SSGA: Synthetic Scene Graph Augmentation via Multiple Pipeline Variants

Kenta Tsukahara, Ryogo Yamamoto, Kanji Tanaka, Tomoe Hiroki
University of Fukui, 3-9-1 Bunkyo, Fukui City, Fukui 910-0017, Japan

Keywords: Cross-View Robot Localization, View Synthesis, Scene Graph Classifier.

Abstract: Cross-view image localization, which involves predicting the view of a robot with respect to a single-view landmark image, is important in landmark-sparse and mapless navigation scenarios such as image-goal navigation. Typical scene graph-based methods assume that all objects in a landmark image are visible in the query image and cannot address view inconsistencies between the query and landmark images. We observed that scene graph augmentation (SGA), a technique that has recently emerged to address scene graph-specific data augmentation, is particularly relevant to our problem. However, the existing SGA methods rely on the availability of rich multi-view training images and are not suitable for single-view setups. In this study, we introduce a new SGA method tailored for cross-view scenarios where scene graph generation and scene synthesis are intertwined. We begin with the fundamental pipeline of cross-view self-localization, and without loss of generality, identify several pipeline variants. These pipeline variants are used as supervision cues to improve robustness and discriminability. Evaluation in an image-goal navigation scenario demonstrates that the proposed approach yields significant and consistent improvements in accuracy and robustness.

1 INTRODUCTION

Neural radiance field (NeRF) and other view synthesis methods have made rapid progress in recent years (Zhan et al., 2023), becoming dominant approaches in the field of robot self-localization. These approaches are particularly powerful in the context of cross-view localization, which involves large viewpoint changes, because they can generate synthetic images that show unfamiliar scenes from different viewpoints. Such cross-view localization is particularly relevant for embodied AI scenarios, such as the recently emerged image goal navigation. For example, in (Mezghani et al., 2022), only one landmark image (i.e., the goal) is provided at the start of navigation, and the goal is to find the desired goal pose based on that landmark image. Thus, the final stage of navigation requires the robot to identify a landmark view and localize its viewpoint relative to the landmark.

An alternative state-of-the-art approach to cross-view self-localization is the scene graph approach. Graph-based scene representations have recently proven effective in such challenging cross-view settings. In the field of robot self-localization, several prior studies have reported that describing a scene using a collection of scene parts rather than a single scene-wide feature is more robust to viewpoint changes, and scene graphs can be seen as an extension of the former approach to describe the relationships between scene parts. For example, in (Parihar et al., 2021), the authors studied self-localization from opposite viewing directions as a typical cross-view self-localization scenario in robot car applications and proved that scene graph-like descriptors are effective in dealing with such challenging cross-view settings.

NeRF and scene graphs have developed independently, and approaches that integrate both to tackle the unified problem of cross-view localization have been largely overlooked. Although these two approaches have conceptually complementary attributes, namely view synthesis and part-based representation, fusing the two is not a simple problem. Common scene graph-based methods assume that all objects in the landmark image are visible in the query image. However, it is not uncommon for certain objects to be missing in the query image, leading to mismatches in scene parsing between training and test images. For example, (Gawel et al., 2018) formulated scene graph recognition as a graph matching problem and presented an approach that is robust to missing objects. However, the proposed method is based on a random walk on the scene graph, and as the authors

Tsukahara, K., Yamamoto, R., Tanaka, K. and Hiroki, T.
SSGA: Synthetic Scene Graph Augmentation via Multiple Pipeline Variants.
DOI: 10.5220/0013098100003912

In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025) - Volume 2: VISAPP*, pages 833-840
ISBN: 978-989-758-728-3; ISSN: 2184-4321

also point out, the computational complexity grows infinitely with the environment scale. In contrast, we considered a scalable formulation, scene graph classification, that is computationally efficient and maintains robustness.

In this study, we consider the scene graph approach from a new perspective, namely graph data augmentation (GDA). GDA is a variant of data augmentation techniques that have emerged in recent years to improve the generalization ability of graph machine learning under uncertainty. Recently, in (Knyazev et al., 2021), scene graph specific augmentation (SGA) was first explored for the task of perturbing real scene graphs to increase the diversity of training distributions and improve the generalization of scene graph inference. However, training their generative adversarial network (GAN)-based models may require expensive annotated datasets. Moreover, their application is limited to two-dimensional (2D) scene understanding, and cross-view settings with 3D viewpoint changes remain unsolved. In contrast, cross-view image localization requires considering the interactions between multiple components, including scene parsing, scene description, and scene classification. This makes existing GDA/SGA techniques inapplicable directly.

