2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Semantic Mapping with Confidence Scores through Metric Embeddings and Gaussian Process Classification

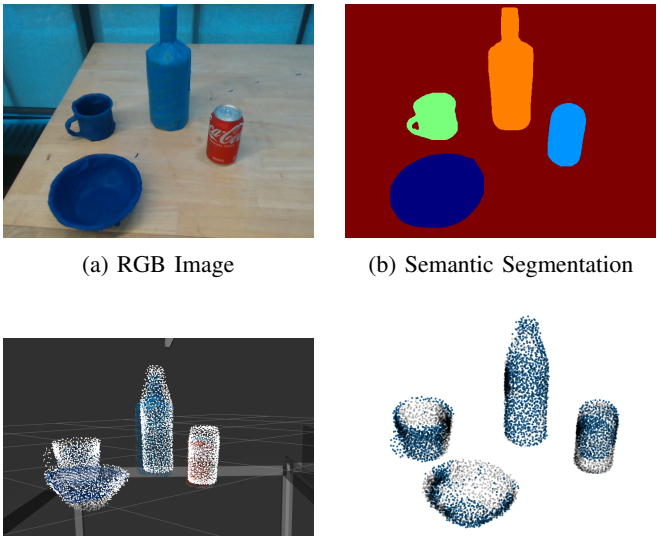
Jungseok Hong[1,2*], Suveer Garg[1], Volkan Isler[1,2*]

***Abstract*— Recent advances in robotic mapping enable robots to use both semantic and geometric understanding of their surroundings to perform complex tasks. Current methods are optimized for reconstruction quality, but they do not provide a measure of how certain they are of their outputs. Therefore, algorithms that use these maps do not have a way of assessing how much they can trust the outputs. We present a mapping approach that unifies semantic information and shape completion inferred from RGBD images and computes confidence scores for its predictions. We use a Gaussian Process (GP) classification model to merge confidence scores (if available) for the given information. A novel aspect of our method is that we lift the measurement to a learned metric space over which the GP parameters are learned. After training, we can evaluate the uncertainty of objects' completed shapes with their semantic information. We show that our approach can achieve more accurate predictions than a classic GP model and provide robots with the flexibility to decide whether they can trust the estimate at a given location using the confidence scores.**

I. Introduction

Mapping a robot's environment with both semantic and geometric information has become significant in recent years as the demand for performing complex robotic tasks has increased. For example, a robot cannot perform tasks such as autonomous driving and manipulation without a semantic understanding of its surroundings. Recent advances in computer vision enable a robot to infer more accurate semantic information from RGBD images. With semantic information, significant progress has been made in the semantic mapping problem [1]. Latest studies [2]–[4] proposed a probabilistic approach to merging incoming semantic information into a map. However, fusing semantic information from different sources still remains a challenge especially when only some of the data sources provide a measure of their uncertainty and others do not.

In this paper, we present a probabilistic mapping approach for scenarios where only partial views of objects are visible to robots. We add a semantic segmentation module to obtain a semantic understanding of the scene, and a shape completion module to obtain the complete shapes of the objects from their partial views (Fig. 1c). To evaluate the reliability of these inferences, our approach uses a *Metric GP*, a GP classification model with metric embedding defined in a unit-sphere metric space. We then unify the semantic information and shape completion using the evaluated values called confidence scores.



(a) RGB Image (b) Semantic Segmentation

(c) Shape Completion (d) Uncertainty Evaluation

Fig. 1: An example scenario of real-world experiments: (a) An RGB image only provides a partial view of the scene, (b) Semantic labels for observed objects, (c) Predicted complete shapes (white color), (d) Complete shapes are filtered with confidence scores. Predictions from (c) are colored in gray/black, and uncertain parts of the complete shapes are marked with blue spheres.

More specifically, when a robot observes objects in its environment via an RGBD camera, we infer their semantic segmentations (Fig. 1b) and confidence scores from an RGB image (Fig. 1a) while predicting the complete shapes (Fig. 1c) of the objects from an RGBD image. Once we have information from both sources, our metric learning model produces embeddings for each segmented pointclouds for training the metric GP model. The GP model performs inferences for the semantic class of complete shape points and their confidence scores. The confidence scores from the metric GP and segmentation models are multiplied for each point, providing confidence scores for all the complete shape points. With these scores, our method enables robots to filter predictions (Fig. 1d) with confidence scores. The filtered points can be used to map the environment.

