Homework 3

Release: 02/23/2022
Due: Wed. 03/16/2022, 11:59 PM

- You are **allowed** to consult any external resources but you must cite them. You are also **allowed** to discuss with each other but you need to acknowledge them. However, your submission must be your own work; specifically, you must **not** share your code or proof.
- The homework contains two problems. The first problem is on 3D pose estimation, and the second is on novel view synthesis with neural radiance fields (NeRF). You are free to **choose and finish one of them**.
- Your submission has 3 parts. The first part is a single PDF file. The second part is a zip file containing your code. For Option 1, the third part is a submission to the leaderboard. For Option 2, the third part is the a zip file containing synthesized images.
- This homework is worth 35/100 of your final grade.
- This homework itself contains 35 points and 5 bonus points.

Option 1 (Pose Estimation). [35 points] In this problem, you need to implement a full pose estimation pipeline on our synthetic dataset. *The objective is to predict object poses for all the objects of interest in the scene given the image and the depth map (no semantic segmentation provided). You need to recognize objects in the image first.*
Read the following carefully and ask on Piazza immediately if you have questions.

1. Overview instruction. *The basic idea is to first use a segmentation network (e.g., U-Net) to segment all objects in an image, and then use a learning-based method, ICP, or a combination to estimate the pose for each segmented object (done in HW2).* Note that semantic segmentation is enough for our case, since there is at most one instance of object appears in one image. Instance segmentation is not required. Given a predicted segmentation mask per object, you can crop corresponding point cloud from the image, and estimate its pose. The instruction above is a straightforward baseline, while you can use any model you like. There exist other advanced pipelines, like PVN3D.
2. Dataset and leaderboard submission format. The training data is the same as HW2. The test data has the same format, but without ground truth segmentation mask. For the dataset description, please refer to HW2. The (leaderboard) submission format is the same as HW2; however, note that there is a new level, **level 3**. We provide an additional test set "testing_data_final(2022).zip", which contains 200 images per level. The link to the dataset does not change. You can download the data or directly use it in Google Colab by mounting the folder.
3. Quantitative evaluation. [10 points + 5 bonus points]

   Your score is computed as the higher of the absolute and relative scores. The absolute score is pose accuracy (*5 degree 1 cm*) averaged across all 3 levels.

   - 10 points + 5 bonus points: if you achieve higher than 80% pose accuracy.
   - 10 points + 4 bonus points: if you achieve higher than 70% pose accuracy.
   - 10 points + 3 bonus points: if you achieve higher than 65% pose accuracy.

- 10 points + 2 bonus points: if you achieve higher than 60% pose accuracy.
- 10 points + 1 bonus points: if you achieve higher than 55% pose accuracy.
- 10 points: if you achieve higher than 50% pose accuracy.
- 9 points: if you achieve higher than 40% pose accuracy.
- 8 points: if you achieve higher than 30% pose accuracy.
- 7 points: if you achieve higher than 20% pose accuracy.
- 6 points: if you achieve higher than 10% pose accuracy.
- 5 points: make a submission

4. Report. [25 points]

   The report should contain explanation of your method and experiments. 2-3 pages with several figures should be enough. The basic guidance is provided, but you can add more sections.

   - Method description [10 points]
     - Method overview
     - Network architecture [5 points]: backbone, segmentation/detection network, pose estimation network. It is suggested to use figures to illustrate network architectures.
     - Losses [5 points]: representation of outputs and how the loss is designed to train pose estimation.
   - Experiments [15 points]
     - Data [3 points]: how many data you use to train networks, and what kind of pre-processing and post-processing you apply to your data.
     - Training details [2 points]: learning rate, batch size, optimization iterations and other hyperparameters.
     - Ablation studies [5 points]: effect on performance of different backbones, effect on performance of different batch sizes or other hyperparamters, runtime analysis w.r.t different design choices. You can use performance on the validation set to report. You can also investigate failure cases.
     - Visualization [5 points]: showcase examples of your segmentation and pose estimation results. You can use utilities provided in HW2. The visualization can be done along with ablation studies.
     - Performance: please mention your final performance on the benchmark.

5. Submission. The submission has 3 parts

   - A short report describing your method and result. (A short version of “our method” and “experiments” section of academic publications.
   - A .zip archive containing your code.
   - You need to submit your results on testing data to our internal benchmark.
     https://storage1.ucsd.edu/cse291ibenchmark/benchmark3
     A benchmark submission will take about 20s, please be patient and please avoid trying to submit multiple times.

6. You should always assume each neural network experiment will take 1 whole day on a GPU. So you should start solving the problems **as early as possible**. If you are using the free version of Colab, make sure you save and load the network weights often as there is a running time limit.

7. Hints:

   - Data pre-processing might save you time.
   - Debug with simple cases (fitting a small subset of data).
   - If you use a direct approach to estimate poses, e.g., applying a PointNet on segmented point clouds to regress poses, you might think over how to prepare training data for pose estimation. For example, it might be better to use point clouds segmented by your segmentation network instead of GT masks. Otherwise, there will be a domain gap during inference.

