Beyond Geometry: Leveraging Pose and Visual Scene Understanding for Indoor Room Segmentation in Mobile Robots

Guilherme G. Zanetti*, Elio D. T. Rodriguez*, Thiago M. Paixão†, Filipe W. Mutz*, Claudine S. Badue*, Alberto F. De Souza*, Anselmo Frizera*, Thiago Oliveira-Santos*
* Universidade Federal do Espírito Santo (UFES); † Instituto Federal do Espírito Santo (IFES)

***Abstract*—Accurate residential indoor room segmentation is crucial for mobile robots, but traditional geometric methods often fail in cluttered environments, while vision-based approaches are still dependent on geometric analysis, and therefore limited to simple environments with clear geometric boundaries. We propose a framework that can handle complex room boundaries by using a robot's visual input to simultaneously segment and classify rooms. This is achieved by projecting image-based room predictions onto a 2D occupancy map using spatial aggregation techniques. By leveraging the vision cone of the robot and powerful image classification models such as CNNs, ViTs and VLMs, the map is segmented into rooms with a higher Intersection over Union (IoU) than traditional methods while handling environments with complex geometry. The results demonstrate that our method using VLMs and vision cone aggregation outperforms other approaches, achieving up to 76.22% classification accuracy on ArchitectThor room segmentation without any task-specific training, offering superior accuracy and flexibility for robotic semantic mapping, while overcoming the geometry-dependency of traditional methods. We also provide the code and improved dataset.**

I. Introduction

Mobile robots are increasingly common in domestic environments, particularly cleaning robots, vacuum cleaners, and robotic lawn mowers. These robots need to navigate and understand their surroundings to operate effectively in complex indoor environments [1]. Central to this spatial awareness is the transition from raw geometric data to semantic knowledge, a process known as Room Segmentation or Room-Type Detection [2], [3]. This task automates the partitioning of occupancy grid maps or floor plans into discrete functional zones (rooms), extracting the boundaries and semantic labels of each room, enabling high-level task planning.

Traditional room segmentation methods partition 2D occupancy grid maps by relying on their geometric and topological features. The most common techniques include Voronoi-based approaches [4], [5] that identify narrow passages like doors and structural walls. Similarly, the ROSE² algorithm [6] employs a frequency-based method to filter out non-structural clutter from the map before segmentation begins. A primary drawback of these geometric methods is their unreliability in cluttered environments and in complex room geometries, leading to inaccurate room divisions [6].

To address the shortcomings of purely geometric techniques, recent work has focused on integrating semantic information into the segmentation process. The SeLRoS framework [7], for instance, refines an initial classic geometric segmentation by using a Large Language Model (LLM) to analyze objects detected within each partitioned space, allowing it to merge erroneously divided rooms and assign functional labels like "Kitchen". However, this method depends on an initial geometric segmentation, and is limited to analyzing objects within each pre-segmented room, missing room divisions which are not obvious from the geometric structure.

To address these limitations, we propose a method that uses image classification models, such as Convolutional Neural Networks (CNNs), Vision Transformers (ViTs), and Visual Language Models (VLMs), to classify viewpoint images captured by a mobile robot inside a residential indoor environment. Our method generates a 2D segmented map with room labels by combining predicted room labels from the viewpoint images with the robot's position. Procedurally and human-generated simulation environments are used for experimental evaluation. We provide the code and improved dataset. [1]

The main contributions of this paper are:

- A method that is agnostic to camera type and SLAM method that performs room segmentation and classification simultaneously using visual information;
- This work provides the complete codebase and manually annotated ground truth room segmentations for the ArchitectThor dataset;
- An empirical analysis of Visual Language Model generalization capabilities for room classification in mobile robotics.

