Occupancy Map Prediction for Improved Indoor Robot Navigation

Vishnu D. Sharma, Jingxi Chen, Abhinav Shrivastava, and Pratap Tokekar

Abstract—In the typical path planning pipeline for a ground robot, we build a map (e.g., an occupancy grid) of the environment as the robot moves around. While navigating indoors, a ground robot’s knowledge about the environment may be limited by the occlusions in its surroundings. Therefore, the map will have many as-yet-unknown regions that may need to be avoided by a conservative planner. Instead, if a robot is able to correctly infer what its surroundings and occluded regions look like, the navigation can be further optimized. In this work, we propose an approach using pix2pix and UNet to infer the occupancy grid in unseen areas near the robot as an image-to-image translation task. Our approach simplifies the task of occupancy map prediction for the deep learning network and reduces the amount of data required compared to similar existing methods. We show that the predicted map improves the navigation time in simulations over the existing approaches.

I. INTRODUCTION

To navigate from one location to another, a mobile robot needs to know the map of the environment to find the path while avoiding obstacles. This information can either be provided as prior information by mapping the environment beforehand, or the robot can build a map in an online fashion using the sensors available to it. Occupancy maps are often used for this purpose, which provide probabilistic estimates about the free (Navigable) and occupied (Non-navigable) areas in the scene. The estimates can be updated as the robot navigates and gains more information about the environment.

Online mapping reduces the deployment time and is especially suitable for navigating in previously unexplored environments. RGBD cameras are widely used for mobile robot sensing and the recent advancement in computer vision, fuelled by deep learning, have improved the understanding of the scene for the robots. However, the mapping and navigation capability of a robot can be limited by the information captured by the onboard sensors. The onboard sensors like cameras are generally limited by the field-of-view and occlusions when mapping the environments. This visibility-based limitation results in the difficulty of using the camera to map the cluttered environment (Figure 1). Lack of knowledge limits the robot’s speed as well since the robot cannot move fast on a path passing through unknown areas lest it collides with an obstacle before being able to brake and come to halt. To move quickly, the robot should have more knowledge about the environment. While adding external cameras to the environment can help with these challenges, it requires the user to have access to the environments and also increases the cost of deployment.

In this work, we propose a deep learning-based approach to extend the sensed occupancy information to nearby areas. Specifically, we use an inpainting network over the occupancy map generated by the mobile robot and predict the occupancy information about the scene as if the robot has access to two cameras on its sides (Figure 2). We limit the region for prediction to nearby areas to simplify the task of inpainting by focusing only on the areas which overlap with the sensed region and thus the robot has some prior information about them. Since we do not force the model to make predictions without context, we lower the chances of overfitting and erroneous predictions.

The input and target maps for the deep learning model are generated by processing the images captured and eliminates the need of manual annotation that similar tasks may require. This process is also efficient as the robot does not require addition of a new hardware and can collect the require information by simple movement. The proposed setting further reduces the data requirement as the robot doesn’t need to explore the whole environment to collect the ground truth information.

Following are our main contributions in this work:
1) We present inpainting models for indoor occupancy maps and provide comparative study of their efficacy in terms of prediction accuracy and extent of inpainting to highlight the effect of different training methods.
2) We simulate the mobile robot navigation utilizing the predicted occupancy map and compare against existing
methods to highlight the advantages of our approach. Our experiments show that with the proposed approach, we can obtain an appreciably well performing prediction model that reduces the distance traversed and results in more successful navigation compared to other methods.

II. RELATED WORK

Mapping the environment is a standard preparation step for autonomous navigation. The mapping problem is coupled with the localization problem and is usually known as Simultaneous Localization and Mapping (SLAM). The traditional mapping approaches use local sensor observations, from Cameras, Lidar or Sonar, and turn them into some occupancy measure to represent the environment. The representative works include GMapping [1], a Rao-Blackwellized particle filter for occupancy-grid mapping, and Cartographer [2], a graph-based SLAM technique.

The problem of inpainting the occupancy map falls under the umbrella of data imputation, which has been studied extensively by the computer vision community. The closely aligned sub-topic of image inpainting [3] has seen tremendous progress and improvement over the past year, owing to the advancements by deep learning. Most of these works focus primarily on color images, but similar architectures have been adapted for different input domains like medical images [4], 3D point clouds [5], optical flow [6], etc. The mobile robot typically requires 3-dimensional information about the scene for navigation, obtained by depth image, and can be combined with color and semantic information for richer representation. This information can be represented by point clouds or 3d voxels, and then further distilled into occupancy maps for navigation. Occlusion in the scene and the robot’s field-of-view constraints can lead to missing information in these representations.