By using the metric embeddings of pointclouds for building the GP classification model, we show that the metric GP can model correlations between the semantic and geometric information of pointclouds better than a classic GP model,

[1] are with the Samsung AI Center New York, 837 Washington St, New York, NY 10014. [2] are with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, 55455. * Work performed solely as a member of Samsung AI Center NY despite some authors being affiliated with a university.

which uses only the (x, y, z) coordinates to train the GP. With the learned correlation, the metric GP can evaluate the uncertainty of the objects' complete shapes as confidence scores. This enables us to compute confidence scores even if inputs do not provide the scores of their predictions. In this way, we can use the estimated scores for shape completion with given confidence scores for semantic segmentation together to yield the final confidence scores of the merged information.

We evaluate our approach with both simulation and real-world settings. In the simulation setup where we have the ground truth, we compare the prediction performance of the metric GP model against the classic GP model with and without applying the shape completion modules. We also qualitatively evaluate our method in a real-world setup. Our results demonstrate that the metric GP model with shape completion can predict unseen parts of objects better than the classic GP model.

The main contributions of our study are:

- We present a probabilistic approach to merge various types of information from multiple sources in a unified fashion.
- Our approach outputs confidence scores which can be used as a measure of uncertainty across the entire scene.
- We introduce a way to obtain metric embeddings of given pointclouds in a unit-sphere metric space.
- We model the correlation between semantic and geometric information with the metric GP model.
- We evaluate our method in both simulation and a real-world robotic arm platform.

II. Related Work

Many researchers have studied robotic mapping [1] over several decades. Seminal techniques such as occupancy grid-based representations [5]–[7], Simultaneous Localization And Mapping (SLAM) [8] have been introduced in the last few decades. Geometric maps resulting from such methods allow robots to navigate their environments accurately, but some early works [5], [6] use a fixed grid size. However, memory consumption from such methods becomes a bottleneck when robots need finer resolution maps or are deployed in large-scale environments. Hornung *et al.* [7] introduced Octomap which uses an Octree data structure and voxel representation to address these issues. Other representations have also been explored to address the memory bottleneck. Newcombe *et al.* [9] proposed KinectFusion which uses a Truncated Signed Distance Field (TSDF) representation to build a map as a part of their SLAM method. Henry *et al.* [10] presented RGB-D Mapping with surfel [11] representation. These are powerful geometric representations, but do not store semantic information.

To allow robots to perform higher-level tasks (*e.g.,* navigate and grasp a target object), researchers proposed approaches to create a map containing both semantic and spatial information of given environments (*i.e.,* semantic mapping [12]–[16] and semantic SLAM [17]–[19]). Semantic mapping approaches [16], [18] generally add semantic information on top of the existing mapping framework without considering the dependency between spatial and semantic information.

Recent advances in deep learning models [20], [21] provide accurate scene understanding from an RGB image. Merging the semantic information of images from different views for mapping has not received much attention. The merging process needs to be robust to both drastic view changes and the lack of generalization of deep neural network approaches. To handle such challenging cases, it is necessary to consider both geometric and semantic features during the merging process.

Researchers [22]–[24] use Gaussian Process (GP) [25] to model the underlying data correlation for the occupancy mapping problem. They formulate occupancy mapping as a regression problem and infer a map with GP. Their results [23], [24] show better mapping performance with GP than standard occupancy mapping [26], which assumes independence between neighboring cells. The success of GP in occupancy mapping motivated several efforts to apply it for semantic mapping. Unlike GP regression, GP multi-class classification requires approximate techniques for computing the latent variables' posterior distribution [27]. This is one of the challenges for adopting GP classification, and researchers reformulate the multi-class problem as multiple binary classifiers [28] or regression problem [29], [30].

Recently, there has been increasing interest [2]–[4] in applying a probabilistic approach to merge semantic information for mapping. Bowman *et al.* [2] associate metric and semantic class probability to formulate an optimization problem for SLAM. Doherty *et al.* [3], [4] considers uncertainty when fusing semantic and geometric information. For these approaches, uncertainty is provided with information from different sources. In addition, they are tested with large-scale and long-trajectory datasets (*e.g.,* KITTI [31]), which have relatively similar camera view angles across frames.

In our approach, we address how to evaluate the uncertainty of predictions with a metric GP model. Additionally, we merge information from multiple sources using the uncertainty and present combined information with its uncertainty. Furthermore, we focus on mapping environments for robotic arms. Such environments are usually small-scale and may have a limited number of views due to physical constraints (*e.g.,* collision, space, etc.).

