Neural Implicit Vision-Language Feature Fields

Kenneth Blomqvist[1], Francesco Milano[1], Jen Jen Chung[2], Lionel Ott[1] and Roland Siegwart[1]

***Abstract*— Recently, groundbreaking results have been presented on open-vocabulary semantic image segmentation. Such methods segment each pixel in an image into arbitrary categories provided at run-time in the form of text prompts, as opposed to a fixed set of classes defined at training time. In this work, we present a zero-shot *volumetric* open-vocabulary semantic scene segmentation method. Our method builds on the insight that we can fuse image features from a vision-language model into a neural implicit representation. We show that the resulting feature field can be segmented into different classes by assigning points to natural language text prompts. The implicit volumetric representation enables us to segment the scene both in 3D and 2D by rendering feature maps from any given viewpoint of the scene. We show that our method works on noisy real-world data and can run in real-time on live sensor data dynamically adjusting to text prompts. We also present quantitative comparisons on the ScanNet dataset.**

I. INTRODUCTION

A key component of building intelligent robots capable of operating in unstructured and cluttered human environments is the representation used to model the robot's surroundings. Often times representations have to trade-off properties which depend on the usage scenario. These properties include the quality of the reconstruction, the ability to integrate sensor data continuously, and the computational complexity to query the representation. The importance of these aspects differs based on what components of a robotic system needs to use the representation, dictating the requirements for available capabilities, sensor data throughput, or query latency. For instance, an obstacle avoidance system needs to query for occupancy at high frequency, while a high-level planning system needs access to semantic knowledge, and finally a grasp planning system requires fine-grained segmentation information.

While in the past occupancy was the main information of interest, robotics has moved towards richer representations using semantics in recent years. A challenge is that most semantic approaches use a fixed, closed set, of pre-determined semantic labels. However, real environments contain more than a few dozen classes, and thus methods capable of handling arbitrary semantic classes, i.e. open set, are desirable. Additionally, objects in an environment do not necessarily belong to distinct, mutually exclusive classes. Certain objects might belong to several classes. A bookshelf is also a piece of furniture, for example. For high-level planning purposes, being able to reason about relations between their semantics might also be useful.

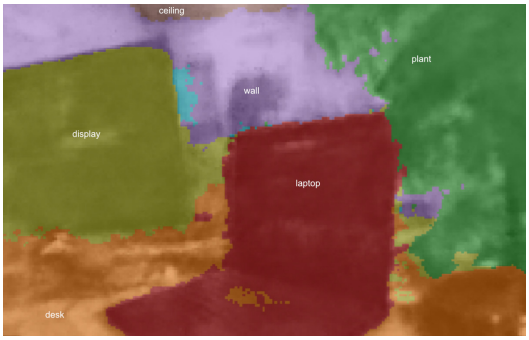

Fig. 1. Our method enables real-time segmentation of scenes into arbitrary text classes provided at run-time.

An environment representation that has wide applicability has several desirable properties, including: (1) can be built incrementally as the robot explores the environment, (2) enables real-time integration of new measurements, (3) has a compact memory footprint, (4) represents geometry at a high-level of detail, (5) is differentiable, (6) supports open set semantic queries, and (7) allows fast querying by downstream modules. Previously introduced 3D semantic scene representations are either built from global scene information [1], use closed set semantics [2]–[5], operate on a fixed level of detail [2], [3], or are not differentiable [2], [3]. In this paper, we take a step towards a representation which has the above-mentioned properties.

Vision-language models (VLM) have shown remarkable performance on open vocabulary object detection [6], [7]. Recently, these results have been extended to dense semantic segmentation [8]–[11]. Some of these methods [8], [9] associate each pixel with a semantically meaningful vector, which is embedded in the same high-dimensional vector space as natural language prompts through a text encoder. This allows direct computation of the similarity between text prompts and image features at run-time.

As vision-language models can be trained on massive web-scale datasets that can be collected automatically without human supervision, they often show better generalization capabilities than models trained on smaller closed-set manually curated datasets. Additionally, VLMs can capture the long tail of scenarios and classes that are so rare that they are unlikely to be included in curated datasets. These properties offer great promise for applications in robotics, where we might want our robots to be able to perform new tasks in never-before-seen environments.

In this paper, we present a method for grounding dense

[1]Autonomous Systems Lab, Swiss Federal Institute of Technology in Zürich, Switzerland. kblomqvist@mavt.ethz.ch

[2]School of ITEE, The University of Queensland, Australia.