Option 2 (Novel View Synthesis). [35 points] The neural radiance field (NeRF) is a very recent method for scene representation. Unlike most explicit scene representations relying on meshes or voxels, neural radiance fields encode all information explicitly with a multi-layer perceptron (MLP), which yields to smoother scene representation. The NeRF is most commonly used to generate photo-realistic novel views of a scene. Please refer to the original NeRF paper (link) for more details. In this problem, you will fit a NeRF model with given posed images. And then you are required to synthesize novel views of the scene with your model. The grades of this problem includes 3 parts: (1) Evaluation [10 points + 5 bonus points], (2) Questions [13 points], and (3) Report [12 points].
Read the following carefully and ask on Piazza immediately if you have questions.

1. The task is to fit a scene using a NeRF model with the bottles dataset (link). You are then required to synthesize novel views of the scene.

2. Data description. In training data, there are two folders: `rgb` contains training images of the scene, and `pose` contains corresponding 4×4 camera extrinsic matrices. `intrinsics.txt` is the 3×3 camera intrinsic matrix. You are free to use all 200 images to fit the model (`0_train` and `1_val` prefix). `pose` folder also contains camera extrinsic matrices for test novel views (`2_test` prefix).

3. Output format. Your model should generate images sized 800×800.

4. Evaluation [10 points + 5 bonus points]. Your assignment is graded based on the quality of synthesized images. Specifically, you are required to generate image id `2_test_0000`, `2_test_0016`, `2_test_0055`, `2_test_0093`, and `2_test_0160` with your model. Your score is computed based on the average PSNR of these generated images.

   - 10 points + 5 bonus points: if you achieve higher than 35.0 average PSNR.
   - 10 points + 4 bonus points: if you achieve higher than 34.0 average PSNR.
   - 10 points + 3 bonus points: if you achieve higher than 33.0 average PSNR.
   - 10 points + 2 bonus points: if you achieve higher than 32.0 average PSNR.
   - 10 points + 1 bonus point: if you achieve higher than 31.0 average PSNR.
   - 10 points: if you achieve higher than 30.0 average PSNR.
   - 9 points: if you achieve higher than 28.0 average PSNR.
   - 8 points: if you achieve higher than 26.0 average PSNR.
   - 7 points: if you achieve higher than 24.0 average PSNR.
   - 6 points: if you achieve higher than 20.0 average PSNR.
   - 5 points: if you achieve higher than 16.0 average PSNR.

5. Questions [13 points]. Answer these questions in the beginning of your report:

(a) What is a radiance field? What information is included in a radiance field? How is *neural* radiance field (NeRF) different? [1 points]

(b) What is ray marching? Given a radiance field, how is each pixel calculated? (This is called the render equation.) Write down your render equation in a concrete math expression with clarified notations. [2 points]

(c) What is positional encoding? What is the purpose of positional encoding in NeRF models? Train your model without positional encoding and compare the results. You need to show at least two pairs of examples. [4 points]

(d) Is it possible to extract scene geometry (e.g. depth) information from a trained NeRF model? Describe your method in detail. Implement your method and show two depth maps generated by your method. [4 points]

(e) What are the major issues you find when using NeRF? List at least 2 drawbacks. For each of them, propose a possible improvement. You are encouraged to check follow-up papers of NeRF, but you should cite these works if borrowing their ideas. [2 points]

6. Report [12 points]. The report should contain explanation of your model and experiments. 1 or 2 pages should be enough. Things to include: models details, network architecture, training and test settings, validation and test scores, visualizations, and references. You are required to show the quality improvement of validation images during training process (e.g. visualize a evaluation image at 10k, 20k, 30k, ... training steps). You are encouraged but not required to find more scenes and visualize your results on these scenes. [12 points]

7. Submission. The submission has 3 parts

   - A PDF containing the answers and a short report.
   - A .zip archive containing your code.
   - A .zip archive containing synthesized images with id `2_test_0000`, `2_test_0016`, `2_test_0055`, `2_test_0093`, and `2_test_0160`.

8. You should always assume each neural network experiment will take 1 whole day on a GPU. So you should start solving the problems **as early as possible**. If you are using the free version of Colab, make sure you save and load the network weights often as there is a running time limit.

9. *Hints:*

   (a) You can first train and evaluate your model on down-sampled 200×200 images to validate the entire pipeline before perform experiments on full size images.

   (b) You should split the 200 images into a training set and a validation set to track your progress while prevent over-fitting.

   (c) A full model of the original NeRF could already give you full evaluation points with some bonus points.

   (d) You can follow these step-by-step guides to implement your own NeRF model and get a better understanding of NeRF: NeRF Tutotial in Keras and Fit a simple Neural Radiance Field via raymarching. You are allowed to refer to public implementations but now allowed to copy their code directly.