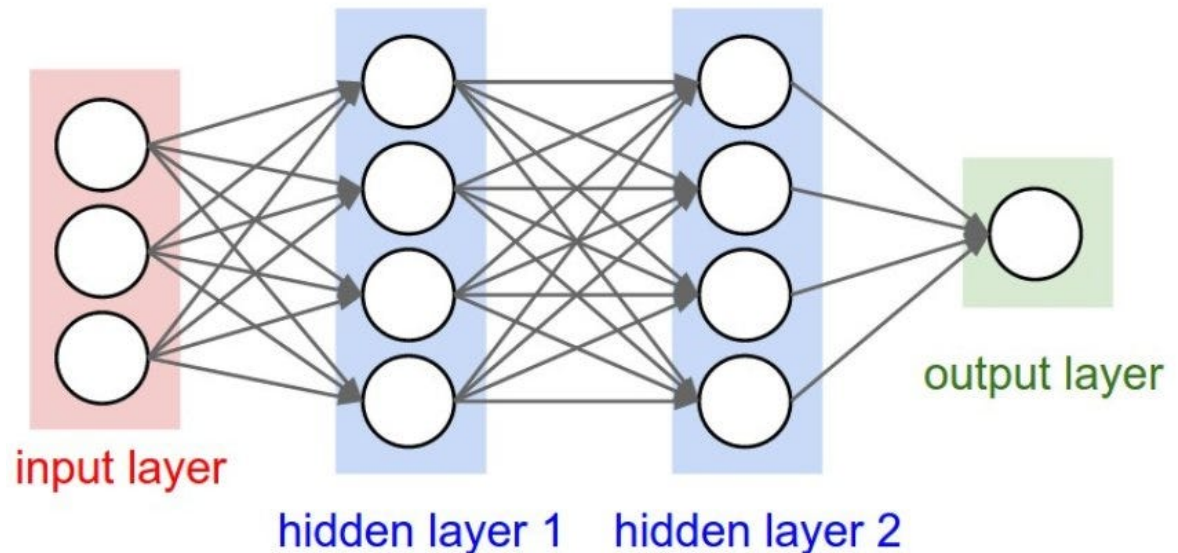


# 1. 原理介紹

- Deep learning

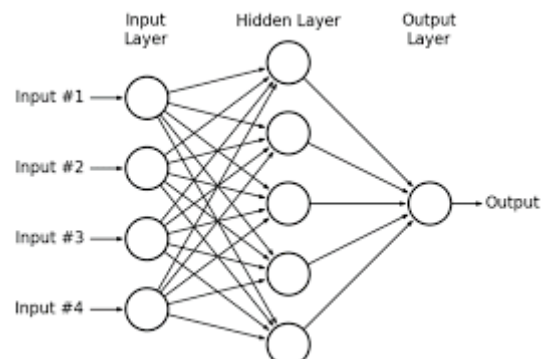
是一種基於人工神經網路的機器學習技術，可以用來從大量的資料中學習並進行模式識別、預測和決策等任務。深度學習模型由多個神經元和層組成，通過多次迭代優化模型的權重和參數，從而實現高度自動化的特徵提取和複雜的非線性建模。深度學習已經在圖像識別、語音識別、自然語言處理、機器翻譯、推薦系統等領域取得了卓越的成果，成為當今人工智能領域的重要分支之一。