To address this challenge, we extend SGA to include a mixed scenario of scene graph generation and synthetic views. Starting from a basic pipeline consisting of scene parsing (P), scene description (D), and scene synthesis (S), we investigate potential misalignments of each pipeline component (P/D/S). We address potential failure modes of individual components and explore different subsets and permutations of pipeline components to create a diverse set of pipeline variants, as indicated by the colored lines with arrows in Figure 1. Specifically, we start with a basic pipeline for cross-view image localization. We then argue, without loss of generality, that different pipeline variants exist and propose to use these pipeline variants as supervision cues to improve robustness and discriminability. The final localization decision is made by consensus of the pipeline ensemble. Our approach achieves significant and consistent improvements in accuracy and robustness when evaluated on the photorealistic Habitat-Sim workspace (Szot et al., 2021).

The main contributions of this work are: (1) We consider a novel approach, called Synthetic Scene Graph (SSG), which extends scene graph descriptors, which have proven effective for cross-view self-localization, with view synthesis techniques. (2) Starting from the base pipeline of SSG, we argue that diverse pipeline variants exist and propose to use these pipeline variants as clues to improve robustness and discriminability. (3) We validate the effectiveness of the proposed method through thorough performance comparison and ablation studies.

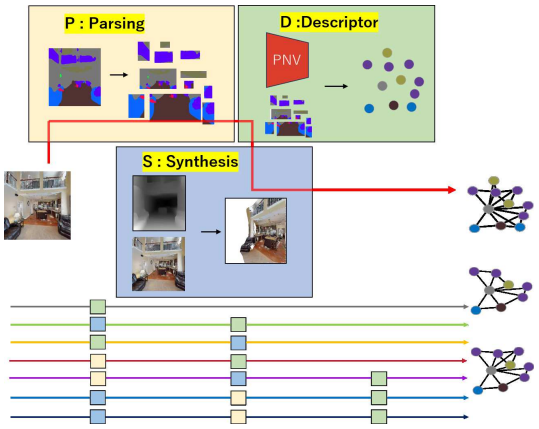

Figure 1: Starting from P-D-S, a basic pipeline for cross-view self-localization, we argue that a variety of pipeline variants exist and propose to use these pipeline variants as supervision cues to improve robustness and discriminability. Throughout this manuscript, P, D, and S are used to represent parsing, descriptor, and synthesis.

2 RELATED WORK

2.1 Cross-View Robot Localization

Cross-view robot localization involves predicting the robot's pose relative to a landmark image under significant viewpoint variations (Zhang et al., 2021). Most localization frameworks, including image retrieval, fail due to assuming that a landmark image similar to the query exists in training. Recent view synthesis methods like neural radiance fields (NeRF) and structure-from-motion have advanced cross-view localization but require spatially dense multi-view training images, making them unsuitable for single-view landmark images. In (Tourani et al., 2021), a typical cross-view robot car scenario was studied, using only training images with opposite orientations to the query. While their scene description is robust, it assumes a fixed 180 deg orientation difference, limiting generality. By contrast, this study introduces a generic cross-view scene-graph setup using multiple landmark images, which need not be spatially dense and may have arbitrary viewing directions.

2.2 Graph Data Augmentation (GDA)

Recently, interest in graph machine learning has grown, leading to the emergence of a new data augmentation field called GDA (Ding et al., 2022). GDA aims to address the gap between observed and actual graphs by perturbing available training samples to create a diverse training set. The non-Euclidean nature of graph data makes this more challenging than data augmentation for image and other data types. Since the process of observing graphs varies widely across applications (e.g., molecular graphs and social networks), there is no universal solution, and application-specific methods have been explored. The most relevant study is the SGA reported in (Knyazev et al., 2021), which considered pipeline processing with a graph neural network and formulated data augmentation as a sampling process using GANs by perturbing intermediate results. However, their work focused on 2D scene understanding, not cross-view scenarios. Additionally, their GAN training requires large annotated datasets, making it unsuitable for several applications, including the sparse training image setup in this study. In contrast, applications involving 3D scene understanding, like the cross-view setup used here, are largely unexplored. To address this, we propose a new SGA method specifically designed for a cross-view robot localization pipeline that includes scene parsing, synthesis, and description.

