MONOCULAR DEPTH ESTIMATION

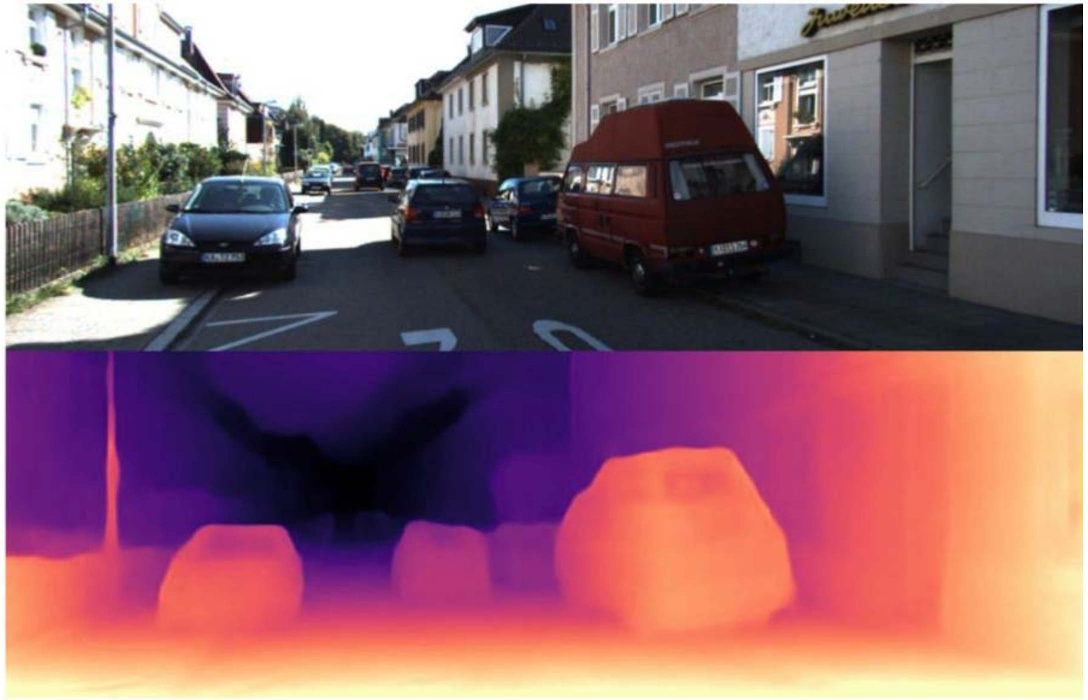
FINAL PRESENTATION, TEAM: ENIGMA

IRFAN ALI SADAB

MD ARAFAT ISLAM

WHAT IS MONOCULAR DEPTH ESTIMATION?

- Monocular depth estimation is the process of determining the distance of objects in a scene using only a single 2D image. This task is challenging as it typically requires stereo vision, but monocular methods aim to infer depth from a single viewpoint. Traditional techniques use geometric cues like focus, motion, or shading, while modern approaches leverage deep learning to directly predict depth from images. These methods find applications in autonomous navigation, augmented reality, and robotics, enabling machines to perceive and interact with their environment more effectively.