II. Image-Based Room Segmentation

The proposed method for room segmentation leverages visual information captured by mobile robots with any monocular RGB camera to perform simultaneous room segmentation and classification. The approach (overview in Figure 1) classifies a sparse set of viewpoint images which serve as seeds for a K-Nearest Neighbors (KNN) based algorithm to generate segmented maps with room labels. The complete pipeline consists of four main stages: **(i) data collection**: the robot captures viewpoint images with respective position and orientation; **(ii) viewpoint classification**: the sparse set of

[1] https://github.com/GuilhermeGZanetti/beyond-geometry-code

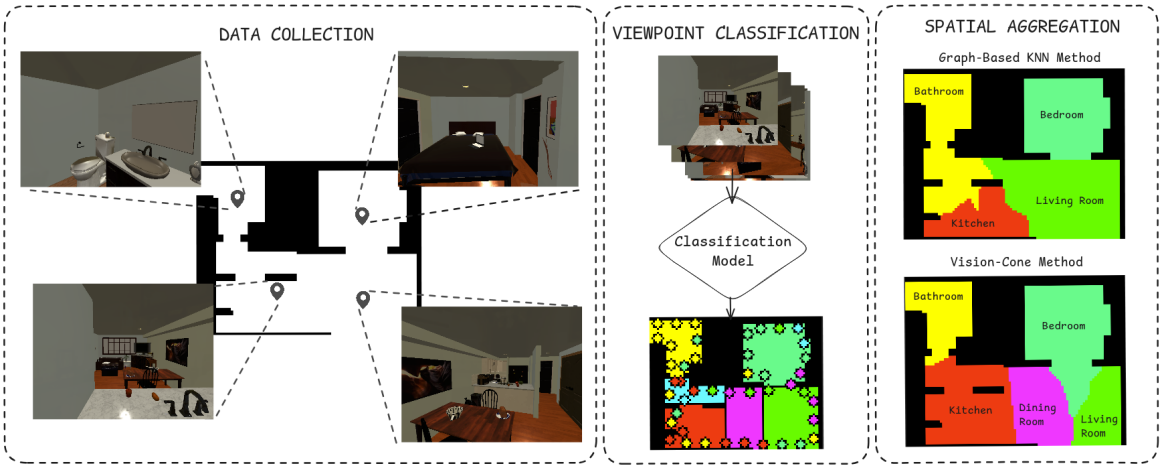

Fig. 1. Overview of our approach. First, the robot captures images along its path. Then, the images are classified into room types. The classification results are then used to segment the occupancy map into rooms with two different methods: (i) Graph-Based K-Nearest Neighbors and (ii) Vision Cone Segmentation. The examples in the figure were generated using the VLM classifier.

viewpoint images are classified into room types; **(iii) spatial aggregation**: the room type is projected onto the occupancy map via two different spatial aggregation methods; **(iv) segmentation post-processing**: the final map is post-processed to remove noise and regularize the region shapes.

A. Data Collection

The proposed method is robot-agnostic, requiring three inputs: (1) a **2D Occupancy Grid Map** defining navigable regions; (2) **Viewpoint Images** captured by the robot throughout the environment; and (3) **Pose Information** (coordinates and orientation) for each image to enable projection onto map locations.

To determine the robot trajectory for capturing the viewpoints, an automatic wall-following autotrace viewpoint generation algorithm is proposed. It follows the walls of the occupancy map to create a path that covers all navigable areas. The algorithm extracts the navigable-area contours, creates an inward-offset path away from walls, and uniformly samples viewpoints along this path with multiple orientations to ensure comprehensive coverage. An example of the resulting paths can be seen in Figure 2. An additional manual teleoperation strategy is also used in the experiments as a baseline. Pseudocode for the method is available at the github repository.

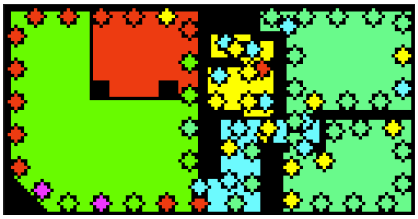

Fig. 2. Example of wall-following autotrace viewpoint generation. Each sampled point is marked in the image with a colored circle, with the color correspoinding to the most common predicted classification label on the point.

B. Viewpoint Classification

The image classification step aims to classify individual images captured by the robot as specific room types or locations within the indoor environment. Three classification approaches are explored:

*a) **CNNs:*** A ResNet-50 [8] architecture is finetuned from pre-trained *IMAGENET1K_V2* weights[2] on seven room classes. Training uses data augmentation (rotation, perspective distortion, cropping, flipping) and ImageNet normalization. At inference, images are resized to 224×224 and normalized.

*b) **ViT (DINOv2):*** DINOv2 [9] extracts 1024-dimensional feature embeddings from 224×224 normalized images. A three-layer MLP classifier is trained on these frozen features for the seven room classes.

