Computer Vision Fall 2025 Final Project

Final Project

The Final Project is a valuable opportunity for you to apply the concepts and techniques learned throughout the course to a computer vision problem of your own interest. We highly encourage you to work in teams—each team may consist of up to four members.

You are free to choose from the suggested project topics listed below. Topics 1–3 focus on classical computer vision tasks, while topics 4–8 explore more advanced and cutting-edge research directions. To encourage exploration of frontier areas, projects based on topics 4–8 will be eligible for 10 bonus points, meaning the maximum project score will be 110 instead of the standard 100.

We recommend selecting a topic that not only challenges your technical skills but also aligns with your interests and future goals.

Project Topics

1. Face recognition

A facial recognition system is a technology capable of matching a human face from a digital image or a video frame against a database of faces. Such a system is typically employed to authenticate users through ID verification services, and works by pinpointing and measuring facial features from a given image.

In this project, your task is to build a simple face recognition system using the following dataset and github repository. You can use only the images in the dataset for testing and training data, but if you are able to input your own(a small amount of) images for face recognition and identification, we will consider giving extra credit.

- Datasets: VGG Face2
- Github: https://github.com/serengil/deepface

2. Single object tracking

The single object tracking system is a technology capable of matching a target that is given by a person in the first frame of a video with the subsequent video sequence. This kind of system is usually used to security monitor, and captures the target by calculating the similarity between the reference frame target and the search frame target.

Task: Collect multiple videos by yourself, use the labeling tool to mark the object that needs to be tracked in the first frame, design tracking algorithm to track and visualize the tracking results.

- Survey:
  - https://blog.csdn.net/qq_37002417/article/details/108141409
  - https://zhuanlan.zhihu.com/p/503735985
- Dataset: http://got-10k.aitestunion.com/
- Github:

- https://github.com/visionml/pytracking
- https://github.com/heartexlabs/labelImg

3. Semantic Segmentation

Semantic segmentation is a typical computer vision problem that involves taking as input some raw data and converting them into a mask with highlighted regions of interest, with applications such as scene understanding, medical image analysis, robotic perception, video surveillance, augmented reality, and image compression, among many others.

Task: Train a semantic segmentation model(not limited to the PASCAL VOC dataset) and collect some interesting scenarios yourself to see how well the segmentation works. Also, we encourage you to try your model for image segmentation on videos and see what difficulties you get.

- Dataset: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/index.html#devkit
- Github: https://github.com/usuyama/pytorch-unet (This is a relatively simple method, and you are free to choose the method you like)

---

4. Segmentation in 3D Gaussian Splatting ⭐

3D Gaussian Splatting (3DGS) has garnered significant attention in the field of 3D reconstruction, novel view synthesis, and related domains due to its efficient and differentiable rendering capabilities. Notably, **Object-Level Segmentation** within 3DGS plays a pivotal role in various downstream applications, including scene editing, scene understanding, and embodied intelligence.

Project Tasks

This project consists of two main components:

1. **Method Reproduction**
   - **Object Segmentation:** Perform object-level segmentation on 3DGS scenes.
   - **Downstream Applications:** Based on the segmented objects, explore and implement several possible tasks, including:
     - 3D Object Removal
     - 3D Object Inpainting
     - 3D Object Style Transfer
     - 3D Multi-Object Editing
     - Or other creative and meaningful applications you come up with
   - **Dataset Requirement:** Conduct experiments on **at least three different scenes** from the provided datasets.
2. **Your Contribution**
   - **Custom Data Validation:** Capture and use your own real-world data to test the same pipeline.

- **Method Enhancements:** Make any improvements or modifications to the segmentation process or the downstream applications. This could be as simple as parameter tuning for a specific scenario—novel ideas are welcome but not required.

Note: The emphasis of this project lies in the **process** of trying and validating, rather than achieving optimal results.
Recourse:

- Dataset:
  - LERF-MASK: https://github.com/lkeab/gaussian-grouping/blob/main/docs/dataset.md
  - Mip-NeRF360: https://jonbarron.info/mipnerf360/
- Github: https://github.com/lkeab/gaussian-grouping (Free to choose other methods.)
- 3DGS Viewer: https://superspl.at/editor

5. Image Captioning ⭐

Image captioning is the process of generating a natural language description for an image. It involves understanding the contents of an image and translating that understanding into words. This task combines techniques from computer vision and natural language
processing.

Project Tasks

In this project, you will build a model capable of generating relevant and accurate captions for images. You must use COCO datasets, which contain images and their respective captions. The goal is to train a vision-language model to generate captions that accurately describe the contents of the images.