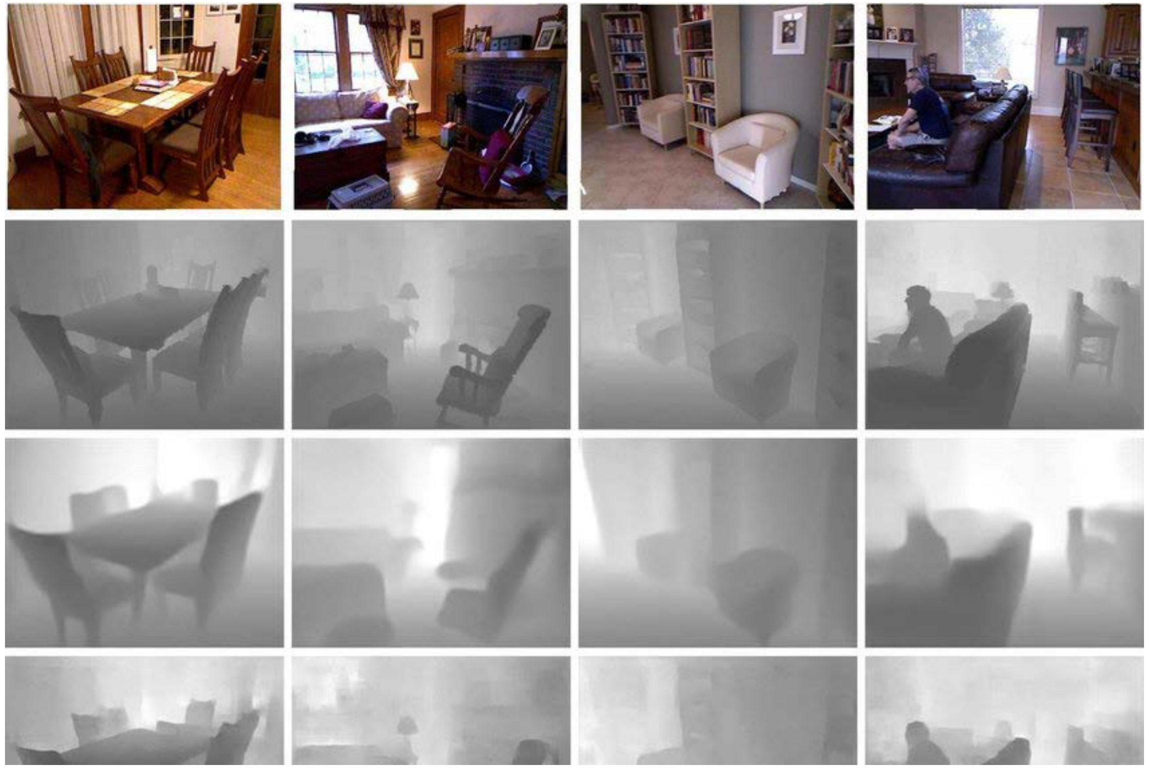
MOTIVATION

1. Monocular depth estimation, which infers depth from a single image, is crucial for applications in autonomous driving, augmented reality, and robotics. Traditional methods relying on stereo vision or LiDAR are expensive and complex, while monocular techniques offer a cost-effective and accessible alternative. This project aims to develop robust monocular depth estimation models, enhancing spatial awareness and enabling innovative applications across various fields, thereby democratizing access to depth information and contributing to technological advancements.

MOTIVATION

DATASET & PREPROCESSING:

- For this project, we decided to work on NUY V2 dataset.
- The NYU Depth V2 dataset is a widely-used benchmark dataset for depth estimation and indoor scene understanding. It consists of RGB-D images captured by Kinect sensors in indoor environments, providing both color and depth information for each scene. The dataset contains over 1,400 densely labeled scenes with diverse indoor scenes, furniture arrangements, and lighting conditions. Each scene includes aligned RGB images, depth maps, and semantic annotations, making it suitable for tasks like depth estimation, semantic segmentation, and 3D reconstruction.



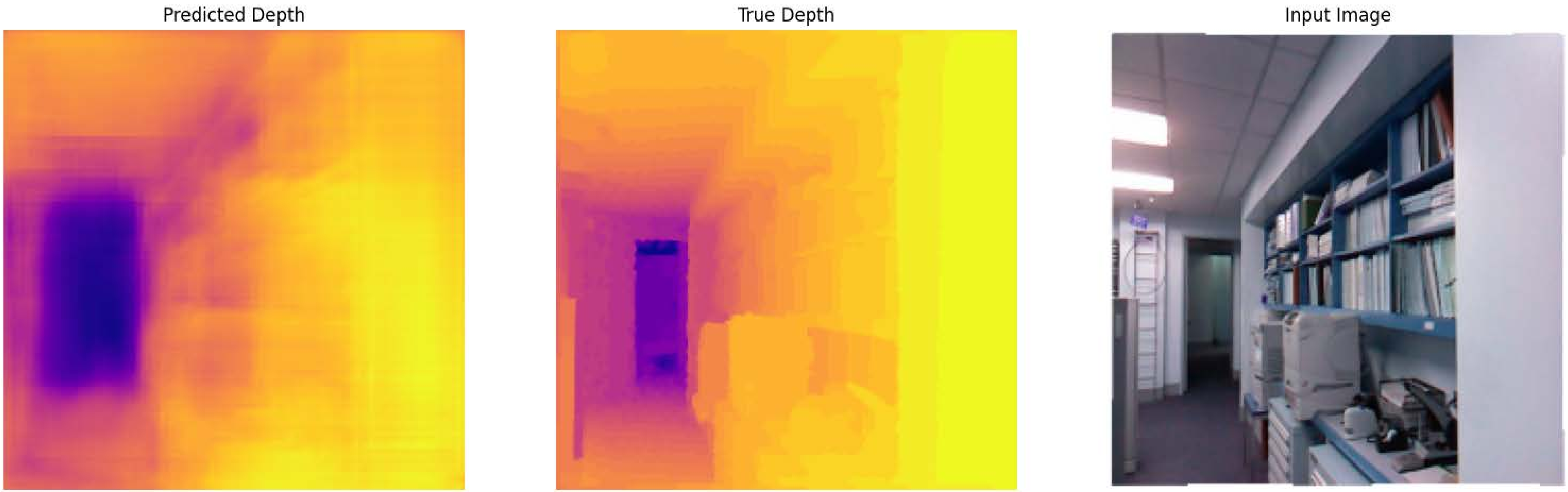
MODEL ARCHITECTURE(U-NET):

