1. Hướng dẫn chạy code Demo trên Google colab
2. Chạy file notebook “Nerfies\_Capture\_Processing\_v2.ipynb” để xử lý dữ liệu từ video ra tập ảnh.

Bước 1: Chạy code trong mục “Install dependencies” để cài đặt các thư viện cần thiết.

Bước 2: Sửa 2 đường dẫn thư mực là save\_dir (thư mục dùng để lưu trữ các thư mục dữ liệu) và capture\_name ( thư mục được sử dụng để lưu dữ liệu sau khi xử lý video sau đó) và sau đó chạy code trong mục “Configure dataset directories” để cài đặt các thư viện cần thiết.

Bước 3: Chạy block trong mục “Upload video file” và chọn “Choose Files”. Sau đó, chọn 1 video selfies có trên máy tính để tải lên drive.

Bước 4: Sửa các chỉ số là max\_scale, fps (lượng frame trên 1 giây), target\_num\_frames (lượng frame tối thiểu có trong video), overwrite (lựa chọn ghi đè) và chạy code trong mục “Flatten into images” để xử lý video thành các ảnh.

Bước 5: Sửa lại lựa chọn image\_scales. Đây là một chuỗi gồm các số là giá trị mà ảnh được giảm chiều xuống. Sau đó chạy code trong thư mục “Resize images into different scales” để thực hiện việc giảm chiều của ảnh và lưu các ảnh đó.

Bước 6: Sửa lại lựa chọn colmap\_image\_scale thành 4 rồi chạy code trong mục “Extract features” để trích xuất các đặc trưng trên ảnh đã được xử lý từ video.

Bước 7: Sửa lại lựa chọn match\_method. Chọn “exhaustive” nếu có ít ảnh và chọn “vocab\_tree” nếu ngược lại. Sau đó chạy code ở mục “Match features” để ghép nối các đặc trưng giữa các ảnh.

Bước 8: Chạy code trong mục “Reconstruction” để tính toán các tham số camera.

Bước 9: Chạy code trong mục “Verify that SfM worked” để kiểm tra liệu code có chạy đúng không.

Bước 10: Chạy code trong các mục còn lại theo thứ tự từ trên xuống để lưu các ảnh và các thông tin khác của dữ liệu gồm các file scene.json, dataset.json và metadata.json.

1. Chạy file notebook “Nerfies\_Training\_v2.ipynb” để huấn luyện mô hình trên tập dữ liệu.

Bước 1: Chạy code trong mục “Environment Setup” để cài đặt các thư viện cần thiết. Cần đảm bảo các thư viện flax có phiên bản 0.5.3, jax và jaxlib đều có phiên bản 0.4.6. Khởi động lại thời gian chạy nếu được yêu cầu.

Bước 2: Chạy code trong mục “Configure notebook runtime” để cấu hình loại thời gian chạy của notebook

Bước 3: Chạy code trong mục “Mount Google Drive” để truy cập đến các thư mục có trong Google drive.

Bước 4: Chạy code trong mục “Define imports and utility functions” để nhập các thư viện cần thiết để chạy notebook.

Bước 5: Sửa các lựa chọn là train\_dir là thư mục dùng để lưu trữ các mô hình sau khi huấn luyện, còn data\_dir là thư mực chứa dữ liệu ảnh. Khi chạy nếu gặp lỗi sau: “ImportError: cannot import name 'nn' from 'flax'(/usr/local/lib/python3.9/dist-packages/flax/\_\_init\_\_.py)” thì ấn vào đường dẫn “[/usr/local/lib/python3.9/dist-packages/nerfies/configs.py](https://localhost:8080/#)”. Sau đó, file config.py cùng với dòng lỗi sẽ hiện ra. Sửa dòng lỗi như sau: “from flax import linen as nn”, sau đó lưu bằng cách ấn Ctrl + S, khởi động lại thời gian chạy và chạy lại code của các mục trước đó của notebook.