Some existing works employ deep learning to learn or enhance the representations to improve navigation. Song et al. [7] proposed SSCNet to predict volumetric and semantic information from a single depth image. Kniaz et al. [8] proposed an approach to extract similar information with a RGB image. Missing information in the 3D presentations can be inpainted with an encoder-decoder network like PCN [9], or with generative models like Point Encoder GAN [5]. For ground robot navigation, information about the 2D ground plane is enough, and hence the inpainting can be reduced to a 2D data inpainting problem, simplifying the problem and reducing the computational requirements. We also focus on 2D occupancy map inpainting.

Katyal et al. [10] compare the efficacy of networks based on ResNet [11], UNet [12] and GAN [13] for 2D occupancy map inpainting, and show that U-Net outperforms the others. They later use the U-Net-based approach to also represent the prediction uncertainties. However, these approaches rely on expensive LIDAR sensors for data collection. Ramakrishnan et al. [14] use RGBD cameras for sensing and use U-Net for inpainting the occupancy maps. However, the robot needs to explore the whole environment to build the map which is then used as the ground truth for training. This approach is specifically time-consuming when the trained network is to be fine-tuned on the real-world data for deployment.

Wei et al. [15] present an alternative approach by limiting the region-of-interest for inpainting. In this approach the robot already has a camera at the height of 1.5m, tilted towards the ground and another camera is placed at the height of 2m with tilt. The model is trained to use the occupancy map obtained from only the lower camera as input and predict the combination of the maps from both the cameras. The motivation for this approach is to get a view similar to a human being. The network, also U-Net, thus learns to predict the area occupied by the obstacles. This approach
is thus data-efficient, does not require manual annotation, and is shown to work well on real robots. However, this approach trained the model in a supervised fashion to predict the information added by the higher camera, given the occupancy map from the lower camera. This setting is not able to predict the edge-like obstacles well. Also, the real-world deployment of this method requires adding a camera to the robot for collecting data for fine-tuning. Lastly, titled cameras limit the information about the scene ahead as compared to straight, forward-looking cameras.

To overcome the aforementioned issues, we adopt the approach by Wei et al. [15] for predicting the occupancy maps obtained by cameras in the same horizontal plane. We use additional cameras to the left and right side of the robot-mounted camera, at the same height and looking straight. This makes our approach similar to traditional occupancy inpainting methods, but we require a lesser amount of data without the need for manual annotation. Given, the efficacy of UNet for the occupancy map inpainting task, we also use UNet as part of a GAN networks to predict the occupancy map resulting from the combination of the information for the three cameras. When deploying in the real world, the left and right camera information can be obtained by moving the robot at the corresponding location and capturing the RGBD images, and thus they do not impose any additional cost.

III. APPROACH

In this work, we consider a ground robot equipped with an RGBD camera in indoor environments. We used simulation environment to get multiple camera views, as well as the depth maps. In a real-world setting, this can be achieved by driving the robot around in the environment. In the following subsections, we detail the data collection process, network architecture, and training details.

A. Data Collection

We used AI2THOR [16] simulator which provides photorealistic scenes, corresponding depth and segmentation maps, and the flexibility to add multiple cameras to the environment. We add an RGBD camera, referred to as CamCenter, at the height of 0.5m from the ground and at the same location as the robot. Two more cameras are added to this setup, one towards the left (CamLeft) and the other towards the right side of the robot (CamRight), as shown in Figure 2a at the same height h = 0.5m, but a distance d = 0.3m away horizontally from the robot. However, each camera is rotated by 30 deg from its principal axis to look towards the area directly in front of the robot. This is done to capture extra information about the scene, while also making sure that the cameras on sides have some overlap with CamCenter. The rotation of the cameras virtually increases the field-of-view for the robot and the translation makes sure than the robot is able to learn to look around the corners rather than simply rotation at its location.

Each camera captures the depth image and the instance segmentation image. While the depth image helps with creating a 3D re-projection of the scene into point clouds, the latter helps with identifying the ceiling, which is excluded from the occupancy map generation process, and the floor, which acts as the free/navigable area. The rest acts as the occupied/non-navigable area for the robot. This segmentation-based processing step is used to process the information precisely and can be replaced with identifying the ceiling and floor using the height of the points after the re-projection. The point clouds are then projected back to the top-down view as we consider the problem of navigation on the 2D plane. For CamLeft and CamRight, these views are first transformed from camera-frame to robot-frame using rotation and translation. Then we limit the maps to 5m × 5m area in front of the robot and convert them to 256 × 256 images to use in the network. Each point belonging to the obstacles contributes to the addition of 1 point to the corresponding cell and the points for floor contribution to a subtraction of 1 point. The number of points in each bin is multiplied with a factor \( m = 0.01 \) converting the map to an occupancy map with log-odds. To prevent the map from having too large log-odds values, we clip the number of points in each to the range \([-10, 10]\). The resulting map from CamCenter, referred to as \( O_c \), acts as the input to the network. Similar to Wei et al., [15] we construct the ground truth map \( O^* \), as the combination of the maps from the three cameras as follows:

\[
O^* = \max\{abs(O_c), abs(O_l), abs(O_r)\} \cdot \text{sign}(O_c + O_l + O_r)
\]