- Encoders:
  - MobileNetV2
  - MobileNetV2(attention mechanism)
  - DenseNet121
  - DenseNet169
  - EfficientNet
- Custom Encoder & Decoder:
  - Dense_Unet(with attention mechanism)
  - Res_Unet(with attention mechanism)
  - U-model

MOBILENETV2

```
Epoch 1/10
1000/1000 [==============================] - 3089s 3s/step - loss: 0.1082 - val_loss: 0.1864
Epoch 2/10
1000/1000 [==============================] - 1192s 1s/step - loss: 0.0828 - val_loss: 0.1872
Epoch 3/10
1000/1000 [==============================] - 1284s 1s/step - loss: 0.0732 - val_loss: 0.1019
Epoch 4/10
1000/1000 [==============================] - 1282s 1s/step - loss: 0.0653 - val_loss: 0.0975
Epoch 5/10
1000/1000 [==============================] - 1226s 1s/step - loss: 0.0598 - val_loss: 0.0800
Epoch 6/10
1000/1000 [==============================] - 1226s 1s/step - loss: 0.0539 - val_loss: 0.0884
Epoch 7/10
1000/1000 [==============================] - 1324s 1s/step - loss: 0.0512 - val_loss: 0.0700
Epoch 8/10
1000/1000 [==============================] - 1284s 1s/step - loss: 0.0467 - val_loss: 0.0654
Epoch 9/10
1000/1000 [==============================] - 1260s 1s/step - loss: 0.0440 - val_loss: 0.0582
Epoch 10/10
1000/1000 [==============================] - 1233s 1s/step - loss: 0.0407 - val_loss: 0.0585
```

MOBILENETV2

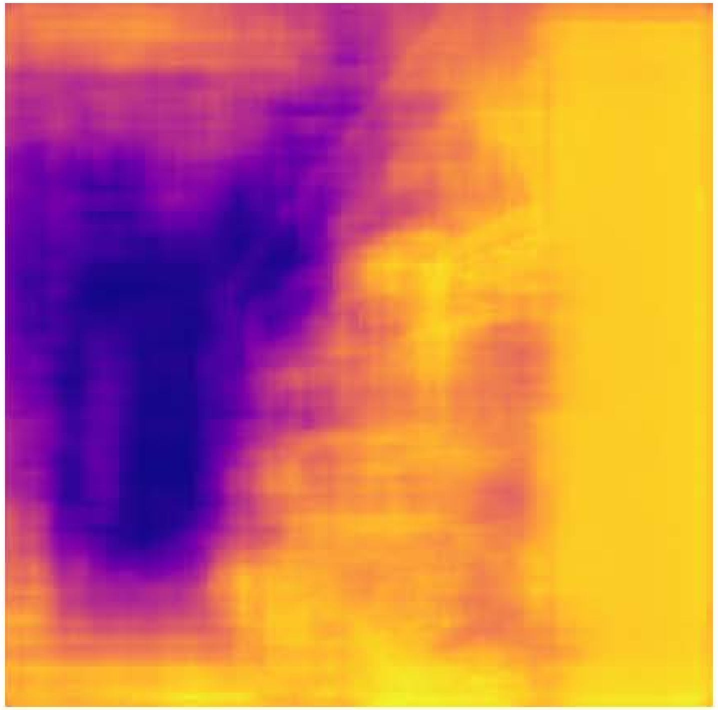


MOBILENETV2(ATTENTION MECHANISM)

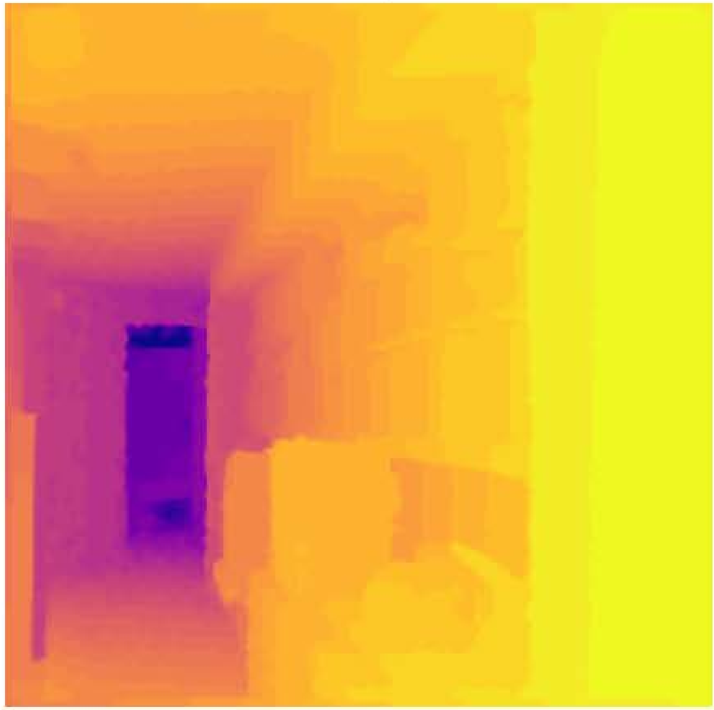
```
Epoch 1/10
1000/1000 [==============================] - 6846s 7s/step - loss: 0.1214 - val_loss: 0.1700
Epoch 2/10
1000/1000 [==============================] - 1290s 1s/step - loss: 0.0976 - val_loss: 0.2133
Epoch 3/10
1000/1000 [==============================] - 1291s 1s/step - loss: 0.0898 - val_loss: 0.0964
Epoch 4/10
1000/1000 [==============================] - 1302s 1s/step - loss: 0.0861 - val_loss: 0.1141
Epoch 5/10
1000/1000 [==============================] - 1366s 1s/step - loss: 0.0817 - val_loss: 0.1149
Epoch 6/10
1000/1000 [==============================] - 1333s 1s/step - loss: 0.0820 - val_loss: 0.1429
Epoch 7/10
1000/1000 [==============================] - 1373s 1s/step - loss: 0.0801 - val_loss: 0.0856
Epoch 8/10
1000/1000 [==============================] - 1300s 1s/step - loss: 0.0751 - val_loss: 0.0830
Epoch 9/10
1000/1000 [==============================] - 1293s 1s/step - loss: 0.0754 - val_loss: 0.0951
Epoch 10/10
1000/1000 [==============================] - 1363s 1s/step - loss: 0.0759 - val_loss: 0.0779
```