圖一、深度學習架構圖

- Multilayer perceptron

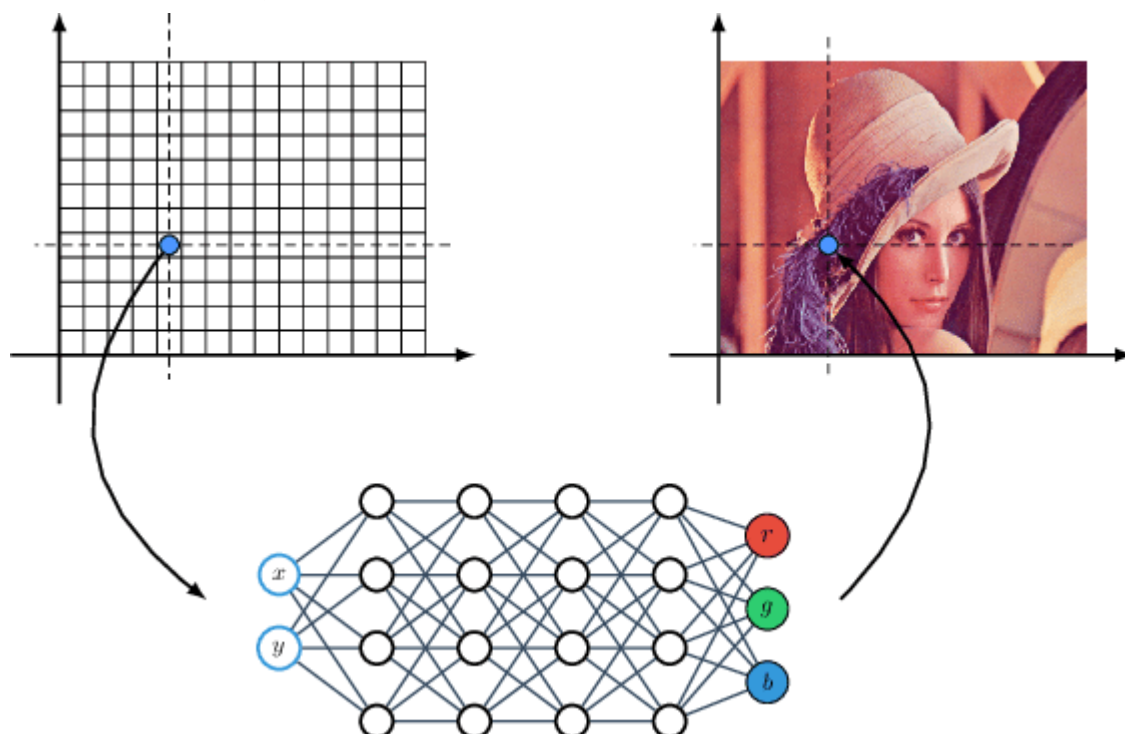
簡稱 MLP，是一種基於人工神經網路的監督式學習模型，由多個全連接層組成，可以用於解決分類和回歸等問題。MLP 的每一層由多個神經元組成，前一層的輸出作為下一層的輸入，通過反向傳播算法學習權重和偏差參數，從而實現非線性映射和特徵提取。相比於簡單的單層感知機模型，MLP 可以更好地擬合複雜的非線性關係，並且能夠處理高維度的特徵空間。MLP 是深度學習的基礎，常用於圖像識別、自然語言處理、推薦系統等領域。



圖二、多程感應器

- Implicit Neural Representation

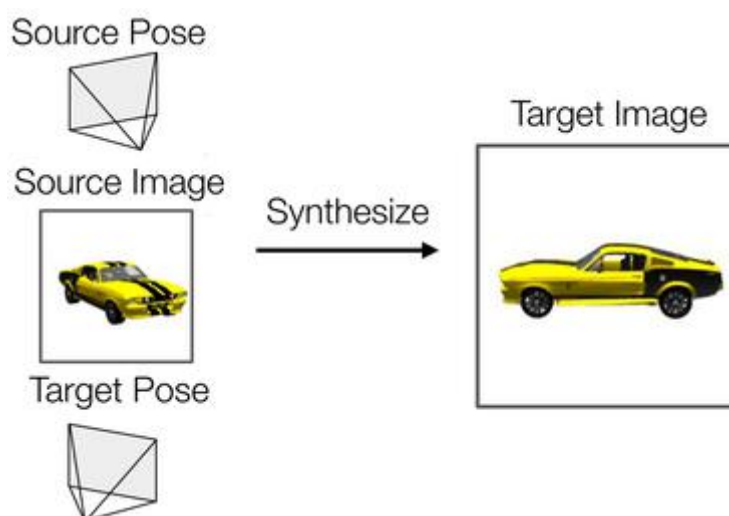
是一種基於神經網路的表示方法，用於捕捉幾何形狀和物體外觀等複雜的非線性變換。與傳統的顯式表示不同，隱式神經表達將物體的表面或體積表示為神經網路的權重和參數，而不是通過解析的公式或數學函數進行表示。隱式神經表達具有高度自由度和柔性，能夠應對任意形狀和複雜度的物體，並且不需要經過手工設計的特徵提取。近年來，隱式神經表達在計算機圖形學、機器人學和虛擬現實等領域得到廣泛應用。