3 APPROACH

The cross-view image localization framework consists of two independent but interacting modules: scene graph generation (parsing 'P' and description 'D') and scene graph synthesis (synthesis 'S'). The scene graph generation module includes a scene parsing step, which converts the scene into a set of nodes and edges, and a scene description step, which generates node attribute descriptors from the nodes. The scene synthesis module transforms the original view into a synthetic view from a given viewpoint. With the synthetic scene graph descriptors produced by these two modules, cross-view image localization can be treated as an image retrieval problem in the space of synthetic scene graph descriptors. In this section, we begin with a base pipeline (3.1) and extend it to introduce a scene-graph augmentation framework (3.2).

3.1 Base Pipeline

The base pipeline (PDS) performs these steps in the following order: parsing (P) - description (D) - synthesis (S). For brevity, each submodule will be abbreviated using its respective symbol (i.e., 'P', 'D', and 'S') and the pipeline using the string of symbols (e.g., PDS).

3.1.1 Scene Parsing

Specifically, if rectangles overlap, they are considered close. In addition, even if they do not overlap, if the distance between the pixels (for input images with size 256×256) of the rectangles is within 20 pixels, the rectangles are considered to be close. For scene parsing, we employed a traditional two-step method for scene graph generation. First, scene parts (i.e., nodes) were extracted from the input image. Then, these nodes were connected via edges. Three methods?cascade segmentation (Zhou et al., 2017), SLICO (Lei et al., 2021), and Detic (Zhou et al., 2022), which will be described later?were considered for node detection. Spatial proximity was used for edge connections, determined by bounding box proximity. Specifically, rectangles were considered close if they overlapped. Additionally, even if they did not overlap, rectangles were deemed close if the distance between their pixels (for input images of size 256×256) was within 20 pixels.

Cascade segmentation (Zhou et al., 2017) is used for semantic segmentation because it predicts pixel-wise semantic labels and exhibits view invariance, making it effective for cross-view image localization. This method takes a 256×256 RGB image as input, with ResNet50 as the encoder and a Pyramid Pooling Module-based model for the decoder.

SLICO (Lei et al., 2021) was used for appearance-based segmentation, as it provides useful cues independent of semantic segmentation, particularly with region decomposition resembling regular grids that are view-independent.

Detic (Zhou et al., 2022) (model "lvis") was employed for object-level region segmentation, providing area boundaries independent of the other two methods. Trained on a large dataset detecting over 20,000 objects, Detic may be affected by errors due to bounding box shape approximation.

3.1.2 Scene Description

The scene description step aims to describe each node region of a scene graph using image descriptors with discriminability and invariance. We used two types of image descriptors: PatchNetVLAD (PNV) local feature descriptors and convolutional neural network (CNN) global features.

PNV (Hausler et al., 2021) was used as a local feature descriptor because it extracts patch-level

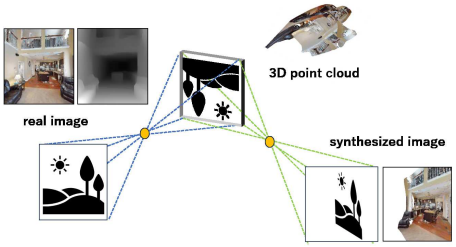

Figure 2: Scene synthesis based on monocular depth estimation and pinhole camera model.

features from NetVLAD residuals and combines the global feature's conditional invariance with the local feature's view invariance. Specifically, 144 PNV features of 512 dimensions were extracted per 256×256 image. For each region (node), the PNV features are aggregated into a bag-of-words (BoW) histogram (i.e., a node feature). BoW uses a prototype dictionary of size $k = 100$, reciprocal rank (RR)-weighted (Cormack et al., 2009), and a naive Bayes nearest-neighbor (NBNN) (Tommasi and Caputo, 2013) similarity measure.