MOBILENETV2(ATTENTION MECHANISM)

Predicted Depth



True Depth



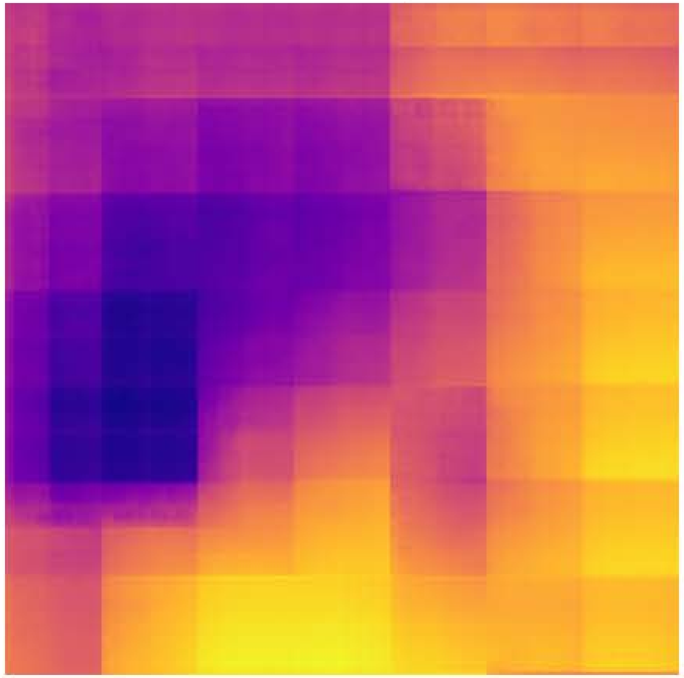
Input Image

DENSENET121

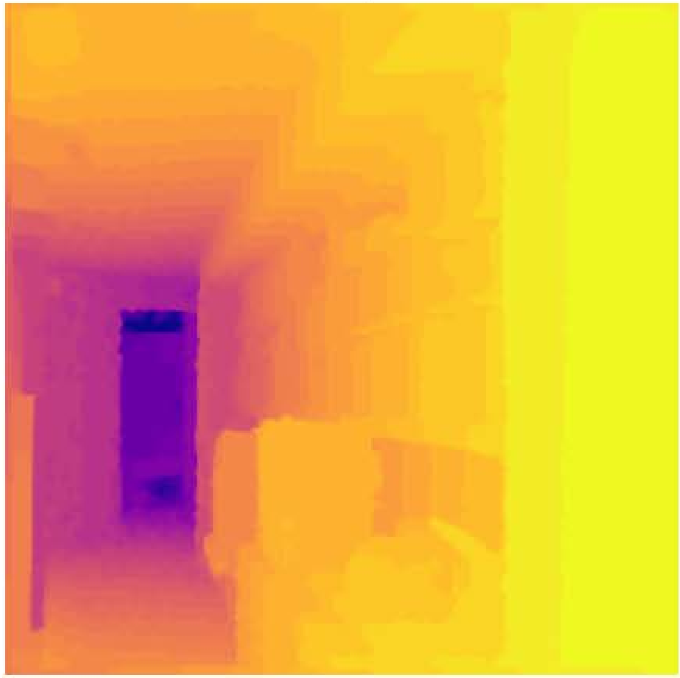
```
Epoch 1/10
1000/1000 [==============================] - 4336s 4s/step - loss: 0.1222 - val_loss: 0.1031
Epoch 2/10
1000/1000 [==============================] - 929s 928ms/step - loss: 0.1070 - val_loss: 0.1115
Epoch 3/10
1000/1000 [==============================] - 925s 925ms/step - loss: 0.1038 - val_loss: 0.0999
Epoch 4/10
1000/1000 [==============================] - 920s 920ms/step - loss: 0.1035 - val_loss: 0.1057
Epoch 5/10
1000/1000 [==============================] - 923s 923ms/step - loss: 0.1009 - val_loss: 0.0996
Epoch 6/10
1000/1000 [==============================] - 925s 925ms/step - loss: 0.1007 - val_loss: 0.0971
Epoch 7/10
1000/1000 [==============================] - 926s 927ms/step - loss: 0.1005 - val_loss: 0.0962
Epoch 8/10
1000/1000 [==============================] - 883s 883ms/step - loss: 0.0997 - val_loss: 0.0953
Epoch 9/10
1000/1000 [==============================] - 910s 910ms/step - loss: 0.0990 - val_loss: 0.0973
Epoch 10/10
1000/1000 [==============================] - 910s 910ms/step - loss: 0.0991 - val_loss: 0.0972
```