III. Methodology

Our proposed pipeline consists of four main components: Semantic Map module, Shape Completion module, Metric Embedding neural network module, and Gaussian Process (GP) Inference module. The pipeline takes RGBD images as inputs (Fig. 2) and makes predictions for semantic labels and shape completion from each view. The confidence scores for semantic labels are given by the Mask R-CNN, and the ones for shape completion are computed by the metric GP model. The outputs from the pipeline have predictions for unseen parts of the objects, their confidence scores, and semantic labels. The final confidence scores are used to filter

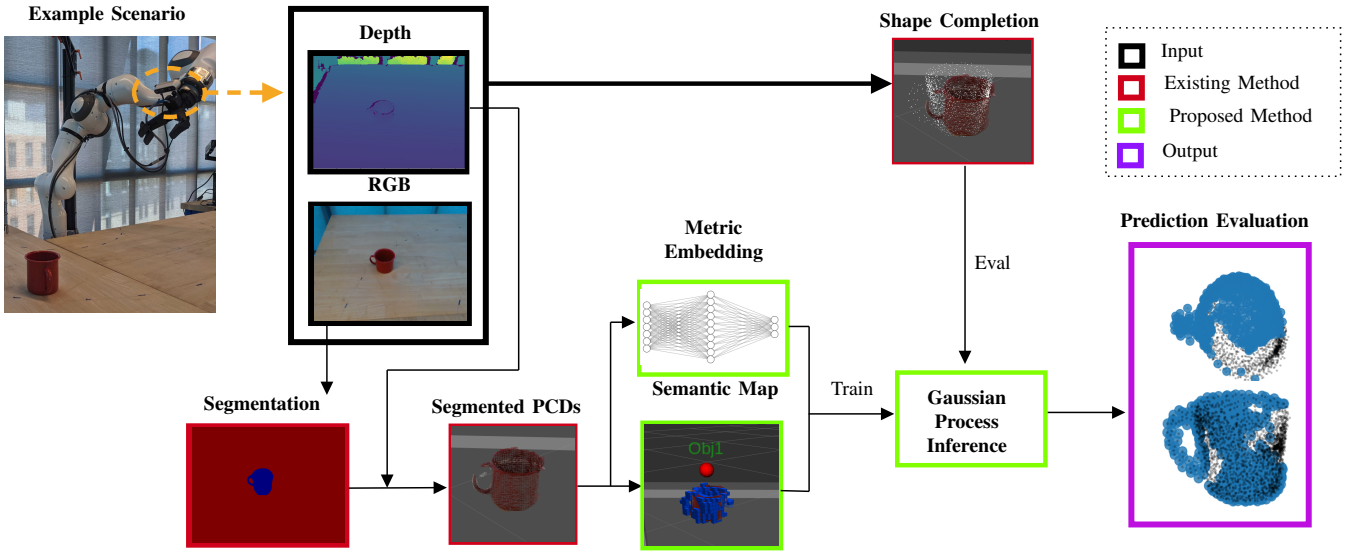


Fig. 2: Proposed Pipeline with an example scenario: our pipeline takes RGBD images and predicts objects' complete shapes with relevant confidence scores. Each component of the pipeline (segmentation, metric embedding, semantic map, GP model, shape completion, prediction evaluation) is presented with example results. The predicted mug pointclouds are colored in black. The predicted points with lower confidence scores than the desired confidence score are marked in blue.

TABLE I: Attributes stored in each voxel of a semantic map

No.	Attribute
1	x, y, z
2	R, G, B
3	Semantic Class (Mode of the histogram)
4	Confidence Score
5	Object ID

predictions, and the filtered point can be used to improve the map. In each section below, we discuss the details of each component.

A. Semantic Map

We design a semantic map module to extract geometric information from the inputs (RGBD images) and maintain relevant semantic information such as *object category, instance ID,* and *a confidence score* (Table I). We select Octomap [7] as our starting point to build the module since it is efficient and flexible due to Octree data structure. From the inputs, RGB images are used by Segmentation Network (MaskRCNN [32]) to predict instance segmentation for each pixel of the RGB images and obtain confidence scores of the prediction. By combining the segmentation outputs and depth images, we obtain segmented pointclouds. The pointclouds are then fed to: 1) Semantic mapping module, and 2) Metric embedding network (refer to Section III-C).