圖三、隱式神經表達

- View synthesis

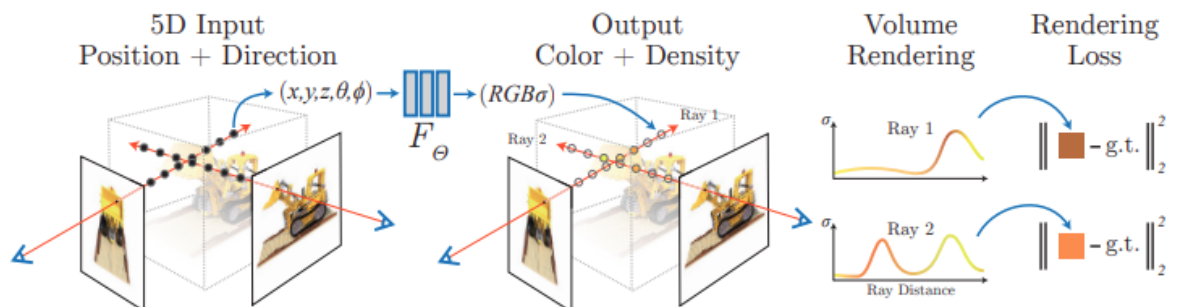
是一種基於深度學習的計算機圖形學技術，用於從現有的圖像或視訊序列中合成出新的視角或場景。該技術通過學習不同視角之間的深度和位移信息，將原始圖像或視訊序列進行重新投影和合成，以生成新的視角或場景。View synthesis 可以應用於虛擬現實、遊戲開發、視訊編碼和多視頻流傳輸等領域，為用戶提供更真實的視覺體驗和更高效的視訊傳輸方式。



圖四、視角合成

- NeRF (Neural Radiance Fields)

是一種基於深度學習的 3D 建模技術，用於捕捉物體的形狀和紋理等細節，並以三維空間中的 radiance field 的形式進行表示。它可以使用從單個或多個圖像中獲取的 2D 視角信息，經過深度神經網路的訓練來估計 3D 空間中的 radiance field，從而實現對物體的精確建模。與傳統的 3D 建模方法不同，NeRF 可以以高質量、高保真的方式呈現真實世界中的物體，並且不需要大量的人工標註或先驗知識。NeRF 技術已經應用於虛擬現實、增強現實、計算機圖形學和機器人學等領域。



圖五、NERF示意圖

## 2. 實做方法

- Colmap:

