Homework #4

Deep Learning for Computer Vision

NTU, Fall 2023

112/11/28

112/12/12 (Tue.) 11:59 PM (GMT+8) due

Update

1. 12/2: In provided dataset.py line 230 & line 277, change f'rgba_{image_id:05d}.png' to f'{image_id:05d}.png'. This is filenames of those images in dataset/

2. 12/2: hw4.sh $1 $2. $1 is the **folder** that contains metadata.json

3. 12/3: please use the latest PyTorch Lightning version, specifically version 2.1.2, as you install from our requirement.txt. Older versions contain tools that are no longer available. If unnecessary, feel free to comment out those lines.
4. 12/4: if using nerf_pl repository, instead of torchsearchsorted package as suggested from their repo, use torch.searchsorted() function. [Link]

Outline

- Problems & Grading
- Dataset
- Submission & Rules
- Supplementary

Problems – Overview

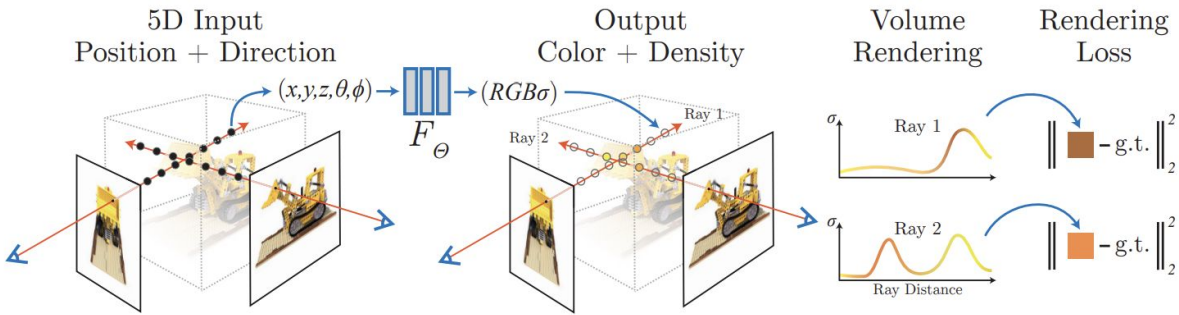
- **Problem :** 3D Novel View Synthesis [dataset/*.png]

Please refer to “Dataset” section for more details about datasets.

Problem: 3D Novel View Synthesis

In this problem, you will have to train your own **NeRF** model on **a multi-object scene**.
To be more specific, given a set of training images (with camera pose) of this scene, make your model fit this scene. After that, given a test set camera pose, the model should be able to synthesize these unseen novel views of this scene.

- Grading:
  - Report (60%)
  - Model Performance (40%)



Reference: [NeRF: Representing Scene as Neural Radiance Fields for View Synthesis]
[Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction]

Problem: 3D Novel View Synthesis

1. (15%) Please explain:

   a. the NeRF idea in your own words

   b. which part of NeRF do you think is the most important

   c. compare NeRF's pros/cons w.r.t. other novel view synthesis work

Please read through ALL reference paper to realize their ideas.

Reference: NeRF: Representing Scene as Neural Radiance Fields for View Synthesis
Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction (DVGO)
Instant Neural Graphics Primitives with a Multiresolution Hash Encoding (InstantNGP)

Problem: 3D Novel View Synthesis

2. (15%) Describe the implementation details of ***your NeRF model*** for the given dataset. You need to explain your ideas completely.
   - **You have to train a novel view synthesis model on the given dataset without loading pretrained weights.**
   - **We provide the following Github links for your reference.**
     - **NeRF** - **LINK1**/**LINK2**(recommended)

