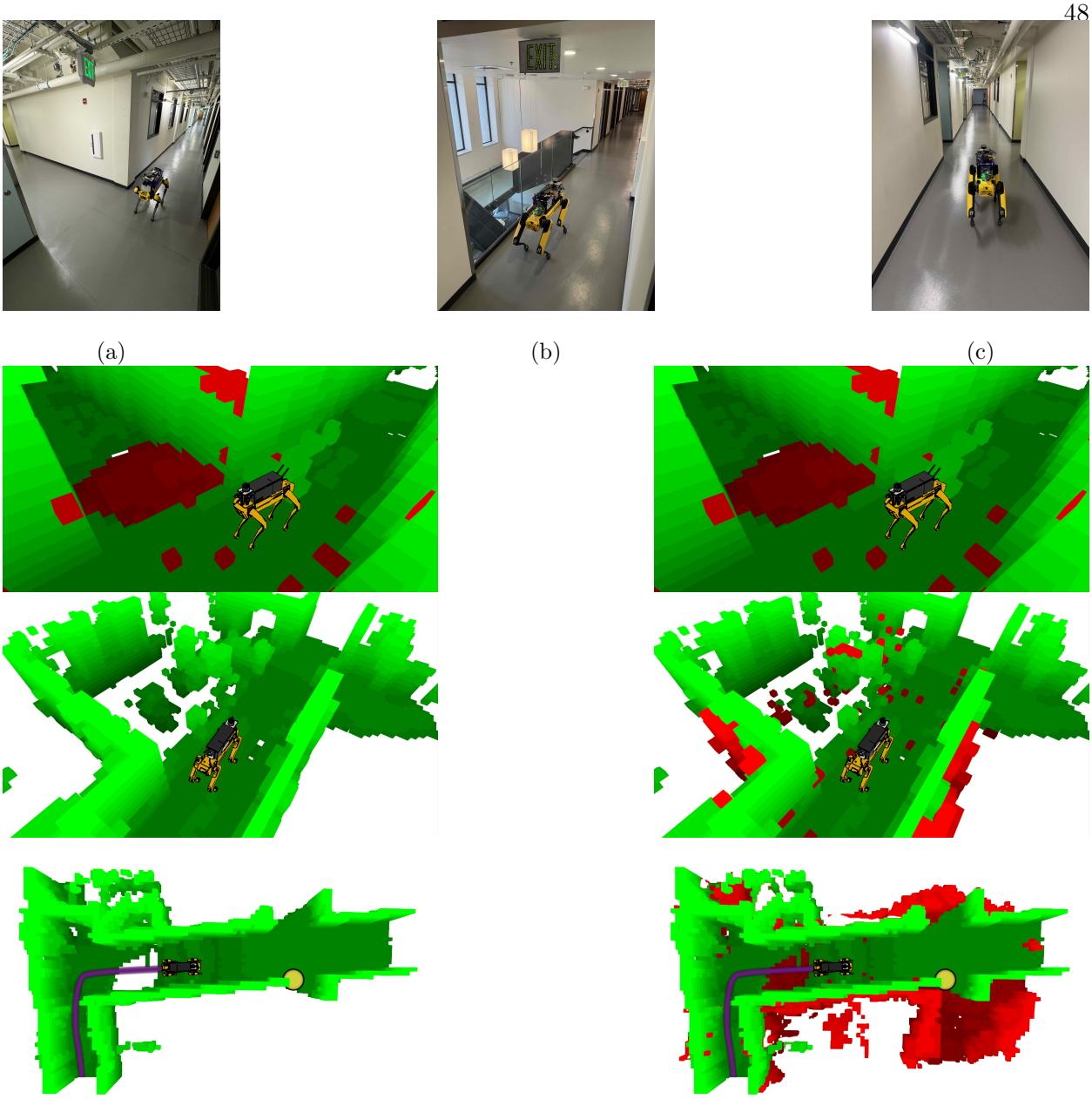


Figure 4.5: **Example Occupancy Predictions**: Scene images at the top of the figure correspond to the 3 pairs of occupancy maps, where (a) corresponds to the top pair of occupancy maps. The left column of occupancy maps shows the vision only map, while the the right column shows the merged vision and 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 as we well as missing wall information that was not observed. **(b)** Spot navigates down a hallway and enter area with a glass railing above the stairs. **(c)**. Spot navigates down a hallway generating predictions along the way.

Results Discussion. The results in Table 4.2 show that RC predictions are quite similar between OSMM and PMM approaches, reducing the FID of the environments by an average of 25% and 28.5% for OSMM and PMM respectively. These results are similar to the simulation-based results presented in [138]. 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 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 the FID [15]. 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 RC and FC results when employing the two map merging methods are due to the distribution of possible predictions given the observed occupancy data. We predict that as the number of unknown voxels grows, 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 4.6

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

Figure 4.6: **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.

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.

Implementation Discussion. The primary challenge associated with adapting SceneSense from simulation to real-world was the noise introduced in the lidar scans, and in particular how errant scans impacted the Octomap ray-casting algorithm. With the current implementation of **occupancy inpainting**, SceneSense is designed to treat observed voxels as ground truth occupancy, propagating any vision errors to the occupancy prediction. Not only does this error propagation keep SceneSense from predicting occupancy in desirable locations, if enough errors accumulate it can result in inpainting that moves the occupancy prediction out of the training distribution, resulting in worse predictions. These vision errors become more common as the lidar beam length increases. We found through testing that $7m$ was the maximum beam length where we could consistently generate reasonable maps.

4.5 Conclusions

In this work we present key architectural changes to the SceneSense [138] 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 probabilistic 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.

Chapter 5

Planning and Control over Predicted Maps

5.1 Introduction

Efficient and intelligent robotic exploration is fundamental to the practical deployment of robotic systems for use in real-world scenarios. In order to intelligently explore, an autonomous agent must have a useful representation of its surroundings and be able to localize within the environment. Robotic agents deployed in search and rescue [151, 12] or disaster response missions [87, 126, 94] rarely have a known map as a prior. This results in slower than desired exploration because the agent can only operate over terrain in the sensor field-of-view.

Given this lack of prior information about the scene agents are constantly operating on incomplete information. In addition to limited field-of-view for mapping, there is also sensor noise, unpredictable dynamic obstacles, obstruction of key information, and visual degradation that an agent may need to reason over to complete a given task. While human agents can filter out noise and fill in gaps in observations by drawing on prior experiences, robots are generally limited to using direct observations without the benefits of prior knowledge or intuition. This results in robotic agents that only take action based on direct observations, a process that reduces efficiency of exploration because of time consuming replanning when informative sensor updates occur.

In related fields, such as self-driving cars, researchers attempt to eliminate the mapping uncertainty using pre-compiled high-definition (HD) maps [110]. This approach provides a good method of localization at the global level, but there still exists uncertainty and missing information locally. Furthermore, this approach scales poorly, as new HD maps are required for any new operating

environment. As discussed previously there are many applications where collating an HD map before navigation is not possible, such as in search and rescue scenarios [38] or extra-terrestrial exploration [173].

With recent breakthroughs in generative AI, neural networks are able to generate convincing human-like results in various modalities, such as text [168], images [145, 85], and audio [47]. We leverage these models to enable robots to make common-sense inference of geometry that is occluded from view. Our model enables a robotic agent to infer geometry through occupancy prediction beyond an obstruction so that the agent can plan a trajectory beyond the field-of-view similar to the way a human would navigate occlusions using intuition.

In this paper we use the occupancy prediction diffusion network SceneSense [138] to generate occupancy information in key areas during exploration. We design a robotic system to run SceneSense online during exploration and develop the probabilistic update rules to merge the diffusion information with a running observed map. We show that with key modifications we use SceneSense to generate realistic occupancy predication at any location in a map, rather than just local to the platform. Further we show the enhanced maps that merge the predicated occupancy simply robotic exploration, solve key perennial issues for autonomous platforms that previously required tedious fine tuning or parameters.

5.2 Related Works

5.2.1 Scene Exploration

An autonomous agent must have a useful representation of its surroundings and be able to localize within the environment to efficiently and intelligently explore. A proven strategy for mapping complex environments for use with autonomous systems is 3D volumetric mapping [1, 2]. OctoMap [77] is a popular framework for 3D mapping due to its memory efficiency and wide adoption in the robotics community [44]. In the DARPA Subterranean (SubT) Challenge [38], Team MARBLE [13] utilized OctoMap on a Boston Dynamics Spot robot, and finished 3rd overall.

Other works [190, 192, 113, 64] have shown considerable success using frontier based exploration strategies. In [57], a frontier-based approach is utilized to demonstrate the efficacy of biasing towards unseen areas of a map for more effective mobile robot exploration. Further efficiency improvements to the frontier based exploration strategy are introduced in [89]. In [64], a frontier based method is proposed for safe and efficient map-building by defining a "safe region" around the robot, allowing it to explore and gather new information while ensuring obstacle-free navigation, although it is limited to mapping at a fixed height, making it unsuitable for more complex environments.

In [169], a volumetric map is incrementally generated from depth sensors and a graph based planner [44] reasons over that map to determine if the terrain is traversable for each vertex of a four-legged robot. This graph traversal requires considerable computation time for determining the status of voxels beneath and around the robot platform. Additionally, this method does not address the issues with path planning if there are obstacles or geometric occlusions in the sensor field of view. Despite the improvements in mapping and frontier based exploration strategies, Team MARBLE [13] reported that there are still challenges with efficient exploration. During the SubT challenge, exploring robots frequently wasted valuable time attempting to explore in previously seen areas rather than extending their trajectories beyond their current frontiers [13]. Influxes of new information from onboard sensors that came when turning corners or exploring new rooms caused the robots to pause for extended periods of time for re-planning, slowing the pace of exploration considerably [13]. The challenge outcomes demonstrated the utility of 3D volumetric mapping using frontier-based exploration for discovery, but also elucidated the need for increased efficiency in the planning and exploration process for time critical missions. Our novel approach improves the efficiency of exploration by reducing ambiguity that leads to long re-planning delays for the robots.

5.2.2 Occupancy Prediction

[177] attempts to address challenges of autonomous navigation in environments with occlusions by utilizing a deep learning-based approach to predict the occupancy distribution of portions of

the environment. Their method, the Occupancy Prediction Network (OPNet), predicts obstacle distributions, utilizing a self-supervised learning technique that generates training data through simulated navigation trajectories with simulated sensor noise. This approach relies on removing portions of the data from the Matterport3D [22] dataset to train the neural network which biases the model to predict occupancy only for the types of occlusions removed from the original dataset. This approach does not scale well to large unseen sections of an environment, is limited to static updates, and does not generalize due to its reliance on a specially curated dataset.

More recently, [79] introduced SelfOcc, a self-supervised method for 3D occupancy prediction using video sequences. SelfOcc converts 2D images into 3D representations with deformable attention layers and uses signed distance fields (SDF) for regularization and occupancy boundary determination. Although SelfOcc performs well with surrounding cameras, it cannot predict occupancy beyond the camera view due to the limitations of SDF in handling occluded geometry.

5.2.3 Scene Synthesis

Diffusion models [158, 72], are a popular and promising technique which have demonstrated impressive results in image [145], video [69], and natural language [78]. Based on these successes, diffusion models are being extended to 3D scene and shape generation. Recent work [108] demonstrates the use of diffusion models for 3D point cloud generation for simple shapes and objects (e.g. tables, chairs). [91] shows successful 3D shape generation from 2D content such as images, and [170] demonstrates similar 3D shape generation but using point cloud datasets rather than images. In LegoNet [201], diffusion models are used to rearrange objects in a 3D scene. In DiffuScene [165], 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 the local robot field of view, thereby bridging the gap between 3D scene generation and practical robotics applications.

