AI3064 Computer Vision

Final Project

Report Due: Jan. 5, 2025

The objective of the final project is to make use of what you have learned during this course to explore potential solutions to problems using computer vision. The final deliverables will be

- A written report, which is limited to eight pages, including figures and tables, in the CVPR style. Additional pages containing only cited references are allowed.
- Source code of your implementation, which should be executable.

You will work in **groups of no more than 4**. To facilitate forming teams, we will create a team search form on Canvas. You could either look for a team or look for more students.

Only one member of your team will need to submit your work on Canvas, but make sure that this member successfully adds all team members on Canvas.

Below we will define some example projects that you can work on. The goal of the project is to explore and to compare different approaches to whatever problem you choose to work on. Grading will reflect the thoroughness and quality of that exploration.

We will ask the following questions:

- How challenging is this problem?
- What approaches did you take to solve this problem, and why?
- How did you explore the space of solutions -- architectures, hyperparameters, training methods etc. -- your chosen approach(es) presented?
- How did you evaluate the performance of the final approach(es) you investigated?

The key point here is that we're not just interested in the absolute performance of what you produce, but also the thought process that led you there. Ideally, you will have done some comparative performance analysis to back up the choices you made.

Grading will be based on the completeness of the project, the clarity of the writeup, the level of complexity/difficulty of the project, and your ability to justify the choices you made.

Schedule

Last two weeks of class: Each team makes a short (< 10 minute) presentation on your project.

Jan. 5 (TBD): Final deliverable submission due.

Final Report Format

Your final report should document the following:

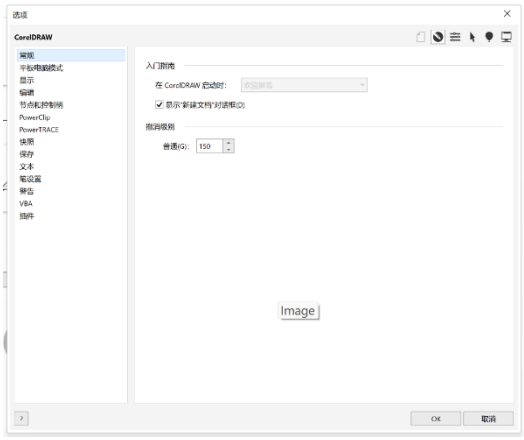
Introduction

Related work, Methodology, Experimental results, and Conclusion.

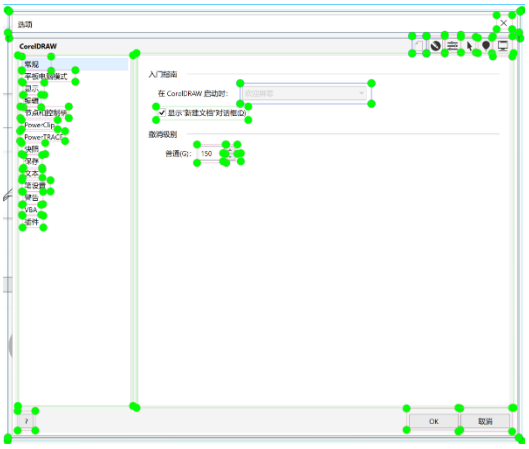
Projects

UI Control Recognition

In today's fast-paced digital world, the rapid growth of the application industry has highlighted the importance of the **User Interface (UI)**, which organizes and visually represents the layout and structure of different controls within applications. As AI technology progresses, the ability to automatically recognize these controls has become increasingly essential. For instance, software like Adobe Premiere Pro and Adobe Photoshop offer sophisticated video and image editing functionalities, but their extensive feature sets present a steep learning curve for new users, who often spend significant time familiarizing themselves with complex desktop applications. Developing an automated software agent capable of executing application operations independently could alleviate the need for extensive software learning, thus enhancing work productivity.



Screenshot



Recognition Result

Unlike mobile applications, however, it is challenging for users to access control information on desktop UIs. These applications lack a unified framework and, due to privacy constraints, often don't expose a comprehensive control tree (layout information), which restricts the agent's ability to automate operations on these applications. As a result, **vision-based UI content parsing technology** plays a critical role in addressing this challenge.

In this project, your task can be concluded as follows:

What you already have: A range of available object detection algorithms and UI recognition literature to inform the task. Access to open-source projects (e.g., MobileAgent, Open-GroundingDino, Yolo, PaddleOCR, APPAgent, OS-world).

What you need to do: Design an algorithm to automatically recognize control information (position and category) in screenshots of PC software and convert it into a standardized layout file format.

Your evaluation metric: Precision, Recall, F1-Score (Download the scripts)

Your datasets: Training data, testing data.

Tips:

You can choose one or more datasets to evaluate your model. It is encouraged to adopt more datasets. Some other datasets are accepted.

In your final report, you should submit executable codes. The results in your report must be reproducible.

Related Open-Source Projects

MobileAgent, Open-GroundingDino, Yolo, PaddleOCR, APPAgent, OS-world

References:

- Rico: A Mobile App Dataset for Building Data-Driven Design Applications
- Mobile-Agent-v2: Mobile Device Operation Assistant with Effective Navigation via Multi-Agent Collaboration
- ASSISTGUI: Task-Oriented PC Graphical User Interface Automation
- YOLO-World: Real-Time Open-Vocabulary Object Detection
- Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments
- Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection.

Indoor scene reconstruction

Scene reconstruction is the process of reconstructing a digital version of a real-world object from pictures or scans of the object. It is a very complex problem with many research histories, open problems, and possible solutions. Currently, most approaches to this problem can be broken down into two broad phases: first, you take your input and turn it into a cloud of points on the object's surface, then reconstruct a mesh from this point cloud. These approaches are summarized as SFM (structure from motion) and MVS (multi-view stereo). Recently, neural rendering-based methods are also popular in reconstructing scenes. Neural rendering is an emerging class of deep image, and video generation approaches that enable control of scene properties such as illumination, camera parameters, pose, geometry, appearance, and semantic structure. It combines machine learning techniques with physical knowledge from computer graphics to obtain controllable and photo-realistic models of scenes.

