Integrating Symbolic Reasoning Into Neural Generative Models For 3D-Space Design Generation

Xiangrui Kong
Purdue University

Abstract

This paper stands for the final report of CS 497, the Honor Research Course. It is mainly about my experience in working on the 3D version of the SPRING project, which expands on the pre-print 2D model. Started by learning and exploring the existing 3D demo of the SPRING structure, I was able to apprehend and replicate the basic pipeline of the perception module in a python notebook, which provides me the visualizations of each intermediate results. Then I learned linear programming along with the SRM module, the backbone of the SPRING project. Then towards the end of the semester, I switched back to explore and experiment more on the perception module pipeline and improved some of the key sub-routines. Furthermore, I also made efforts on learning the math and implementations of Stable Diffusion, the state-of-art text-to-image generative model, which helped facilitate my understanding of the VEG module in SPRING model. Other than the SPRING project, attending Professor Xue's weekly meeting with all his intelligent PhD students provided me valuable chances of appreciating the most cutting-edge researches in the field as well as getting plenty of practical, real-life research tips which would be precious and meaningful for my future academic journey.

1 Introduction

The field of text-to-image models has seen significant growth and focus with the continuing development of deep learning networks, leading to remarkable advancements that have the potential to revolutionize indoor design. These neural network models are now capable of transforming specific user prompts or requirements into detailed images, thereby streamlining the design process. This technology not only enhances creativity but also makes it more accessible, allowing for the easy visualization of design concepts. However, that's only a half of the story. Achieving good design generation requires tight integration of neural and symbolic reasoning, as the generated deign should be able to not only reflect reasonable rules for utility and aesthetics, but also must satisfy essential user needs. Currently, generative design applications driven by neural networks are capable for creating pleasing designs but failed for meet necessary user specifications. At the same time, other non-neural network models that programmed with hard symbolic constraints sacrifice the ability of absorbing low-level implicit information such as aesthetics and physical legitimacy. To bridge this gap, the original 2D version of the Spatial Reasoning Integrated Generator (SPRING)[2] was introduced for design generation. SPRING embedded a neural and symbolic integrated spatial reasoning module inside the deep generative network, which the neural generative network responsible for deciding the location of objects, while the symbolic filter for achieving constraint satisfactions.

My work for this semester built on the existing demo of the 3D SPRING implemented by my PhD group leader, which is the expansion of the original 2D SPRING. The former version of SPRING takes in a 2D image, processing it within a two-dimensional domain, and rendering a modified 2D image that aligns with specific user requirements. However, given the intrinsic three-dimensional nature of indoor design, the task of interpreting and manipulating 3D objects within the confined 2D space presents certain limitations and it performs the task distanced from the realistic scenario. Therefore, capturing, integrating, and encapsulating meaningful 3D information into the reasoning module of SPRING seems a reasonable next step. Thus, we extend the idea of SPRING into three-dimensional world, and proposed a new version of SPRING. The novel SPRINGV2, which stands for SPRING version 2, is designed to ingest 2D images, transform and extrapolate them in the three-dimensional space, and finally compress the 3D representation into 2D results. With

this advanced pipeline, SPRING model would be capable of providing more reliable and practical predictions of object locations in a real-world scenario, which has the potential facilitating and contributing to the field of indoor design.

2 SPRINGV2 Structure

The newly implemented SPRINGV2 consists of three modules. The first perception module is responsible for extracting features from the input background images, such as potential boundaries, objects' positions, and depth estimations. Then spatial reasoning module (SRM) is designed to learn the implicit three-dimensional space and generate bounding cuboids using neural and symbolic integrated approaches. Finally, the visual element generator(VEG) utilized the state-of-art generative image inpainting model Stable Diffusion to render the actual object back in a two-dimensional space.

2.1 Perception Module

To extract features from the input image, firstly the perception module utilize the pre-trained DETR50 [1] model for object detection, and to further facilitate the process, we use Segment Anything Model (SAM) [4] for image segmentation. Then the segmentation is used for object and boundary detection. After collecting image information, we use Depth estimation models [3, 5, 7] to transform 2D inputs to 3D space. The choices for a certain 3D model is currently not settled and needs future experiments. Then the perception module converts 3D representation with features into accessible forms such as point clouds and Json files for future use.