CNN was used as the global feature descriptor because it provides information about scene layout and image regions. Existing CNN-based approaches in visual self-localization generally use either the fully connected layer of CNN as an image feature or the final CNN layer for image classification. We use a hybrid approach. After training an image classifier with the Vgg16 CNN model, we translate the output classification results into class-specific reciprocal rank features (RRF) and use RRF features as graph node features.

3.1.3 Scene Synthesis

The scene synthesis step converts a real image into a 3D point cloud and generates a synthetic scene image for a given virtual viewpoint from any point in the 3D point cloud. The virtual-viewpoint image generation process is shown in Fig. 2. First, the RGB image (256×256) is converted into a depth image using MiDaS (Ranftl et al., 2022), a monocular depth estimation model. The depth image is then converted into a 3D point cloud using a pinhole camera model. The calibration parameters for this model were learned using independent training data with public parameter values[1] and internal parameters provided in Habitat's API (Szot et al., 2021). While recent studies show the effectiveness of instant calibration adaptation through few-shot learning, this supervised method is not applicable in our self-supervised setup, requiring offline calibration. As a result, depth prediction is less reliable compared to few-shot adaptation. MiDaS (Ranftl et al., 2022) was used because it is one of the few monocular depth estimation models designed for generalization across domains.

[1]https://aihabitat.org/docs/habitat-api/view-transform-warp.html

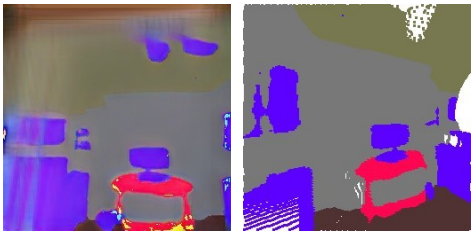

Figure 3: View synthesis results by SynSin (Wiles et al., 2020) (left) and the proposed view-synthesis (right).

Figure 4: Experimental environments.

An example of the resulting synthetic image is shown in Fig. 3, comparing the proposed method with SynSin (Wiles et al., 2020). Unlike NeRF and its variants, which require many training images with dense viewpoints, SynSin only requires one viewpoint image for training, making it suitable for our single-view training setup. However, SynSin's synthetic images exhibit GAN-specific artifacts, as shown in the figure. In contrast, the proposed view synthesis method generates artifact-free virtual viewpoint images.

3.2 Scene Graph Augmentation (SGA)

In this study, we argue that there is not only one dominant pipeline (e.g., PDS), but also other multiple possible pipelines (e.g., PD) and their pipeline variants, and that it is not necessarily obvious which of these variants is optimal (Fig. 1). The center of gravity and the endpoints of the bounding box of each scene part in the synthetic scene graph are determined by the coordinate transformation of the original scene graph. Each pipeline comprises the following independent modules: scene parsing (P), scene description (D), and scene synthesis (S). For instance, pipeline variants that reorder the processing steps (SPD, PSD, and PDS) and pipeline variants that remove some processing steps (S and P) are also formally valid. We observed that these pipeline variants are often not only formally valid but also promising in terms of performance. That is, in comparison with the original pipeline PDS, a SPD pipeline variant with reordered processing steps is attractive because scene parsing is

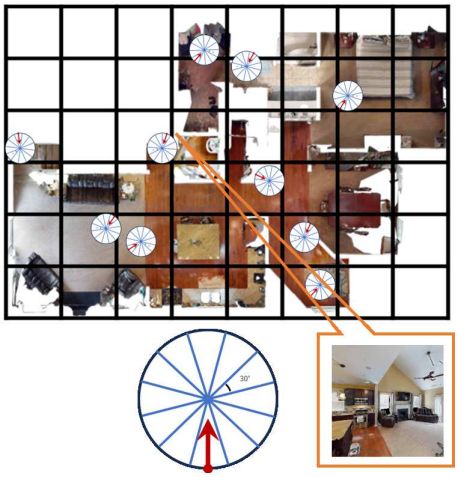

Figure 5: Experimental setup. The workspace is first partitioned into a grid of location cells. Each location cell is then further partitioned into 12 place classes with different orientations.

performed in a synthetic view, and the scene parts can reflect the layout of the synthetic view. DSP and SPD differ in that the former uses descriptors derived from the original image, whereas the latter uses descriptors derived from the synthetic image. In contrast, the pipeline variant PD that skips the scene synthesis step S performs scene description based only on the original view. Therefore, it is not susceptible to scene parsing errors and may be a better choice for scenes with complex layouts that are difficult to parse. In addition, the pipeline variant DS, which skips the parsing step P and describes the scene with only a single whole image node without decomposing the scene into nodes and edges, is attractive because by definition, it is not affected by scene parsing errors.