(1)

where \( O_c, O_l \) and \( O_r \) refer to the occupancy maps generated by CamCenter, CamLeft, and CamRight, respectively. These log-odds maps are converted to probability maps before being used for network training.

AI2THOR provides different types of rooms. We use living rooms only as they have a larger size and contain more obstacles compared to others. There are 30 such rooms available, of which we use the first 20 rooms for training and validation, and the rest for testing. To collect the maps, we divide the floor into square grids of size 0.5m and capture data by rotating the cameras to 360 deg in steps of 45 deg. As some maps do not contain a lot of information to predict due to the robot being closed to the walls, we filter out the pairs of the map where the number of occupied cells in \( O^* \) is more than 20%. This process provides us with 6000 map pairs for training and 2000 pairs for testing the network.

B. Network Architecture and Training Details

We use PyTorch implementation of UNet to take the \( O_c \) as input and predict \( O^* \), after converting them from log-odds maps to probability maps. Since the underlying task is learning shape priors with a limited amount of data, we treat it as an image-to-image translation task and use pix2pix [17] architecture. In this generative adversarial network (GAN), we use UNet for generator and discriminator due to its success with pixel-to-pixel prediction problems like segmentation due to its ability to predict the boundaries well using connections between encoder and decoder layers. Our network consists of a five-block encoder and a five-block
The training objective is to achieve a generator $G$ and discriminator $D$ over input $x$ and the target map $y$ as defined below

$$L_{L1}(G) = \mathbb{E}_x[\|y - G(x)\|_1]$$

$$L_{GAN}(G, D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_x[\log(1 - D(G(x)))]$$

The training objective to achieve a generator $G^*$ such that

$$G^* = \arg\min_G \max_D \mathbb{E}_y[\log D(y)] + \lambda \mathbb{E}_x[\log(1 - D(G(x)))]$$

As the inputs and output maps are probabilistic, for training UNet-pred we use KL-divergence as loss function. KL-divergence over the the target distribution $p(x)$ and the input distribution $q(x)$ is defined as $KL(p||q) = \sum_x p(x) \log(\frac{p(x)}{q(x)})$, where $X$ is the target data distribution. Assuming each map as a multivariate Bernoulli distribution parametrized by the probability of each cell, the KL-divergence can be written as $KL(p||q) = \sum_i p_i \log \left( \frac{p_i}{q_i} \right) + \sum_i (1 - p_i) \log \left( \frac{1 - p_i}{1 - q_i} \right)$ which can be further simplified as the binary cross-entropy (BCE) loss by eliminating the parts dependent only on $p(x)$ as they do not depend on the network parameters. This problem can also be treated as a regression problem, with the target probabilities acting as the target values, thus we also trained a model with Mean Squared Error (MSE) loss functions.

IV. EXPERIMENTS & RESULTS

A. Inpainting Network and Evaluation

The network is trained on a GeForce RTX 2080 GPU, with a batch size of 4 for UNet-GAN and 16 for UNet-Pred. Early stopping is used to avoid overfitting the network with the maximum number of epochs set to 300. $\lambda$ is set to 10 for training GAN-UNet.

To evaluate the performance we consider only the predictions for which we have an entry in the corresponding ground truth map. We call them inpainted cells. A cell is considered to be free if the probability $p$ in this cell is lesser than 0.495. Similarly, a cell with $p \geq 0.505$ is considered to be occupied. The remaining cells are treated as unknown and are not considered in the evaluations. We apply this treatment to both the predicted map and the ground truth map and then find the accuracy of correctly identifying the free and occupied cells. Note that this step is also needed as the deep learning model may not be able to predict an exact value of 0.5 for unknown areas, as used in the input maps.