In this project, your task can be concluded as follows:

What you already have: Calibrated cameras' positions and rotations, posed scene images (RGB or RGB-D), any on-the-shelf pretrained model.

What you need to do: Reconstruct indoor scenes with respect to mesh.

Your evaluation metric: Accuracy, completeness, precision, recall and F-score. (You may find the definition of these metrics in Atlas)

Your datasets: Replica. Replica is a reconstruction-based 3D dataset of 18 high fidelity scenes with dense geometry, HDR textures, and semantic annotations. A separate metrics report is encouraged. You can download the entire data from Omnidata (https://github.com/EPFL-VILAB/omnidata/tree/main/omnidata_tools/dataset#readme).

Tips:

You can choose one or more datasets to evaluate your model. It is encouraged to adopt more datasets. Some other datasets are accepted.

Both RGB-based methods and RGB-D-based methods are acceptable. However, RGB-D-based methods would be more demanding.

Images are provided with poses, while there would be a bonus if you estimate the poses from images and compare them with the ground truth.

In your final report, you should submit executable codes. The results in your report must be reproducible. A detailed table for performance among different scenes must be in the report. Reconstructed meshes ought to be attached as supplementary for visualization.

References:

- Murez, Zak, et al. "Atlas: End-to-end 3d scene reconstruction from posed images." European conference on computer vision. Springer, Cham, 2020.
- Schonberger, Johannes L., and Jan-Michael Frahm. "Structure-from-motion revisited." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- Yao, Yao, et al. "Mvsnet: Depth inference for unstructured multi-view stereo." Proceedings of the European conference on computer vision (ECCV). 2018.
- Huang, Po-Han, et al. "Deepmvs: Learning multi-view stereopsis." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
- Sucar, Edgar, et al. "iMAP: Implicit mapping and positioning in real-time." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.
- Zhu, Zihan, et al. "Nice-slam: Neural implicit scalable encoding for slam." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.
- Azinović, Dejan, et al. "Neural RGB-D surface reconstruction." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.
- Yariv, Lior, et al. "Volume rendering of neural implicit surfaces." Advances in Neural Information Processing Systems 34 (2021): 4805-4815.
- Wang, Peng, et al. "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction." arXiv preprint arXiv:2106.10689 (2021).
- Eftekhar, Ainaz, et al. "Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

Medical Image Segmentation Based on Large Models

Image segmentation is a classic problem in computer vision research and has become a major focus within the field of image understanding. It is the foundational step in image analysis and a key component of computer vision, representing one of the most challenging tasks in image processing. Image segmentation involves dividing an image into multiple non-overlapping regions based on features like grayscale, color, spatial texture, and geometric shape, ensuring that these features are consistent or similar within each region but markedly different between regions. Simply put, it separates the target object from the background in an image. Due to the intrinsic importance and difficulty of this problem, segmentation has attracted significant research attention since the 1990s. With advancements in deep learning and large models (especially segmentation-focused models like the Segment Anything Model, SAM), substantial progress has been made in segmentation tasks.

In this project, your task can be concluded as follows:

What you already have: SAM (Segment Anything Model) for segmentation, including any available pretrained versions.

What you need to do: Implement medical image segmentation using the SAM model and fine-tune SAM using efficient fine-tuning techniques (e.g., LoRA, Adapters) to fit the specific requirements of the medical imaging dataset.

Your evaluation metric: Accuracy (zero-shot and fine-tuning results).

Your datasets: MICCAI FLARE22 Challenge Dataset.

Tips:

You can choose one or more datasets to evaluate your model. It is encouraged to adopt more datasets. Some other datasets are accepted.

In your final report, you should submit executable codes. The results in your report must be reproducible. A detailed table for performance among different datasets must be in the report.

Relevant Open-Source Projects:

- https://github.com/bowang-lab/MedSAM
- https://github.com/MedicineToken/Medical-SAM-Adapter
- https://github.com/wangsssky/SonarSAM
- https://github.com/tianrun-chen/SAM-Adapter-PyTorch

References:

- A Comprehensive Survey on Segment Anything Model for Vision and Beyond, Arxiv 2024.
- Towards Segment Anything Model (SAM) for Medical Image Segmentation: A Survey, CBM, 2024.
- Segment Anything, ICCV (2023) Best Paper Honorable Mention.
- SAM 2: Segment Anything in Images and Videos, Arxiv, 2024.
- SegGPT: Segmenting Everything In Context, ICCV, 2023.
- Segment Everything Everywhere All at Once, NeurIPS, 2023.
- CAD: Memory Efficient Convolutional Adapter for Segment Anything, Arxiv, 2024.
- Tri-Plane Mamba: Efficiently Adapting Segment Anything Model for 3D Medical Images, Arxiv, 2024.
- A Federated Learning-Friendly Approach for Parameter-Efficient Fine-Tuning of SAM in 3D Segmentation, Arxiv, 2024.
- MedSAGa: Few-shot Memory Efficient Medical Image Segmentation using Gradient Low-Rank Projection in SAM.
- Gradient-based Parameter Selection for Efficient Fine-Tuning.
- How to Efficiently Adapt Large Segmentation Model (SAM) to Medical Images, Arxiv, 2023.
- Parameter-Efficient Transfer Learning for NLP, ICML, 2019.
- BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language Models, ACL, 2022.
- LoRA: Low-Rank Adaptation of Large Language Models, ICLR, 2022.