This project has received funding from EU Horizon 2020 program, project PILOTING H2020-ICT-2019-2 871542.

vision-language features into a 3D implicit neural representation that can be built up incrementally, in real-time, as new observations come in. We jointly model radiance, vision-language model features, and density in the scene using an implicit neural representation. Our representation can be incrementally built up given posed images of the scene and a pre-trained language model. We can directly compute the similarity between natural language text prompts and either 3D points or 2D image coordinates for any given viewpoint of the scene through volumetric rendering. This enables semantically segmenting a scene zero-shot into text categories provided at run-time, without having to fine-tune the system on any domain specific semantics.

In experiments, we showcase results in real-world experiments where we build up our scene representation in real-time on a real system, and demonstrate the ability to segment the scene into different classes provided as natural language prompts at run-time. We additionally present quantitative segmentation results on the large and diverse ScanNet dataset. To the best of our knowledge, our method is the first real-time capable 3D vision-language neural implicit representation. Our implementation will be made available through the Autolabel project [1].

II. RELATED WORK

Open Vocabulary Semantic Segmentation and Vision-Language Models

CLIP [12] introduced a visual-language model capable of mapping images into the same vector space as natural language queries by correlating images to their text descriptions mined from the open web. Open vocabulary segmentation methods typically learn dense features which are compared to text queries given at run-time [8], [9]. Others take a multi-task learning approach, fusing a task prompt with the architecture [10], [11]. Other methods such as Clippy [13] explored learning pixel-aligned visual-language models from large scale web datasets without requiring segmentation labels, potentially enabling large-scale open set training, if the results can be extended to full semantic segmentation.

Language Models in Robotics

Large language models have been explored as an approach to high-level planning [14]–[18] and scene understanding [19], [20]. Vision-language models embedding image features into the same space as text have been applied to open vocabulary object detection [16], [17], natural language maps [15], [17], [21]–[23], and for language-informed navigation [24]–[26].

Recent methods have explored fusing global CLIP features [22], image caption embeddings [27], or dense pixel-aligned [1] visual-language model features into a point cloud representation for scene understanding. Concurrent work ConceptFusion [28] explores building multi-modal semantic maps by fusing features from vision-language models as well as audio into a reconstructed 3D point cloud. Similar to these, we also fuse VLM features into a 3D representation. Unlike [1], [22], [28], we use a continuous neural representation of geometry and semantics which we learn jointly through volumetric rendering. [1], [27] fuse image features from a pre-built point cloud using a multi-view fusion method and learn a 3D convolutional network to map scene points to dense features. Our representation can be built incrementally as measurements are collected and does not require global scene geometry upfront.

Semantic Scene Representations

Voxel-based map representations have been proposed to store semantic information about a scene [2], [3], [29]–[31]. These methods assign a semantic class to each individual voxel in the scene. Voxel-based dense semantic representations typically operate on static scenes, but some have explored modeling dynamic objects [32], [33].

Scene graphs [34]–[36] have also been proposed as a candidate for a semantic scene representation that can be built-up online. Such methods decompose the scene into a graph where edges model relations between parts of the scene. The geometry of the parts are typically represented as a signed distance functions stored in a voxel grid [15].

Neural implicit representations infer scene semantics [4], [5], [37]–[42] jointly with geometry using a multi-layer perceptron or similar parametric model. These have been extended to dynamic scenes [43]. Neural feature fields [5], [38], [44], [45] are neural implicit representations which map continuous 3D coordinates to vector-valued features. Such representations have shown remarkable ability at scene segmentation and editing. [44] also presented some initial results on combining feature fields with vision-language features, motivating their use for language driven semantic segmentation and scene composition.

III. METHOD

Our method consists of two components: i) a NeRF-like feature field mapping points in a volume to color, density, and feature vector and ii) a vision-language model which both extracts features from image frames and can embed text prompts into the same vector space.

A. Volumetric Scene Representation

We want to associate 3D points in the volume of our scene to density, color, and a feature vector. From this, we can render corresponding maps of color, depth, and feature vectors through a NeRF-like [46] volumetric rendering function, visualized in Figure 2. We model these maps using a positional encoding function and three multilayer perceptrons (MLP). The first MLP, indicated as (1), outputs density and a geometric code. The second MLP, labeled (2), outputs color from the geometric code and an encoded viewing direction. The third MLP, denoted by (3), takes the geometric code and outputs the feature vector.