Bước 6: Chạy code trong mục “Create datasource and show an example” và mục “Create training iterators” để tạo dữ liệu. Khi gặp phải lỗi sau:”AttributeError: module 'jax.tree\_util' has no attribute 'tree\_multimap'” thì ấn vào đường dẫn “[/usr/local/lib/python3.9/dist-packages/nerfies/utils.py](https://localhost:8080/#)” . Sau đó, file utils.py sẽ cùng với dòng lỗi sẽ hiện ra. Sửa dòng lỗi như sau: “return tree\_util.tree\_map(lambda \*x: np.stack(x), \*list\_of\_pytrees’, sau đó lưu bằng cách ấn Ctrl + S, khởi động lại thời gian chạy và chạy lại code của các mục trước đó của notebook.

Bước 7: Chạy code trong mục “Initialize model” để khởi tạo mô hình. Ấn vào lựa chọn restore\_checkpoint để lấy mô hình (nếu có sẵn) ra huấn luyện tiếp. Khi gặp phải lỗi sau: “TypeError: broadcast\_to requires ndarray or scalar arguments, got <class 'list'> at position 0.” thì ấn vào đường dẫn :”[/usr/local/lib/python3.9/dist-packages/nerfies/model\_utils.py](https://localhost:8080/#)” có trong truy xuất lỗi của hệ thống. Sau đó, file model\_utils.py cùng với dòng lỗi sẽ hiện ra. Sửa lại dòng lỗi như sau: “ jnp.broadcast\_to(jnp.array([last\_sample\_z]), z\_vals[..., :1].shape)“, sau đó lưu bằng cách ấn Ctrl + S, khởi động lại thời gian chạy và chạy lại code của các mục trước đó của notebook.

Bước 8: Chạy code trong mục “Train a Nerfies!” để huấn luyện mô hình. Khi gặp lỗi sau: “AttributeError: module 'jax' has no attribute 'tree\_multimap'” thì ấn vào đường dẫn “[/usr/local/lib/python3.9/dist-packages/nerfies/evaluation.py](https://localhost:8080/#)” trong truy xuất lỗi của hệ thống. Sau đó, file evaluation.py cùng với dòng lỗi sẽ hiện ra. Sửa lại dòng lỗi như sau: “ ret\_map = jax.tree\_map(lambda x: utils.unshard(x, padding), ret\_map)” , sau đó lưu bằng cách ấn Ctrl + S, khởi động lại thời gian chạy và chạy lại code của các mục trước đó của notebook.

1. Chạy file notebook “Nerfies\_Training\_v2.ipynb” để huấn luyện mô hình trên tập dữ liệu.

Bước 1: Chạy code trong mục “Environment Setup” để cài đặt các thư viện cần thiết. Cần đảm bảo các thư viện flax có phiên bản 0.5.3, jax và jaxlib đều có phiên bản 0.4.6. Khởi động lại thời gian chạy nếu được yêu cầu.

Bước 2: Chạy code trong mục “Configure notebook runtime” để cấu hình loại thời gian chạy của notebook.

Bước 3: Chạy code trong mục “Mount Google Drive” để truy cập đến các thư mục có trong Google drive.

Bước 4: Chạy code trong mục “Define imports and utility functions” để nhập các thư viện cần thiết để chạy notebook.

Bước 5: Sửa các lựa chọn là train\_dir là thư mục dùng để lưu trữ các mô hình sau khi huấn luyện, còn data\_dir là thư mực chứa dữ liệu ảnh trong mục “Model and dataset configuration”. Khi chạy nếu gặp lỗi sau: “ImportError: cannot import name 'nn' from 'flax'(/usr/local/lib/python3.9/dist-packages/flax/\_\_init\_\_.py)” thì ấn vào đường dẫn “[/usr/local/lib/python3.9/dist-packages/nerfies/configs.py](https://localhost:8080/#)”. Sau đó, file config.py cùng với dòng lỗi sẽ hiện ra. Sửa dòng lỗi như sau: “from flax import linen as nn”, sau đó lưu bằng cách ấn Ctrl + S, khởi động lại thời gian chạy và chạy lại code trong các mục trước đó của notebook.