由於OS的環境是Linux 22.04.1-Ubuntu，所以參考助教給的[網址](#)，依照上述的步驟安裝Colmap。

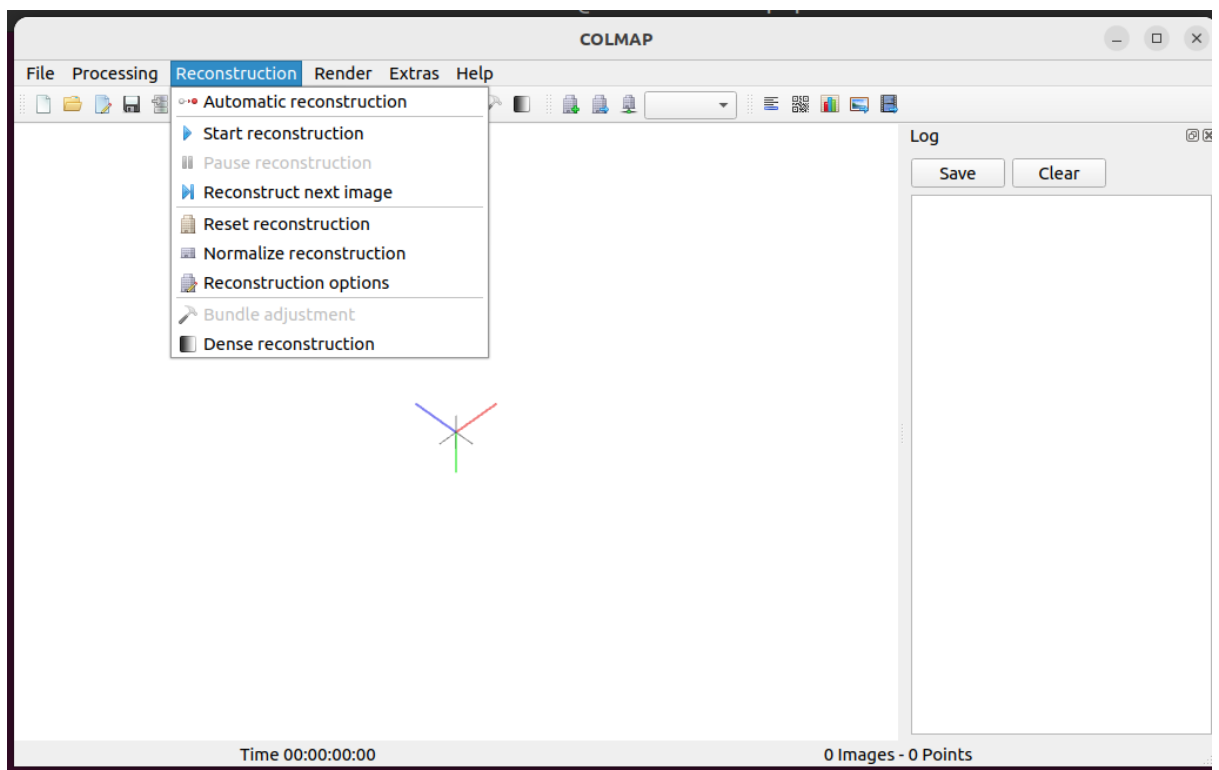
```
sudo apt-get install \
  git \
  cmake \
  ninja-build \
  build-essential \
  libboost-program-options-dev \
  libboost-filesystem-dev \
  libboost-graph-dev \
  libboost-system-dev \
  libboost-test-dev \
  libeigen3-dev \
  libflann-dev \
  libfreeimage-dev \
  libmetis-dev \
  libgoogle-glog-dev \
  libgflags-dev \
  libsqlite3-dev \
  libglew-dev \
  qtbase5-dev \
  libqt5opengl5-dev \
  libcgald-dev \
  libceres-dev
```

```
git clone https://github.com/colmap/colmap.git
cd colmap
git checkout dev
mkdir build
cd build
cmake .. -GNinja
ninja
sudo ninja install
```

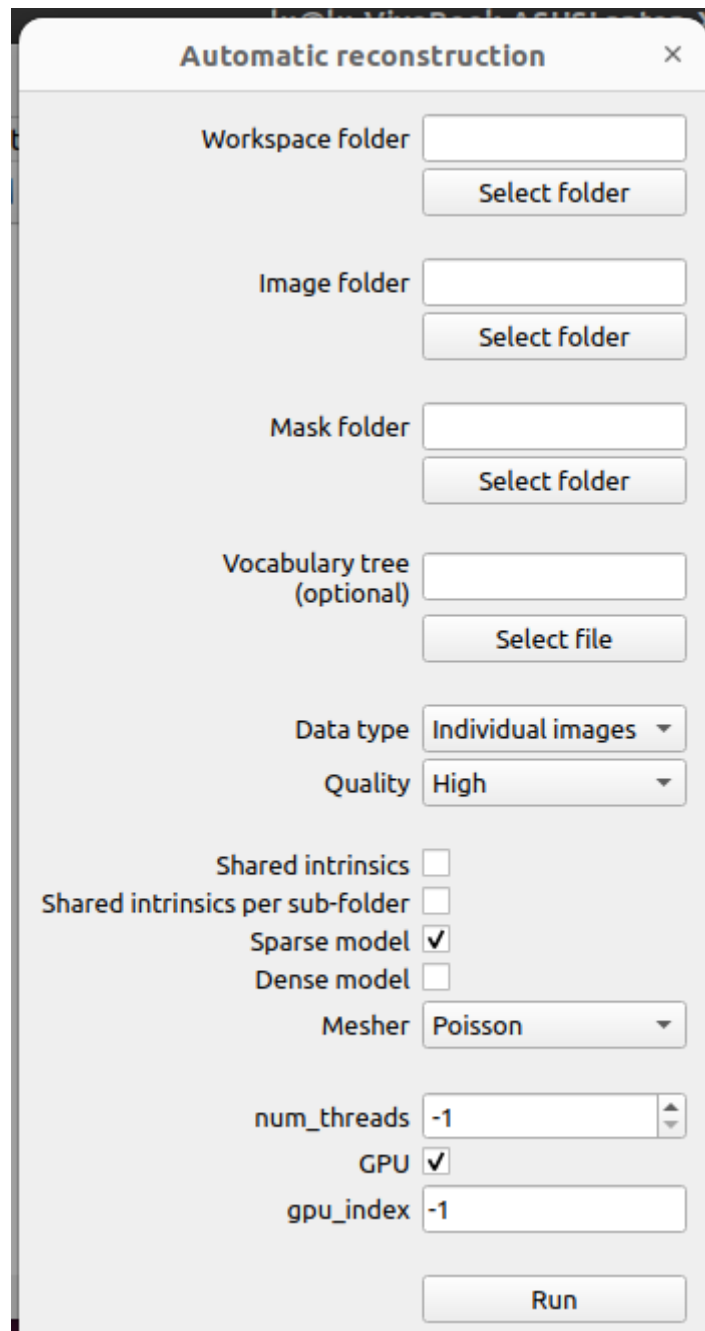
之後再按照指示打開Colmap的圖形化界面

```
colmap -h
colmap gui
```