To encode the x, y, and z position in the volume, we use the hybrid positional encoding introduced by [38]. We concatenate the vector valued hashgrid encoding introduced

[1] https://github.com/ethz-asl/autolabel

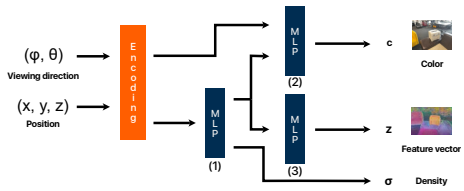

Fig. 2. A diagram of the model used for our feature field.

in [47] with the low-frequency values of traditional NeRF [46] frequency encoding with $L = 2$. The low-frequency components allows us to model the coarse spatial location in the scene, whereas the parameters in the hashgrid grid allow us to quickly learn high-frequency details.

The resulting encoding is fed into an MLP, (1) in Figure 2, which outputs a 15-dimensional geometric code vector and scalar density σ. The geometric code is fed into two different MLPs. The first one outputs a feature vector $\boldsymbol{f}$. The other one takes as additional input the encoded viewing direction and outputs a color vector $\boldsymbol{c}$. To encode the viewing direction, we use the same spherical harmonic encoding as [46].

We use these outputs to volumetrically render color images and feature outputs using the rendering function:

$$R(\mathbf{r}, h) = \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i)) h(\mathbf{x}_i), \tag{1}$$

$$T_i = \exp\left(- \sum_{j=1}^{i-1} \sigma_j \delta_j \right), \tag{2}$$

where h is a function outputting a vector or scalar quantity for points $\mathbf{x}_i$ within the volume, T_i is the transmittance function, δ_j is the distance between samples and σ_i is the predicted density for encoded point samples $\mathbf{x}_i$ along a ray $\mathbf{r}$. We use R to produce rendered quantities:

$$\begin{aligned} \hat{\mathbf{c}}(\mathbf{r}) &= R(\mathbf{r}, \mathbf{c}), \\ \hat{d}(\mathbf{r}) &= R(\mathbf{r}, z), \\ \hat{\mathbf{f}}(\mathbf{r}) &= R(\mathbf{r}, \mathbf{f}), \end{aligned} \tag{3}$$

using z for the depth component of samples, $\mathbf{c}$ for the color MLP output and $\mathbf{f}$ for the feature vector output of our MLP.

These quantities are learned by optimizing photometric, depth, and feature rendering error terms:

$$\mathcal{L}_{rgb}(\mathbf{r}) = \left\| \hat{\mathbf{c}}(\mathbf{r}) - \bar{\mathbf{c}}(\mathbf{r}) \right\|_2^2, \tag{4}$$

$$\mathcal{L}_{d}(\mathbf{r}) = \begin{cases} \left\| \hat{d}(\mathbf{r}) - \bar{d}(\mathbf{r}) \right\|_1, & \text{if } \bar{d} \text{ is defined for } \mathbf{r} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$\mathcal{L}_{f}(\mathbf{r}) = \left\| \hat{\mathbf{f}}(\mathbf{r}) - \bar{\mathbf{f}}(\mathbf{r}) \right\|_2^2 / D \tag{6}$$

where $\bar{\mathbf{c}}(\mathbf{r})$ is the ground truth and $\hat{\mathbf{c}}(\mathbf{r})$ the predicted color for ray $\mathbf{r}$, $\bar{d}(\mathbf{r})$ is the ground truth depth (if available), $\hat{d}(\mathbf{r})$ the predicted depth predictions along ray $\mathbf{r}$, $\hat{\mathbf{f}}$ rendered feature outputs, $\bar{\mathbf{f}}$ extracted image features for ray $\mathbf{r}$, and D the dimensionality of the image features.

The parameters in the hashgrid encoding volume and in the MLPs are jointly learned by optimizing the objective:

$$\mathcal{L}(\mathbf{r}) = \mathcal{L}_{rgb}(\mathbf{r}) + \lambda_d \mathcal{L}_d(\mathbf{r}) + \lambda_f \mathcal{L}_f(\mathbf{r}) \tag{7}$$

using stochastic gradient descent on a set of rays sampled uniformly from input images $\mathbf{I}$ along with corresponding feature vectors $\bar{\mathbf{f}}$. The parameters λ_d and λ_f are weighting parameters to weight the different components of the loss function. To learn the representation online, while our robot is exploring the environment, keyframes with image features can be added to the image set as they are captured.