5.2.4 Generative AI in Robotics

The body of work devoted to applications of generative AI in robotics is steadily growing. [203] demonstrates that hierarchical generative modeling, inspired by human motor control, can enable autonomous robots to effectively perform complex tasks with robust performance even in challenging conditions. DALL-E-Bot [85], uses web-scale diffusion models for generating an image from a text prompt which the robot utilizes to rearrange real objects in accordance with the image. Diffusion models were also used in [21] to improve robot motion planning by learning priors on trajectory distributions from previously successful plans, a methodology which shows good generalization capabilities in environments with previously unseen obstacles. More recently, diffusion models were shown to successfully generate terrain predictions behind occluded geometries in indoor environments using a single RGB sensor mounted on a mobile robot platform [138]. We build on this work extending the practical applications of diffusion models by demonstrating a novel method for occupancy prediction which shows significant efficiency gains during exploration.

5.3 Methods

5.3.1 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 $m \in \mathbb{M}_{unknown}$, representing free, occupied and unknown space respectively. We seek to identify and evaluate frontiers in $\mathbb{M}$ that can enhance the robot's decision making for potential exploration. In general, "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 [13, 44].

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 $x \subset \mathbb{M}$.

Exploration. During Robotic Exploration we seek to generate the complete map $\mathbb{M}_{comp}$ of a

target area defined as $\mathbb{M}_{target}$ constructed from successive sensor measurements from an onboard sensor $\mathbb{S}$ and odometer measurements $\mathbb{O}$. While completion and alignment of $\mathbb{M}_{comp}$ when compared to $\mathbb{M}_{target}$ is the primary goal of robotic exploration, speed of exploration and naturally by extension volumetric gain per second are secondary goals during autonomous exploration, so long as these goal do not compromised the safety of the system.

5.3.2 SceneSense Architecture

Denoising Network. The denoising network in our method is a modification of the network presented [138]. As the results of the ablation study presented in [138] show little to no reduction in accuracy of prediction by remove the RGBD conditioning data we design our denoising network as an unconditional diffusion model where the only guidance will be occupancy inpainting at runtime. Our denoising network is a U-net constructed from the HuggingFace Diffusers library of blocks [174] and consists of Resnet [70] downsampling/upsampling blocks.

Occupancy Mapping. Occupancy mapping allows for platforms to build a running map of areas that have been measured to contain matter using onboard sensors like lidar or RGB-D cameras. For our framework we use the popular occupancy mapping framework Octomap [77] to generate an occupancy map as the platform explores the environment. Importantly, Octomap provides a probability of occupancy $o \in [0, 1]$ for every voxel in the map that has been observed using pose ray casting. This means that as we explore we will be maintaining not only a map of occupied areas M_o, but also a map of areas that have been measured to not contain any data M_u. These maps will later be used to inform SceneSense where occupancy predictions should be made.

5.3.2.1 Training

During training, we generate a noisy local occupancy map x_t where $t \in [1, T]$ from a ground truth local occupancy map x. We train the diffusion model f_θ to predict the noise applied to x_t.

Data Augmentation. In previous SceneSense implementations, the primary area for prediction

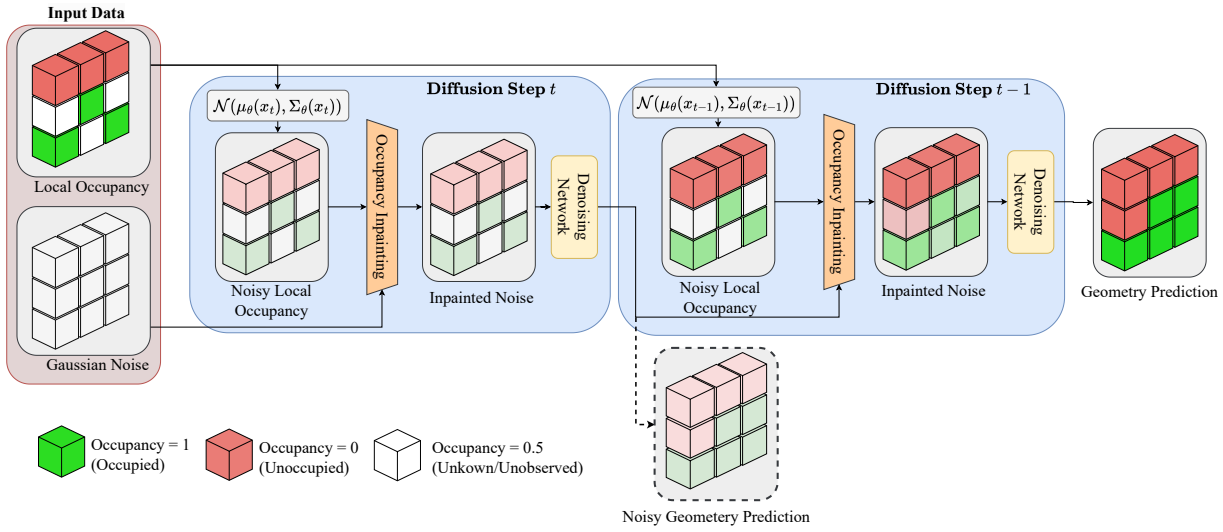


Figure 5.1: **Reverse Diffusion Process**: The reverse diffusion process takes the local occupancy information and the Gaussian noise of the area to be diffused over. Noise commensurate with the current diffusion step is added to the local occupancy information, which includes observed occupied (green) and observed unoccupied (red) data. The result is inpainted into the noisy local occupancy prediction. The inpainted noise data and the feature vectors generated by the perception backbone are provided to the denoising network which generates a new noisy geometry prediction at $t-1$. This process is repeated as the starting noise x_T is iteratively denoised to x_0 which is the final geometry prediction from the framework.

was centered around the robot. In this paradigm the SceneSense training data was collected as local occupancy data collected around the robot. However this training approach struggles to capture the variety of poses possible when doing occupancy predictions at the frontier. The graph based planner where poses are selected can be much closer to structures like walls, and at a wider variety of rotation poses. To account fo this in training we augment the training data to closer mimic that of the graph based poses. For each pose we generate 10 noisy pose samples where we add noise from a uniform distribution of $[-1, 1]$ in the x,y direction and $[0, 2\pi]$ in the rotational plane. This modification increases the training set by $10x$ but yields much more constant results at inference time when performing occupancy prediction around graph nodes.

Occupancy Corruption. To corrupt each ground truth local occupancy map x to train the network we add Gaussian noise to x to generate x_t. This corruption process is defined in eq. (3.3) where the intensity of the noise is controlled by α_t which is configured by a linear noise scheduler [72].

Loss Function. The network f_θ is trained using the calculated l_2 loss between the denoised x_t prediction and the associated ground truth data x. l_2 loss is a popular diffusion loss function, however other loss functions such as cross-entropy loss or mean squared error can be applied and have had some success in similar diffusion frameworks [124, 146, 26, 83].

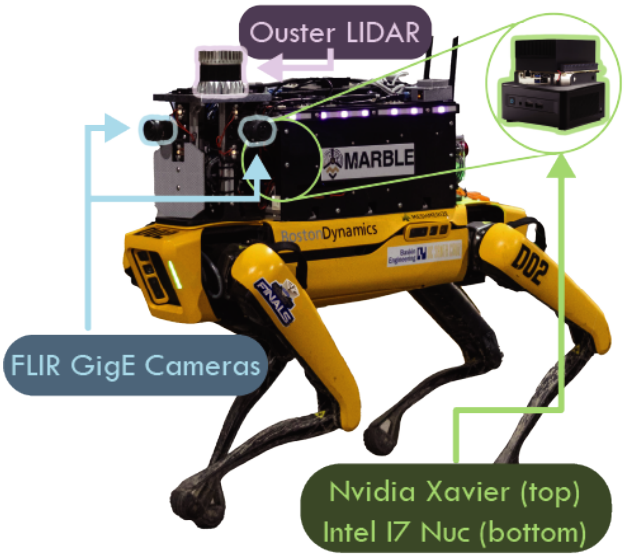


Figure 5.2: Spot robotic platform with onboard compute and sensors. Sensor suite consists of a 64 Beam OS1 lidar as well a 3 FLIR GigE Cameras.

5.3.2.2 Inference

Sampling Process The trained noise prediction network f_θ takes isotropic Gaussian noise $\mathcal{N}(0, \mathcal{I})$ as the starting point x_T to begin the reverse diffusion process. The noise is iteratively removed by using f_θ to compute x_{t-1}.

Occupancy Inpainting. Our method of occupancy inpainting ensures observed space is never overwritten with SceneSense predictions. Additionally occupancy inpainting enhances the predictions from the models seen in fig. 3.6 (c). Inspired by image inpainting methods seen in image diffusion [146] and guided image synthesis methods [116], occupancy inpainting continuously applies the known occupancy information to the diffusion target during inference. To perform occupancy inpainting we select our target region x as a sub map of the full occupancy map. From x we build a occupied map M_o and unoccupied map M_u composed of observed occupied voxels and observed unoccupied voxels respectively. From M_u and M_o we generate noisy representations commensurate with the current diffusion step t using Eq. 3.3, M_{un} and M_{on} respectively. Finally the data from M_{un} and M_{on} replaces the data in the diffusion target at the associated coordinates. This process is

repeated for each inference step and can be seen in fig. 3.2. This method both increases the fidelity of the scene predictions and ensures the diffusion model does not predict or modify geometry in space that has already been observed.

Multiple Prediction. Diffusion is a noisy process that can generate different results given the same context. In image generation this is a desirable characteristic as the framework can generate different results given the same prompt, increasing the diversity of the generated data. As shown in fig. 3.3 SceneSense has the same behavior as these networks and can generate different reasonable predictions based on the same input information. Further there is no compute time increase (assuming enough compute is available) as multiple predictions can be done in parallel. For simplicity we simply generate one prediction at each pose, but additional heuristics or voting schemes could be added to the system to score multiple outputs and select preferable predictions.

5.3.3 Robotic System Architecture

Our robotic system is constructed as a quadruped (Spot) as shown in Figure 4.1 as well as an off-board high performance computer to handle computationally expensive requests. A block diagram of the system is shown in Figure 4.2.

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, 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 [152] to provide localization at run-time. To improve localization performance both the IMU and lidar sensor were fasted to a 6061 aluminum sensor plate. This allowed for a high-precision relative transform between the sensors, reducing the necessity on extrinsic calibration. Further LIO-SAM requires aligned sensor

timestamps and sensor publish rates to be constant. The LORD IMU however allows for large fluctuation in publish rate due to the USB transmission delays. TO reduce the sensitivity the IMU timesamps are adjusted when messages are not received within 15% of the nominal rate. Additionally the lidar sensor uses PTP to synchronize with the onboard computer. These modifications allowed for consistent localization using LIO-SAM.

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 [44]. 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:

$$\textbf{ExplorationGain}(\sigma_i) \quad = \quad e^{-\gamma_{\mathcal{S}}\mathcal{S}(\sigma_i,\sigma_{exp})} \quad \cdot \quad \sum_{j=1}^{m_i} \textbf{VolumetricGain}(v_j^i) e^{-\gamma_{\mathcal{D}}\mathcal{D}(v_1^i,v_j^i)}, \quad (5.1)$$

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

Exploration gain is used to rank nodes at which occupancy prediction should run. Given a minimum node spacing d_m and a maximum number of frontier prediction nodes n_{max}, SceneSense generates occupancy predictions at range, centered around the identified frontiers.