打開圖形化界面後，選取"Reconstruction"，在點選"Automatic reconstruction"



之後在"Workspace folder"那行填入要輸出的資料夾，"Image folder"填入放照片的資料夾，填寫完之後就可以點選"Run"開始執行



The image shows a software dialog box titled "Automatic reconstruction" with a close button (X) in the top right corner. The dialog contains several configuration options:

- Workspace folder:** A text input field with a "Select folder" button below it.
- Image folder:** A text input field with a "Select folder" button below it.
- Mask folder:** A text input field with a "Select folder" button below it.
- Vocabulary tree (optional):** A text input field with a "Select file" button below it.
- Data type:** A dropdown menu currently set to "Individual images".
- Quality:** A dropdown menu currently set to "High".
- Shared intrinsics:** An unchecked checkbox.
- Shared intrinsics per sub-folder:** An unchecked checkbox.
- Sparse model:** A checked checkbox.
- Dense model:** An unchecked checkbox.
- Mesher:** A dropdown menu currently set to "Poisson".
- num\_threads:** A text input field with the value "-1" and a spin button to its right.
- GPU:** A checked checkbox.
- gpu\_index:** A text input field with the value "-1".
- Run:** A large button at the bottom right of the dialog.

- 深度學習網路

根據助教提供的[網址](#)前往github, 然後要轉換到"DEV"Branch, 因為"Master" Branch的版本太舊, 會跟現在硬體的GPU不相容

The screenshot shows the GitHub repository page for `kwea123/nerf_pl`. The main content area displays a list of files and their commit history:

File	Commit Message	Time Ago
<code>README_Unity.md</code>	Update README_Unity.md	3 years ago
<code>README_mesh.md</code>	Update README_mesh.md	3 years ago
<code>eval.py</code>	upd	3 years ago
<code>extract_color_mesh.py</code>	use PIL to read	2 years ago
<code>extract_mesh.ipynb</code>	add more comments	3 years ago
<code>losses.py</code>	training passed, waiting for result	3 years ago
<code>metrics.py</code>	upd	3 years ago
<code>opt.py</code>	add resume from checkpoint	3 years ago
<code>requirements.txt</code>	Update requirements.txt	3 years ago
<code>test.ipynb</code>	increase test time speed, reduce memory	3 years ago
<code>train.py</code>	upd	3 years ago

Below the file list is the `README.md` content:

### nerf\_pl

Update: NVIDIA open-sourced a lightning-fast version of NeRF: [NGP](#). I re-implemented in pytorch [here](#). This version is ~100x faster than this repo with also better quality!

Update: an improved [NSFF](#) implementation to handle dynamic scene is [open!](#)

Update: [NeRF-W](#) (NeRF in the Wild) implementation is added to [nerfw](#) branch!