B. Vision-language Features and Zero-shot Segmentation

Our framework presented above is capable of making use of arbitrary feature maps. Thus, we can use features from any feature extractor that produces dense pixel-aligned feature maps from images. To enable open set semantic queries in both 2D and 3D at run-time, we choose to use learned features for which the similarity with text prompts can be computed through a simple dot product. LSeg [9] and OpenSeg [8] are both suitable candidates for this purpose. In our experiments, we use LSeg features, as pretrained models are readily available. The model comes both with an image feature extractor $\bar{\mathbf{F}}$ and text encoder $\mathbf{E}$.

Given a pose in the world frame of the volume, we can render color, depth, and feature maps using volumetric rendering, using equations 1 and 3. We compute the semantic class by assigning the feature $\hat{\mathbf{f}}$ to the most similar class given a set of user defined natural language class descriptions $t_i \in \mathcal{T}$ into which we want to segment our scene:

$$\hat{s}(\mathbf{r}) = \text{argmax}_i \mathbf{E}(t_i) \cdot \hat{\mathbf{f}}(\mathbf{r}). \tag{8}$$

For 3D queries at point $\mathbf{x}$, we can simply evaluate the feature MLP at $\mathbf{x}$, i.e.:

$$s(\mathbf{x}) = \text{argmax}_i \mathbf{E}(t_i) \cdot \mathbf{f}(\mathbf{x}). \tag{9}$$

IV. EXPERIMENTAL RESULTS

In the following experiments we provide quantitative results on the ScanNet dataset. We compare our method to the OpenScene [1] work in terms of mean intersection over union (mIoU) and mean accuracy (mAcc). Then, to highlight the utility of our approach in robotics application we integrate our approach with a SLAM framework. Finally, we report run-time information to demonstrate the feasibility of running our algorithm on a real robotic system.

In all our experiments, we use LSeg features [9] trained on the ADE20k dataset [48]. For the loss function, we use $\lambda_d = 0.1$ and $\lambda_f = 0.5$ throughout all experiments. Having tried a range of different values, we found that they perform similarly and settled on these values in the middle of the range. In case less noisy and more accurate depth measurements are available, a higher λ_d value might yield better results.

A. ScanNet

On the ScanNet dataset we perform evaluation both in 3D, by segmenting the provided ground truth point cloud, as well as in 2D by comparing our rendered segmentation maps to the ones provided in the dataset. We use the 20 classes from the ScanNet benchmark. Points or pixels that do not belong to these classes are ignored.

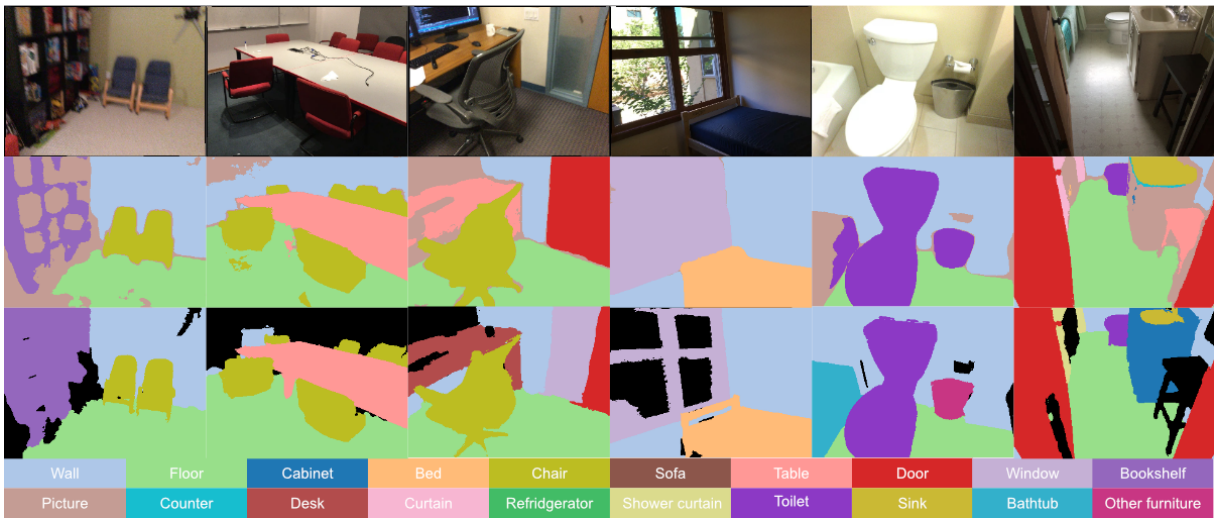

Fig. 3. Randomly sampled 2D segmentation examples from the ScanNet validation set. Top row shows the original RGB images, second row shows our segmentation and the bottom row shows the ground truth segmentation from the ScanNet dataset. Black pixels in the ground truth segmentation correspond to classes not included in the 20 ScanNet evaluation classes.