*c) **Visual Language Models (VLMs):*** A 4-bit quantized Qwen2.5-VL-7B-Instruct model [10] performs zero-shot classification using textual prompts specifying the seven room categories. The classification prompt is available in the code repository.

Both CNN and ViT (DINOv2) are trained on seven classes (described in Section III-A) from the Places-2 dataset [11], while the VLM uses its pre-trained weights.

C. Spatial Aggregation

Image labels from classified viewpoints are aggregated with their map positions to segment the occupancy map into rooms. Each walkable pixel is assigned a room label based on nearby classified viewpoints. Two variants are used to propagate labels from viewpoints to map pixels: **(i) Graph-Based KNN with Path Distance** and **(ii) Vision Cone Segmentation**.

1) Graph-Based KNN with Path Distance: A simple Euclidean-distance KNN can cross walls and obstacles, yielding invalid assignments. To address this, a graph-based variant models the occupancy map as a grid graph over walkable cells

[2]https://docs.pytorch.org/vision/main/models/generated/torchvision.models.resnext50_32x4d.html#torchvision.models.ResNeXt50_32X4D_Weights

and uses shortest-path (geodesic) distances that respect occupancy constraints. The distances are calculated from each grid cell to every cell containing a labeled viewpoint using Breadth-first Search (BFS). For each pixel, class scores are computed from the K nearest viewpoints, and the most common class label is assigned.

The computational cost is dominated by BFS runs from each labeled grid cell, which scales with the number of labeled viewpoints and the walkable area.

2) Vision Cone Segmentation: This proposed variant uses camera pose and field of view to constrain label propagation to observable regions and aggregates per-pixel votes across viewpoints. Each viewpoint first undergoes a forward wall-distance check: if a short ray intersects an obstacle, the viewpoint is discarded. For each valid viewpoint, candidate pixels are selected from a cone of 90° originating from the position and direction of the camera, extending to the maximum range of 500 pixels. For each candidate, line-of-sight is verified by ray-tracing; only non-occluded pixels are retained. Each retained pixel receives a vote for the viewpoint's label (the result of the image classification model for that viewpoint). After all viewpoints are processed, the final label at each pixel is obtained by the most frequent vote in a 5 by 5 grid around it.

For a more formal specification, pseudocode for the graph-based KNN and the vision-cone aggregation are available in the project repository referenced in the Introduction.

D. Segmentation Post-processing

The described spatial aggregation methods often produce noisy segmentation maps with small label islands, holes, and jagged borders due to viewpoint sparsity and occasional misclassifications. To improve region consistency while strictly respecting navigability constraints, we apply an image-based post-processing pipeline to all predicted segmentation maps. The pipeline takes as input the binary occupancy map (walkable vs. non-walkable) and the predicted segmentation (color-coded labels), and outputs a cleaned segmentation in the same format.

The proposed post-processing pipeline is as follows: (i) normalize map resolution to 0.1 m/px using nearest-neighbor rescaling; (ii) identify the label set and infer a background label from pixels outside the walkable area; (iii) apply an iterative sliding-window mode filter to locally smooth label noise (excluding background by default); (iv) remove small connected components per label and mark removed pixels as unlabeled; (v) iteratively fill unlabeled interior pixels by 3×3 neighbor-majority voting; and (vi) regularize borders by approximating label masks with polygons using Douglas–Peucker algorithm [12]. We evaluate the impact of this post-processing in Section IV-C.

III. Evaluation Setup

All experiments were conducted on a Linux server equipped with a NVIDIA RTX 4090 GPU. This setup was used because of the VLM-based classification, which requires substantial computational resources.

A. Datasets

The evaluation is conducted using three distinct datasets. The first (Places-2) is used to train and evaluate the image classification models. The other two datasets (ProcThor-10k and ArchitectThor) are used in the final application, i.e. to measure and evaluate the quality of segmentation and room classification on residential homes and apartments.

a) ***Places-2*** *[11]:* This dataset contains 1,000 images for each of 365 scene categories. From them, 7 categories are selected based on their relevance to residential indoor environments: balcony, bathroom, bedroom, corridor, dining_room, kitchen, and living_room. Categories such as army_base, jail_cell, and park are excluded as they are not applicable to the target domain.

b) ***ProcThor-10k*** *[13]:* A large-scale synthetic dataset containing ten thousand procedurally generated houses with ground truth room segmentations. The procedurally generated houses may contain unnatural house and room layouts. The evaluation was limited to the first 100 houses from the train dataset due to computational restrictions.

c) ***ArchitectThor*** *[13]:* A smaller but more realistic dataset comprising 10 human-generated houses. One of our contributions is the manual annotations of the room segmentations, segementing the rooms of each house into the seven possible classes of the Places-2 dataset, which can be found in the code repository. The available houses have more natural and detailed ambients when compared to ProcThor.