While our main focus is on the uncertainty evaluation of predictions, we develop a basic tracker for a semantic map to aid semantic information retrieval. With the tracker, the semantic map can maintain consistent object IDs over observed objects unless they are heavily cluttered. This is useful functionality to recover the objects' semantic information if a robot's motion causes drastic camera view changes (*e.g.,* no overlapping observations between two sequential views). Object tracking is a well-studied problem. Existing image-based object trackers [33] generally rely on overlapping information between different views, and they lose tracking information if the views do not share any visual cue. We propose a 3D geometry-based tracker to handle this issue. When we observe an object from each view, we calculate a centroid of the object's pointclouds. We then use the Euclidean distance between the new centroid and existing centroids to update the set of centroids (*i.e.,* delete, append, and modify centroids). We then assign an object ID to each centroid in the set and store this information in our semantic map along with the semantic information in each voxel. In addition, we maintain a histogram for each semantic class in the voxel map to address temporary false prediction caused by a misclassification or noise from a camera.

The functionalities introduced above enable us to query the information at the object level from a semantic map (Fig. 3). The information includes *semantic label, confidence score*, and *object ID* for every voxel belonging to any observed object. Although a semantic map contains information about the observed pointclouds, it cannot reason any unseen parts of the objects. To address such limits, we use a shape completion module and predict unseen parts of the objects.

B. Shape Completion

Shape completion is a task to predict the shape of a given object when only partial observations are available for the object. It has gained increasing attention in recent years [34]. However, most of the previous studies focus on predicting an object's shape in an object frame. This object frame-based shape completion requires an additional pose estimation of the object for robot manipulation. This is an

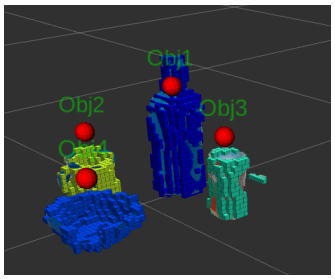


Fig. 3: A semantic map of the example scene introduced in Fig. 1a. Each object's semantic labels are represented with different colors, and object trackers (red spheres) are visualized with their IDs.

issue when an object's pose is not available. To avoid this issue, we choose CenterSnap [35] to complete an object's shape in a camera frame, which can then be transformed to the world frame with a known robot pose. The outputs of this network are the 3D shape reconstruction for all the objects in the current scene and their categorical 6 degree-of-freedom poses and size estimates. The network is trained on the NOCS dataset [36] that contains six object classes. The input to the network is an RGB and depth image. The output is shape completed pointclouds and bounding boxes in the camera frame. Using the camera extrinsics previously calculated via the robot, we transform these into the world frame, which is defined at the base of the robot arm. While this module can predict unseen parts of objects, a way to evaluate each predicted point does not yet exist. To fill such gap, we introduce metric embedding for a GP classification model.

C. Metric Embedding

Embeddings refer to intermediate results that we can obtain from neural networks before they make predictions at the last layer. The intermediate results are often called *embeddings* since they *embed* some information; *semantics* and *geometric* information in our case. To obtain such embeddings for the segmented pointclouds ((x, y, z) coordinates), we use two neural networks sequentially (Fig. 4): 1) DGCNN [37], and 2) Multilayer Perceptron (MLP) network. First, we feed segmented pointclouds to DGCNN and take the $7D$ embeddings from the last dense layer of DGCNN. Next, we pass the embeddings to the MLP network that we designed to enforce embeddings to be in a unit-sphere metric space. The MLP consists of four dense layers $((7 \times 250), (250 \times 100), (100 \times 50), (50 \times 3))$, and is trained with A-Softmax loss [38]. This loss forces the MLP to generate embeddings on a unit-sphere surface upon training the network. As a result, we yield $3D$ metric embeddings for segmented pointclouds from this step. Free space does not have any geometry, so its embedding cannot be learned. Therefore, we assign a zero embedding to free space explicitly. To make objects' embeddings equidistant from the free space embedding, we constrain them to lie on a unit-sphere.