We first fit our representation using the given RGB, depth frames and camera poses using 20 000 optimization iterations. For 3D point cloud segmentation, we look up the feature vector for each point in the point cloud and assign it to the nearest text class using the ScanNet class label names as the text prompts. For 2D segmentation, we segment feature maps from each viewpoint in each scan and compare against the reference segmentation map.

	ScanNet mIoU	ScanNet mAcc
OpenScene - LSeg (3D)	54.2	66.6
OpenScene - OpenSeg (3D)	47.5	70.7
Ours - LSeg (3D)	47.4	55.8
Ours - LSeg (2D)	62.5	80.2

TABLE I

MEAN INTERSECTION-OVER-UNION AGREEMENT WITH THE SCANNET VALIDATION SET.

Table I shows mean intersection-over-union (mIoU) results on the ScanNet validation set, averaging over scenes and classes. LSeg [9]/OpenSeg [8] denotes the 2D image features used. 3D denotes segmentation agreement on the given ground truth point cloud whereas 2D shows agreement against the semantic segmentation maps.

OpenScene [1] performs better overall, but it should be noted that it makes use of the ground truth scene point cloud, whereas we only use the color and depth frames and implicitly reconstruct the geometry. We only use the scene point cloud for evaluation. We additionally show 2D segmentation results compared with the ground truth segmentation frames in the dataset. As OpenScene only segments the point cloud, only 3D segmentation accuracy is shown.

Figure 3 shows qualitative 2D segmentation masks. Our method mostly performs well, but often struggles to distinguish between semantically similar classes such as "desk" and "table" or "curtain" and "shower curtain" in the ScanNet evaluation, as we do not make use of any tuning to align the semantics of the dataset with the semantics of the vision-language vector space. The ScanNet label quality is also not perfect and our method often gets details correct which are missed by the ScanNet ground-truth labels, such as legs of tables and chairs or other thin structures.

B. Real-time SLAM Experiment

To test our scene representation in a real-world robotics scenario, we integrate our system with a SLAM pipeline [2] using a Luxonis OAK-D Pro stereo camera. While the system is running, we integrate color, depth, and features extracted using LSeg from keyframes at 5 Hz with poses obtained from the SLAM system. In experiments, we use either the left (grayscale) camera image or RGB camera. Depth is computed using stereo matching and aligned to the keyframe camera's frame.

To test our system we give it classes in the form of text prompts while it is running and inspect the quality of the segmentation. Using the odometry poses provided by the SLAM system, we render color, depth maps and segmentation maps from the current camera viewpoint in real-time, segmenting the camera image into the given classes.

Figure 5 shows snapshots of a real-time experiment performed with a handheld camera in a regular office environment. The prompts used to produce the segmentation map are shown, but note that these can be changed at run-time to re-segment the scene. Figure 4 shows how quickly our representation is able to fit to a new scene when learned from scratch and integrating frames in real-time. After a dozen seconds, our method is able to produce good segmentation maps and scene reconstructions.

[2]Specifically the SpectacularAI SDK available here: https://github.com/SpectacularAI/sdk

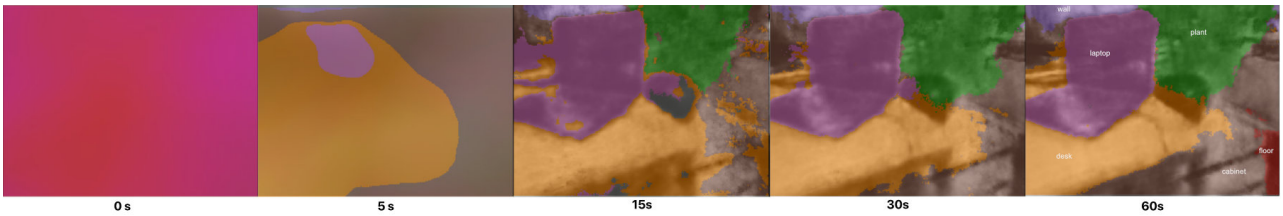

Fig. 4. Snapshots from real-time zero shot volumetric segmentations from a fixed viewpoint at given intervals. Our representation is able to learn in real-time and is already useful after a dozen seconds. Each image shows RGB rendering output for the viewpoint, overlayed with the semantic segmentation given the 6 class prompts shown.