Thus, we integrated the baseline PDS with other possible pipeline variants: PSD, SPD, DSP, PD, DS, SD, and D, using a graph convolutional neural network (GCN) (Kipf and Welling, 2017). Similar use of GCN as an integration tool was explored in previous work on SGA (Knyazev et al., 2021). However, in the previous study, the scene synthesis problem was not addressed. Consequently, the ensemble of pipeline variants generated by the interaction of scene synthesis and scene graph generation was not considered. Therefore, the proposed approach is different. In the proposed approach, each pipeline-specific GCN maps an input image onto a class-specific probability map. Subsequently, a late fusion step is performed to fuse the class-specific probability maps from all pipeline-specific GCNs using the RR fusion in the spirit of majority voting ensemble strategy. In this framework, we observed that it is more effective to assign higher weights to whole-image nodes than to component nodes. For all K nodes, that is, part nodes, excluding the whole image node, a weight adjustment is performed such that the node feature vector is multiplied by $1/K$. When a system is composed of multiple pipeline variants, the class-specific probability map output from each pipeline variant is converted into a class-specific reciprocal rank vector, and then a single classification is created by reciprocal rank fusion, which makes the final classification result of the system.

4 EXPERIMENTS

We conducted evaluation experiments in a virtual workspace constructed using Habitat-Sim (Szot et al., 2021) and the Habitat-Matterport 3D Research Dataset (HM3D) (Ramakrishnan et al., 2021). This virtual workspace provides photo-realistic images unique to diverse viewpoints with ground-truth annotations. The Habitat-Sim workspace is popular and proven in recent embodied AI applications such as image goal navigation (Mezghani et al., 2022) and thus is considered an important application of cross-view localization.

4.1 Dataset

Habitat-Sim is a flexible high-performance 3D simulator with configurable agents, sensors, and general 3D dataset processing. It prioritizes simulation speed over the breadth of simulation capabilities, achieving thousands of frames per second (FPS) on a single thread and 10,000 FPS processes on a single GPU when rendering scenes from the HM3D dataset. HM3D is a large-scale 3D indoor space dataset, generated from real-world environments, and there are 1000 types of scenes, such as residences, commercial facilities, and public facilities. We experimented using three environments with workspace names “00800-TEEsavR23oF”, “00801-HaxA7YrQdEC”, and “00802-wcojb4TFT35” from the workspaces of the HM3D dataset. A bird’s eye view of the workspace is shown in Fig. 4. The size of the images acquired by the robot is set to 256×256. Additionally, the dataset contains information on the viewpoint’s location (x, y) and orientation θ associated with each image.

First, the workspace was partitioned into a grid of location cells with dimensions of 2[m]×2[m]. Each view image in the workspace was considered to belong to a location cell if the visibility cone, determined by its viewpoint, included the centroid of the location cell. If multiple such location cells exist, the

Figure 6: Example results. In the upper row, from left to right, each panel is a training image, 2 level 2 test images, and 2 level 3 test images. The bottom row is the corresponding composite image of each virtual viewpoint.

cell with the center of gravity closest to its viewpoint is selected. In this experiment, we randomly selected ten location cells for each workspace. Furthermore, as shown in Fig. 5, 12 place classes with different orientations were defined for each location cell. Therefore, the total number of place classes for each workspace was $12 \times 10 = 120$.

To investigate cross-view localization performance, *only one* training image was given for each of the 10 location cells. That is, of the 12 place classes belonging to one location cell, a training image is available for one class, and is not available for the remaining 11 classes. There is only one place class for which a real training image is available, and in addition, care is taken to ensure that different viewpoints are sampled as the test images' viewpoints for this class. For classes for which there are no training images, methods with the 'S' (Synthesis) module can synthesize pseudo-training images from available training images, while methods without the 'S' module have no choice but to simply use available training images as they are. 1500 test images independent of training images were sampled for each class.