B. Evaluation Metrics

The performance of the room segmentation methods is assessed using standard computer vision and robotics metrics:

Intersection over Union (IoU): Measures the overlap between predicted and ground truth regions. For each predicted room segment A and its best matching ground truth segment B, IoU is calculated as $\frac{|A \cap B|}{|A \cup B|}$. This was calculated for each predicted room segment, and then averaged over all segments.

Precision and Recall: For the same predicted and ground truth segments, precision is computed as $\frac{|A \cap B|}{|A|}$ and recall as $\frac{|A \cap B|}{|B|}$, and averaged over all segments.

Classification Accuracy: Measures the percentage of pixels matching the ground truth labels (pixel accuracy).

C. Simulation Environment

The viewpoints are generated in ArchitectThor by using the Habitat-Sim simulator [14]–[16] and in ProcThor by using the AI2-THOR simulator [17]. In the following experiments, we evaluate two different viewpoint generation strategies in the ArchitectThor dataset: (i) manual teleoperation with visual feedback and (ii) the automatic wall-following autotrace described in Section II-A.

When generating the viewpoints through manual teleoperation, a member of our research team operates the robot throughout each simulated house to observe all rooms, ensuring comprehensive coverage of the environment. This manual procedure is used only as a baseline in the viewpoint-generation comparison experiment.

IV. Experimental Results and Discussion

A. Image Classification Performance

The first experiment evaluates the capabilities of the image classification models (CNN, DINOv2, and VLM) to classify the 7 room types from the Places-2 dataset correctly. The Places-2 dataset is split into training (28,000 samples), validation (3,500 samples) and test (3,500 samples) sets. The validation set was used to calibrate training hyperparameters and chose the best checkpoint and the test set was used to evaluate the classification performance of the models after training and the zero-shot capabilities of the VLM.

Table I shows the results of the classification models on the Places-2 test split. It can be seen that DINOv2 with a lightweight MLP outperformed the fine-tuned CNN, while the zero-shot VLM matched DINOv2 despite no task-specific training. This demonstrates that VLMs carry substantial prior knowledge of indoor semantics, making them a strong default when no training budget or in-domain labels are available.

TABLE I
Classification metrics on the Places-2 test set (macro-averaged in %).

Model	Precision	Recall	Accuracy	F1
CNN	87.17	87.08	87.03	87.06
DINOv2	92.37	92.42	92.40	92.38
VLM	92.32	92.38	92.37	92.32

B. Effect of Viewpoint Generation

To measure the sensitivity of the pipeline to the robot exploration strategy, it was conducted an additional experiment on ArchitectThor comparing the two viewpoint generation strategies: (i) manual teleoperation with visual feedback and (ii) automatic wall-following autotrace. Post-processing is not performed in this experiment.

Table II compares ArchitectThor results with the two viewpoint generation strategies. Overall, the automatic strategy is competitive with manual collection and, for the best-performing configuration (VLM + vision-cone voting), slightly improves IoU and accuracy. Since autotrace achieves the best results while eliminating human teleoperation cost, we adopt wall-following autotrace for viewpoint generation in all remaining experiments for both datasets.

TABLE II
ArchitectThor results with two viewpoint generation strategies: manual and wall-following autotrace.

	Manual capture		**Wall-following autotrace**	
Method	IoU (%)	Acc. (%)	IoU (%)	Acc. (%)
VLM KNN	48.80	62.82	47.77	62.33
VLM Cone	**54.04**	**74.30**	**55.38**	**74.84**
CNN KNN	41.22	33.95	42.55	33.34
CNN Cone	46.04	42.71	41.39	40.87
DINOv2 KNN	48.53	58.57	51.29	58.86
DINOv2 Cone	53.97	72.15	53.46	73.46