Mapping. The probabilistic volumetric mapping method Octomap [77] was selected as the mapping framework. Octomap was adopted for its log-odds update method for predicting occupied and unoccupied cells. This approach allows for elegant fusion of observed occupancy and predicted occupancy maps. Further discussion on map fusion is provided in Section 5.3.4.

5.3.4 Probabilistic Map Merging

In previous work, predicted occupancy was merged into the running occupancy map in a "fire and forget" approach [138]. A given occupancy prediction was temporarily merged into the running occupancy map by setting the predicted 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 [77].

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 [138], 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.

$$P(m \mid j_{1:t}) = \begin{cases} \left[1 + \frac{1-P(m|d_t)}{P(m|d_t)} \frac{1-P(m|j_{1:t-1})}{P(m|j_{1:t-1})} \frac{P(m)}{1-P(m)}\right]^{-1} & \text{if } m \notin \mathcal{O} \\ \left[1 + \frac{1-P(m|z_t)}{P(m|z_t)} \frac{1-P(m|j_{1:t-1})}{P(m|j_{1:t-1})} \frac{P(m)}{1-P(m)}\right]^{-1} & \text{if } m \in \mathcal{O}. \end{cases} \tag{5.2}$$

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 5.3, 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 [15, 145]. 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. 5.2 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.

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. 5.2 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.

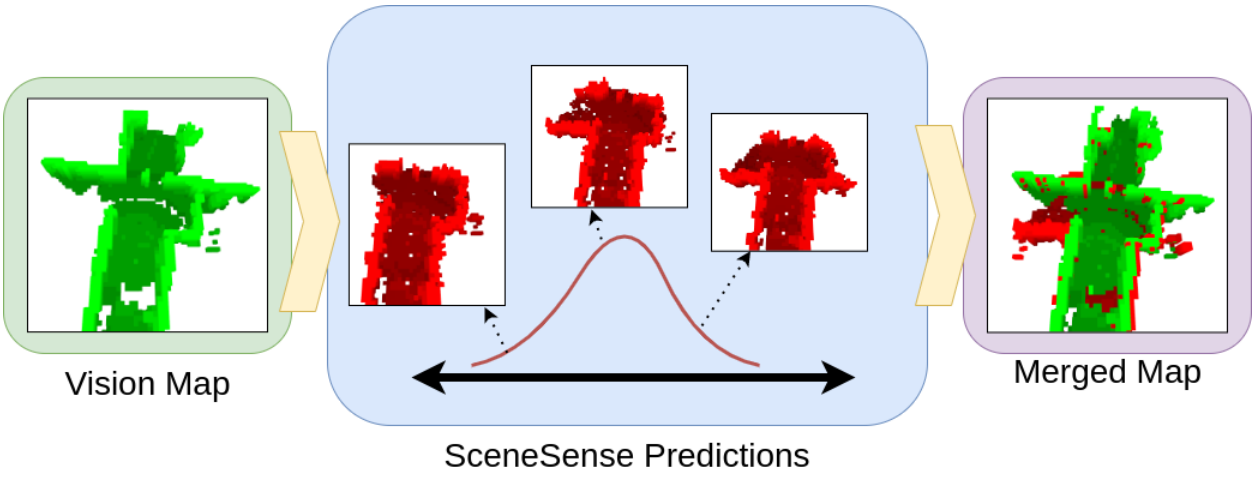


Figure 5.3: **Multi-Prediction Occupancy Merging**: SceneSense predicts various occupancy maps based on equivalent input data that form a distribution. This distribution forms a curve where more likely predictions occur more often, and less likely predictions occur infrequently. These predictions are fused into the merged map using Eq. 5.2. The resulting merged map naturally filters out the unlikely voxel predictions, forming an extended occupancy map.

Exploration Our exploration system is the spot robot shown in figure 5.2. To identify and evaluate potential areas of exploration and generate paths to these target points we use the GBPlanner [44] previously discussed for frontier identification. After GBPlanner publishes the target path for traversal, A simple path follower is used to publish control commands to the spot system. The existing Boston Dynamics Control API is utilized to send simple throttle or turn commands to the spot robot.

5.4 Results

In this section we provide results and evaluations of SceneSense in the real world as well as results and evaluation of the exploration system as a whole. In particular we are interested in measuring the impact that SceneSense has on the exploration system in both select areas where sensor measurements are occluded and general exploration.

5.4.1 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. In previous SceneSense implementations, the primary area for prediction was centered around the robot. In this paradigm the SceneSense training data was collected as local occupancy data collected around the robot. However this training approach struggles to capture the variety of poses possible when doing occupancy predictions at the frontier. The graph based planner where poses are selected can be much closer to structures like walls, and at a wider variety of rotation poses. To account fo this in training we augment the training data to closer mimic that of the graph based poses. For each pose we generate 10 noisy pose samples where we add noise from a uniform distribution of $[-1, 1]$ in the x,y direction and $[0, 2\pi]$ in the rotational plane. This modification increases the training set by $10x$ but yields much more consistent results at inference time when performing occupancy prediction around graph nodes.

We implement the same denoising network structure presented in [138]. It is a U-net constructed from the HuggingFace Diffusers library of blocks [174] and consists of Resnet [70] 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 [88] 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.

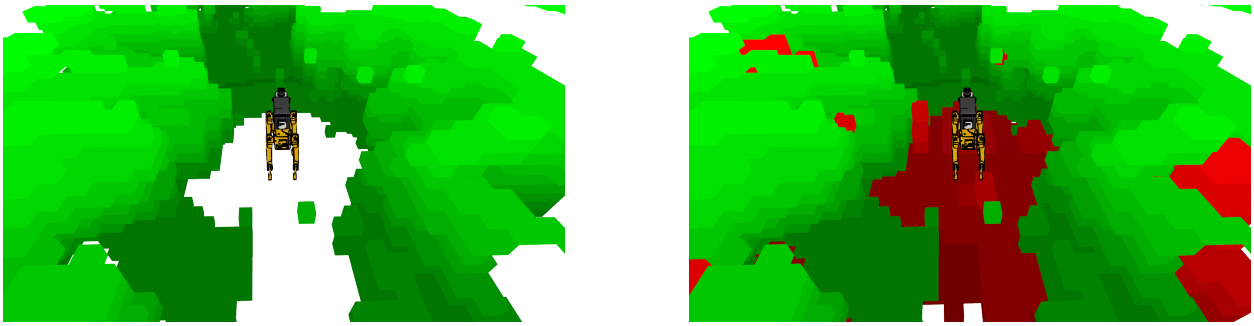
At inference time we evaluate our dataset using an RTX 4070 TI Super GPU for acceleration. The number of diffusion steps is configured to 30 steps.

5.4.2 Solved Robotic Exploration Tasks

Robotic systems often exhibit poor or unpredictable performance in edge cases, particularly when encountering rare or unforeseen scenarios. Many edge cases in robotics are known, and need special error handling processes to circumvent errors when in these configurations. With the addition of SceneSense to our robotic framework we alleviate the need for special error handling for some common edge cases seen during deployment. In particular, the common sense occupancy filling alleviates cases where robotic systems cannot generate plans over ares in which they cannot directly observe although it would be common sense that a traversable path exists. We provide two examples of configurations where planning failed without the probabilistic map merging and one example where traditional terrain filling methods would fail where SceneSense does not.

5.4.2.1 Start Up

. The first configuration where the robotic planner fails in during start up. Given the mounting

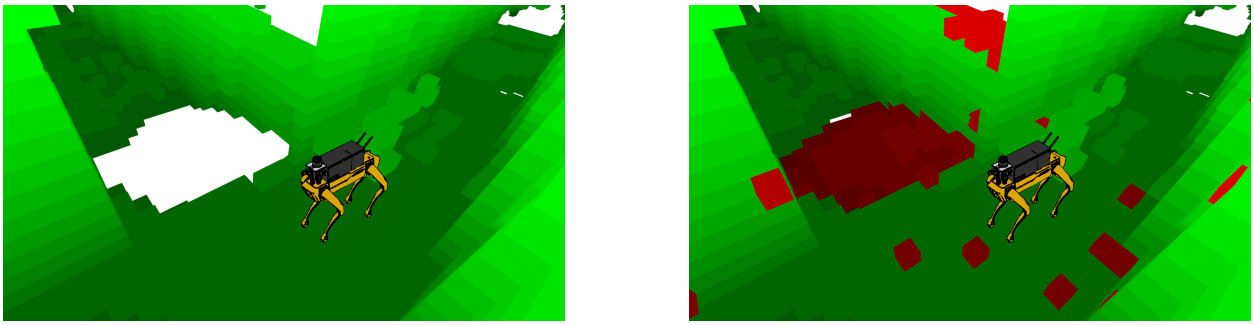


(a) (b)

Figure 5.4: 5.4a: At startup the robot cannot observe the ground directly under it due to the mounting location of the lidar. 5.4b: SceneSense generates occupancy predictions (Red Voxels) that fill in the hole under the robot as well as some of the vertical occluded geometry. With the additional predictions a plan can now be generated and navigation can start.

point of the lidar and height of the robot, there are several meters around the platform that cannot be observed directly by the lidar. At startup this results in a large hole around the platform, which the planner sees as untraversable terrain and therefore is unable to form a plan for traversable as

shown in figure 5.4. Traditionally this problem could be addressed with algorithmic hole filling methods but these methods are brittle as they can fail in unexpected ways when traversing new environments, and require online tuning for fill size. Further these method require specific handling of the "start" configuration requiring additions to the robot exploration method. In addition algorithmic filling methods it can be common to add additional depth sensors intended to point directly at what the robot is walking on. However this approach requires the purchase of additional sensor which can be prohibitively expensive, and would require additional ongoing calibration between the sensor to ensure alignment of measurements. Alternatively, SceneSense is able to fill the hole at startup and immediately begin planning, without special configuration handling.



(a) Subfigure Caption 1

(b) Subfigure Caption 2

Figure 5.5: 5.5a: During traversal, the robot fails to observe the ground directly to its right which will result in the inability to plan down the hallway for exploration 5.5b: SceneSense generates occupancy predictions (Red Voxels) that fill in the hole to the right of robot as well as some of the vertical occluded geometry. With the additional predictions a plan can now be generated and the robot can continue navigation down the hallway.

5.4.2.2 Narrow Hallways and Corners

. During Traversal, perception systems can miss measurements of necessary information to continue exploration. In our case, due the the high mounting point of our lidar sensor, traversal of narrow hallways can lead to missing ground measurements necessary to continue exploration as shown in figure 5.5a. In some sampling based planners for ground systems, it is not allowed to sample points above open space, and therefore the robot will never traverse down the hallway to

continue exploration. SceneSense provides realistic filling in this scenario and allows the planner to continue exploration down the hallway.