Update: The latest code (using the latest libraries) will be updated to [dev](#) branch. The master branch remains to support the colab files. If you don't use colab, it is recommended to switch to dev branch. Only issues of the dev and nerfw branch will be considered currently.

The sidebar on the right contains the following sections:

- Releases**: [NeRF-W on brandenburg gate](#) (Latest) on Jan 27, 2021. + 7 releases.
- Sponsor this project**: [kwea123](#) AI獎. [Sponsor](#). [Learn more about GitHub Sponsors](#).
- Packages**: No packages published.
- Languages**: [Jupyter Notebook](#) 67.7%, [Python](#) 32.3%.

接著按照作者的操作安裝環境

## Software

- Clone this repo by `git clone --recursive https://github.com/kwea123/nerf_pl`
- Python>=3.7 (installation via [anaconda](#) is recommended, use `conda create -n nerf_pl python=3.7` to create a conda environment and activate it by `conda activate nerf_pl`)
- Python libraries
  - Install core requirements by `pip install -r requirements.txt`

之後再按照作者給的方法輸入train.py 跟eval.py就可以完成這次作業的基本操作

## Training model

Run (example)

```
python train.py \
  --dataset_name llff \
  --root_dir $LLFF_DIR \
  --N_importance 64 --img_wh 504 378 \
  --num_epochs 30 --batch_size 1024 \
  --optimizer adam --lr 5e-4 \
  --lr_scheduler steplr --decay_step 10 20 --decay_gamma 0.5 \
  --exp_name exp
```

These parameters are chosen to best mimic the training settings in the original repo. See [opt.py](#) for all configurations.

You can monitor the training process by `tensorboard --logdir logs/` and go to `localhost:6006` in your browser.

## Your own data

### ▼ Steps

1. Install [COLMAP](#) following [installation guide](#)
2. Prepare your images in a folder (around 20 to 30 for forward facing, and 40 to 50 for 360 inward-facing)
3. Run COLMAP **sparse** reconstruction.
4. Train the model using the same command as in [LLFF](#). If the scene is captured in a 360 inward-facing manner, add `--spheric` argument.

```
python eval.py \
  --root_dir $BLENDER \
  --dataset_name blender --scene_name lego \
  --img_wh 400 400 --N_importance 64 --ckpt_path $CKPT_PATH
```

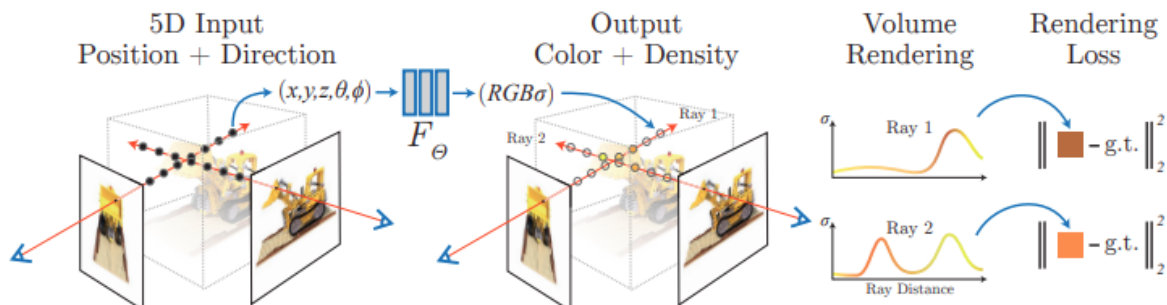
- 以下為深度網路模型解釋以及參數解釋

參數名字	功能
--root_dir	root directory of dataset
--dataset_name	which dataset to train/val
--img_wh	resolution (img_w, img_h) of the image
--spheric_poses	whether images are taken in spheric poses (for llff)
--N_emb_xyz	number of frequencies in xyz positional encoding
--N_emb_dir	number of frequencies in dir

	positional encoding
--N_samples	number of coarse samples
--N_importance	number of additional fine samples
--use_disp	use disparity depth sampling
--perturb	factor to perturb depth sampling points
--noise_std	std dev of noise added to regularize sigma
--batch_size	batch size
--chunk	chunk size to split the input to avoid OOM
--num_epochs	number of training epoch
--num_gpus	number of gpus
--ckpt_path	pretrained checkpoint to load (including optimizers, etc)
--prefixes_to_ignore	the prefixes to ignore in the checkpoint state dict
--optimizer	optimizer type
--lr	learning rate
--momentum	learning rate momentum
--weight_decay	weight decay
--lr_scheduler	scheduler type
scheduler type	lr is multiplied by this factor after --warmup_epochs
--decay_step	scheduler decay step
--decay_gamma	learning rate decay amount
--poly_exp	exponent for polynomial