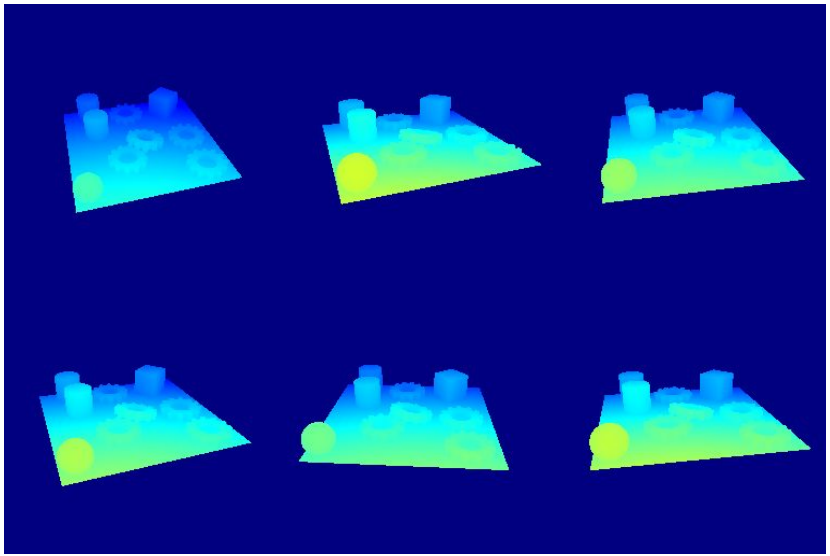
Problem: 3D Novel View Synthesis

3. (15%) Given novel view camera pose from **metadata.json**, your model should render novel view images. Please evaluate your generated images and ground truth images with the following three metrics (mentioned in the NeRF paper). Try to use at least **three** different hyperparameter settings and discuss/analyze the results.
   - **Please report the PSNR/SSIM/LPIPS on the validation set.**
   - **You also need to explain the meaning of these metrics.**
   - **Different configuration settings such as MLP and embedding size, etc.**

Setting	PSNR	SSIM	LPIPS (vgg)
Setting 1 (You need to write your setting)	*TODO*	*TODO*	*TODO*
Setting 2 (You need to write your setting) …	*TODO*	*TODO*	*TODO*

Problem: 3D Novel View Synthesis

4. (15%) With your trained NeRF, please implement depth rendering in your own way and visualize your results.
   - (Reference usage: **cv2.applyColorMap(depth, COLORMAP_JET)**) [link]



Problem: 3D Novel View Synthesis

Model Performance (40%)

- PSNR and SSIM scores should **both** be above the baseline scores to get points
  - Public baseline (on the validation set):
    - Baseline (20%):
      - PSNR: **35**
      - SSIM: **0.85**
  - Private baseline (on the test set):
    - Baseline (20%):
      - PSNR: **TBD**
      - SSIM: **TBD**

Outline

- Problems & Grading
- Dataset
- Submission & Rules
- Supplementary

Problem Dataset: multi-object scene

- The dataset consists of **101** RGB images of size 256 X 256 of a scene.

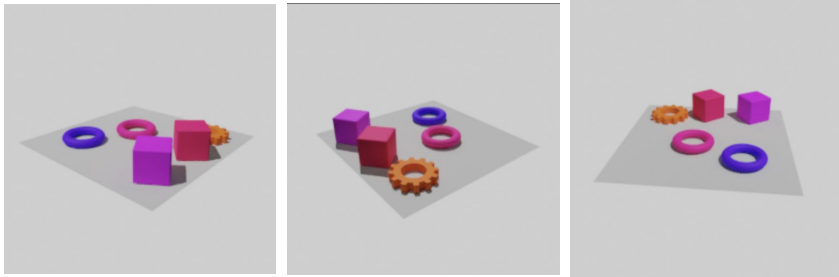
- In the dataset, you will get

dataset/

xxxxx.png	# training images (81 images)
…	…
xxxxx.png	# validation images (20 images)
metadata.json	# including camera pose, train/val/test split. Explain in next page.

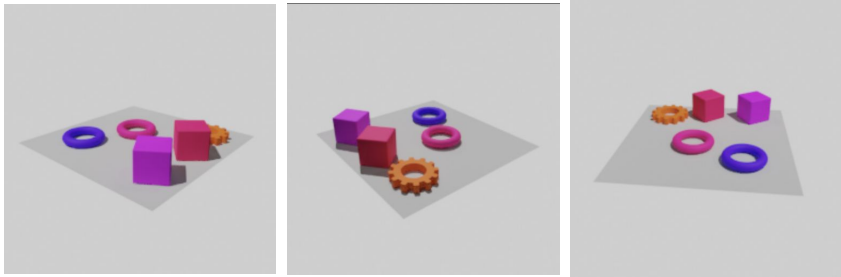
- You **CANNOT** use validation data for training purposes.