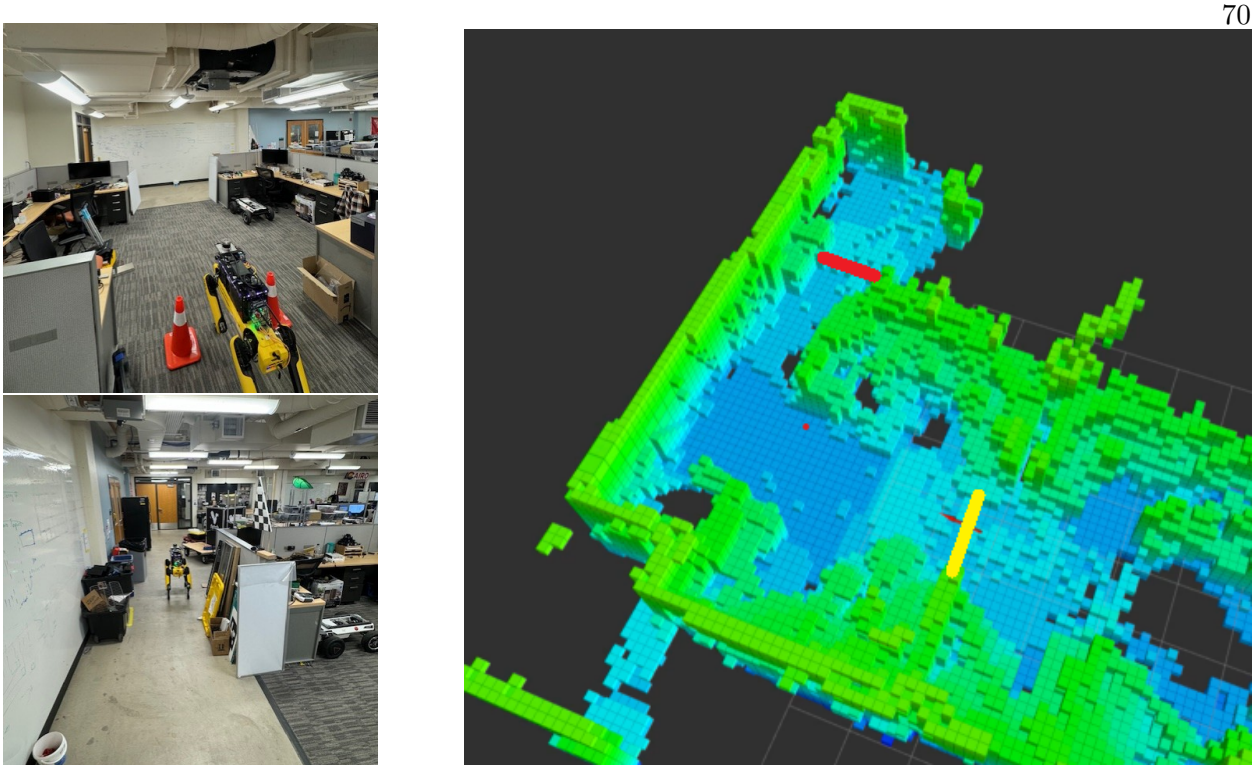
In this scenario we can directly measure the impact that SceneSense has on robot navigation. We provide 2 experiments to evaluate the SceneSense occupancy enhancements when traversing narrow corners. First we run 10 corner traversals in the same location to evaluate consonant of the enhancements. Second, We select 10 different narrow hallway scenarios to evaluate the SceneSense predictions in different environments and generalization abilities. In both scenarios we set a start and end point during the hallway exploration and set the robot to explore. We evaluate the following systems for comparison.

(1) **User Operation Ground Truth**: The system is teleoperated by the user to generate a ground truth result for human-like exploration.

(2) **GBPlanner [44]**: baseline GBPlanner is used to explore the narrow hallway environments.

(3) **GBPlanner + Local SceneSense** : baseline GBPlanner is used to explore the narrow hallway environments. SceneSense predictions are merged with the running occupancy map as the system explores. SceneSense only makes local occupancy predictions (Predictions centered around the robot).

(4) **GBPlanner + Frontier SceneSense** : baseline Planner is used to explore the narrow hallway environments. SceneSense predictions are merged with the running occupancy map as the system explores. SceneSense makes predictions at range for 1 target point in the graph with the highest exploration gain.



(a) (b)

Figure 5.6: **Test Environment**: The test environment used to produce Table 5.1. Figure 5.6a show the starting position of the robot (top) and the goal pose of the robot (bottom). 5.6b is the complete occupancy map representation with the start and finish line shown in yellow and red respectively.

Table 5.1: Single hallway traversal results averaged over 8 runs per configuration. All time measurements are in seconds and distance measurements are in meters

	Exp. Time	Min Time	Max Time	σ_{exp}	Failures
Human Operator	9.35	9.00	9.70	0.34	0
GBPlanner [44]	25.98	18.33	33.77	5.18	1
GBPlanner + Local SS	23.77	18.48	29.27	3.81	0
GBPlanner + Frontier SS	21.10	16.35	28.17	3.53	0

Discussion: As shown in Table 5.1, enhancing the mapping method with local SceneSense results in the best autonomous performance of the system. The performance of GBPlanner + Local SS is

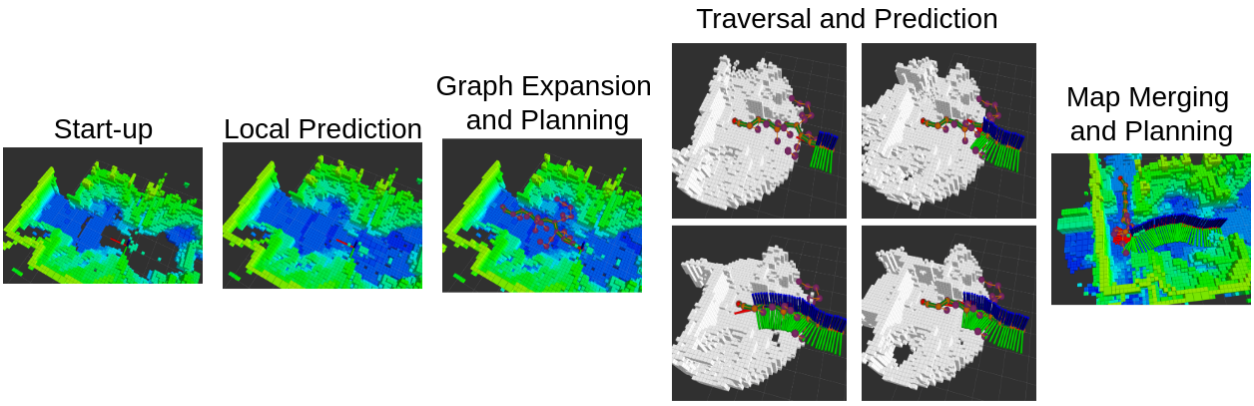
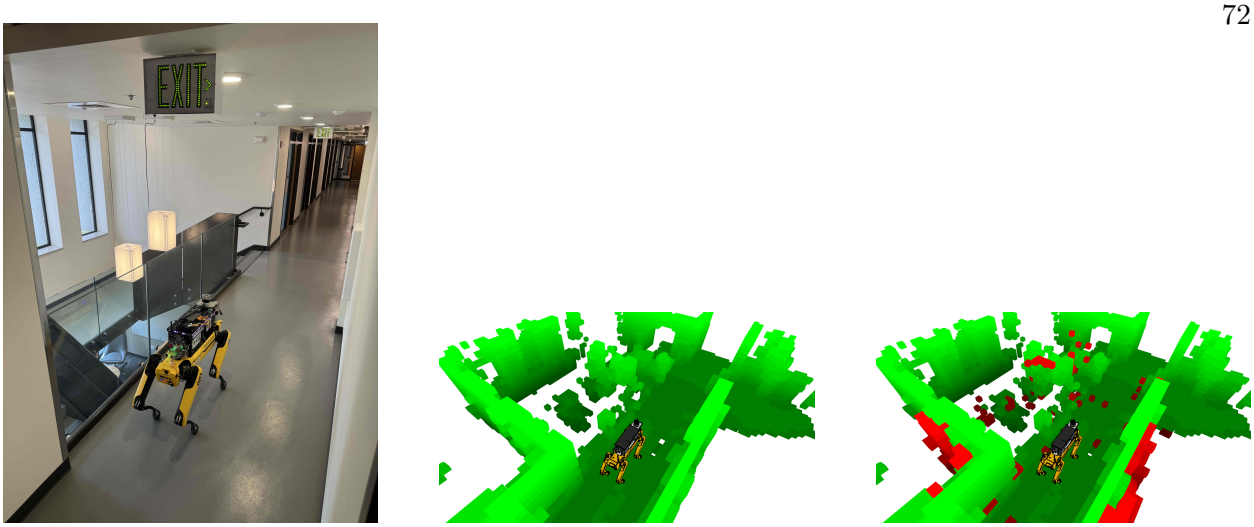


Figure 5.7: **Example traversal of GBPlanner + Frontier SS**. AT startup local diffusion is run to allow for planning from startup. One a graph is constructed and a path is planned the robots begins both traversing the path and making predictions of the frontier occupancy. Upon reaching its destination the graph is extended based on the observed and predicted occupancy and the robot continues exploration.

not only the lowest mean exploration time and path length, but also has a lower variance than the other methods.Further we see that that the frontier centric predictions of SceneSense can produce the fastest single shot exploration, however due to the noise of the diffusion process, also produces the highest variance of results. In general we see that when obstructing obstacles are predicted by frontier predictions it takes a few observations to negotiate the predicted obstacles out of the map, which can result in unpredictable behavior.

5.4.2.3 SceneSense Robustness

While promising, the results presented in Sections 5.4.2.1 and 5.4.2.2 could be solved with simple brute force solution such as removing the ground checking from the graph sampler. To further differentiate SceneSense's from potential brute force solutions we identify adversarial scenarios where areas of the map are not traversable. One such scenario is shown in figure 5.8, where the robot traverses near a glass railing unobservable to the Lidar. While a naive unconstrained sampling approach could sample points and paths above the empty space, SceneSense is able to differentiate this space from other potential ground areas and not predict ground over the edge.



(a) (b) (c)

Figure 5.8: 5.8a: The robot position during exploration near a glass railing, an adversarial scenario for planner that do not take into account traversablity of ground. 5.8c The running occupancy map shows that the ground suddenly ends to the right of the robot, where the railing is protecting navigating off the edge. Samplers that do not check traversablity of space that could have solved the scenarios in Figures 5.8 and 5.4 would sample and allow for paths constructed in open space. 5.8c the red voxels are the predictions generated from SceneSense, showing that while the model will fill unobserved ground in relevant scenarios, it will be able to negation potential adversarial situations such as this glass railing

5.4.3 Full Robot Exploration

In this section we explore the end effects of SceneSense when applied to a robotic exploration platform. For these experiments SceneSense is considered a "drop-in" methods and does not require custom planning or mapping frameworks.

Experimental Setup.

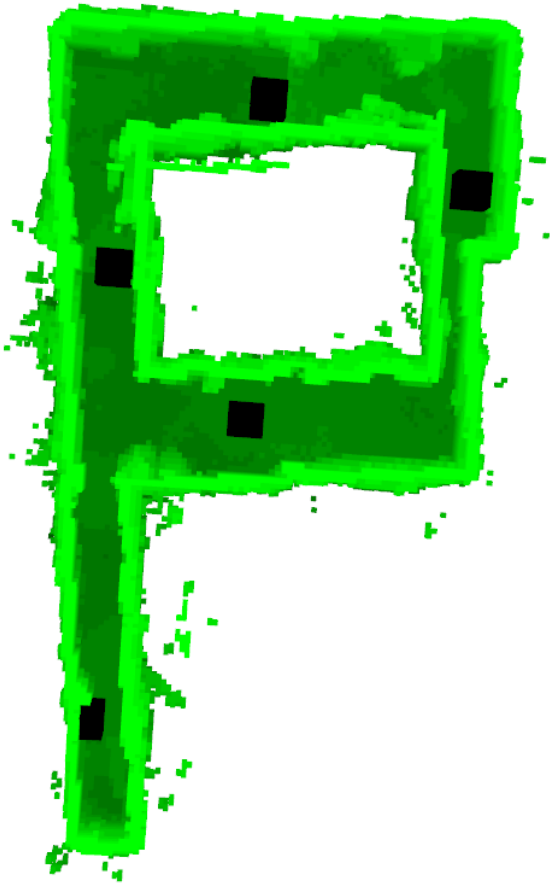


Figure 5.9: **Birds-eye-view of test environment 1**: This is the generated occupancy grid after a full environment exploration. The shade of green of the voxel is mapped to the z-axis of the given voxel, getting lighter as the height increases.