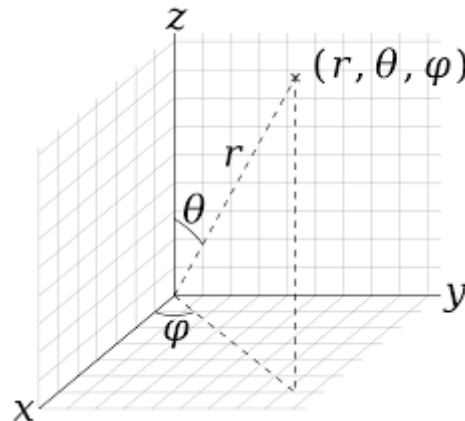
	learning rate decay
--exp_name	experiment name



→5D Input Position+Direction:

$(x,y,z)$  是三維世界的座標(用的坐標系是三維笛卡爾單位向量)

$(\theta, \varphi)$  是指觀察者(或相機)觀察一個三維場景時, 視線的方向。



→ $F_{\theta}$  :

此函數為多程感應器, 用來估計五維的連續場景表示。我們可以藉由優化此函數的參數來找到五維座標對應到的體積密度跟發光顏色。

→Output

Color 是用RGB來表示。

$\sigma$  是體積密度, 表示光線在穿過該位置時所受到的阻礙程度如果體積密度較高, 光線穿過該位置時會被吸收或散射, 影響像素的顏色和亮度值。

→Volume Rendering

體積繪製用於將學習到的神經輻射場(Neural Radiance Fields)轉換為渲染的圖像。具體來說, NERF使用神經網路學習場景的輻射場和密度函數, 將每個像素的顏色和亮度計算為通過每個像素的光線與場景中的神經輻射場相交時的累積輻射強度。然後, 通過卷積這些輻射強度和密度函數, 我們可以計算出場景中每個點的顏色和亮度。

→Rendering loss

用來評估渲染結果跟目標結果的差異，並使用這個差異來優化模型的參數以獲得更好得結果。

$$\mathcal{L} = \sum_{\mathbf{r} \in \mathcal{R}} \left[ \left\| \hat{C}_c(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 + \left\| \hat{C}_f(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 \right]$$