DENSENET121

Predicted Depth



True Depth

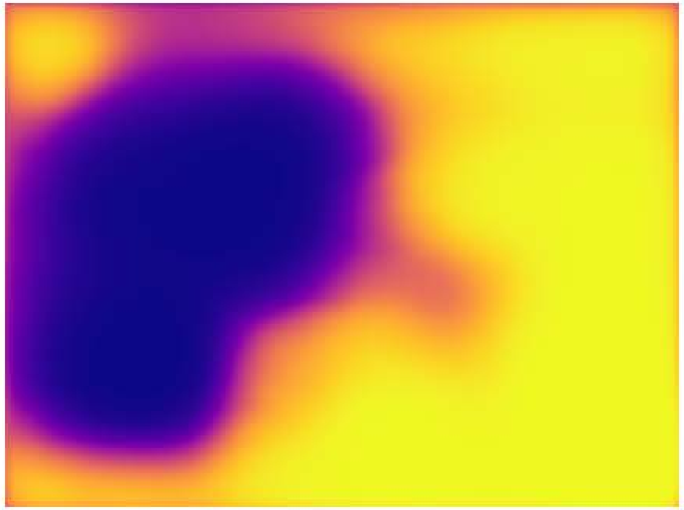


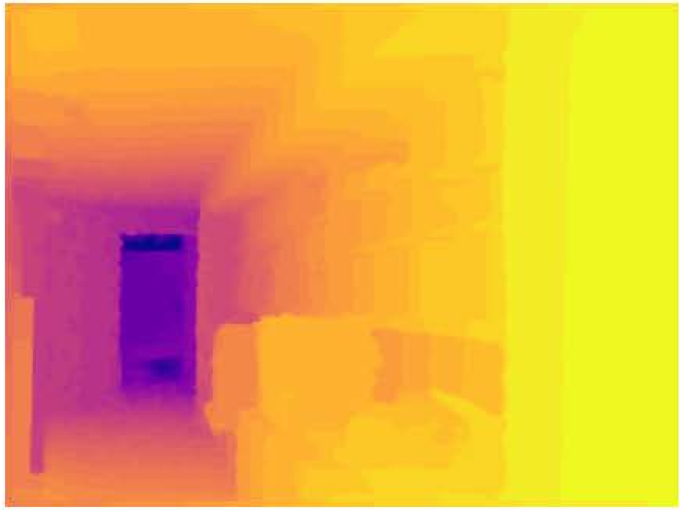
Input Image

DENSENET169

```
Epoch 1/3
100/100 [==============================] - 480s 4s/step - loss: 0.1933 - accuracy_function: 0.7067 - val_loss: 0.2543 - val_accuracy_function: 0.6460
Epoch 2/3
100/100 [==============================] - 96s 963ms/step - loss: 0.1606 - accuracy_function: 0.7738 - val_loss: 0.1830 - val_accuracy_function: 0.7517
Epoch 3/3
100/100 [==============================] - 96s 963ms/step - loss: 0.1446 - accuracy_function: 0.8181 - val_loss: 0.1790 - val_accuracy_function: 0.7317
```