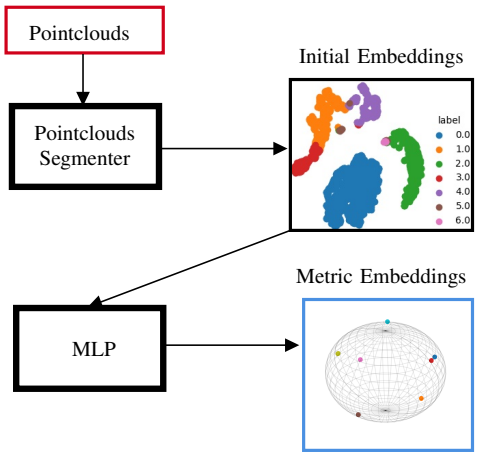


Fig. 4: Our metric learning method: It takes pointclouds from partial views as inputs (red) and outputs learned metric embeddings (blue).

TABLE II: Two types of inputs for the GP model

Type	Train X	Train Y	Test X
Classic GP	x, y, z	Semantic Label	x, y, z
Metric GP	Metric Embedding	Semantic Label	x, y, z

D. Gaussian Process

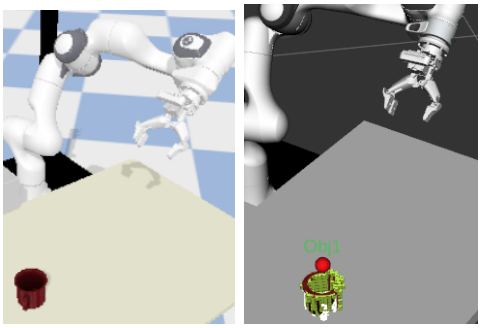
With the metric embeddings of the segmented pointclouds, we select the GP-based model for the prediction since GP [25] is well-known for its ability to capture the underlying structure from training data points and use the structure to make predictions for test data points. Particularly, we choose the Dirichlet GP classification model [39] to make predictions from our embeddings. This model solves a classification problem by transforming class labels with Dirichlet distribution. With the metric embedding as training points, we call this GP model *metric GP*. However, its computational cost often becomes a bottleneck as the size of the data increases. A few libraries [40]–[42] have been introduced to alleviate this problem by optimizing the framework. Among them, we use GPytorch [40], an open-source library that allows us to run fast GP inference on GPUs using Pytorch [43] and CUDA. In addition, it provides recent advanced GP models.

With the selected GP classification model, we present two approaches with two types of inputs (Table II): 1) Classic GP, and 2) Metric GP. For the classic GP, we use pointclouds (x, y, z coordinates of each point) to construct the covariance matrices with a kernel. This approach relies on the geometric distances between training and test points to make inferences, which may not be able to exploit underlying semantic information from the training data.

The GP model can reason underlying data features better if the training points embeds semantic information. This motivated us to construct the GP model using training points obtained in a learning metric embedding space, which is a

TABLE III: The results of baseline experiments for mug and bowl in Simulation

		TP	TN	FP	FN	Acc.
Mug	Classic GP with SC	0.066	0.793	0.125	0.016	0.345
	Metric GP with SC (Ours)	0.161	0.711	0.083	0.045	**0.659**
	Classic GP without SC	0.356	0.139	0.414	0.091	0.462
	Metric GP without SC	0.404	0.161	0.345	0.090	**0.539**
		TP	TN	FP	FN	Acc.
Bowl	Classic GP with SC	0.265	0.024	0.598	0.113	0.307
	Metric GP with SC (Ours)	0.406	0.012	0.502	0.080	**0.447**
	Classic GP without SC	0.246	0.206	0.477	0.071	0.340
	Metric GP without SC	0.249	0.244	0.445	0.062	**0.358**



(a) Simulation (b) Visualization

Fig. 5: An example scenario of our simulation experiments: (a) A mug on the table rendered with Pybullet, (b) Visualization results in Rviz: It shows the mug pointclouds and semantic map with its object tracker.

unit-sphere metric space in our case.

Once we obtain the metric embeddings for the training points, we append embeddings to relevant points. As a result, the training data becomes 6-dimensional (x, y, z, e_1, e_2, e_3). Eq. 1 shows the covariance matrix K for the GP model.

$$K = \begin{pmatrix} K(X,X) & K(X,X_*)^T_* \\ K(X,X_*) & K(X_*,X_*) \end{pmatrix} \tag{1}$$

where X is the training points, and X_* is the test points.

For our GP model training, we construct the $K(X,X)$ matrix using the metric embeddings (e_1, e_2, e_3) of the training points. For the inference, we calculate $K(X,X_*)$ with $3D$ coordinates of both training and test points. Due to the fact we do not have any prior information (*i.e.,* colors, cluster shapes) about the test points, which are unknown except for their coordinates, we use the 3D coordinates of the training points to find the correlation between the training and test points via $K(X,X_*)$. Similarly, we compute $K(X_*,X_*)$ with $3D$ coordinates of the test points.