UNet-GAN model achieved an accuracy of 86.27%, whereas UNet-Pred trained with MSE loss achieves 83.34% compared to 79.89% accuracy of UNet-Pred with the BCE loss. UNet-GAN is better at making high confidence prediction in the new cells as marked by higher number of cells inpainted by it compared to the other two models. While UNet-Pred is able to inpaint only 4.34% and 4.58% of the cells in the input map, UNet-GAN inpaints 6.95% cells, which is relatively closer to number of cells added when using additional cameras (8.51%). Figure 3 shows that it is able to make highly accurate predictions. Figure 5 shows some example input maps, ground truth maps, and prediction by the three models. We observed that UNet-GAN is able to predict the edges of the obstacles much better than the other models. On the other hand, both the UNet-Pred models predict blurry maps with the prediction condensed near the robot (left edge of the image). They also seem to extend the area near the field of view edges in the inputs maps. The blurring effect is much more pronounced with MSE loss compare to BCE. IN comparison, UNet-GAN predictions give a sense of shape completions as observed in the sofa at the top in the first example in Figure 5.

B. Navigation Simulation and Evaluation

We perform the navigation simulation in AI2THOR living room FloorPlan227 due to its larger size and more number of obstacles compared to the other rooms we set aside for network testing. In each simulation we first randomly generate a start location (src), a destination (dst) and an initial yaw for the robot. At each step, the robot generates an occupancy map using the RGB-D camera. The robot also

| Method       | Accuracy | Cells Inpainted |
|--------------|----------|-----------------|
| UNet-Pred (BCE) | 79.89%   | 4.34%           |
| UNet-Pred (MSE) | 83.34%   | 4.58%           |
| UNet-GAN     | 86.27%   | 6.95%           |

Fig. 3: Histogram of the prediction accuracy (%) for UNet-GAN
keeps a global occupancy map in the memory. We use the perceived height to filter out the ceiling and other obstacle situated at a height higher than the robot, rather than using the ground truth semantic maps for filtering. We assume that the robot knows its global location and yaw without error. The robot keeps a global. An augmentation method, described below, modifies the occupancy map and updates its global map using the same method as used for the ground truth generation (Eq. 1). This global map is transformed into a cost map for for the path planner. The path planner uses Dijkstra’s algorithm to find the shortest path between $src$ and $dst$. The robot then navigates to the next waypoint prescribed by planner, moving in the steps of 20cm, if it is within 1° of the robot’s line-of-sight. Otherwise, the robot rotates to face the waypoint, and then moves. The room is discretized into a grid of square cells with side 0.2m. The simulation ends either if the $dst$ is within one diagonal cell at max, or if the robot has moved and rotated $S_{max}=100$ times, which we found is enough to reach the $dst$ if the robot doesn’t get stuck.

For path planning, we use inpainted maps instead of predicted maps, i.e. the predicted state of a cell is overwritten by the corresponding observed state. The sensed occupied and free areas are given preference over the predicted counterparts in the maps and the costs are accordingly weighted in the cost maps. The observed free and occupied areas have 1 and 100000 as weights, respectively, compared to 2 and 1000 for the predicted equivalents. The unknown areas carry a weight of 10. We keep the speed of the robot proportional to its confidence about the free cells on the predicted path. Thus the robot moves faster when it is confident about not colliding with an obstacles on the path. If there are possible obstacles on the path, it should keep its speed lower to be able to break and stop before collisions. The cost of the traversal is equal to time taken to reach the goal.

We compare the proposed occupancy map inpainting methods $UNet-GAN$ and $UNet-Pred$ with MSE, due to its higher prediction accuracy, against an navigation approach relying solely on the occupancy map generated by $Cam_{Cent}$, referred to as $Normal$. Additionally, we compare these methods against the ground truth approach used in data collection i.e. when $Cam_{Left}$ and $Cam_{Right}$ provides additional information about the scene. As the metric we use a modified form of SPL, $Success$ weighted by (normalized inverse) $Path$ Length[18], calling it $Success$ weighted by (normalized inverse) $Path$ Duration (SPD). We define it as

$$SPD = \frac{1}{N} \sum_{i=1}^{N} S_i \frac{l_i}{\max(p_i, l_i)}$$ (5)
where, $N$ is the number of test episode, $S_i$ a binary variable indicating success in the $i^{th}$ episode, $t_i$ is the traversal time on the shortest path between the source and destination and $p_i$ is the measured traversal time by the robot in this episode. In our case, $S_i$ is 1 only when the robot is within one cell away, number of total steps don’t exceed $S_{\text{max}}$, and the simulator doesn’t run into error during simulation. $l_i$ is found by running simulation on the global map generated using the reachable positions at the start of the simulations. This map doesn’t depend on sensing and thus isn’t updated during an episode. $l_i$ is calculated as the time taken for traversal by the robot in each simulation. A higher value of SPD indicates higher success rate and the $p_i$ being closer to $l_i$, thus a method with higher SPD is preferred over others.