4.2 Performance Index

In this experiment, we used Top-1 Accuracy, Top-5 Accuracy, and mean reciprocal rank (MRR) as performance indicators. Top-1 accuracy is useful because it reflects the percentage of test data in which both location and orientation are correct. Top-5 and MRR are useful for determining the percentage of test data in which only either location or orientation is correct. This latter type of performance metric is particularly important in the context of multi-view self-localization, multi-hypothesis tracking, and map-based navigation, where it is not always necessary to uniquely refine both location and orientation from a single-view observation.

Because the difficulty of cross-view image localization is significantly affected by the difference in viewpoint between the query and test images, we define three levels of difficulty and use them to investigate their relationship with the estimation accuracy. Specifically, view overlap rate (VO) [%], the area of the intersection of 2D visibility area on the bird's-eye view 2D plane between the query and test viewpoints, was calculated from the ground-truth (provided by Habitat-Sim) of the robot's viewpoints and occluding obstacles. Then, the difficulty levels were classified as follows: level 1: VO$>$60, level 2: 30$<$VO$\leq$60, and level 3: VO$\leq$30.

4.3 Results

Table 1 compares the performance of the proposed method with that of several baseline and ablation methods. From the table, it is evident that compared to the baseline methods, the proposed method ("ours") exhibits superior performance overall across most dataset levels. Cross-view image localization, the primary focus of this research, operates under the assumption of "level 2" and "level 3" difficulty levels. In "level 2" and "level 3", the proposed method outperforms the baseline methods in the performance index Top-1 Accuracy. In addition, the performance indicators Top-5 Accuracy and MRR are significantly higher. Therefore, the effectiveness of the proposed method was confirmed.

When comparing the two descriptors, CNN (DS) and PNV (DS, SD), it is observed that for Top-1 Accuracy, the performance of CNN tends to be higher than that of PNV, whereas for Top-5 Accuracy, the performance of PNV tends to be higher than that of CNN. This is because CNN, which is a global feature descriptor (from the entire image), exhibits strong characteristics for identifying the entire image. Therefore, it can predict location classes with higher accuracy than PNV. In contrast to global feature descriptors, PNV, which is a local feature descriptor (with partial features), is resistant to changes due to rotation and translation. Therefore, it can narrow down the correct location class candidates with higher accuracy than CNN. Comparing the cases with synthesis (DS, SD) and without synthesis (D), we can confirm the effectiveness of the scene synthesis.

Regarding the method with a synthesis module ('S'), an interesting difference in behavior was observed between the case of transferring the real image descriptor to the synthetic viewpoint (DS) and the case of using the synthetic image descriptor (SD). . It can be seen that SD tends to have better overall performance than DS. DS tends to perform better than SD for top-1 accuracy, and DS performs better for top-5 accuracy. This shows that DS is better than SD in terms of accuracy, but the opposite is true in terms of robustness.

The pipeline variant SD may be effective in pre-

Table 1: Performance results.