2.2 Spatial Reasoning Module (SRM)

The centre of SPRINGV2 lies in the SRM module. The spatial reasoning module decides the spatial position of each object. Firstly, it transform the Json file containing feature information into a graph, then utilize Graphical Convolution Network as feature extractor. Then to generate object positions, the SRM module utilize a RNN structure, more specifically, implementing using a GRU, which takes the input of the background image, the object identifiers, and outputs each object position. For each object, the RNN structure generates corresponding decision scores for all spatial variables over 3 actions, 'left' - which takes the first half of the range(ex: if available range is [0, 1000], this action will take [0, 500]), 'right' - which takes the second half of the range, and 'end' - which takes the mid-point of the range and continue with the next variable. After that, the SRM module performs a forward checking by reducing the decision score problem into a mixed integer programming problem and adding specific user constraints to the linear programming solver. The forward checking process gets a global optimal object locations of the input image while satisfying user requirements. This is the symbolic aspects of the model.

2.3 Visual Element Generator

The visual element generator of the original SPRING model is responsible for generating actual visual elements using state-of-art model, such as Stable Diffusion model. Thus, after the SRM model predicted/ generated the output for the bounding cuboids, the Visual Element Generator would converts the 3D information back to 2D settings, then utilized the Stable Diffusion model to outputs the actual inpaintings.

3 Results

Over the past semester, I replicated and tried to improve the extended version of SPRINGV2 using Jupyter Notebook, which allows me to see intermediate results for every operation of the model and helps me better understand the model pipeline. In addition, during implementation, I noticed some of the subroutines in the perception module did not perform as expected or as well as we previously thought. Therefore, I also targeted those functions and carried out my own experiment in order to improve the overall performance.

3.1 Replicating Perception Model

For the perception module, I firstly utilized the pre-trained DETR50[1] object detection model to get the initial list of detected objects. I found out that the DETR50 could only identify object labels previously existed in the model's label set, which cause significant inaccuracy of the performance. In addition, the hyperparameters during post-processing of object filtering also contributes largely when identifying objects, which leaves spaces for further experiments and research.

After using object detection model, as implemented, I also utilized the pre-trained Segment Anything Model[4] to obtain the segmentation masks of the input image. Using the list of masks, the perception model could potentially identify more undetected objects which the DETR50 model was unable to distinguish. Furthermore, the obtained segmentation masks are essential to detect boundaries, including 'walls', 'ceilings', and 'floors'.

Figure 1. The result image from the pre-trained DETR50 object detection, which identifies certain objects in the model's label set.

These boundaries are keys for placing objects in SRM, since by nature of gravitational force, objects are either placed on the floor, mounted on the wall, or hanged over the ceiling. Therefore, if the module could not "observe" potential boundaries in the input image, the final placement of the objects would simultaneously become unrealistic. Following the segmentation, the perception model performs post processing procedures to sorted and combined similar or overlapping masks, and finally identified potential boundaries in the below metric: if the area of a specific mask is greater than the threshold we defined, the module considers the mask as boundaries. Then by checking the maximum or minimum pixels of the mask, the module labels the mask as one of the three boundaries.

Figure 2. The result overlay image from the pre-trained SAM model

After that, I utilized the pre-trained depth estimation models, including Depth-Anything, GLPN image processor and DPT image processor[3, 5, 7]. Then I followed the implemented 3D SPRING source code to convert the depth estimation into point cloud based on objects, planes and the entire image. After that, these format supports me to use MeshLab software to observe the results. Finally, the perception model converts the depth representations into Json files for future use.