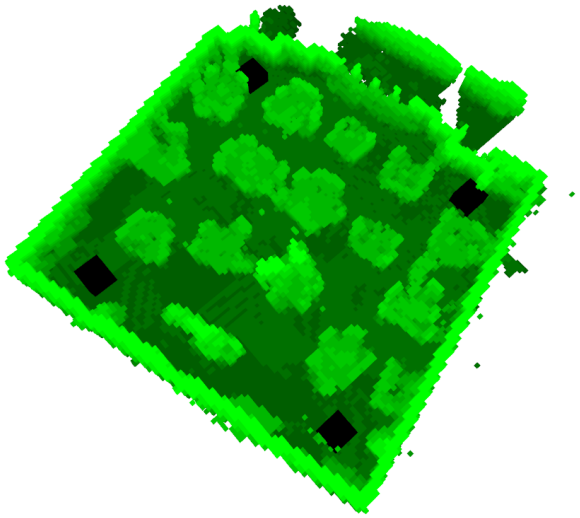


Figure 5.10: **Birds-eye-view of test environment 2**: This is the generated occupancy grid after a full environment exploration. The shade of green of the voxel is mapped to the z-axis of the given voxel, getting lighter as the height increases.

To evaluate the exploration speed of the system we deployed the platform in in a test environments shown in Figure 5.9and 5.10. The robot is started once at each position indicated by the black boxes in the associated figures (5 position in environment 1, 4 positions in environment 2). One run per starting position is then run per configuration and the results are averaged to generate Figures 5.12 and The same testing configurations are used that were used in Section 5.4.2.2. The results of this testing is shown in Figures 5.12 and 5.11 and Tables 5.2 5.3.

Table 5.2: **Exploration Results**: Results from exploration of the environment shown in Figure 5.9. All results are calculated to 95 % of the environment being explored.

	Exp. Time	min Exp. Time	Max Exp. time	σ_{exp}	Avg. $\frac{\text{exp \%}}{sec}$
Human Operator	64.03	36.75	75.05	8.717	1.327
GBPlanner [44]	152.27	57.75	148.45	13.027	0.529
GBPlanner + Local SS	151.83	91.65	141.99	9.629	0.565
GBPlanner + Frontier SS	N.A.	N.A.	N.A.	7.86	N.A.

Table 5.3: **Exploration Results**: Results from exploration of the environment shown in Figure 5.10. All results are calculated to 95 % of the environment being explored.

	Exp. Time	Min Exp. Time	Max Exp. Time	σ_{exp}	Avg. $\frac{\text{exp \%}}{sec}$
Human Operator	55.762	40.950	62.55	10.445	1.516
GBPlanner [44]	106.458	69.679	117.552	5.394	0.624
GBPlanner + Local SS	81.676	74.35	91.05	5.685	0.866
GBPlanner + Frontier SS	N.A.	N.A.	N.A.	6.993	N.A.

Discussion. In both testing environments, The SceneSense enhanced map out performs the vision only map in average exploration time, max exploration time, and average exploration % gain per second. Interestingly the minimum exploration time is faster when operating over the vision only map in both cases. Notability, frontier-centric SceneSense fails to complete the exploration of the space. While the initial rates of exploration with frontier-centric SceneSense are promising, the planning stacks inablitity to disambiguate predicted space and observed space leads to the planner believe the space is fully observed before exploration is actually complete.

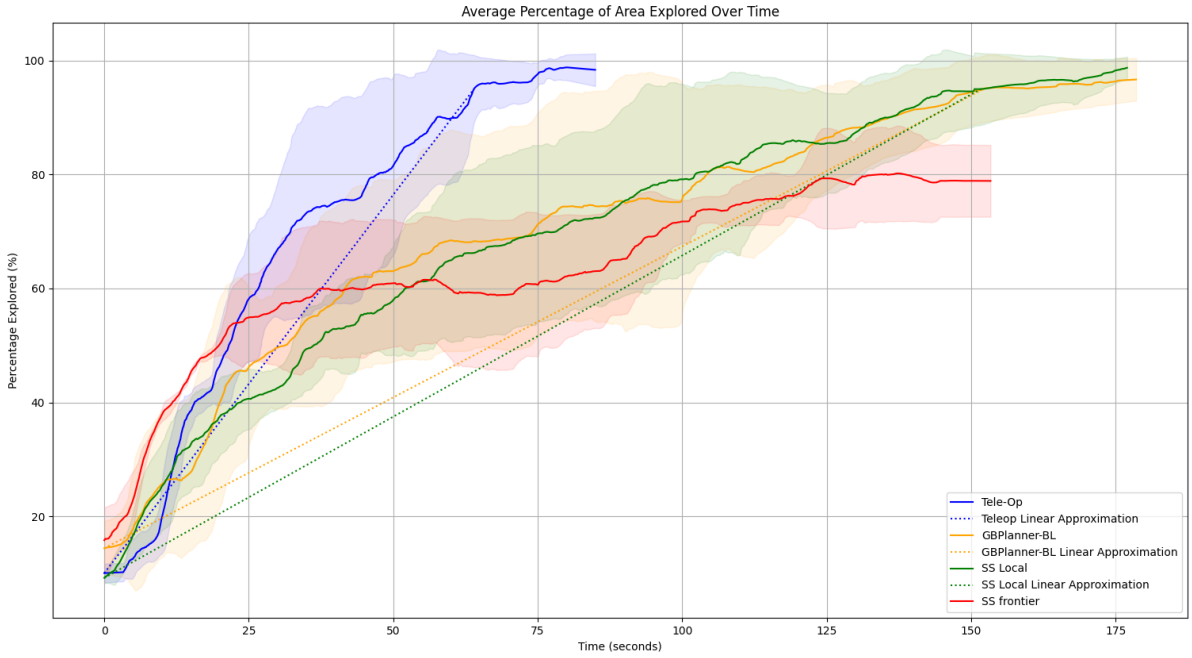


Figure 5.11: **Exploration Percentage over Time env. 1**: Given 5 total runs per configuration, the robot is started once at each of the starting positions shown in Figure 5.9. The average exploration % is shown for each configuration, where the shaded region is the standard deviation across the 5 runs at that timestep. Additionally linear approximations are shown to capture the average exploration rate to achieve 95% exploration of the testing environment.

The minimum time in environment 1 is attributed to the vision only GBPlanner implementation navigating straight down the hallway from the when starting from the dead-end instead of turning right at the first junction as shown in Figure 5.9. With the more complete ground representation, the SceneSense enhanced GBPlanner navigates down the larger hallway to the right first, increasing its overall completion time. Without this outlier existing in the environment 2 data collection, the exploration rate of the SceneSense enhanced planner is increased by 32%.

It is our hypothesis that the substantial exploration rate gains are actually attributed to the startup filling. As shown in Figure 5.13b the startup filling has a lingering effect on the map even after some initial exploration. It is likely the exploration gain in these areas is still significant, resulting

in some planning to want to double back to explore the starting position. Further, the existence of holes in the map can result in unintuitive planning results. As shown in Figure 5.14, a hole in the map leads to a path with a loop back around the hole, configuring the robot in a poor position to continue exploration. By filling these holes during occupancy prediction the planner has a cleaner space to generate potential exploration samples, as well as a reduction in overall explorable space in areas that have already been traversed.

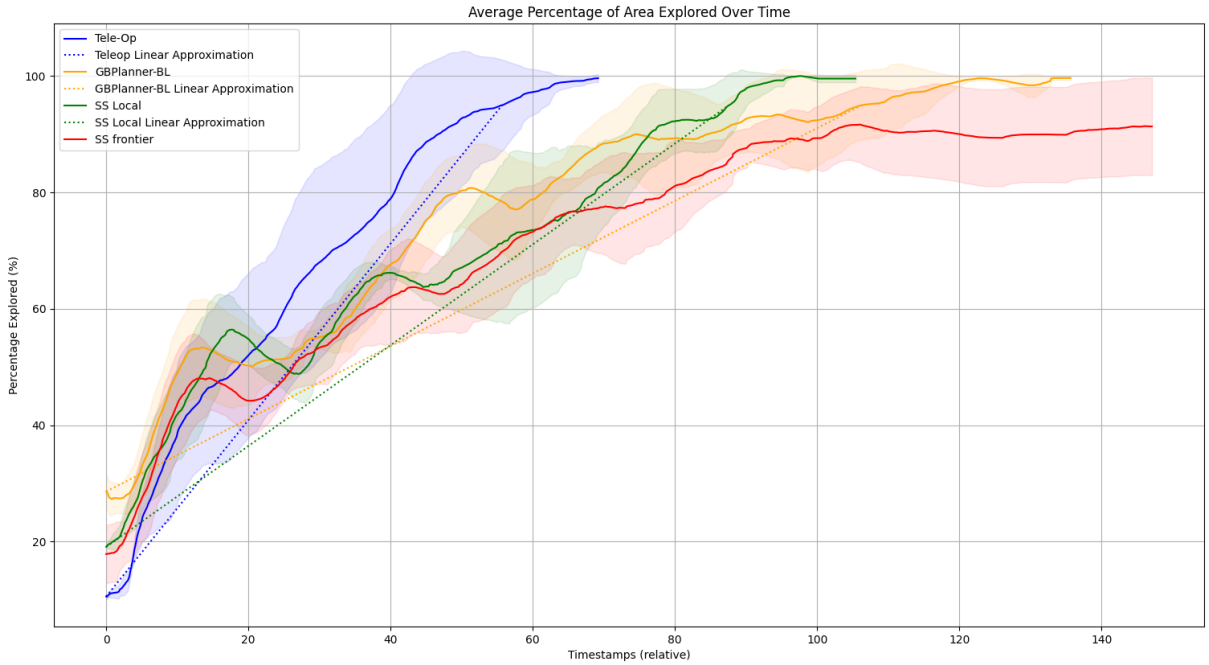
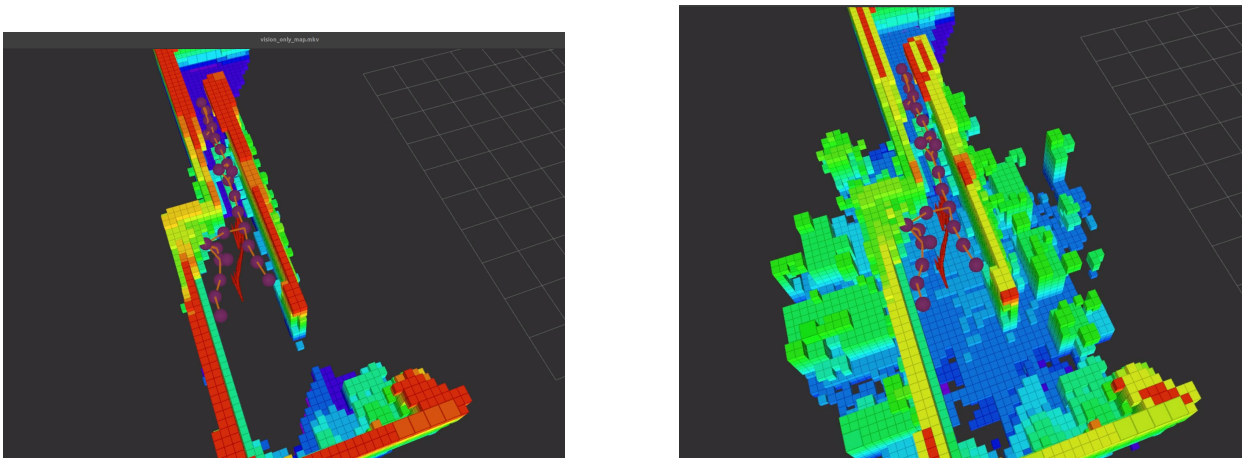


Figure 5.12: **Exploration Percentage over Time env.2**: Given 5 total runs per configuration, the robot is started once at each of the starting positions shown in Figure 5.9. The average exploration % is shown for each configuration, where the shaded region is the standard deviation across the 5 runs at that timestep. Additionally linear approximations are shown to capture the average exploration rate to achieve 95% exploration of the testing environment.