When the GP model makes inferences, it provides variance (σ) along with predictions. We use the variance to output the raw confidence score (s in Eq. 2) of each prediction by measuring the variance reduction. Then we normalize s to obtain the score $\tilde{s}$ ranging between 0 and 1.

$$s = \frac{1-\sigma}{1-\sigma_{min}} \tag{2}$$

$$\tilde{s} = \frac{s - s_{min}}{s_{max} - s_{min}} \tag{3}$$

To compute our final prediction confidence scores for shape completion parts and their semantics, we multiply the normalized confidence score (Eq. 3) by the confidence score given by the segmenter.

IV. Experiments and Results

A. System Overview

We set up our experiments in both simulation and real-world settings to evaluate our approach both quantitatively and qualitatively. Our simulator is built using Pybullet [44]. For real experiments, we evaluate our approach with a Panda Arm robot, which has 7 degrees-of-freedom. We mount a Realsense D-435 (RGBD) camera on the wrist of the robot. For both the simulation and real setup, we use Rviz to visualize the outputs from our algorithms. We use open source libraries in ROS2 Foxy with Ubuntu 20.04 OS. The underlying communication layer is ROS2, allows for seamless communication between the robot (both in simulation and real), our algorithm, and visualization modules.

B. Evaluation Metrics

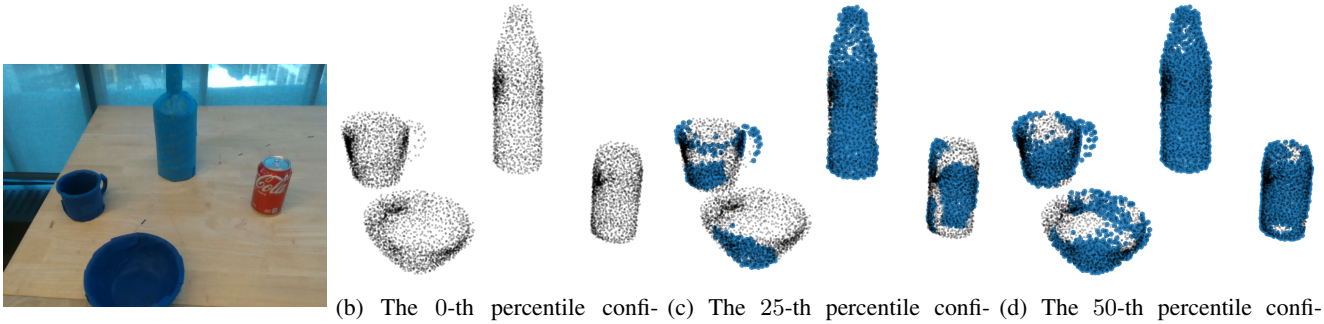
To evaluate our approach, we build a confusion matrix to evaluate the prediction accuracy. The matrix has four elements, and they are defined as follows for our experiments:

- True Positive (TP): Points predicted by GP as part of the object that lie on the ground truth object mesh.
- True Negative (TN): Points predicted by GP as free space that doe not lie on the ground truth object mesh.
- False Positive (FP): Points predicted by GP as part of the object that do not lie on the ground truth object mesh.
- False Negative (FN): Points predicted by GP as free space but lie on the ground truth object mesh.

C. Baselines

Due to sensor noise and lighting conditions, it is generally infeasible to obtain ground truth pointclouds for objects in a real-world experiment setup. We ran our baseline experiments in simulation to evaluate our method with ground truth pointclouds sampled from known object meshes as shown in Fig. 5. For the simulation experiments, we use two types of objects (*e.g.,* bowl and mug) while for the real robot evaluation we added multiple objects in the scene including a bottle and a can.

We designed our experiments to compare the performance of our proposed pipeline where we apply the metric GP



(a) An RGB Image (b) The 0-th percentile confidence score (c) The 25-th percentile confidence score (d) The 50-th percentile confidence score

Fig. 6: An example of real-world evaluation on the robotic platform. The original predicted shape completion points are colored in black. The predicted points with lower confidence scores than the desired confidence score are marked in blue. As the desired confidence score increases from (b) to (d), the number of uncertain points are increasing visually (*e.g.* the size of the blue region grows).

model and the classic GP model. The metric GP model uses metric embeddings of the observed pointclouds to train the model, and the classic GP takes the pointclouds without any further processing. Additionally, the pipeline is tested with and without the shape completion (SC) module for both the GP models to see if the performance of each GP model relies on the shape completion module's presence. For the case without the SC module, we sample test points for inference randomly in the object bounding box. For the other case, we use points from the SC module as test points. In total, we ran four runs for a mug and bowl, respectively.