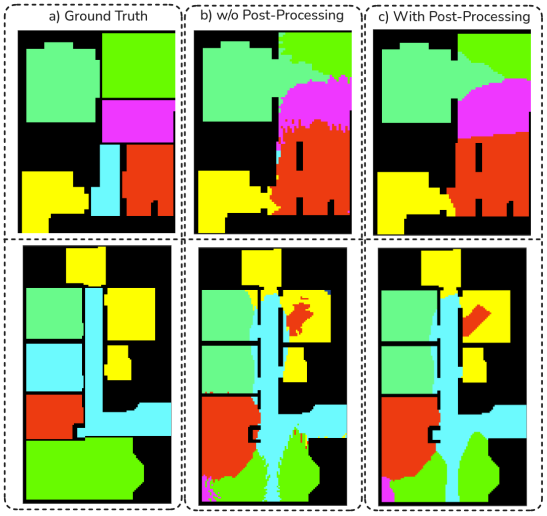

Fig. 3. Qualitative effect of segmentation post-processing. Columns show (a) ground truth, (b) prediction without post-processing, and (c) prediction with post-processing. Post-processing reduces isolated label noise and produces more consistent, regular room regions while preserving non-walkable constraints.

C. Effect of Segmentation Post-processing

This experiment evaluates the post-processing pipeline described in Section II-D to reduce label noise and regularize region shapes. To quantify its effect independently of the classifier or aggregation variant, the predicted maps are compared *with* and *without* post-processing, reporting mean IoU and pixel classification accuracy averaged across all model+aggregation combinations for each dataset. This experiment uses the wall-following autotrace viewpoint generation strategy.

Table III reports the impact of the proposed post-processing pipeline. Post-processing consistently improves IoU on both ProcThor and ArchitectThor, indicating cleaner, more spatially consistent regions. Pixel classification accuracy changes only slightly, but also shows an improvement, suggesting that post-processing primarily reduces boundary noise and small spurious components rather than altering large-scale label assignments. Figure 3 shows qualitative examples of this effect, where small label islands are removed and room borders become more regular. Because of its benefits, the post-processing pipeline is used in the remaining experiments.

TABLE III
Impact of segmentation post-processing (mean).

Dataset	Post-processing	IoU (%)	Acc. (%)
ProcThor	None	49.31	52.99
ProcThor	Mode filter + polygons	**55.12**	**53.85**
ArchitectThor	None	48.64	57.28
ArchitectThor	Mode filter + polygons	**53.82**	**57.82**

D. Room Segmentation Performance

This experiment is conducted to assess the complete pipeline on the two different house datasets (ProcThor and

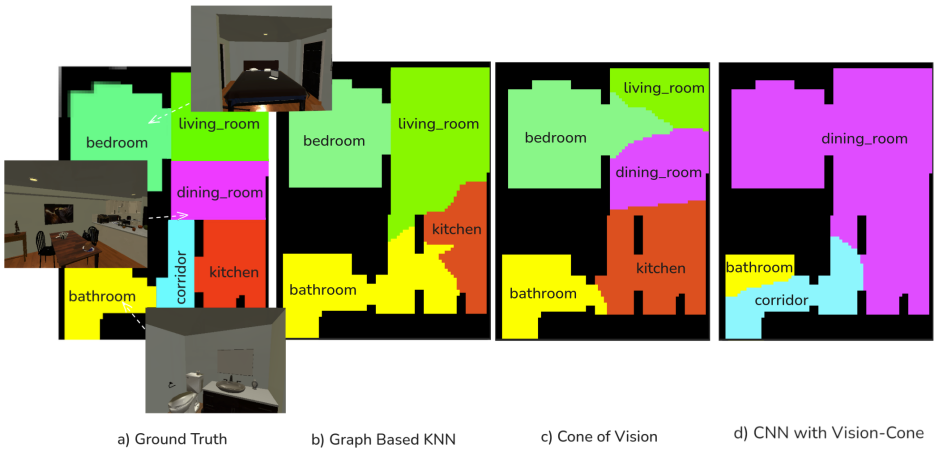

Fig. 4. Room segmentation results on an ArchitectThor environment using the VLM. (a) Ground truth segmentation with sample camera viewpoints showing the actual room appearances. (b) Graph-based KNN aggregation results. (c) Vision-cone voting aggregation results. Both methods successfully identify major room boundaries, with vision-cone voting showing slightly better performance in resolving ambiguous regions between dining room and kitchen areas. (d) Results using the CNN model with Vision-Cone Voting.