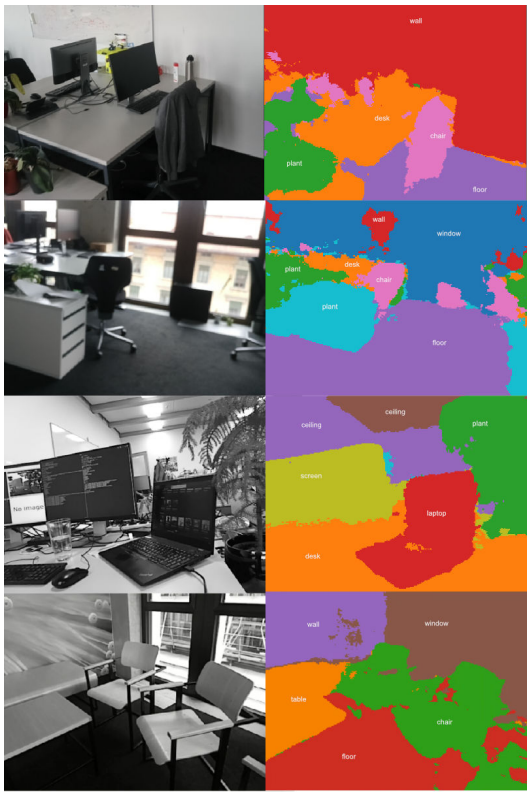

Fig. 5. RGB renderings and semantic segmentation maps of our representation from our real-time experiment in an office environment given the prompts shown below the images.

C. Query Performance

We time the latency and throughput of queries performed with our implementation on an Nvidia RTX 3070 GPU. 3D semantic and density point queries can be performed at over 7 million lookups per second with a latency of less than 10 milliseconds. 2D ray queries can be rendered and segmented at roughly 30 000 pixels per second using 256 samples per ray, but this can be adjusted to to suit the desired fidelity.

V. Discussion and Conclusions

While the results obtained are an encouraging first step towards open-set 3D semantic segmentation there are still many open questions to improve such approaches, some of which we discuss in the following.

Currently, the largest factor limiting segmentation performance is the quality of the vision-language features. While LSeg uses natural language features from CLIP trained on a very large dataset, the visual encoder is trained on the small closed-set ADE20K dataset. If we were able to compute dense pixel-aligned visual-language features from open-set web scraped data without requiring any human annotations, we believe that results could eventually surpass supervised learning methods. [13] presented some promising initial results on learning pixel aligned features without using segmentation masks or other expert annotations.

In real-time experiments, our system relied on poses coming from a SLAM system. If many bad poses are computed by the SLAM system, the 3D representation could become corrupted by bad updates. Possible solutions include treating the sparse SLAM poses as initial guesses and optimizing the poses jointly with scene geometry, as in [49], [50], or bad poses could be filtered out by analyzing the photometric or geometric error across frames.

In robotics, downstream modules, such as motion planners and high-level planning systems, might benefit from a more explicit and principled representation of geometry than what we presented in this paper. For example, signed distance function based approaches [51] might provide better surface and occupancy reconstruction and have other favorable properties, such as the ability to compute the normal of a surface by differentiating through the distance function. For the time being, our method is limited to static scenes. Dealing with moving objects within scenes remains an open problem, but promising recent research [43] suggests that extending neural implicit representations to dynamic scenes might be feasible.

To conclude, we proposed a volumetric neural representation which is able to jointly learn geometry, radiance, and semantic feature information of a scene. We have shown that by using dense pixel-aligned vision-language features, our resulting learned representation can be used to volumetrically segment scenes into, at run-time, user defined categories. We have also shown how the representation can be used to produce dense 2D segmentation maps for queried viewpoints. Experiments on the ScanNet dataset showed competitive performance and our real-world experiments demonstrate that the method could be run onboard a robotic system.

References

[1] S. Peng, K. Genova, C. Jiang, A. Tagliasacchi, M. Pollefeys, T. Funkhouser, *et al.*, "OpenScene: 3D Scene Understanding with Open Vocabularies," *arXiv preprint arXiv:2211.15654*, 2022.

[2] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, "Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery," *IEEE Robotics and Automation Letters*, 2019.

[3] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping," in *IEEE ICRA*, 2020.

[4] S. Zhi, E. Sucar, A. Mouton, I. Haughton, T. Laidlow, and A. J. Davison, "iLabel: Revealing Objects in Neural Fields," *IEEE Robotics and Automation Letters*, 2022.

[5] K. Mazur, E. Sucar, and A. J. Davison, "Feature-Realistic Neural Fusion for Real-Time, Open Set Scene Understanding," *IEEE ICRA*, 2023.