- Dataset code is provided in hw4 github



Problem Dataset: metadata.json

- metadata.json:
  - “camera”: {focal length, sensor width ,field of view, **301** camera positions and quaternions}
    - Camera Pose = Camera Position + Quaternion -> Camera Extrinsic Matrix
  - “scene boundaries”: the scene is bounded in the given boxes.
  - “split_ids”: 81 train_ids, 20 val_ids, 200 test_ids.
    - You have the corresponding images for 101 images (train_ids + val_ids), but no test set images.
    - You do have the camera poses of all 301 images.
    - Following the Dataset code and make your model be able to predict images from the test set camera poses.



Tools for Dataset

- Download the dataset https://drive.google.com/file/d/1hF4z9U-xaoV4qaq9DbhTP-KKJTlwOUv_/view?usp=sharing

Outline

- Problems & Grading
- Dataset
- Submission & Rules
- Supplementary

Submission

- Click the following link and sign in to your GitHub account to get your submission repository
    - https://classroom.github.com/a/6iIKHp0j
    - You should connect your Github account to the classroom with your **student ID**
    - If you cannot find your student ID in the list, please contact us (ntudlcv@gmail.com)
- By default, we will only grade your last submission before the deadline (**NOT** your last submission). Please e-mail the TAs if you'd like to submit another version of your repository, and let us know which commit to grade.
- We will clone the **main** branch of your repository.

Submission

- Your GitHub repository should include the following files
  - hw4_<studentID>.pdf (report)
  - hw4.sh (for Problem 1)
  - Python files (e.g., training code & inference code & visualization code)
  - Model files (can be loaded by your python file)
- **Don’t push the dataset to your repo.**
- If any of the file format is wrong, you will get zero point.

Bash Script - Problem 1

- Please provide a script to the specified json file which contain camera pose, and save the generated images into the specified directory.
- TA will run your code as shown below target
  - bash hw4.sh $1 $2
    - $1: path to the **folder** of **metadata.json** (e.g., */*/metadata.json)

      contains camera poses with the same format as in metadata.json, you should predict novel views base on the test split of this file.
    - $2: path of the **folder** to put output **images** (e.g., xxxxx.png, please follows metadata.josn[“split_ids”][“test”] to name your output images.)

      the filename should be xxxxx.png, e.g., 00000.png, 00013.png. The image size should be the same as training set, 256x256 pixel.
- Please follow the naming rules strickly
- Note that you should **NOT** hard code any path in your file or script.
- Your testing code have to be finished in **20 mins.**
- We provide the grade.py for grading, see the supplementary page

Bash Script (cont'd)

- You must **not** use commands such as **rm**, **sudo**, **CUDA_VISIBLE_DEVICES**, **cp**, **mv**, **mkdir**, **cd**, **pip** or other commands to change the Linux environment.
- In your submitted script, please use the command **python3** to execute your testing python files.
  - For example: python3 test.py $1 $2
- We will execute you code on **Linux** system, so try to make sure you code can be executed on Linux system before submitting your homework.

Rules – Submission

- If your model checkpoints are larger than GitHub's maximum capacity (50 MB), you could download and preprocess (e.g. unzip, tar zxf, etc.) them in hw4_download.sh.
  - TAs will run `bash hw4_download.sh` prior to any inference if the download script exists, i.e. it is **NOT** necessary to create a blank `hw4_download.sh` file.
- Do **NOT** delete your model checkpoints before the TAs release your score and before you have ensured that your score is correct.

Rules – Submission

- Please use **wget** to download the model checkpoints from cloud drive (e.g. Dropbox) or your working station.
    - You should use **-O argument** to specify the filename of the downloaded checkpoint.
- Please refer to this Dropbox Guide for a detailed tutorial.
- Google Drive is a widely used cloud drive, so it is allowed to use **gdown** to download your checkpoints from your drive.
    - It is also recommended to use -O argument to specify the filename.
    - Remember to set the permission visible to public, otherwise TAs are unable to grade your submission, resulting in zero point.