圖六、Rendering Loss函數

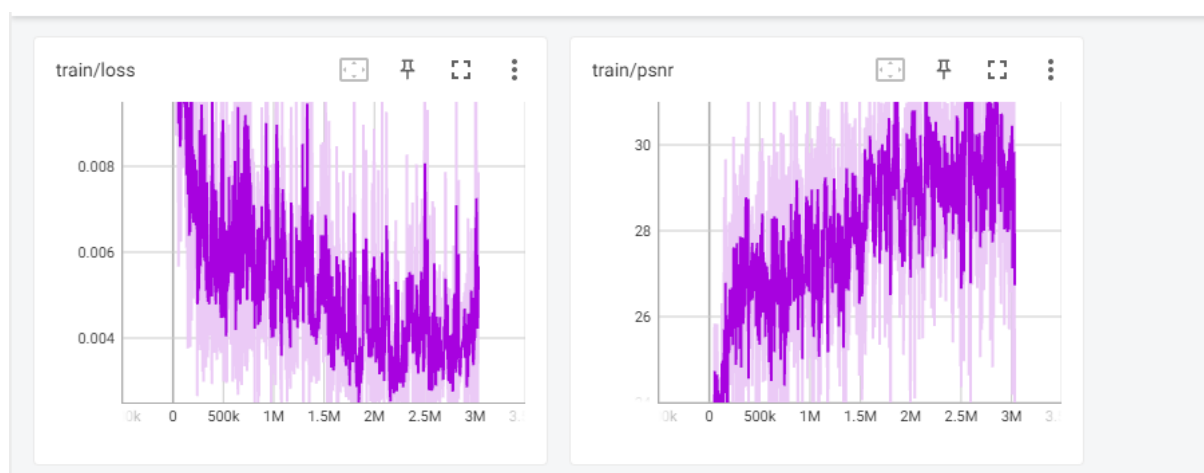
### 3. 深度學習模型訓練及評估結果

-Trainging

	epoch10	epoch15	epoch20
loss	0.009571	0.004383	0.005674
psnr	27.75	30.63	26.72

觀察：

從上方圖表來看，模型好像沒有訓練到，但是去tensorboard看，會發現其實整體的趨勢是對的(train愈來愈小，psnr愈來愈來大)，因此可以推斷訓練還是有成功，只是有些訓練的hyperparameter要改



圖六、訓練loss跟psnr在tensorboard的圖表

-Testing

→ epoch 10

周圍的背景還是有些微的模糊，且環境會出現奇怪的黑影。但是在目標物體上不會出現上述現象，不過物體的畫質相較於原圖還要更加粗糙。

→ epoch 15

周圍的背景相較於epoch10沒有更加清楚，不過最後產生的gif檔是三者裡面最精緻的。

→ epoch 20

周圍的環境相較於epoch10、epoch15的結果沒什麼區別，但是在目標物體上的畫質明顯比epoch10精緻，但略輸epoch15的結果(因為震盪的關係)。

## 4. 對作業整體結果分析與討論

- 作業整體結果分析

從tensorboard的觀察可以看出，整體的loss有在下降，且整體的psnr的也有在上升，但兩者的數值震盪還是很大。代表這個模型的訓練過程可能需要進一步調整。這可能涉及到調整超參數、更改優化算法或調整訓練數據等步驟，以幫助模型更穩定地訓練，從而提高其性能和準確性。

- 作業整體結果討論

選擇NeRF生成結果對作業結果造成很大的影響。在作業的結果中，可以清楚看到模型的優缺點。

優點：

- 高精度：Nerf模型可以生成高品質、高分辨率的三維重建結果，對場景的細節和顏色進行了準確的捕捉，具有很高的幾何和視覺質量。
- 彈性：Nerf可以從不同的視角、角度和光照條件下捕獲場景，並且可以處理不同的物體形狀和複雜性。
- 無需顯式網格：Nerf模型使用連續的函數來表示場景的密度和顏色，因此無需顯式的網格表示，可以減少計算和存儲的成本。

缺點：

- 訓練時間長：由於Nerf模型需要在大量視角和光照條件下對場景進行渲染和重建，因此需要耗費大量的時間和計算資源來訓練模型。
- 資料收集困難：Nerf需要大量的高質量圖像和相應的相機和光照信息，因此資料收集和處理可能比較困難。
- 計算成本高：Nerf需要大量的計算資源來進行訓練和推斷，並且對GPU和記憶體的要求比較高，因此在實際應用中可能會面臨計算成本較高的問題。

## 5. 自由選擇加分項

- 改進方法：

NERF++採用了以下方法來生成更逼真和準確的三維渲染結果。

→多尺度取樣：NERF++ 引入了多尺度取樣技術，能夠更加準確地重建不同尺度的物體，對於場景複雜度較高時能夠獲得更好的渲染效果。

→空間變化 NeRF：NERF++ 引入了空間變化 NeRF 技術，可以更靈活地處理場景中的各種幾何形狀和光照條件，能夠捕捉場景中的空間變化信息並進行自適應取樣和優化。

→隱式微分渲染器：NERF++ 引入了隱式微分渲染器，能夠更高效地計算渲染過程中的梯度和反向傳播，可以加速模型的訓練和推理過程，提高模型的效率和穩定性。

→隱式三維-二維對齊：NERF++ 利用隱式函數的連續性，實現了隱式三維-二維對齊，能夠更加準確地重建三維場景，可以捕捉二維圖像和三維場景之間的映射關係，並進行一致性檢查和優化。

- 參考文獻：

[NeRF++: Analyzing and Improving Neural Radiance Fields](#)