[6] A. Zareian, K. D. Rosa, D. H. Hu, and S.-F. Chang, "Open-Vocabulary Object Detection Using Captions," in *IEEE/CVF CVPR*, 2021.

[7] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui, "Open-vocabulary Object Detection via Vision and Language Knowledge Distillation," *arXiv preprint arXiv:2104.13921*, 2021.

[8] G. Ghiasi, X. Gu, Y. Cui, and T.-Y. Lin, "Scaling Open-Vocabulary Image Segmentation with Image-Level Labels," in *ECCV*, 2022.

[9] B. Li, K. Q. Weinberger, S. Belongie, V. Koltun, and R. Ranftl, "Language-driven Semantic Segmentation," *arXiv preprint arXiv:2201.03546*, 2022.

[10] H. Zhang, P. Zhang, X. Hu, Y. Chen, L. Li, X. Dai, L. Wang, L. Yuan, J. Hwang, and J. Gao, "GLIPv2: Unifying Localization and Vision-Language Understanding," in *NeurIPS*, 2022.

[11] X. Zou, Z.-Y. Dou, J. Yang, Z. Gan, L. Li, C. Li, X. Dai, H. Behl, J. Wang, L. Yuan, *et al.*, "Generalized Decoding for Pixel, Image, and Language," *arXiv preprint arXiv:2212.11270*, 2022.

[12] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning Transferable Visual Models From Natural Language Supervision," in *International Conference on Machine Learning*, 2021.

[13] K. Ranasinghe, B. McKinzie, S. Ravi, Y. Yang, A. Toshev, and J. Shlens, "Perceptual Grouping in Vision-Language Models," *arXiv preprint arXiv:2210.09996*, 2022.

[14] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, *et al.*, "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[15] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual Language Maps for Robot Navigation," *arXiv preprint arXiv:2210.05714*, 2022.

[16] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su, "LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models," *arXiv preprint arXiv:2212.04088*, 2022.

[17] B. Chen, F. Xia, B. Ichter, K. Rao, K. Gopalakrishnan, M. Ryoo, A. Stone, and D. Kappler, "Open-vocabulary Queryable Scene Representations for Real World Planning," *arxiv preprint arXiv:2209.09874*, 2022.

[18] S. S. Raman, V. Cohen, E. Rosen, I. Idrees, D. Paulius, and S. Tellex, "Planning with Large Language Models via Corrective Re-prompting," *arXiv preprint arXiv:2211.09935*, 2022.

[19] W. Chen, S. Hu, R. Talak, and L. Carlone, "Leveraging Large Language Models for Robot 3D Scene Understanding," *arXiv preprint arXiv:2209.05629*, 2022.

[20] H. Ha and S. Song, "Semantic Abstraction: Open-World 3D Scene Understanding from 2D Vision-Language Models," in *Conference on Robot Learning*, 2022.

[21] V. Blukis, R. Knepper, and Y. Artzi, "Few-shot Object Grounding and Mapping for Natural Language Robot Instruction Following," in *Conference on Robot Learning*, 2021.

[22] N. M. M. Shafiullah, C. Paxton, L. Pinto, S. Chintala, and A. Szlam, "CLIP-Fields: Weakly Supervised Semantic Fields for Robotic Memory," *arXiv preprint arXiv:2210.05663*, 2022.

[23] S. Tan, M. Ge, D. Guo, H. Liu, and F. Sun, "Self-supervised 3D Semantic Representation Learning for Vision-and-Language Navigation," *arXiv preprint arXiv:2201.10788*, 2022.

[24] D. Shah, B. Osinski, B. Ichter, and S. Levine, "LM-Nav: Robotic Navigation with Large Pre-Trained Models of Language, Vision, and Action," *arXiv preprint arXiv:2207.04429*, 2022.

[25] Z. Wang, M. Li, M. Wu, M.-F. Moens, and T. Tuytelaars, "Find a Way Forward: a Language-Guided Semantic Map Navigator," *arXiv preprint arXiv:2203.03183*, 2022.

[26] A. Majumdar, G. Aggarwal, B. Devnani, J. Hoffman, and D. Batra, "ZSON: Zero-Shot Object-Goal Navigation using Multimodal Goal Embeddings," *arXiv preprint arXiv:2206.12403*, 2022.

[27] R. Ding, J. Yang, C. Xue, W. Zhang, S. Bai, and X. Qi, "Language-driven Open-Vocabulary 3D Scene Understanding," *arXiv preprint arXiv:2211.16312*, 2022.