D. Results

For our experiments, we trained the Mask R-CNN with the NOCS real and camera datasets with 10 epochs on NVIDIA 1080 TI GPU. Additionally, we trained our metric embedding model with $30,000$ scenes where each scene has $2,048$ data points on NVIDIA 3090 GPU. Pretrained weights are used for the shape completion network(CenterSnap). Lastly, we built both GP models with $3,600$ training points and queried $1,000$ test points from them using Matern Kernel.

The results of the baseline are shown in Table III. The confusion matrix results for each case are normalized as all evaluations are done on 1000 points. The results demonstrate that our method outperforms all the other baseline experiments for both mug and bowl classes. We noticed that the shape completion (SC) module improved the performance of our method regardless of the choice of the GP model. Additionally, the mug class shows higher accuracy in general than the bowl class. The mug model in our simulation has a more similar shape to the mug model in the NOCS dataset compared to the bowl case.

We also qualitatively evaluate our proposed approach in a real-world environment. Fig. 6 shows the results, where the blue-colored points represent the points with lower confidence scores than the threshold confidence score. In other words, any point that has a lower confidence score than the desired confidence score is colored blue. From the observed scene, our approach predicted shape completion and semantic information. Based on the observed prior (Fig. 6a), each point in the prediction has different confidence scores. The complete shape points are then filtered with the $n-$th percentile confidence scores (*i.e.,* threshold confidence score) in Fig. 6b-6d. As the threshold confidence becomes higher, we observed that the number of uncertain predicted points is increasing. Observed points from the RGB image tend to be more certain points. Also, the bottle class has more uncertain points compared to other classes since its predicted complete shape is quite different from the shape of the bottle we used for the experiment. This shows the difference between the object model predicted from the SC module and the observed model can degrade the performance of our approach. As our qualitative result shows, the capability to filter predictions with confidence scores will give a robot to adjust the threshold confidence score based on its task for mapping its environment. flexibility to a robot to decide how much predicted information it will use for a mapping. In other words, the robot can use higher confidence scores to filter the prediction with higher-risk tasks while lower confidence scores can be used for lower-risk tasks.

V. Conclusions

This paper presents our novel mapping method that unifies semantic information and shape completion predicted from partial-view RGBD images and calculates confidence scores for its predictions. Our metric Gaussian Process (GP) classification model merges confidence scores (if available) for the given information. A novel aspect of our method is to transfer sensor measurements to a learned metric space for training the GP model. From this metric GP model, we can measure the uncertainty of the unified information and make predictions more accurately than a classic GP model. The results demonstrate that our approach successfully merges the given information (*e.g.,* semantic information and shape completion) from partial views and computes their confidence scores. Future work will include building a grasping algorithm that relies on our modules and demonstrate its robustness to occlusions in the scene. In the long term, we plan to focus on generalizing our approach so that it can incorporate a larger number of sensor inputs.

REFERENCES

[1] S. Garg, N. Sünderhauf, F. Dayoub, D. Morrison, A. Cosgun, G. Carneiro, Q. Wu, T.-J. Chin, I. Reid, S. Gould, *et al.*, "Semantics for robotic mapping, perception and interaction: A survey," *Foundations and Trends® in Robotics*, vol. 8, no. 1–2, pp. 1–224, 2020.

[2] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 1722–1729.

[3] K. Doherty, D. Fourie, and J. Leonard, "Multimodal semantic slam with probabilistic data association," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 2419–2425.

[4] K. J. Doherty, D. P. Baxter, E. Schneeweiss, and J. J. Leonard, "Probabilistic data association via mixture models for robust semantic slam," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1098–1104.

[5] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," in *Sensor devices and systems for robotics*. Springer, 1989, pp. 253–276.

[6] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 249–265, 1987.

[7] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, no. 3, pp. 189–206, 2013.

[8] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[9] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *2011 10th IEEE international symposium on mixed and augmented reality*. Ieee, 2011, pp. 127–136.

[10] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments," *The international journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, 2012.