This project consist of two main components:

1. **Build and train the VLM**
   - Start from a pretrained vision encoder(ResNet,ViT etc) and a pretrained large language model(LLaMA, Qwen2/Qwen2.5, etc), and use a connector (simply MLP as defalut) to form a vision-language model.
   - Prepare your dataset pipeline and train the vision-language model on the COCO caption dataset. It is recommended to use the `transformers` package of huggingface to build the project. You can refer to implementation style of `llava` for more details.
   - Evaluate your trained model on the testset of COCO, report the `BLEU` score and `Cider` score. You can use the `pycocoevalcap` package to compute the two metrics.
2. **Ablation study**
   - Explore the effects of different visual encoder architectures(for example CNNs, transformer-based, mamba-based) on final performance, and report your findings.
   - Explore the effects of different connector on the final performance. You can simply replace the MLP as other modules like Q-former or other modules you want. If you designed a new modules, please explain your design principles.

    - (Optional for bonus) Explore the effects of transformer decoder-based language models and mamba-based language models on the final performance. Besides, compare their time consumption under different input sequence lengths, draw the time-sequence length curve and explore the possible reasons.

Note: The emphasis of this project lies in the process of trying and validating, rather than achieving optimal results.

Hint: If your computing resources are limited, you can start with a smaller vision model and language model. For example GPT-2, Mamba-130M, Qwen3-0.6B and so on.

Resource:

- For COCO caption annotations, you can find it on HuggingFace or you can simply download the preprocessed COCO2014 caption annotation file here. Note that you still need to download the image file.
- Implementation of LLaVA style: https://github.com/haotian-liu/LLaVA
- transformers tutorial: https://huggingface.co/docs/transformers/index

6. Human-to-Humanoid Motion Retargeting ⭐

Human-to-Humanoid motion retargeting aims to transfer 3D human motion sequences (captured from datasets like AMASS) to humanoid robots (e.g., Unitree H1) with different kinematic structures, enabling robots to mimic natural human movements while maintaining physical stability. This task requires solving challenges such as joint freedom mismatch, dynamic balance, and real-time control.

Project Task

Design an optimization algorithm to map SMPL-based human motion parameters to Humanoid joint control signals. Validate generated motions in Isaac Gym platform using Humanoid robotics.

This project needs to be completed according to the following steps:

1. **Data acquisition**: Download the AMASS dataset to understand the relevant knowledge of SMPL data and rigid body motion transformation
2. **Algorithm design**: Design a mapping algorithm from human motion data to humanoid robot data. You can refer to **MimickingBench** this article to learn about retargeting methods
3. **Visualization display**: Use Unitree H1 or other humanoid robots on the Isaac Gym platform to display data after reorientation
4. **Experimental evaluation**: Design corresponding indicators to evaluate the accuracy and other metrics of the algorithm, etc

- Dataset:
    - AMASS: https://amass.is.tue.mpg.de Large-scale human motion capture dataset with SMPL parameters.
- Github:
    - UH-1: https://github.com/sihengz02/UH-1
    - InterMimic： https://github.com/Sirui-Xu/InterMimic

- Isaac Gym：https://developer.nvidia.com/isaac-gym

7. R1 Reasoning ⭐

DeepSeek-R1 has demonstrated impressive reasoning abilities in solving complex problems, sparking significant research interest in the field of LLM-based reasoning. In this project, you will explore **R1-style reasoning** within the Countdown task — where, given a target number and a set of N numbers, the goal is to generate a valid equation that reaches the target using the provided numbers and operations.
Note: Access to at least one GPU with **≥40GB memory** is recommended.

Project Tasks

This project consists of two main components:
1. Method Reproduction

- **Countdown Reasoning:** Reproduce the training pipeline for solving the Countdown task. Your implementation should enable the model to reason step-by-step to generate valid equations that solve the target.
- **Evaluation & Comparison:** Compare the reproduced model's performance with baseline variants (e.g., pre-trained models without reasoning-specific training).

2. Your Contribution

- **Result Analysis:** Analyze how the post-training affects model performance on the Countdown task. This includes both:
  - **Quantitative analysis** (e.g., accuracy, reward metrics)
  - **Qualitative analysis** (e.g., generated outputs, error patterns)
    Since the primary objective is to explore reasoning ability rather than maximize accuracy, your insights into learning trends and failure modes are especially valuable. If performance does not improve, discuss plausible reasons such as data scarcity or unstable training. You are encouraged to visualize trends from the training logs (e.g., reward progression over time).
- **Custom Task Extension:**
  - **Custom Task:** Optionally explore the model's reasoning capability on tasks beyond Countdown. You may use a simple symbolic reasoning task, arithmetic chain-of-thought, or another structured logic task.
  - **Multimodal Reasoning:** Optionally, if you have additional time and computional resources, consider extending reasoning to multimodal tasks. For example, Geometry3K involves interpreting diagrams alongside problem statements.