[28] K. M. Jatavallabhula, A. Kuwajerwala, Q. Gu, M. Omama, T. Chen, S. Li, G. Iyer, S. Saryazdi, N. Keetha, A. Tewari, *et al.*, "ConceptFusion: Open-set Multimodal 3D Mapping," *arXiv preprint arXiv:2302.07241*, 2023.

[29] M. Strecke and J. Stuckler, "EM-Fusion: Dynamic Object-Level SLAM With Probabilistic Data Association," in *IEEE/CVF ICCV*, 2019.

[30] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji, "PanopticFusion: Online Volumetric Semantic Mapping at the Level of Stuff and Things," in *IROS*, 2019.

[31] L. Schmid, J. Delmerico, J. L. Schönberger, J. Nieto, M. Pollefeys, R. Siegwart, and C. Cadena, "Panoptic Multi-TSDFs: a Flexible Representation for Online Multi-resolution Volumetric Mapping and Long-term Dynamic Scene Consistency," in *IEEE ICRA*, 2022.

[32] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger, "MID-Fusion: Octree-based Object-Level Multi-Instance Dynamic SLAM," in *IEEE ICRA*, 2019.

[33] M. Grinvald, F. Tombari, R. Siegwart, and J. Nieto, "TSDF++: A Multi-Object Formulation for Dynamic Object Tracking and Reconstruction," in *IEEE ICRA*, 2021.

[34] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese, "3D Scene Graph: A Structure for Unified Semantics, 3D Space, and Camera," in *IEEE/CVF ICCV*, 2019.

[35] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization," in *Robotics: Science and Systems*, 2022.

[36] S.-C. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari, "SceneGraphFusion: Incremental 3D Scene Graph Prediction from RGB-D Sequences," in *IEEE/CVF CVPR*, 2021.

[37] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison, "In-Place Scene Labelling and Understanding with Implicit Scene Representation," in *IEEE/CVF International Conference on Computer Vision*, 2021.

[38] K. Blomqvist, L. Ott, J. J. Chung, and R. Siegwart, "Baking in the Feature: Accelerating Volumetric Segmentation by Rendering Feature Maps," *arXiv preprint arXiv:2209.12744*, 2022.

[39] X. Fu, S. Zhang, T. Chen, Y. Lu, L. Zhu, X. Zhou, A. Geiger, and Y. Liao, "Panoptic NeRF: 3D-to-2D label transfer for panoptic urban scene segmentation," *arXiv preprint arXiv:2203.15224*, 2022.

[40] Y. Siddiqui, L. Porzi, S. R. Buló, N. Müller, M. Nießner, A. Dai, and P. Kontschieder, "Panoptic Lifting for 3D Scene Understanding with Neural Fields," *arXiv preprint arXiv:2212.09802*, 2022.

[41] Z. Liu, F. Milano, J. Frey, M. Hutter, R. Siegwart, H. Blum, and C. Cadena, "Unsupervised Continual Semantic Adaptation through Neural Rendering," *arXiv preprint arXiv:2211.13969*, 2022.

[42] A. Kundu, K. Genova, X. Yin, A. Fathi, C. Pantofaru, L. J. Guibas, A. Tagliasacchi, F. Dellaert, and T. Funkhouser, "Panoptic Neural Fields: A Semantic Object-Aware Neural Scene Representation," in *IEEE/CVF CVPR*, 2022.

[43] X. Kong, S. Liu, M. Taher, and A. Davison, "vMAP: Vectorised Object Mapping for Neural Field SLAM," *arxiv preprint arXiv:2302.01838*, 2023.

[44] S. Kobayashi, E. Matsumoto, and V. Sitzmann, "Decomposing NeRF for Editing via Feature Field Distillation," in *NeurIPS*, 2022.

[45] V. Tschernezki, I. L. D. Larlus, and A. Vedaldi, "Neural Feature Fusion Fields: 3D Distillation of Self-Supervised 2D Image Representations," in *Conference on 3D Vision*, 2022.

[46] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," in *ECCV*, 2020.

[47] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding," *arXiv preprint arXiv:2201.05989*, 2022.

[48] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene Parsing through ADE20K Dataset," in *IEEE CVPR*, 2017.

[49] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "iMAP: Implicit Mapping and Positioning in Real-Time," in *IEEE/CVF ICCV*, 2021.

[50] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "NICE-SLAM: Neural Implicit Scalable Encoding for SLAM," in *IEEE/CVF CVPR*, 2022.

[51] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction," *arXiv preprint arXiv:2106.10689*, 2021.