[11] H. Pfister, M. Zwicker, J. Van Baar, and M. Gross, "Surfels: Surface elements as rendering primitives," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 335–342.

[12] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J.-A. Fernandez-Madrigal, and J. González, "Multi-hierarchical semantic maps for mobile robotics," in *2005 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2005, pp. 2278–2283.

[13] H. Zender, O. M. Mozos, P. Jensfelt, G.-J. Kruijff, and W. Burgard, "Conceptual spatial representations for indoor mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 6, pp. 493–502, 2008.

[14] Z. Wei, W. Chen, and J. Wang, "3d semantic map-based shared control for smart wheelchair," in *International Conference on Intelligent Robotics and Applications*. Springer, 2012, pp. 41–51.

[15] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "Semanticfusion: Dense 3d semantic mapping with convolutional neural networks," in *2017 IEEE International Conference on Robotics and automation (ICRA)*. IEEE, 2017, pp. 4628–4635.

[16] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid, "Meaningful maps with object-oriented semantic mapping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 5079–5085.

[17] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. M. M. Montiel, "Towards semantic slam using a monocular camera," in *2011 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2011, pp. 1277–1284.

[18] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.

[19] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1689–1696.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[21] F. Lateef and Y. Ruichek, "Survey on semantic segmentation using deep learning techniques," *Neurocomputing*, vol. 338, pp. 321–348, 2019.

[22] S. O'Callaghan, F. T. Ramos, and H. Durrant-Whyte, "Contextual occupancy maps using gaussian processes," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 1054–1060.

[23] S. T. O'Callaghan and F. T. Ramos, "Gaussian process occupancy maps," *The International Journal of Robotics Research*, vol. 31, no. 1, pp. 42–62, 2012.

[24] J. Wang and B. Englot, "Fast, accurate gaussian process occupancy maps via test-data octrees and nested bayesian fusion," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1003–1010.

[25] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[26] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[27] C. Villacampa-Calvo, B. Zaldivar, E. C. Garrido-Merchán, and D. Hernández-Lobato, "Multi-class gaussian process classification with noisy inputs," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 1696–1747, 2021.

[28] M. G. Jadidi, L. Gan, S. A. Parkison, J. Li, and R. M. Eustice, "Gaussian processes semantic map representation," *ArXiv*, vol. abs/1707.01532, 2017.

[29] E. Zobeidi, A. Koppel, and N. Atanasov, "Dense incremental metric-semantic mapping via sparse gaussian process regression," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6180–6187.

[30] E. Guerrero-Font, F. Bonin-Font, M. Martin-Abadal, Y. Gonzalez-Cid, and G. Oliver-Codina, "Sparse gaussian process for online seagrass semantic mapping," *Expert Systems with Applications*, vol. 170, p. 114478, 2021.

[31] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[32] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017. [Online]. Available: http://arxiv.org/abs/1703.06870

[33] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, "Multiple object tracking: A literature review," *Artificial Intelligence*, vol. 293, p. 103448, 2021.

[34] X.-F. Han, H. Laga, and M. Bennamoun, "Image-based 3d object reconstruction: State-of-the-art and trends in the deep learning era," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 5, pp. 1578–1604, 2019.

[35] M. Z. Irshad, T. Kollar, M. Laskey, K. Stone, and Z. Kira, "Centersnap: Single-shot multi-object 3d shape reconstruction and categorical 6d pose and size estimation," 2022. [Online]. Available: https://arxiv.org/abs/2203.01929

[36] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[37] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, 2019.

[38] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 212–220.

[39] D. Milios, R. Camoriano, P. Michiardi, L. Rosasco, and M. Filippone, "Dirichlet-based gaussian processes for large-scale calibrated classification," *CoRR*, vol. abs/1805.10915, 2018. [Online]. Available: http://arxiv.org/abs/1805.10915

[40] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "Gpytorch: Blackbox matrix-matrix gaussian process inference with GPU acceleration," *CoRR*, vol. abs/1809.11165, 2018. [Online]. Available: http://arxiv.org/abs/1809.11165

[41] A. G. d. G. Matthews, M. Van Der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman, "Gpflow: A gaussian process library using tensorflow." *J. Mach. Learn. Res.*, vol. 18, no. 40, pp. 1–6, 2017.

[42] GPy, "GPy: A gaussian process framework in python," http://github.com/SheffieldML/GPy, since 2012.

[43] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[44] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.