Bước 6: Chạy code trong mục “Create datasource and show an example.” để tạo bộ dữ liệu cho mô hình.

Bước 7: Chạy code trong mục “Initialize model” để khởi tạo mô hình từ các checkpoint có trong mục exp. Khi gặp phải lỗi sau: “TypeError: broadcast\_to requires ndarray or scalar arguments, got <class 'list'> at position 0.” thì ấn vào đường dẫn :”[/usr/local/lib/python3.9/dist-packages/nerfies/model\_utils.py](https://localhost:8080/#)” có trong truy xuất lỗi của hệ thống. Sau đó, file model\_utils.py cùng với dòng lỗi sẽ hiện ra. Sửa lại dòng lỗi như sau: “ jnp.broadcast\_to(jnp.array([last\_sample\_z]), z\_vals[..., :1].shape)“, sau đó lưu bằng cách ấn Ctrl + S, khởi động lại thời gian chạy và chạy lại code của các mục trước đó của notebook.

Bước 8: Chạy code trong các mục “Define pmapped render function.” và “Load camera” để tạo một số hàm hiển thị ảnh và tải các tham số của camera lên notebook.

Bước 9: Chạy code trong mục “Render video frames” để hiển thị các frame của video sau khi đã được mô hình xử lý. Khi gặp lỗi sau: “AttributeError: module 'jax' has no attribute 'tree\_multimap'” thì ấn vào đường dẫn “[/usr/local/lib/python3.9/dist-packages/nerfies/evaluation.py](https://localhost:8080/#)” trong truy xuất lỗi của hệ thống. Sau đó, file evaluation.py cùng với dòng lỗi sẽ hiện ra. Sửa lại dòng lỗi như sau: “ ret\_map = jax.tree\_map(lambda x: utils.unshard(x, padding), ret\_map)” , sau đó lưu bằng cách ấn Ctrl + S, khởi động lại thời gian chạy và chạy lại code trong các mục trước đó của notebook.

Bước 10: Chạy code trong mục “Show rendered video.” để cho ra video đã được mô hình xử lý.

II. Hướng dẫn train code trên máy tính cá nhân (Chỉ cài trên máy có hệ điều hành Linux)

1. Tải tệp dataset về sau đó cho vào folder “nerfies”

( Tải tệp zip “nerfies-vrig-dataset-v0.1.zip” tại đường dẫn “<https://github.com/google/nerfies/releases/tag/0.1>”)

1. Tạo 1 env riêng với phiên bản python 3.9 (không nên sử dụng python 3.10 và 3.11 vì nó có thể bị lỗi với 1 số package cần cài đặt)
2. Trong file requirements.txt, thay đổi phiên bản của opencv-python thành 4.7.0.72 và thêm package ‘jaxlib’ với phiên bản 0.1.69
3. Kích hoạt môi trường ảo và chạy lệnh “pip install -r requirements.txt” để cài đặt các packages
4. Tiếp theo chạy lệnh “pip install --upgrade "jax[cuda111]" -f <https://storage.googleapis.com/jax-releases/jax_releases.html>” để cài đặt jax
5. Ở trong file test\_vrig.gin nên sửa ‘defaults.gin’ ở dòng “ include 'defaults.gin' “ thành đường dẫn đến file ‘defaults.gin’ trong folder configs chẳng hạn như ‘./defaults.gin’
6. Ở trong file train.py, comment lại dòng số 53

( Dòng jax.config.parse\_flags\_with\_absl() )

1. Tiếp đến để train chúng ta chạy lệnh:

python train.py \

--data\_dir $DATASET\_PATH \

--base\_folder $EXPERIMENT\_PATH \

--gin\_configs configs/test\_vrig.gin

với $DATASET\_PATH là đường dẫn đến tập data muốn sử dụng

$EXPERIMENT\_PATH là đường dẫn đến thư mục muốn lưu kết quả train

configs/test\_vrig.gin có thể sửa thành ./configs/test\_vrig.gin để dẫn đến file test\_vrig.gin trong folder configs