DENSENET169

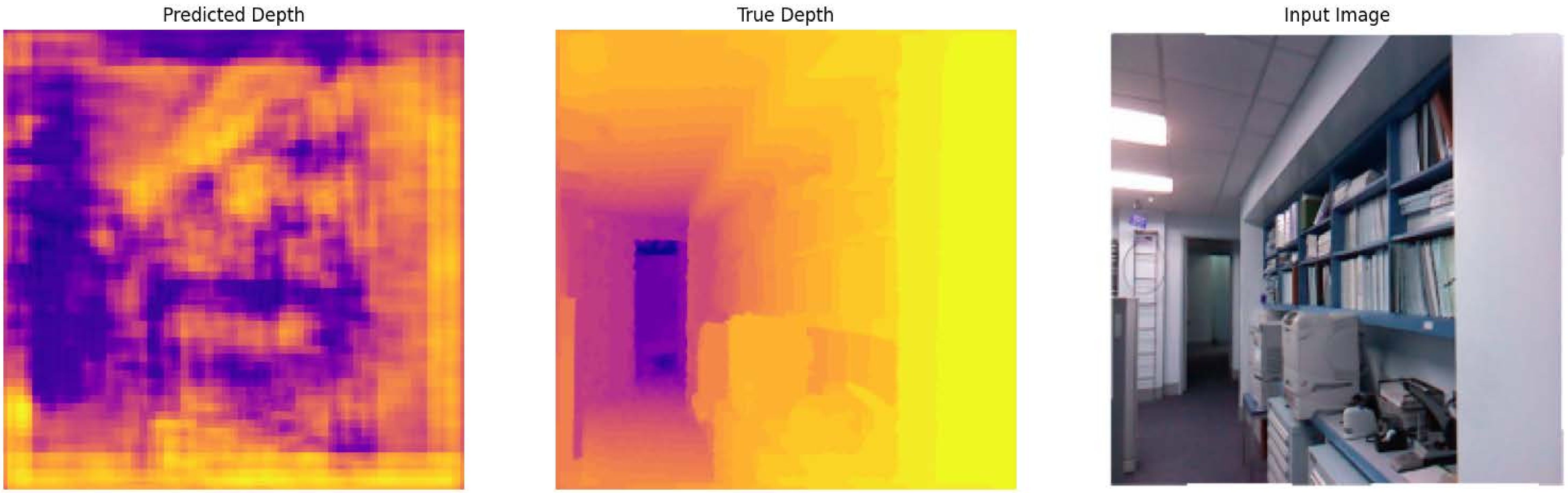




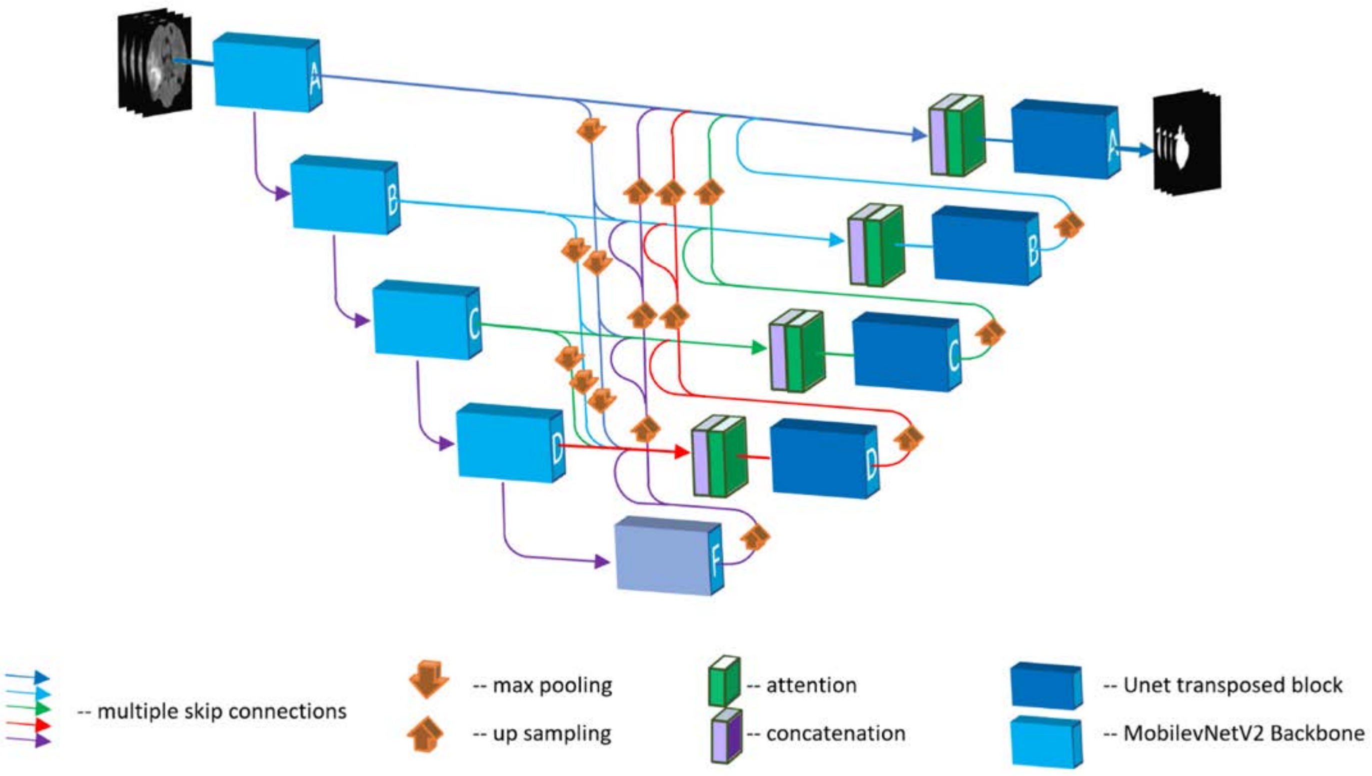
EFFICIENTNET

```
Epoch 1/10
1000/1000 [==============================] - 6798s 7s/step - loss: 1.6047 - val_loss: 1.6865
Epoch 2/10
1000/1000 [==============================] - 988s 988ms/step - loss: 1.5459 - val_loss: 1.6555
Epoch 3/10
1000/1000 [==============================] - 987s 987ms/step - loss: 1.5276 - val_loss: 1.5937
Epoch 4/10
1000/1000 [==============================] - 963s 963ms/step - loss: 1.5188 - val_loss: 1.5283
Epoch 5/10
1000/1000 [==============================] - 973s 973ms/step - loss: 1.5118 - val_loss: 1.5096
Epoch 6/10
1000/1000 [==============================] - 969s 969ms/step - loss: 1.5039 - val_loss: 1.5007
Epoch 7/10
1000/1000 [==============================] - 970s 970ms/step - loss: 1.4979 - val_loss: 1.4886
Epoch 8/10
1000/1000 [==============================] - 978s 978ms/step - loss: 1.4941 - val_loss: 1.4730
Epoch 9/10
1000/1000 [==============================] - 983s 983ms/step - loss: 1.4870 - val_loss: 1.4666
Epoch 10/10
1000/1000 [==============================] - 977s 977ms/step - loss: 1.4813 - val_loss: 1.4721
```