Table II summarizes our findings for the navigation experiments in $N = 100$ episodes. UNet-GAN provides a relative improvement in SPD of 2.97%. It also has a higher success rate compared to other augmentation method. UNet-Pred, on the other hand, is slightly worse than Normal, suggesting that training as a prediction task may not help in learning the shape priors well. While the ground truth approach is much better than the Normal method with a slightly worse than UNet-Pred in training as a prediction task, it induces noise in the output. The ground truth approach is much more difficult to learn due to the non-differentiability of the loss function.

| Method          | SPD   |
|-----------------|-------|
| Normal          | 0.556 |
| Ground Truth    | 0.641 |
| UNet-Pred       | 0.552 |
| UNet-GAN        | 0.572 |

V. Conclusion

In this paper, we investigated the problem of occupancy map enhancement as an image inpainting problem with a GAN and a direct prediction based approach to aid robot navigation in presence of obstacles in the scene. We observed that GAN-based approach result in better inpainting, both qualitatively and quantitatively, compared to supervised-prediction based approach, and helps the robot move faster by making inferences about the part of the maps occluded by the obstacles. We found that the models trained with supervised setting try to be more accurate by performing high confidence predictions in the areas closer to the robot. It may be further improved by adding more information in the regions away from the robot or by applying weights to increase the importance of the areas away from the robot. Our navigation experiments show that good prediction inpainting methods can be realized with a lighter, and easier to collect data collection strategy.

The proposed method can be further extended to study the effect of different placements of CamLeft and CamRight on robot navigation. Other camera placements that do not require additional equipment is also a promising directions to be explored. Lastly, we utilize only the occupancy maps for network training in this work. The method may benefit from leveraging other sensor modalities for prediction and planning.

REFERENCES

[1] G. Grisetti, C. Stachniss, and W. Burgard, “Improved techniques for grid mapping with rao-blackwellized particle filters,” IEEE Transactions on Robotics, vol. 23, no. 1, pp. 34–46, 2007.
[2] W. Hess, D. Kohler, H. Rapp, and D. Andor, “Real-time loop closure in 2d lidar slam,” in 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 1271–1278.
[3] M. Bertalino, G. Sapino, V. Caselles, and C. Ballester, “Image inpainting,” in Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000, pp. 417–424.
[4] K. Armanious, Y. Mecky, S. Gatidis, and B. Yang, “Adversarial inpainting of medical image modalities,” in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 3267–3271.
[5] Y. Yu, Z. Huang, F. Li, H. Zhang, and X. Le, “Point encoder gan: A deep learning model for 3d point cloud inpainting,” Neurocomputing, vol. 384, pp. 192–199, 2020.
[6] K. Luo, C. Wang, N. Ye, S. Liu, and J. Wang, “Occipflow: Occlusion-inpainting optical flow estimation by unsupervised learning,” arXiv preprint arXiv:2006.16637, 2020.
[7] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, “Semantic scene completion from a single depth image,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1746–1754.
[8] V. V. Kniaz, V. A. Kayaz, F. Remondino, A. Bordodymov, and P. Moshkantsev, “Image-to-voxel model translation for 3d scene reconstruction and segmentation,” in European Conference on Computer Vision. Springer, 2020, pp. 105–124.
[9] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, “Pcn: Point completion network,” in 2018 International Conference on 3D Vision (3DV). IEEE, 2018, pp. 728–737.
[10] K. Katyal, K. Popok, C. Paxton, J. Moore, K. Wolfe, P. Burlina, and G. D. Hager, “Occupancy map prediction using generative and fully convolutional networks for robot navigation.”
[11] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” corr abs/1512.03385 (2015), 2015.
[12] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical image computing and computer-assisted intervention. Springer, 2015, pp. 234–241.
[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” Advances in neural information processing systems, vol. 27, 2014.
[14] S. K. Ramakrishnan, Z. Al-Halah, and K. Grauman, “Occupancy anticipation for efficient exploration and navigation,” in European Conference on Computer Vision. Springer, 2020, pp. 400–418.
[15] M. Wei, D. Lee, V. Isler, and D. Lee, “Occupancy map inpainting for online robot navigation,” in 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 8551–8557.
[16] E. Kolbe, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, “Ai2-thor: An interactive 3d environment for visual ai,” arXiv preprint arXiv:1712.05474, 2017.
[17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–1134.
[18] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, et al., “On evaluation of embodied navigation agents,” arXiv preprint arXiv:1807.06757, 2018.