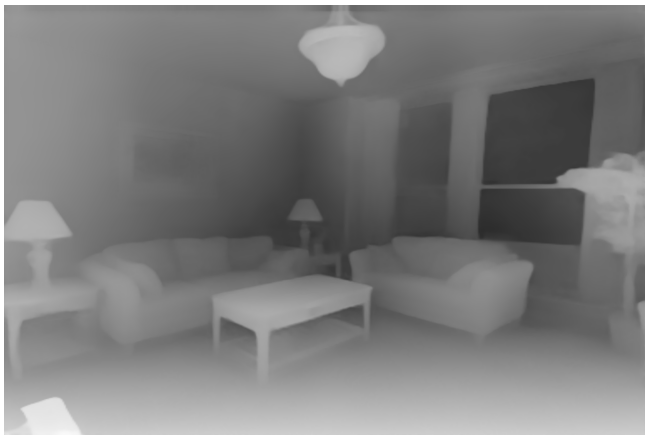

Figure 3. The result image for depth estimation model

3.2 Improving Combining Segmentation

When directly utilizing the pre-trained SAM model to obtain segmentation masks usually faces limitations and flaws as the model providing similar or overlapping masks, which is unintended and troublesome when detecting objects and walls. These repetitive masks could cause the perception module to produce same object or boundary more than once, causing inaccuracy in later operations. Therefore, one of the actions in the post processing procedure after getting the segmentation masks from the SAM model is to combine overlapping masks or removing highly similar masks. The subroutine the module previously implemented as follow: for a specific mask among all the segmented masks, and for every other masks than the current one, if the other masks have overlapping with the current one, combine all such masks as one. Then the function calculates the jaccard index, a statistic used for gauging the similarity and diversity, between them.

$$jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

If the jaccard index exceeds the defined threshold, remove the current mask.

However, during testing, this method of combining segmentation would still output repetitive masks, which by my observation, I assume it was because the existence of outlier pixels in some of the segmentation masks. This would produce lower jaccard index score because it would increase the overall area, the logical or, between them, which bypassed the threshold.

To counter this problem, I borrowed the same thoughts as previous method, but slightly modified the implementation. Instead of combining every other segmentation masks, I used a more straightforward way, which checking the list of masks in a linear way: for a specific mask, checking every other masks, if there exists one masks that produce high jaccard index with the current one, remove the current mask. The idea behind this greed style method is, if such mask to be removed, then there must exists one similar mask that represent the same object or boundary to the removed one, which we will remove all repetitive masks without omitting any mask needed.

To evaluate two methods, I collected a small dataset of 20 images of indoor scenes. The experimental results suggest that the newly implemented method output 0 repetitive masks and 0 omitted masks. On the other hand, the old methods produces approximately 26.8 percent of inaccuracy.

3.3 Improving Wall Detection

Another key subroutine in the post processing procedure is the method of detecting/labeling the potential boundaries, including "walls", "floors", and "ceilings". Since following the perception module, SRM model is responsible for learning the implicit knowledge of indoor object placements, such as a table needs to be placed on the floor or a TV could be mounted on the wall, the detection of boundaries is essential for the correct prediction of objects.

Using the output segmentation masks from SAM model, the previous wall detection method identify potential boundaries as below metrics: for a specific mask in the segmentation list, calculate its area by summing up all its pixels. In addition, we calculate the area of the input image. Then the subroutine would compare the ratio between the area of the mask and the total area of the background image. If the ratio exceeds the defined threshold, the mask would be considered as a potential boundary. Finally the function distinguishes the boundary as "wall", "floor" and "ceiling" by determining whether the mask's centroid y position is exactly at the middle, close to bottom, close to the top. During testing, I found out this brute-force detecting metric acts poorly even for simple backgrounds with sparse objects. The straightforward method is vulnerable to images containing large objects since these objects easily occupying larger areas in the frame, thus exceeding the threshold.

To counter this issue, I designed a new metric of boundary detection, which performs as follows: instead of utilizing segmentation area as criterion, the new method makes use of the "span" of a certain mask, which is the distance between its maximum and minimum x coordinates (and y coordinates respectively). If a specific mask extend over(either x coordinates or y coordinates) the defined threshold, the new method will identify it as either walls, floors or ceilings based on its y coordinate. The intuition behind the new method lies in the observation that a potential boundary does not necessarily occupy a large area, but a wider "span". For floors and ceilings, they normally across wide range of x coordinates at the top of the image, or on the bottom. In addition, for walls, they oftentimes stretch across the horizontal middle line.

Similarly, I collected a small datasets of background images for experiment, and manually labeled their boundaries as ground truths. Then I compared the performances between the two methods, and concluded that the new method achieving strikingly better accuracy than the previous metrics (98% accuracy versus 26% accuracy).