	Top-1[%]			Top-5[%]			MRR[%]		
	00800								
	L1	L2	L3	L1	L2	L3	L1	L2	L3
CNN	31.0	14.7	12.2	35.1	20.5	19.7	35.3	19.7	17.0
P (DS)	17.0	10.9	6.4	52.6	28.9	22.6	32.3	21.1	13.7
P (SD)	28.2	13.2	9.3	44.2	26.6	18.8	35.8	20.2	14.7
P-SE (PDS)	21.6	13.6	3.5	52.6	37.3	23.0	35.9	24.1	12.0
P-SE (PSD)	26.6	14.3	1.2	41.4	26.6	17.9	34.4	21.7	16.0
P-DE (PDS)	20.8	12.6	5.4	52.4	32.6	20.5	34.8	22.6	12.5
P-DE (PSD)	23.3	10.7	7.6	36.3	24.2	12.4	30.9	18.0	11.5
P-SL (PDS)	27.8	10.8	7.9	60.0	30.4	22.7	42.2	21.2	15.0
P-SL (PSD)	29.5	9.9	7.1	47.7	28.7	19.8	38.4	19.9	13.5
P-SL (SPD1)	24.4	14.6	9.3	59.7	36.7	22.9	40.1	25.5	16.1
P-SL (SPD2)	30.2	15.0	7.9	42.8	24.5	15.9	36.9	20.7	12.9
Ensemble (Ours.)	31.4	17.5	13.7	55.8	32.3	23.2	42.5	26.2	18.7
	00801								
CNN	34.8	10.5	4.06	39.3	16.8	6.7	0.385	0.150	0.073
P (DS)	15.0	7.6	1.94	51.6	33.9	14.2	30.6	19.2	8.6
P (SD)	26.7	10.0	5.74	52.3	21.7	8.68	37.8	17.1	8.8
P-SE (PDS)	18.0	9.3	4.8	49.2	31.5	12.7	32.7	20.0	9.8
P-SE (PSD)	28.6	10.1	4.2	43.1	18.4	8.3	36.7	16.2	7.3
P-DE (PDS)	15.8	6.8	2.1	41.7	23.0	9.0	27.5	15.1	6.7
P-DE (PSD)	16.8	7.0	3.0	31.2	14.0	6.6	25.1	12.2	6.2
P-SL (PDS)	17.6	9.7	3.4	55.7	28.1	13.0	33.8	19.6	9.0
P-SL (PSD)	28.3	9.5	3.9	45.8	15.8	7.5	37.1	15.0	7.3
P-SL (SPD1)	20.9	7.1	4.2	50.1	27.1	12.4	34.8	17.6	11.3
P-SL (SPD2)	24.6	8.9	4.7	46.6	22.3	9.5	35.1	16.3	9.7
Ensemble (Ours.)	31.2	11.0	4.75	59.5	30.3	13.1	43.9	20.7	10.0
	00802								
CNN	38.4	10.0	3.67	40.7	15.1	4.1	41.7	14.5	6.4
P (DS)	30.5	15.1	5.72	66.7	37.1	16.2	46.3	25.5	10.7
P (SD)	36.3	11.3	4.20	54.3	26.3	12.2	43.9	20.3	9.5
P-SE (PDS)	29.2	14.1	8.4	56.8	38.8	16.5	42.3	25.9	13.1
P-SE (PSD)	37.6	10.1	3.9	52.0	22.6	13.0	45.0	18.3	9.5
P-DE (PDS)	16.8	9.5	3.3	44.3	28.5	6.6	28.6	19.0	6.7
P-DE (PSD)	26.0	11.2	3.4	43.5	26.3	7.2	43.5	19.0	7.1
P-SL (PDS)	22.7	11.3	8.3	65.1	39.6	20.0	41.1	24.6	14.5
P-SL (PSD)	37.4	10.4	3.9	56.4	31.4	12.0	46.5	20.8	10.2
P-SL (SPD1)	30.9	16.3	7.1	63.3	42.4	16.6	45.6	28.6	12.6
P-SL (SPD2)	38.6	12.8	3.9	61.4	30.9	12.2	48.8	22.1	9.9
Ensemble (Ours.)	40.0	13.2	4.16	65.4	42.5	17.4	51.6	26.2	10.9

00800: 00800-TEEsavR23oF, 00801: 00801-HaxA7YrQdEC,
00802: 00802-wcojb4TFT35, L1: level1, L2: level2, L3: level3,
P: PatchNetVLAD, SL: SLICO, SE: cascade segmentation, DE: Detic

dicting both location and orientation; however, because it relies on the accuracy of synthetic images from a virtual viewpoint, recognition instability tends to be significant. In contrast, the pipeline variant DS does not exhibit such instability in recognition. However, its ability to discriminate both location and orientation tends to be weaker.

We also compared scene graphs containing only whole nodes (e.g., DS) and scene graphs consisting of whole nodes and part group nodes (e.g., PDS). The results show that PDS tends to perform better than DS, confirming the effectiveness of scene parsing. Regarding the performance of each pipeline, it is observed that the pipelines with high performance vary from datasets. Thus, all the eight types of pipelines exhibit their own unique strengths and weaknesses. It can be concluded that combining all the pipeline variants into an ensemble achieves a stable and consistent performance improvement.