(a) (b)

Figure 5.13: **Vision vs SceneSense at Startup**: Maps and plans generated from the SceneSense enhanced map where the vision only map is shown in (a). The current planning graph is also shown in this image as the purple and red graph structures. **(a)** vision only map after some traversal where large hole occurs in map around the robots starting position. Due to the occlusions from the narrow hallway the robot is traversing the hole is not filled. **(b)** Map where local SceneSense has been running over the same vision only map in (a). Holes are filled and the robot can plan over the starting space.

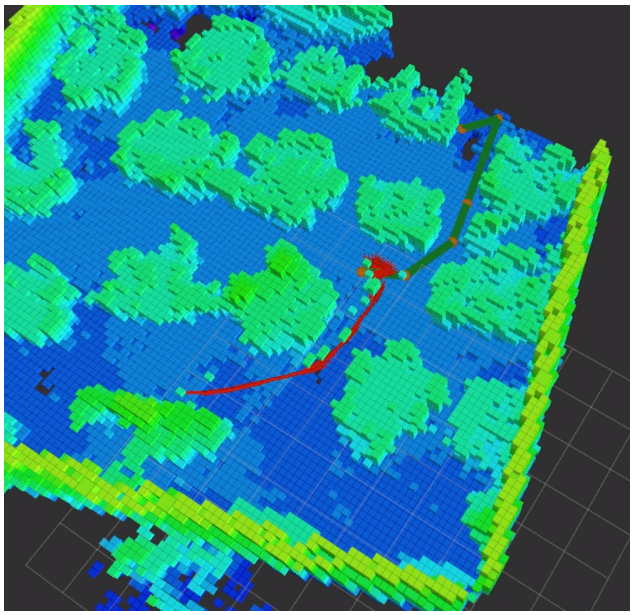


Figure 5.14: **Unintuitive Planning**. As the robot is traversing the classroom environment a hole in the map leads to a loop back path. The robot odometry is shown in read and the planned path is shown in green. Due to the hole in the map the robot plan will configure the robot in a poor position continue exploration.

5.5 Conclusions

In this work we have presented a generative occupancy prediction tool, SceneSense, and implemented the model onboard a real world platform to show the increased effectiveness of a exploration with local Occupancy prediction enabled. While occupancy prediction at range is promising, and has shown the best results in some testing, planners must be developed that can negotiate predicted and observed space in real time to take advantage of occupancy prediction at range. However local occupancy prediction has been shown to both increase the rate of exploration as well as the consistency of exploration.

Chapter 6

Future Work and Conclusions

6.1 Future Work

This work has created a interesting area of research in generative occupancy mapping. SceneSense operates on the assumption the observed occupancy information is to be trusted, and can not modify observed data with the presented occupancy inpainting approach. While this approach provides the convenient guarantee of visual occupancy predictions being enforced, our sensors are known to fail in various scenarios, such as areas of high reflectivity, semi-transparent objects, or in cases of visual degradation [111, 93]. Recent works have used diffusion models for a means of removing corruption introduced in these lidar scans [112]. SceneSense could implement similar methods and in addition to filling and extending scenes, it could remove corruption and noise from running occupancy maps in the same way. Further similar to recent works in on the effectiveness of foundation models [16], SceneSense could be retrained on a broader set of data as a foundation model. This would not only enable further research quickly in other domains but also would like produce better results than our domain specific models.

While SceneSense has been shown to work with an off-the-shelf planner the planner is completely agnostic of whether terrain is observed or predicted. This lack of information could limit the full effectiveness of these methods in more aggressive driving scenarios. More research is needed to develop planners that are conscious of observed space and predicted space, and can make intelligent decision based on it.

6.2 Conclusion

Throughout the course of this work we have developed and evaluated the generative occupancy mapping model *SceneSense*. With these evaluations we have substantiated that SceneSense predictions better represent a fully explored ground truth map than the running occupancy map. Further we have shown that these prediction can be merged into the running maps that are used for exploration planning, and enhance the systems planning and control in particular scenarios. While enhancing our control in targeted scenario, the SceneSense local predictions also have shown to increase the exploration rate and consistency during larger scale exploration test. These results support the robustness of local SceneSense as a "drop-in" planning improvement when integrated with existing planners.

After approaching generative occupancy mapping with modifications to similar models in chapter 2, we adopt diffusion models for occupancy prediction for their realistic generative results shown in the image space. After proving that our model, SceneSense, does enhance the local representation of occupancy maps during robot exploration in chapter 3, we present key modifications to allow for onboard operation at 2Hz on our spot robot in chapter 4. Additionally we present a probabilistic map update method to allow for merging of the predicted occupancy map with the observed map while maintaining the coherency of the overall map. We adopt existing graph-based as a means for identifying and evaluating frontier in the map, which are then used to produce occupancy predictions at range. These predictions are show to have a higher variance in prediction, as the space around the predictions are much more uncertain. These high variance predictions are shown to be successfully negotiated and merged into the running occupancy map using the probabilistic update method. Finally in chapter 5 we perform experiments during robot exploration to evaluate SceneSense's contribution to exploration.

SceneSense solves scenarios where lack of direct observation for map construction interferes with sampling based planners methods for generating reasonable plans. However, a key limitation of this work is the integration of frontier-centric occupancy prediction with existing exploration planners.

We show that while in some scenarios the frontier predictions can be a "drop-in" addition to existing planners, the frontier predictions can de-incentivize exploration in regions of the map where occupancy predictions have been made. These frontier prediction should be differentiated from from the running observed occupancy map to ensure that the planner does not assume exploration has occurred given a frontier predictions.

The work presented in this work validates the thesis statement:

> *By generating unseen geometry from partial observation we can enhance autonomous system's ability to navigate in unknown environments, while maintaining high reliability.*

Bibliography

[1] Shakeeb Ahmad, Andrew B Mills, Eugene R Rush, Eric W Frew, and J Sean Humbert. 3d reactive control and frontier-based exploration for unstructured environments. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 2289–2296. IEEE, 2021.

[2] Shakeeb Ahmad, Zachary N Sunberg, and J Sean Humbert. Apf-pf: Probabilistic depth perception for 3d reactive obstacle avoidance. In 2021 American Control Conference (ACC), pages 32–39. IEEE, 2021.

[3] Tomer Amit, Tal Shaharbany, Eliya Nachmani, and Lior Wolf. Segdiff: Image segmentation with diffusion probabilistic models, 2022.

[4] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023.

[5] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 18208–18218, June 2022.

[6] Zhibin Bao, Sabir Hossain, Haoxiang Lang, and Xianke Lin. A review of high-definition map creation methods for autonomous driving. Engineering Applications of Artificial Intelligence, 122:106125, 2023.

[7] Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In International Conference on Learning Representations, 2022.

[8] G. I. Baylor. Up, up and away. Proc. Roy. Soc., London A, 294:456–475, 1959.

[9] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV), 2019.

[10] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In Proceedings of the IEEE/CVF international conference on computer vision, pages 9297–9307, 2019.

[11] Stuart Bennett. The past of pid controllers. Annual Reviews in Control, 25, 2001.

[12] Markus Bernard, Konstantin Kondak, Ivan Maza, and Anibal Ollero. Autonomous transportation and deployment with aerial robots for search and rescue missions. Journal of Field Robotics, 28(6):914–931, 2011.

[13] Harel Biggie, Eugene R Rush, Danny G Riley, Shakeeb Ahmad, Michael T Ohradzansky, Kyle Harlow, Michael J Miles, Daniel Torres, Steve McGuire, Eric W Frew, et al. Flexible supervised autonomy for exploration in subterranean environments. arXiv preprint arXiv:2301.00771, 2023.

[14] Harel Biggie, Eugene R. Rush, Danny G. Riley, Shakeeb Ahmad, Michael T. Ohradzansky, Kyle Harlow, Michael J. Miles, Daniel Torres, Steve McGuire, Eric W. Frew, Christoffer Heckman, and J. Sean Humbert. Flexible supervised autonomy for exploration in subterranean environments, 2023.

[15] Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In International Conference on Learning Representations, 2018.

[16] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. arXiv preprint arXiv:2212.06817, 2022.

[17] Yingjie Cai, Xuesong Chen, Chao Zhang, Kwan-Yee Lin, Xiaogang Wang, and Hongsheng Li. Semantic scene completion via integrating instances and scene in-the-loop, 2021.

[18] Eduardo F Camacho and Carlos Bordons Alba. Model predictive control. Springer science & business media, 2013.

[19] Anh-Quan Cao and Raoul de Charette. Monoscene: Monocular 3d semantic scene completion. In CVPR, 2022.

[20] Miguel Á. Carreira-Perpiñán. A review of mean-shift algorithms for clustering, 2015.

[21] João Carvalho, An T. Le, Mark Baierl, Dorothea Koert, and Jan Peters. Motion planning diffusion: Learning and planning of robot motions with diffusion models. In 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1916–1923, 2023.

[22] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. arXiv preprint arXiv:1709.06158, 2017.

[23] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. International Conference on 3D Vision (3DV), 2017.

[24] Anas Charroud, Karim El Moutaouakil, Vasile Palade, Ali Yahyaouy, Uche Onyekpe, and Eyo U. Eyo. Localization and mapping for self-driving vehicles: A survey. Machines, 12(2), 2024.

[25] Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling, 2023.

[26] Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. arXiv preprint arXiv:2211.09788, 2022.

[27] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru R. Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions, 2023.

[28] Ting Chen, Lala Li, Saurabh Saxena, Geoffrey Hinton, and David J. Fleet. A Generalist Framework for Panoptic Segmentation of Images and Videos. arXiv e-prints, page arXiv:2210.06366, October 2022.

[29] Ting Chen, Lala Li, Saurabh Saxena, Geoffrey Hinton, and David J. Fleet. A generalist framework for panoptic segmentation of images and videos, 2023.

[30] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning, 2023.

[31] Xiaokang Chen, Kwan-Yee Lin, Chen Qian, Gang Zeng, and Hongsheng Li. 3d sketch-aware semantic scene completion via semi-supervised structure prior, 2020.

[32] Hui-Xian Cheng, Xian-Feng Han, and Guo-Qiang Xiao. Cenet: Toward concise and efficient lidar semantic segmentation for autonomous driving. In 2022 IEEE International Conference on Multimedia and Expo (ICME), pages 01–06. IEEE, 2022.

[33] Ran Cheng, Christopher Agia, Yuan Ren, Xinhai Li, and Liu Bingbing. S3cnet: A sparse semantic scene completion network for lidar point clouds, 2020.