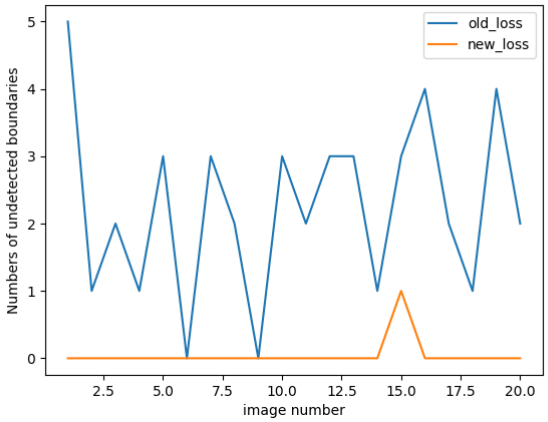

Figure 4. Experiment results of comparing two methods of wall detection, where the blue line representing the old method and orange line for the new method. It is obvious that the new method achieve much higher accuracy in detecting boundaries

4 Plans for future works

As an ongoing project, there are definitely more things worth exploring. During weekly meetings with Professor Xue, he came up with a new idea of embedding physical engine into SPRING model which could help it produce more realistic results and push the research even further. After finishing this semester, I hope I can continue working with the current research group and discover more opportunities around the SPRING project. Furthermore, during this summer, I will stay on campus and ideally work on a new research topic revolving around Large Language Model, and its potential novel applications. To do that, I will start to dive into the API for using such models (ChatGPT for example), and present my findings to the group. Then I will explore the topic throughout this summer.

5 Related works

Recent years, the field of computer vision witness some groundbreaking discoveries, which advanced the capabilities of various models and frameworks. Accentuated in this paper as the pivotal model, The generative model Spatial Reasoning Integrated Generator (SPRING)[2] integrates neural and symbolic spatial reasoning modules, providing aesthetic and practical indoor designs that meet user needs.In SPRING, many aspects of computer vision models are used to facilitate the pipeline. In object detection, the End-to-End Object Detection with Transformers (DETR) model [1] represents a significant milestone. DETR employs the transformer architecture to directly predict object bounds and classes, which simplifies the object detection pipeline and enhances the efficiency and accuracy of the process. The versatility in segmentation is highlighted in the work "Segmenting Anything"[4] which demonstrates a framework capable of accurately identifying and segmenting a diverse range of objects within various scenes. Depth estimation, essential for interpreting three-dimensional spaces from two-dimensional images, has seen innovative approaches such as the Global-Local Path Networks[3] and Vision Transformers for Dense Prediction[5], which leverage the transformer structure for robust depth estimation. Additionally, the recent Depth Anything model[7] further enhances performance in this area. Lastly, the Text-to-Image Generative Model, Stable Diffusion[6], also adds to the spectrum of creative possibilities by enabling high-quality image generation from textual descriptions. Together, these developments create a rich landscape of tools and methodologies that significantly contribute to advancements in the field.

6 Conclusion

In conclusion, this final report encapsulates the motivation and brief introduction of SPRING research project, my work over the semester, and future plans. My involvement in Professor Xue's research group, specifically with PhD student Max, has been a pivotal period of learning and contribution. Even in this brief period, my involvement in this research project has significantly enhanced my understanding of academic research, offering me a valuable opportunity to apply my academic knowledge to broader, more practical contexts. The journey thus far has been immensely enriching, and I look forward to continuing delve deeper into the research, aiming to contribute further to the development of the SPRING project as well as other intriguing topics.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-End Object Detection with Transformers. arXiv:2005.12872 [cs.CV]

[2] Maxwell Joseph Jacobson and Yexiang Xue. 2023. Integrating Symbolic Reasoning into Neural Generative Models for Design Generation. arXiv:2310.09383 [cs.AI]

[3] Doyeon Kim, Woonghyun Ka, Pyungwhan Ahn, Donggyu Joo, Sehwan Chun, and Junmo Kim. 2022. Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth. arXiv:2201.07436 [cs.CV]

[4] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment Anything. *arXiv:2304.02643* (2023).

[5] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. 2021. Vision Transformers for Dense Prediction. arXiv:2103.13413 [cs.CV]

[6] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752 [cs.CV]

[7] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. 2024. Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data. arXiv:2401.10891 [cs.CV]