EFFICIENTNET



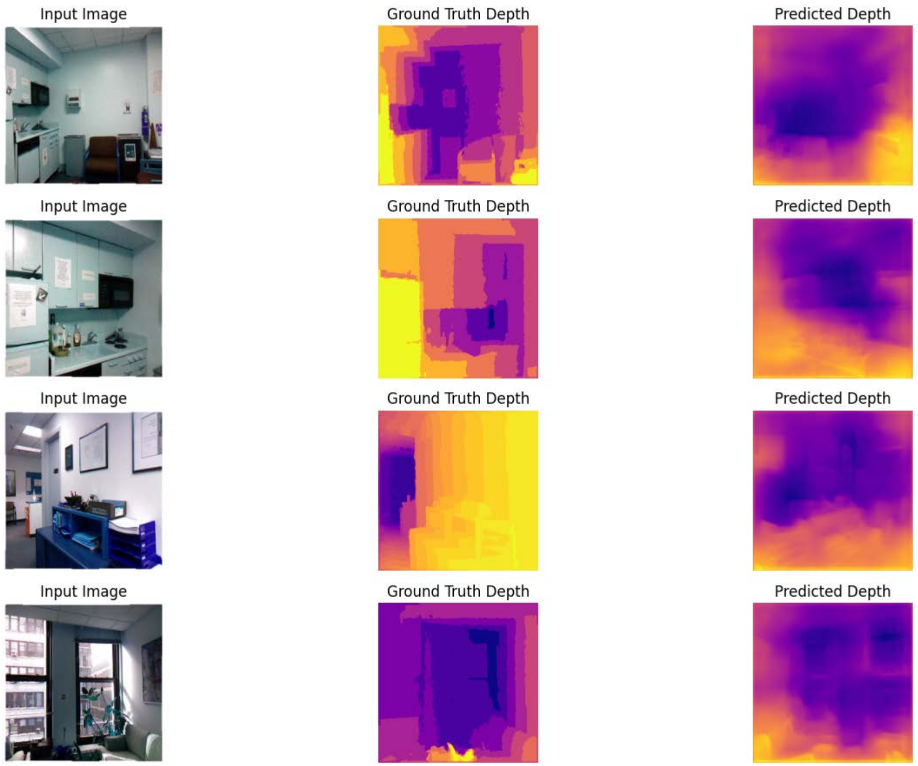
DENSE_U-NET(WITH ATTENTION MECHANISM)



