DLCV 2022 – HW4

Name : 周宇玄
Student ID : R10525104

Problem 1.:

1.

a. Explain the NeRF idea in your own words:

I think NeRF's main idea in one sentence is NeRF representing scenes as neural radiance fields for view synthesis, this means NeRF not just consider view on surface, it make NeRF can represent metal and translucent objects, both objects have the property of refracting light differently from different viewing angles so we can't just consider surface(because every surface not continuously in different angle.).

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

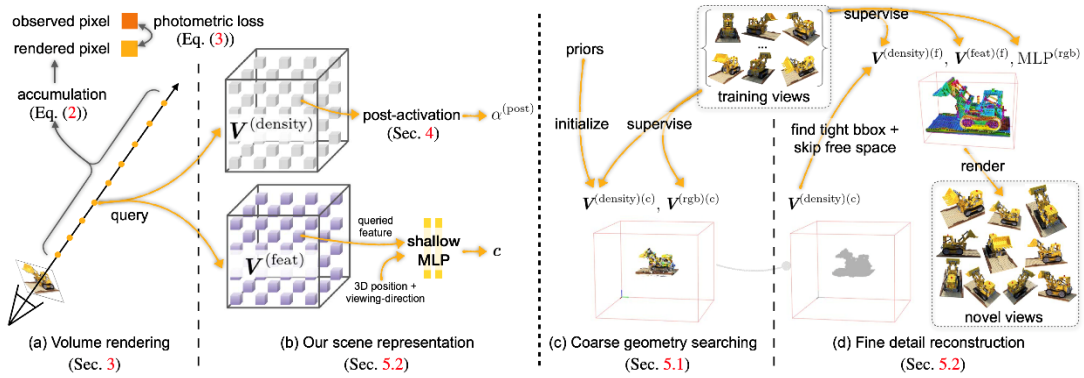
I think the most important part of NeRF is just like I mentioned at question "a", NeRF consider the view as neural radiance field make generate metal and translucent objects become possible, and for implement, NeRF is build by MLP, this advantage make this model can be used in many situations and more easy to training when you have lower level's device.

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

NeRF has good advantage at generate metal and translucent and good at training when you have lower level's device, but it still have limit, for example, if we take different time's image be input, because the lighting will be different so NeRF will "NOT" generate good result, but if we use PixelReNF we can solve this problem, and of course, lighter and faster models will appear over time(for example, DVGO), so I think the most important advantage of NeRF is that it was a completely new idea at the time, and it's a balanced model and it still referenced by many, cons is that every advantage of NeRF they all have better option to replace.

2.

DVGO is a model I just mentioned at 1., it's mentioned for having more speed, why it's more speed is because DVGO consider NeRF by voxel grid, and use two voxel to avoid trapped in local optimal solutions (DVGO just need consider three point to get one voxel), this make DVGO become fast, and by find light bbox + skip free space make it become more fast and efficient(don't consider unnecessary point).



My setting of DVGO is:
Data augmentation : None (Because I think keeping the real image of the different angle is the best option for generating a better result, if I use data augmentation will make generating result become unreal or orderless.)
Learning rate : 0.0001
Num_voxels=160**3
Alpha_init=0.01
Fast_color_thres=0.0001
Step = 0.5

3.

Setting	PSNR	SSIM	LPIPS
Setting 1	35.183	0.974	0.041
Setting 2	34.625	0.970	0.049

The difference of setting 1 and setting 2 is that their num_voxerls setting different number, setting 1 is set by 160 and setting 2 is set by 100. As the result on above we can see that setting 2 has lower PSNR and SSIM, my personal understanding of PSNR and SSIM is that PSNR represents the degree of distortion of a picture, and SSIM represents the similarity between two pictures. So we can say that setting 1 has lower degree of distortion of a picture than setting 2 and more like val-img than setting 2 “in computer”, the setting 2 has higher LPIPS but LPIPS is lower is better, LPIPS is similar with SSIM but LPIPS means setting 1 more like val-img than setting 2 “in human”. So by these three indicators we can know that setting 1 is better than setting 2, which means more num_voxerls will produces better result, because it has more voxel can be used to represent result.

Problem 2.:

1.

My setting of SSL is:
SSL method : BYOL
Data augmentation : RandomHorizontalFlip, RandomAffine, ColorJitter, Normalize.
Learning rate schedule : ExponentialLR
Optimizer : Adam
Batch size : 8
Learning rate : 0.0002
Resize : 128*128
Backbone : ResNet50

2.

Setting	Validation accuracy
A	45.21%
B	47.12%
C	55.17%
D	42.77%
E	42.34%

As result on above, we can see that the C setting has highest accuracy, I think this is we can expect because C setting is pretrained on the dataset "mini", which is similar with dataset "office" 's image so it's easier to fit it, same reason for 2nd best setting B which pretrain on "mini" has higher accuracy then A. By the test-result that worst setting is D and E, we can know that if we fixed the backbone we will get worse performance, and I think it shows that if we only training at a part of model the model will not perfect fit new dataset/task, but fixed the backbone can save the training time and computing resources, so fixed or not fixed depends on the needs of the moments.

Reference :

Hw4_intro

P1 :

https://kakaobrain.github.io/NeRF-Factory/docs/performance/dvgo/
https://sunset1995.github.io/dvgo/
https://github.com/sunset1995/DirectVoxGO
https://arxiv.org/abs/2003.08934
https://arxiv.org/pdf/2111.11215.pdf

P2 :

https://github.com/lucidrains/byol-pytorch
https://arxiv.org/abs/2006.07733