[34] Ran Cheng, Christopher Agia, Yuan Ren, Xinhai Li, and Liu Bingbing. S3cnet: A sparse semantic scene completion network for lidar point clouds. In Conference on Robot Learning, pages 2148–2161. PMLR, 2021.

[35] Ran Cheng, Christopher Agia, Yuan Ren, Xinhai Li, and Liu Bingbing. S3cnet: A sparse semantic scene completion network for lidar point clouds. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, Proceedings of the 2020 Conference on Robot Learning, volume 155 of Proceedings of Machine Learning Research, pages 2148–2161. PMLR, 16–18 Nov 2021.

[36] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2023.

[37] Yen-Yu Chiu, Hiroshi Omura, Hung-en Chen, and Su-Chin Chen. Indicators for post-disaster search and rescue efficiency developed using progressive death tolls. Sustainability, 12, 10 2020.

[38] Timothy H Chung, Viktor Orekhov, and Angela Maio. Into the robotic depths: Analysis and insights from the darpa subterranean challenge. Annual Review of Control, Robotics, and Autonomous Systems, 6:477–502, 2023.

[39] Spconv Contributors. Spconv: Spatially sparse convolution library. `https://github.com/traveller59/spconv`, 2022.

[40] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving, 2020.

[41] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey, 2023.

[42] M. E. Crow. Aerodynamic sound emission as a singular perturbation problem. Stud. Appl. Math., 29:21–44, 1968.

[43] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proc. Computer Vision and Pattern Recognition (CVPR), IEEE, 2017.

[44] Tung Dang, Marco Tranzatto, Shehryar Khattak, Frank Mascarich, Kostas Alexis, and Marco Hutter. Graph-based subterranean exploration path planning using aerial and legged robots. Journal of Field Robotics, 37(8):1363–1388, 2020.

[45] Giannis Daras and Alexandros G. Dimakis. Multiresolution textual inversion, 2022.

[46] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[47] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. arXiv preprint arXiv:2005.00341, 2020.

[48] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021.

[49] Surís Dídac, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. arXiv preprint arXiv:2303.08128, 2023.

[50] Julian D. Dole. Perturbation Methods in Applied Mathematics. Winsdell Publishing Company, 1967.

[51] 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, pages 65–72, 2012.

[52] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[53] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, page 226–231. AAAI Press, 1996.

[54] J. S. Fabnis, H. J. Giblet, and H. McDormand. Navier-stokes analysis of solid rocket motor internal flow. J. Prop. and Power, 2:157–164, 1980.

[55] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image, 2016.

[56] Ben Fei, Weidong Yang, Wen-Ming Chen, Zhijun Li, Yikang Li, Tao Ma, Xing Hu, and Lipeng Ma. Comprehensive review of deep learning-based 3d point cloud completion processing and analysis. IEEE Transactions on Intelligent Transportation Systems, 23(12):22862–22883, 2022.

[57] Luigi Freda and Giuseppe Oriolo. Frontier-based probabilistic strategies for sensor-based exploration. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pages 3881–3887. IEEE, 2005.

[58] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari. Bayesian spatial kernel smoothing for scalable dense semantic mapping. IEEE Robotics and Automation Letters, 5(2):790–797, April 2020.

[59] Kyle Gao, Yina Gao, Hongjie He, Dening Lu, Linlin Xu, and Jonathan Li. Nerf: Neural radiance field in 3d vision, a comprehensive review. arXiv preprint arXiv:2210.00379, 2022.

[60] Alessandro Gasparetto, Paolo Boscariol, Albano Lanzutti, and Renato Vidoni. Path planning and trajectory planning algorithms: A general overview. Motion and Operation Planning of Robotic Systems: Background and Practical Approaches, pages 3–27, 2015.

[61] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pages 3354–3361, 2012.

[62] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. The International Journal of Robotics Research, 32(11):1231–1237, 2013.

[63] Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. Transformer networks for trajectory forecasting. In 2020 25th International Conference on Pattern Recognition (ICPR), pages 10335–10342, 2021.

[64] Héctor H González-Banos and Jean-Claude Latombe. Navigation strategies for exploring indoor environments. The International Journal of Robotics Research, 21(10-11):829–848, 2002.

[65] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 216–224, 2018.

[66] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 10696–10706, June 2022.

[67] F. Guillot and Z. Javalon. Acoustic boundary layers in propellant rocket motors. J. Prop. and Power, 5:331–339, 1989.

[68] Alhasan Hakami, Arun Kumar, Sung J. Shim, and Yousef Abu Nahleh. Application of soft systems methodology in solving disaster emergency logistics problems. World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering, 7:2470–2477, 2013.

[69] William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos, 2022.

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[71] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017.

[72] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[73] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598, 2022.

[74] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models, 2022.

[75] Gabriel M. Hoffmann, Claire J. Tomlin, Michael Montemerlo, and Sebastian Thrun. Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing. In 2007 American Control Conference, pages 2296–2301, 2007.

[76] Sungchul Hong, Antyanta Bangunharcana, Jae-Min Park, Minseong Choi, and Hyu-Soung Shin. Visual slam-based robotic mapping method for planetary construction. Sensors, 21(22), 2021.

[77] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. Autonomous Robots, 2013. Software available at `https://octomap.github.io`.

[78] Rongjie Huang, Zhou Zhao, Huadai Liu, Jinglin Liu, Chenye Cui, and Yi Ren. Prodiff: Progressive fast diffusion model for high-quality text-to-speech, 2022.

[79] Yuanhui Huang, Wenzhao Zheng, Borui Zhang, Jie Zhou, and Jiwen Lu. Selfocc: Self-supervised vision-based 3d occupancy prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 19946–19956, June 2024.

[80] Felipe Inostroza, Isao Parra-Tsunekawa, and Javier Ruiz-del Solar. Robust localization for underground mining vehicles: An application in a room and pillar mine. Sensors, 23(19), 2023.

[81] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In International Conference on Machine Learning, 2022.

[82] Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis, 2022.

[83] Yuanfeng Ji, Zhe Chen, Enze Xie, Lanqing Hong, Xihui Liu, Zhaoqiang Liu, Tong Lu, Zhenguo Li, and Ping Luo. Ddp: Diffusion model for dense visual prediction. arXiv e-prints, pages arXiv–2303, 2023.

[84] R.E. Kalman. A new approach to linear filtering and prediction problems. Journal of Basic Engineering, 82(1):35–45, 1960.

[85] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. IEEE Robotics and Automation Letters, 8(7):3956–3963, 2023.

[86] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. DALL-e-bot: Introducing web-scale diffusion models to robotics. IEEE Robotics and Automation Letters, 8(7):3956–3963, jul 2023.

[87] Shinji Kawatsuma, Mineo Fukushima, and Takashi Okada. Emergency response by robots to fukushima-daiichi accident: summary and lessons learned. Industrial Robot: An International Journal, 39(5):428–435, 2012.

[88] Kate Keahey, Jason Anderson, Zhuo Zhen, Pierre Riteau, Paul Ruth, Dan Stanzione, Mert Cevik, Jacob Colleran, Haryadi S. Gunawi, Cody Hammock, Joe Mambretti, Alexander Barnes, François Halbach, Alex Rocha, and Joe Stubbs. Lessons learned from the chameleon testbed. In Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20). USENIX Association, July 2020.

[89] Matan Keidar and Gal A Kaminka. Efficient frontier detection for robot exploration. The International Journal of Robotics Research, 33(2):215–236, 2014.

[90] Heeseung Kim, Sungwon Kim, and Sungroh Yoon. Guided-TTS: A diffusion model for text-to-speech via classifier guidance. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, Proceedings of the 39th International Conference on Machine Learning, volume 162 of Proceedings of Machine Learning Research, pages 11119–11133. PMLR, 17–23 Jul 2022.

[91] Seung Wook Kim, Bradley Brown, Kangxue Yin, Karsten Kreis, Katja Schwarz, Daiqing Li, Robin Rombach, Antonio Torralba, and Sanja Fidler. Neuralfield-ldm: Scene generation with hierarchical latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8496–8506, 2023.

[92] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. arXiv:2304.02643, 2023.

[93] Lingdong Kong, Youquan Liu, Xin Li, Runnan Chen, Wenwei Zhang, Jiawei Ren, Liang Pan, Kai Chen, and Ziwei Liu. Robo3d: Towards robust and reliable 3d perception against corruptions. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 19994–20006, 2023.

[94] Ivana Kruijff-Korbayová, Robert Grafe, Nils Heidemann, Alexander Berrang, Cai Hussung, Christian Willms, Peter Fettke, Marius Beul, Jan Quenzel, Daniel Schleich, et al. German rescue robotics center (drz): A holistic approach for robotic systems assisting in emergency response. In 2021 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pages 138–145. IEEE, 2021.

[95] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 12689–12697, 2018.

[96] Henry Lao. Linear Acoustic Processes in Rocket Engines. PhD thesis, University of Colorado at Boulder, 1979.

[97] Q. Lao, M. N. Cassoy, and K. Kirkpatrick. Acoustically generated vorticity from internal flow. J. Fluid Mechanics, 2:122–133, 1996.

[98] Q. Lao, D. R. Kassoy, and K. Kirkkopru. Nonlinear acoustic processes in rocket engines. J. Fluid Mechanics, 3:245–261, 1997.

[99] Steven M. LaValle. Rapidly-exploring random trees : a new tool for path planning. The annual research report, 1998.

[100] Jingtao Li, Jian Zhou, Yan Xiong, Xing Chen, and Chaitali Chakrabarti. An adjustable farthest point sampling method for approximately-sorted point cloud data, 2022.

[101] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation, 2022.

[102] Yao-Nan Lien, Hung-Chin Jang, and Tzu-Chieh Tsai. A manet based emergency communication and information system for catastrophic natural disasters. 2009 29th IEEE International Conference on Distributed Computing Systems Workshops, pages 412–417, 2009.

[103] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks. In NIPS, 2017.

[104] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks, 2017.

[105] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.

[106] Zhijian Liu, Haotian Tang, Alexander Amini, Xingyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In IEEE International Conference on Robotics and Automation (ICRA), 2023.

[107] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[108] 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, pages 2837–2845, 2021.

[109] 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 (CVPR), pages 2837–2845, June 2021.

[110] Wei-Chiu Ma, Ignacio Tartavull, Ioan Andrei Bârsan, Shenlong Wang, Min Bai, Gellert Mattyus, Namdar Homayounfar, Shrinidhi Kowshika Lakshmikanth, Andrei Pokrovsky, and Raquel Urtasun. Exploiting sparse semantic hd maps for self-driving vehicle localization. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5304–5311. IEEE, 2019.

[111] K. T. Yasas Mahima, Asanka Perera, Sreenatha Anavatti, and Matt Garratt. Exploring adversarial robustness of lidar semantic segmentation in autonomous driving. Sensors, 23(23), 2023.

[112] KT Yasas Mahima, Asanka G Perera, Sreenatha Anavatti, and Matt Garratt. 3dr-diff: Blind diffusion inpainting for 3d point cloud reconstruction and segmentation. In 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2024.

[113] Alexei A Makarenko, Stefan B Williams, Frederic Bourgault, and Hugh F Durrant-Whyte. An experiment in integrated exploration. In IEEE/RSJ international conference on intelligent robots and systems, volume 1, pages 534–539. IEEE, 2002.

[114] Petri Manninen, Heikki Hyyti, Ville Kyrki, Jyri Maanpää, Josef Taher, and Juha Hyyppä. Towards high-definition maps: a framework leveraging semantic segmentation to improve ndt map compression and descriptivity. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), page 5370–5377. IEEE, October 2022.