ArchitectThor), comparing the two different Spatial Aggregation methods with all three image classification models.

Table IV highlights two consistent trends. First, VLM-based classification yields the best overall segmentation, achieving the top accuracy on both datasets and the best IoU on ArchitectThor, while DINOv2 is a strong second best (especially with vision-cone voting). In contrast, the CNN underperforms across both metrics, suggesting limited transfer to these simulated residential environments. Second, the two aggregation schemes exhibit a clear tradeoff: on ProcThor, graph-based KNN gives higher IoU, while vision-cone voting improves pixel accuracy; on ArchitectThor, vision-cone voting improves both IoU and accuracy, indicating that using camera orientation to restrict label projection better resolves adjacent-room ambiguities.

TABLE IV
ROOM SEGMENTATION RESULTS ON PROCTHOR AND ARCHITECTTHOR.

	ProcThor		**ArchitectThor**	
Model	IoU (%)	Acc. (%)	IoU (%)	Acc. (%)
CNN KNN	52.44	30.92	46.09	34.34
CNN Cone	53.02	29.77	49.09	39.87
DINOv2 KNN	56.49	51.78	51.30	59.60
DINOv2 Cone	47.22	63.32	61.30	73.43
VLM KNN	**61.51**	73.28	50.95	63.46
VLM Cone	60.04	**74.02**	**64.29**	**76.22**

When looking at the example house in Figure 4, it can be seen that the Graph-Based KNN aggregation method confuses the living room and the kitchen, and ommits completely the dining room bettwen them. Meanwhile, the Vision-Cone Voting aggregation method is able to partially segment the dining room and most of the other rooms, although not perfectly matching the ground truth. This example highlights the limitations of the KNN method and traditional geometry-based methods. For example, the robot might be standing on the living room, but looking at the kitchen, and therefore it assings a kitchen label to the living room. The Vision-Cone method diminishes this effect by assigning the label on the pixels in the direction of the robot's gaze, instead of where the robot is standing, therefore being more capable of identifying rooms with ambiguous boundaries.

In Figure 4 (d) the results using the CNN model with Vision-Cone Voting are shown. It can be seen that the CNN model misclassified most of the house as a dining room. This example highlights the limitations of the CNN model, and the influence that the classification model has on the room segmentation.

E. Comparison with Baseline Methods on ProcThor-20

To contextualize our results against established occupancy-map room segmentation methods, this experiment compares the proposed method against Voronoi Random Fields (VRF) [4], SeLRoS [7], and ROSE^2 [6]. Since the SeLRoS repository only provides results for a subset of ProcThor houses and does not enable running the full baseline pipeline across the 100-house evaluation set, this comparison is restricted to the 20 ProcThor houses for which SeLRoS and VRF outputs are available. For fairness, the proposed method and ROSE^2 run on the same 20-house subset.

Table V shows the results of the comparison. Overall, both of our best configurations (using VLMs) outperform both VRF and SeLRoS in IoU while also achieving higher precision and recall. ROSE^2 performs poorly on this subset, suggesting that its structural assumptions and clutter-handling heuristics do not transfer well to ProcThor's house layouts.

TABLE V
COMPARISON WITH PRIOR WORK ON PROCTHOR-20. THE PRESENTED METRICS ARE MEAN IOU, PRECISION (P), AND RECALL (R).

Method	IoU (%)	P (%)	R (%)
SeLRoS [7]	60.15	77.63	74.50
VRF [4]	42.76	77.21	52.82
ROSE^2 [6]	24.42	73.24	27.38
Ours (VLM + post + Graph KNN)	**64.14**	83.61	**76.22**
Ours (VLM + post + Vision cone)	61.64	**83.66**	73.24

It can be seen that both methods based only on geometry (ROSE^2 and VRF) achieved low performance on this experiment, when compared to the two methods that incorporate semantic information (SeLRoS and the proposed method). This confirms the importance and effectiveness of incorporating semantic information during the room segmentation process, specially for complex environments where geometry alone is not enough to accurately segment the rooms.

Beyond the quantitative improvements, the integration of VLMs also enables natural generalization to room types beyond the predefined categories through simple prompt modifications, eliminating the need for retraining when encountering new environments or room classifications. Furthermore, the vision cone aggregation method demonstrates good performance in challenging scenarios, which are hard for geometry based methods. For example, houses having cluttered regions and subtle distinctions between adjacent rooms, such as the observed dining room-kitchen boundary.