Recourse

- **Base Code & Dataset:** TinyZero: https://github.com/Jiayi-Pan/TinyZero
- **Training Method Reference:** GRPO: https://arxiv.org/abs/2402.03300
- **Multimodal Training Code**: https://github.com/hiyouga/EasyR1

8. Multi-view stereo SLAM ⭐

Multi-view stereo (MVS) extends the principles of passive stereo to multiple viewpoints and aims to reconstruct a dense 3D model of a scene from a sequence of images with known camera parameters. Multi-view stereo SLAM extends this progress, to Incrementally estimate camera parameters while reconstructing.

Task: In this project, you need to run a MVS-SLAM pipeline. accepting monocular video as input, and output 3D scene point cloud. Test your result in public dataset, and build a live demo using SUSTech scene (classrooms or laboratories)

Bonus: you can consider use NVS methods (like 3d gaussian splatting) to build a better visual effect of your result.

- Datasets:
  - 7-Scenes: https://www.microsoft.com/en-us/research/project/rgb-d-dataset-7-scenes/ (copy the download link and paste to search bar to download it if you meet problem).
- Github: https://github.com/rmurai0610/MASt3R-SLAM

Note: We do not limit the datasets and github repositories you use, you can use the resources you find to complete the task.

Forming Groups

- Groups can have up to 4 members.
- Reports should be individually submitted and it should highlight the contributions of each team member on a section of the paper.
- Please complete the topic selection and team member information on Tencent Docs: https://docs.qq.com/sheet/DTm5QcEZoWndxVnB3?tab=BB08J2 VFZRXFV?tab=BB08J2. The deadline for grouping is **2025.5.24**.

Report

We've provided a template to help guide your final project report, but feel free to follow your own structure as long as it is clear and well-organized. Latex is also recommended for your report.

Regarding the reports:

- Each group should submit one report.
  - The report should include the names of all the collaborators.
  - You can use word, markdown, or Latex to form your report.
- PDF file should be submitted.

You should describe and evaluate what you did in your project, which may not necessarily be what you hoped to do originally. A small result described and evaluated well will earn more credit than an ambitious result where no aspect was done well. Be accurate in describing the problem you tried to solve. Explain in detail your approach, and specify any simplifications or assumptions you have taken. Also demonstrate the limitations of your approach. When doesn't it work? Why? What steps would you have taken have you continued working on it? Make sure to add references to all related work you reviewed or used.

You are allowed to submit any supplementary material that you think it important to evaluate your work, however we do not guarantee that we will review all of that material, and you should not assume that. The report should be self-contained.

Submission: submit your report to blackboard as a pdf file named groupid_final.pdf. Submit any supplementary material(e.g. videos) as a single zip file named groupid_sup.zip. Add a README file describing the supplemental content. Submit your code as single zip file named groupid_code.zip.

Grading Policy

- Report (60)
  - Introduction (10)
  - Related work (5)
  - Approach (10)
  - Experimental results (20)
  - Conclusion (5)
  - References (5)
  - Overall clarity of the report (5)
  - Contribution of each member (e.g.35% for xxx,25% for xxx): we will add or deduct
- Presentation (40)
  - 15 min per group + 3 min Q&A
- Topic choice bonus(10): As long as you choose a topic from 4 to 8 and complete the project in a satisfactory manner, you will receive 10 bonus points. However, if the quality of the work is too low, the bonus points will not be awarded.

FAQ

- **Q**: Does this major assignment require innovation in the algorithm?
  - **A**: We highly encourage students to innovate in their algorithms. However, due to time constraints, it is not mandatory. You can treat it more as an application-oriented project and focus on demonstrating your engineering efforts and teamwork. Of course, if any student does come up with innovative algorithms that perform well on standard datasets, we will award extra points for that.
- **Q**: How will TA determine the contribution level?
  - **A**: For groups that have open-sourced on GitHub, we can roughly judge the contribution level based on commits. However, we mainly rely on the contributions section in your reports.

Due

- The report is due in **2025.6.21** at the end of day.
- The presentation is in **2025.6.21 2:00pm-4:00pm**.