Rules – Environment

- Ubuntu 20.04.1 LTS
- NVIDIA GeForce RTX 2080 Ti (11 GB)
- GNU bash, version 5.0.17(1)-release
- Python 3.9

Rules – Environment

- Ensure your code can be executed successfully on **Linux** system before your submission.
- Use only **Python3** and **Bash** script conforming to our environment, do not use other languages (e.g., CUDA) and other shell (e.g., zsh, fish) during inference.
    - Use the command **“python3”** to execute your testing python files.
- You must **NOT** use commands such as **sudo, CUDA_VISIBLE_DEVICES** or other commands to interfere with the environment; **any malicious attempt against the environment will lead to zero point in this assignment**.
- You shall **NOT** hardcode any path in your python files or scripts, while the dataset given would be the absolute path to the directory.

Packages

- pip install -r requirements.txt --no-cache-dir
- Others if you need (e.g., tqdm, gdown, glob, yaml, etc.)
- Any dependencies of above packages, and other standard python packages

- **E-mail or ask TA first if you want to import other packages.**

Packages and Reminders

- If you use other github repository other than two reference link, **LINK1**/**LINK2**, please give us your environment's requirements.txt and describe how we can reproduce your environment in your report.
- Python==3.9
- Do not use **imshow()** or **show()** in your code or your code will crash.
- Use **os.path.join** to deal with path as often as possible.
- If you train on GPU ids other than 0, remember to deal with the "**map location**" issue when you load model. (More details about this issue, please refer to https://github.com/pytorch/pytorch/issues/15541)

Deadline and Academic Honesty

- Deadline: **112/12/12 (Tue.) 11:59 PM (GMT+8)**
- Late policy : Up to 3 free late days in a semester. After that, late homework will be deducted 30% each day.
- Taking any unfair advantages over other class members (or letting anyone do so) is strictly prohibited. Violating university policy would result in F for this course.
- Students are encouraged to discuss the homework assignments, but you must complete the assignment by yourself. TA will compare the similarity of everyone's homework. Any form of cheating or plagiarism will not be tolerated, which will also result in F for students with such misconduct.

Penalty

- If we cannot execute your code, TAs will give you a chance to make minor modifications to your code. After you modify your code,
  - If we can execute your code, you will still receive a 30% penalty in your model performance score.
  - If we still cannot execute your code, no point will be given.

Reminder

- Please start working on this homework as early as possible.
- The training may take hours on a GPU or days on CPUs.
- Please read and follow the HW rules carefully.
- If not sure, please ask your TAs!

How to Find Help

- Google!
- Use TA hours (please check course website for time/location).
- Post your question under HW4 discussion section on NTU COOL.
- Contact TAs by e-mail: ntudlcv@gmail.com.

DOs and DONTs for the TAs (& Instructor)

- Do NOT send private messages to TAs via Facebook.
- TAs are happy to help, but they are not your tutors 24/7.
- TAs will NOT debug for you, including addressing coding, environmental, library dependency problems.
- TAs do NOT answer questions not related to the course.
- If you cannot make the TA hours, please email the TAs to schedule an appointment instead of stopping by the lab directly.

Outline

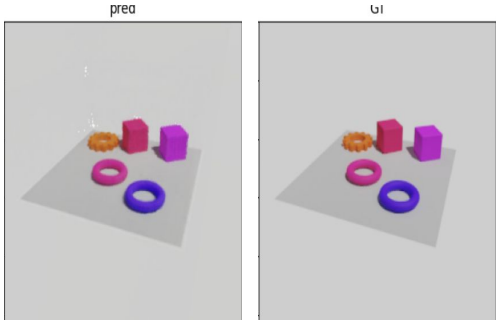
- Problems & Grading
- Dataset
- Submission & Rules
- Supplementary

Supplementary - dataset.py

- This file is on github and it is the one you can load image as pytorch dataset.
  - rays_o: rays origin
  - rays_d: rays direction
  - self.near, self.far: near bound and far bound for a ray
  - c2w: camera to world transformation matrix
  - blender is a coordinate system used in most cases

Supplementary - grade.py

- This file is on github and it is the one we will evaluate your score after running your hw4.sh.
  - python grade.py $1 $2
    - $1 is the path to the folder of generated image
    - $2 is the path to the folder of gt image
  - you can also use this file to evaluate on eval dataset




```
(DVGO) timothy@vll-timothy:~/Desktop/DirectVoxGO$ bash grade.sh
100%|                                                                                | 50/50 [00:53<00:00,  1.06s/it]
Testing psnr 35.379612398147586 (avg)
Testing ssim 0.9752658490162989 (avg)
```