[115] Tad McGeer. Passive Dynamic Walking. The International Journal of Robotics Research, 9(2):62–82, 1990.

[116] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In International Conference on Learning Representations, 2022.

[117] Xiangyun Meng, Nathan Hatch, Alexander Lambert, Anqi Li, Nolan Wagener, Matthew Schmittle, JoonHo Lee, Wentao Yuan, Zoey Chen, Samuel Deng, Greg Okopal, Dieter Fox, Byron Boots, and Amirreza Shaban. Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation, 2023.

[118] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020.

[119] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet ++: Fast and accurate lidar semantic segmentation. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4213–4220, 2019.

[120] F. C. Mulick. Rotational axisymmetric mean flow and damping of acoustic waves in a solid propellant. AIAA J., 3:1062–1063, 1964.

[121] F. C. Mulick. Stability of four-dimensional motions in a combustion chamber. Comb. Sci. Tech., 19:99–124, 1981.

[122] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models, 2020.

[123] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In Marina Meila and Tong Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, volume 139 of Proceedings of Machine Learning Research, pages 8162–8171. PMLR, 18–24 Jul 2021.

[124] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. GLIDE: Towards photorealistic image generation and editing with text-guided diffusion models. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, Proceedings of the 39th International Conference on Machine Learning, volume 162 of Proceedings of Machine Learning Research, pages 16784–16804. PMLR, 17–23 Jul 2022.

[125] OpenAI. Chatgpt: Openai's large language model. `https://openai.com/blog/chatting-with-the-internet/`, 2021. Accessed on May 11, 2023.

[126] Shinsuk Park, Yoojin Oh, and Daehie Hong. Disaster response and recovery from the perspective of robotics. International Journal of Precision Engineering and Manufacturing, 18:1475–1482, 2017.

[127] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In CVPR, 2022.

[128] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

[129] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion, 2022.

[130] Xavi Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Ruslan Partsey, Jimmy Yang, Ruta Desai, Alexander William Clegg, Michal Hlavac, Tiffany Min, Theo Gervet, Vladimir Vondrus, Vincent-Pierre Berges, John Turner, Oleksandr Maksymets, Zsolt Kira, Mrinal Kalakrishnan, Jitendra Malik, Devendra Singh Chaplot, Unnat Jain, Dhruv Batra, Akshara Rai, and Roozbeh Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots, 2023.

[131] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017.

[132] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems, 30, 2017.

[133] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[134] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2021.

[135] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

[136] Alec Reed, Lorin Achey, Brendan Crowe, Bradley Hayes, and Chris Heckman. Online diffusion-based 3d occupancy prediction at the frontier with probabilistic map reconciliation. In Submitted and Under Review, 2024.

[137] Alec Reed, Guillaume O Berger, Sriram Sankaranarayanan, and Chris Heckman. Verified path following using neural control lyapunov functions. In Conference on Robot Learning, pages 1949–1958. PMLR, 2023.

[138] Alec Reed, Brendan Crowe, Doncey Albin, Lorin Achey, Bradley Hayes, and Christoffer Heckman. Scenesense: Diffusion models for 3d occupancy synthesis from partial observation, 2024.

[139] Alec Reed and Christoffer Heckman. Looking around corners: Generative methods in terrain extension. arXiv preprint arXiv:2306.07160, 2023.

[140] R. S. Richards and A. M. Brown. Coupling between acoustic velocity oscillations and solid propellant combustion. J. Prop. and Power, 5:828–837, 1982.

[141] Christoph B. Rist, David Emmerichs, Markus Enzweiler, and Dariu M. Gavrila. Semantic scene completion using local deep implicit functions on lidar data. CoRR, abs/2011.09141, 2020.

[142] Barbara Roessle, Jonathan T. Barron, Ben Mildenhall, Pratul P. Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views, 2022.

[143] Luis Roldao, Raoul de Charette, and Anne Verroust-Blondet. 3d semantic scene completion: a survey, 2021.

[144] Luis Roldao, Raoul De Charette, and Anne Verroust-Blondet. 3d semantic scene completion: A survey. International Journal of Computer Vision, 130(8):1978–2005, 2022.

[145] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022.

[146] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[147] Kallol Saha, Vishal Mandadi, Jayaram Reddy, Ajit Srikanth, Aditya Agarwal, Bipasha Sen, Arun Singh, and Madhava Krishna. Edmp: Ensemble-of-costs-guided diffusion for motion planning, 2023.

[148] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(4):4713–4726, 2023.

[149] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[150] Saurabh Saxena, Abhishek Kar, Mohammad Norouzi, and David J. Fleet. Monocular depth estimation using diffusion models, 2023.

[151] Max Schwarz, Tobias Rodehutskors, David Droeschel, Marius Beul, Michael Schreiber, Nikita Araslanov, Ivan Ivanov, Christian Lenz, Jan Razlaw, Sebastian Schüller, et al. Nimbro rescue: solving disaster-response tasks with the mobile manipulation robot momaro. Journal of Field Robotics, 34(2):400–425, 2017.

[152] Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping, 2020.

[153] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. ECCV (5), 7576:746–760, 2012.

[154] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-a-video: Text-to-video generation without text-video data, 2022.

[155] T. M. Smitty, R. L. Coach, and F. B. Höndra. Unsteady flow in simulated solid rocket motors. In 16st Aerospace Sciences Meeting, number 0112 in 78. AIAA, 1978.

[156] Jarrod M Snider. Automatic steering methods for autonomous automobile path tracking. Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08, 2009.

[157] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR.

[158] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In International conference on machine learning, pages 2256–2265. PMLR, 2015.

[159] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022.

[160] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image, 2016.

[161] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[162] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models, 2020.

[163] Anthony Stentz. Optimal and efficient path planning for partially-known environments. In Proceedings of the 1994 IEEE international conference on robotics and automation, pages 3310–3317. IEEE, 1994.

[164] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[165] Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. Diffuscene: Scene graph denoising diffusion probabilistic model for generative indoor scene synthesis, 2023.

[166] Joseph D. Taum. Investigation of flow turning phenomenon. In 20th Aerospace Sciences Meeting, number 0297 in 82. AIAA, 1982.

[167] Lyne P Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 383–392, 2019.

[168] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971, 2023.

[169] Marco Tranzatto, Mihir Dharmadhikari, Lukas Bernreiter, Marco Camurri, Shehryar Khattak, Frank Mascarich, Patrick Pfreundschuh, David Wisth, Samuel Zimmermann, Mihir Kulkarni, Victor Reijgwart, Benoit Casseau, Timon Homberger, Paolo De Petris, Lionel Ott, Wayne Tubby, Gabriel Waibel, Huan Nguyen, Cesar Cadena, Russell Buchanan, Lorenz Wellhausen, Nikhil Khedekar, Olov Andersson, Lintong Zhang, Takahiro Miki, Tung Dang, Matias Mattamala, Markus Montenegro, Konrad Meyer, Xiangyu Wu, Adrien Briod, Mark Mueller, Maurice Fallon, Roland Siegwart, Marco Hutter, and Kostas Alexis. Team cerberus wins the darpa subterranean challenge: Technical overview and lessons learned, 2022.

[170] Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, Karsten Kreis, et al. Lion: Latent point diffusion models for 3d shape generation. Advances in Neural Information Processing Systems, 35:10021–10039, 2022.

[171] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017.

[172] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[173] Vandi Verma, Jeremy Nash, Lucas Saldyt, Quintin Dwight, Haoda Wang, Steven Myint, Jeffrey Biesiadecki, Mark Maimone, Andrei Tumbar, Adnan Ansar, Gerik Kubiak, and Robert Hogg. Enabling long & precise drives for the perseverance mars rover via onboard global localization. In 2024 IEEE Aerospace Conference, pages 1–18, 2024.

[174] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. `https://github.com/huggingface/diffusers`, 2022.

[175] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection, 2020.

[176] Lizi Wang, Hongkai Ye, Qianhao Wang, Yuman Gao, Chao Xu, and Fei Gao. Learning-based 3d occupancy prediction for autonomous navigation in occluded environments. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4509–4516, 2021.

[177] Lizi Wang, Hongkai Ye, Qianhao Wang, Yuman Gao, Chao Xu, and Fei Gao. Learning-based 3d occupancy prediction for autonomous navigation in occluded environments. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4509–4516. IEEE, 2021.

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

[179] Yida Wang, David Joseph Tan, Nassir Navab, and Federico Tombari. Learning local displacements for point cloud completion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 1568–1577, June 2022.

[180] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds, 2019.

[181] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In The Eleventh International Conference on Learning Representations, 2023.

[182] Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. Lego-net: Learning regular rearrangements of objects in rooms. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 19037–19047, 2023.

[183] Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos A. Theodorou. Aggressive driving with model predictive path integral control. In 2016 IEEE International Conference on Robotics and Automation (ICRA), pages 1433–1440, 2016.

[184] Julia Wolleb, Robin Sandkühler, Florentin Bieder, Philippe Valmaggia, and Philippe C. Cattin. Diffusion models for implicit image segmentation ensembles, 2021.

[185] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1912–1920, 2015.

[186] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1912–1920, 2015.

[187] Zhaoyang Xia, Youquan Liu, Xin Li, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, and Yu Qiao. Scpnet: Semantic scene completion on point cloud. In IEEE Conference on Computer Vision and Pattern Recognition, 2023.

[188] Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In Proceedings of the IEEE/CVF international conference on computer vision, pages 5499–5509, 2021.

[189] Yan Xu, Xinge Zhu, Jianping Shi, Guofeng Zhang, Hujun Bao, and Hongsheng Li. Depth completion from sparse lidar data with depth-normal constraints, 2019.

[190] Brian Yamauchi. A frontier-based approach for autonomous exploration. In Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation', pages 146–151. IEEE, 1997.

[191] Brian Yamauchi. A frontier-based approach for autonomous exploration. Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation', pages 146–151, 1997.

[192] Brian Yamauchi, Alan Schultz, and William Adams. Mobile robot exploration and map-building with continuous localization. In Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146), volume 4, pages 3715–3720. IEEE, 1998.

[193] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 3101–3109, 2021.

[194] Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. Diffsound: Discrete diffusion model for text-to-sound generation, 2023.

[195] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications, 2023.

[196] Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation, 2022.

[197] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation, 2017.

[198] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 11784–11793, June 2021.

[199] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. CVPR, 2021.

[200] Takuma Yoneda, Luzhe Sun, Bradly Stadie, Ge Yang, and Matthew Walter. To the noise and back: Diffusion for shared autonomy. arXiv preprint arXiv:2302.12244, 2023.

[201] Sihao Yu, Fei Sun, Jiafeng Guo, Ruqing Zhang, and Xueqi Cheng. Legonet: A fast and exact unlearning architecture, 2022.

[202] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In Proceedings of the IEEE/CVF international conference on computer vision, pages 12498–12507, 2021.

[203] Kai Yuan, Noor Sajid, Karl Friston, and Zhibin Li. Hierarchical generative modelling for autonomous robots. Nature Machine Intelligence, 5(12):1402–1414, 2023.

[204] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In 2018 international conference on 3D vision (3DV), pages 728–737. IEEE, 2018.

[205] Robert A. Zeddini. Injection-induced flows in porous-walled ducts. AIAA Journal, 14:766–773, 1981.

[206] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation, 2022.

[207] Yiming Zhao, Lin Bai, and Xinming Huang. Fidnet: Lidar point cloud semantic segmentation with fully interpolation decoding. 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4453–4458, 2021.

[208] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion, 2021.