F. Reproducibility Considerations

The preparation of this experiment revealed a significant gap in the literature for standardized evaluation. As far as we know, existing works such as SeLRoS do not provide publicly available code, detailed metrics implementations, or complete datasets used for evaluation, making direct comparison with new methods challenging. To address this limitation and promote reproducible research, this work provides the complete codebase and manually annotated ground truth room segmentations for the ArchitectThor dataset. Additionally, the standardized evaluation using the first 100 houses from the ProcThor training set and the provided ArchitectThor ground truths establishes a consistent benchmark that future works can easily adopt for fair comparison with the proposed method.

V. Conclusions

This paper presented a new state-of-the-art approach for semantic room segmentation in indoor environments using VLMs and spatial aggregation techniques. Experimental results demonstrated that a modern VLM can effectively perform room-type classification and segmentation tasks without requiring task-specific training or extensive labeled datasets. They also offered an indication that the proposed method, specially using a VLM with the Vision Cone aggregation, is promising across diverse environments, from synthetic procedural layouts to realistic manually designed spaces, while requiring no domain-specific training.

A few key limitations were identified in the proposed method, including challenges when encountering room types outside of the predefined seven residential labels. The capabilities of the VLM to generalize to new environments can be evaluated in future works. Additionally, the vision-cone method has a high sensitivity to classification errors, which can lead to propagated errors in noisy environments. The usage of a VLM also poses a limitation, as it requires specialized hardware or API costs to run the viewpoints classification.

Several directions can emerge for extending this work. For example, (i) developing methods to dynamically handle room types beyond the current residential classification set and test VLM's generalization capabilities to new environments, (ii) experimenting with different algorithms for automatic exploration methods to capture the viewpoints, and (iii) deploying and validating the approach on real mobile robots in real indoor environments. These extensions would enhance the practical applicability of the proposed room segmentation framework.

Acknowledgments

This study was financed in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES, Brazil) Finance Code 001; Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq); and Fundação de Amparo à Pesquisa e Inovação do Espírito Santo (FAPES, Brazil) - grant 2021-07KJ2 and additional support for event participation (Edital Fapes N° 02/2026).

References

[1] P. Anderson, *et al.*, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3674–3683.

[2] Z. Xu, N. Jha, S. Mehadi, and M. Mandal, "Automatic floor plan analysis: Datasets, methods, and applications (2000–2025)," *Automation in Construction*, vol. 178, p. 106378, 2025.

[3] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hägele, "Room segmentation: Survey, implementation, and analysis," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1019–1026.

[4] S. Friedman, H. Pasula, and D. Fox, "Voronoi random fields: Extracting topological structure of indoor environments via place labeling." 2007, pp. 2109–2114.

[5] A. Diosi, G. Taylor, and L. Kleeman, "Interactive slam using laser and advanced sonar," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*. IEEE, 2005, pp. 1103–1108.

[6] M. Luperto, T. Kucner, A. Tassi, M. Magnusson, and F. Amigoni, "Robust structure identification and room segmentation of cluttered indoor environments from occupancy grid maps," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7974–7981, 2022.

[7] T. Kim and B.-C. Min, "Semantic layering in room segmentation via llms," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 9831–9838.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[9] M. Oquab, *et al.*, "Dinov2: Learning robust visual features without supervision," 2023.

[10] S. Bai *et al.*, "Qwen2.5-vl technical report," arXiv:2502.13923, 2025.

[11] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[12] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *The Canadian Cartographer*, vol. 10, no. 2, pp. 112–122, 1973.

[13] M. Deitke, *et al.*, "ProcTHOR: Large-Scale Embodied AI Using Procedural Generation," in *NeurIPS*, 2022.

[14] M. Savva, *et al.*, "Habitat: A Platform for Embodied AI Research," in *Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV)*, 2019.

[15] A. Szot, *et al.*, "Habitat 2.0: Training home assistants to rearrange their habitat," in *NeurIPS*, 2021.

[16] X. Puig, *et al.*, "Habitat 3.0: A co-habitat for humans, avatars and robots," 2023.

[17] E. Kolve, *et al.*, "AI2-THOR: An Interactive 3D Environment for Visual AI," *arXiv*, 2017.