5 CONCLUSIONS AND FUTURE WORKS

In this work, we tackled the challenge of cross-view image localization using synthetic scene graphs. Starting with a standard pipeline for scene graph-based localization, we proposed using pipeline variants as supervision cues to enhance robustness and discriminativity. Through pipeline ensembles, ablation studies, and performance validation, we demonstrated that the proposed self-supervision cues consistently improve performance. While the method's effectiveness is clear, further improvements are possible. Both scene graph generation and view synthesis show strong invariance, and we believe the framework can be enhanced by integrating various self-localization techniques. Ensemble learning is a promising direction for future work (Islam et al., 2003).

REFERENCES

Cormack, G. V., Clarke, C. L. A., and Büttcher, S. (2009). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 758–759.

Ding, K., Xu, Z., Tong, H., and Liu, H. (2022). Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 24:61–77.

Gawel, A., Del Don, C., Siegwart, R., Nieto, J., and Cadena, C. (2018). X-view: Graph-based semantic multi-view localization. *IEEE Robotics and Automation Letters*, 3(3):1687–1694.

Hausler, S., Garg, S., Xu, M., Milford, M., and Fischer, T. (2021). Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 14141–14152.

Islam, M. M., Yao, X., and Murase, K. (2003). A constructive algorithm for training cooperative neural network ensembles. *IEEE Transactions on neural networks*, 14(4):820–834.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017*.

Knyazev, B., de Vries, H., Cangea, C., Taylor, G. W., Courville, A., and Belilovsky, E. (2021). Generative compositional augmentations for scene graph prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15827–15837.

Lei, K.-P., Feng, X.-X., and Yu, W.-S. (2021). A shadow detection method based on slico superpixel segmentation. In *2021 International Symposium on Computer Technology and Information Science (ISCTIS)*, pages 294–298. IEEE.

Mezghani, L., Sukhbaatar, S., Lavril, T., Maksymets, O., Batra, D., Bojanowski, P., and Alahari, K. (2022). Memory-augmented reinforcement learning for image-goal navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3316–3323.

Parihar, U. S., Gujarathi, A., Mehta, K., Tourani, S., Garg, S., Milford, M., and Krishna, K. M. (2021). Rord: Rotation-robust descriptors and orthographic views for local feature matching. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1593–1600.

Ramakrishnan, S. K., Gokaslan, A., Wijmans, E., Maksymets, O., Clegg, A., Turner, J. M., Undersander, E., Galuba, W., Westbury, A., Chang, A. X., Savva, M., Zhao, Y., and Batra, D. (2021). Habitat-matterport 3d dataset (HM3D): 1000 large-scale 3d environments for embodied AI. *CoRR*, abs/2109.08238.

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., and Koltun, V. (2022). Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(3):1623–1637.

Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D. S., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier, F., Galuba, W., Chang, A. X., Kira, Z., Koltun, V., Malik, J., Savva, M., and Batra, D. (2021). Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems 34*, pages 251–266.

Tommasi, T. and Caputo, B. (2013). Frustratingly easy NBNN domain adaptation. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 897–904.

Tourani, S., Desai, D., Parihar, U. S., Garg, S., Sarvadevabhatla, R. K., Milford, M., and Krishna, K. M. (2021). Early bird: Loop closures from opposing viewpoints for perceptually-aliased indoor environments. In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 409–416.

Wiles, O., Gkioxari, G., Szeliski, R., and Johnson, J. (2020). Synsin: End-to-end view synthesis from a single image. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7465–7475.

Zhan, F., Yu, Y., Wu, R., Zhang, J., Lu, S., Liu, L., Kortylewski, A., Theobalt, C., and Xing, E. (2023). Multimodal image synthesis and editing: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, X., Wang, L., and Su, Y. (2021). Visual place recognition: A survey from deep learning perspective. *Pattern Recognit.*, 113:107760.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. (2017). Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641.

Zhou, X., Girdhar, R., Joulin, A., Krähenbühl, P., and Misra, I. (2022). Detecting twenty-thousand classes using image-level supervision. In *Computer Vision – ECCV 2022*, pages 350–368, Cham.