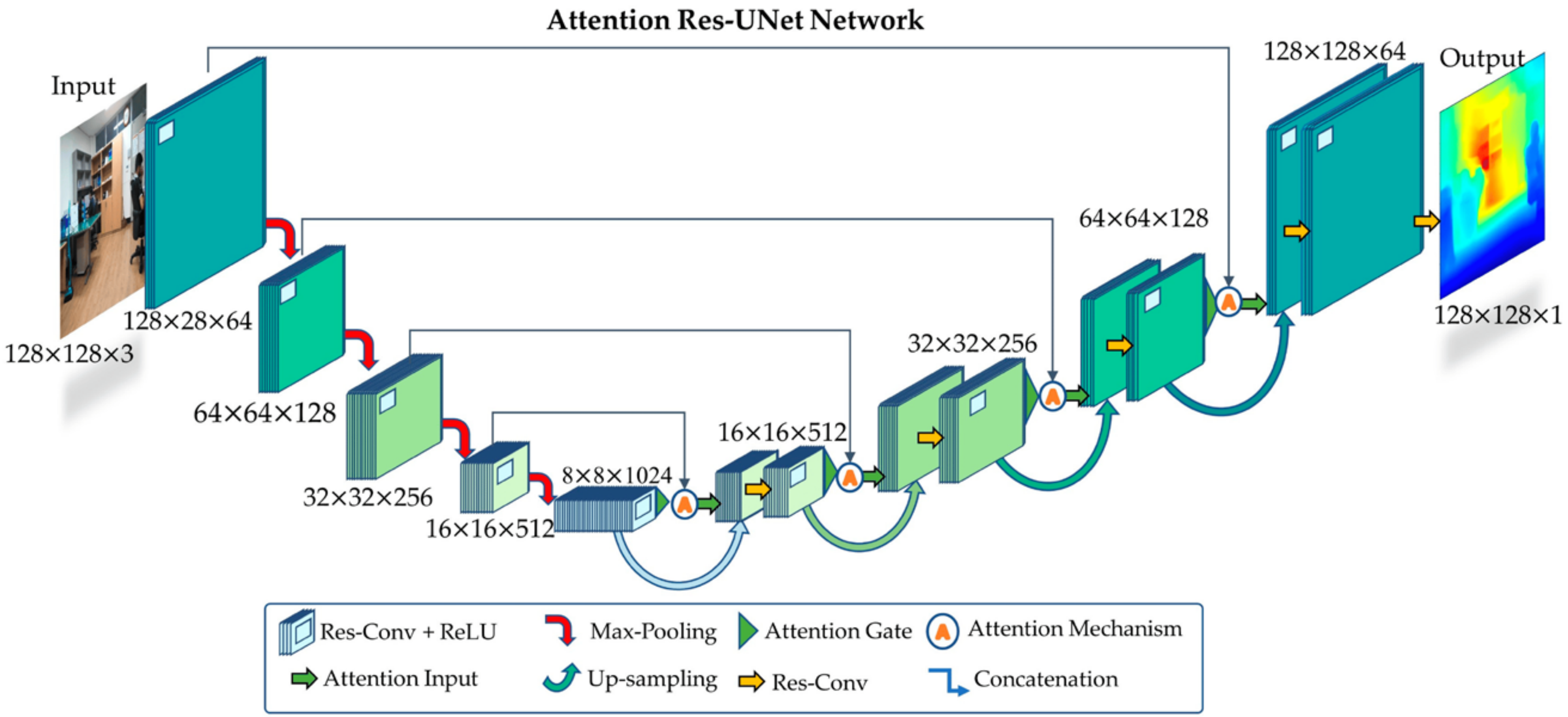
DENSE_U-NET(WITH ATTENTION MECHANISM)

```
  warnings.warn(
Epoch 1/25
c:\Users\User\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\trainers\data_adapters\py_dataset_adapter.py:120: UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs
  self._warn_if_super_not_called()
100/100 ━━━━━━━━━━━━━━━━━━━━ 698s 7s/step - loss: 1.1672 - ssim_loss: 0.4704 - val_loss: 1.0707 - val_ssim_loss: 0.3842
Epoch 2/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 632s 6s/step - loss: 1.0596 - ssim_loss: 0.3786 - val_loss: 1.0608 - val_ssim_loss: 0.3807
Epoch 3/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 631s 6s/step - loss: 1.0526 - ssim_loss: 0.3812 - val_loss: 1.0561 - val_ssim_loss: 0.3791
Epoch 4/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 630s 6s/step - loss: 1.0073 - ssim_loss: 0.3624 - val_loss: 1.0705 - val_ssim_loss: 0.3841
Epoch 5/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 630s 6s/step - loss: 0.9752 - ssim_loss: 0.3345 - val_loss: 1.0640 - val_ssim_loss: 0.3823
Epoch 6/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 630s 6s/step - loss: 0.9783 - ssim_loss: 0.3350 - val_loss: 1.0081 - val_ssim_loss: 0.3588
Epoch 7/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 633s 6s/step - loss: 0.9740 - ssim_loss: 0.3365 - val_loss: 0.9875 - val_ssim_loss: 0.3481
Epoch 8/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 630s 6s/step - loss: 0.9617 - ssim_loss: 0.3252 - val_loss: 0.9853 - val_ssim_loss: 0.3472
Epoch 9/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 630s 6s/step - loss: 0.9726 - ssim_loss: 0.3298 - val_loss: 0.9739 - val_ssim_loss: 0.3409
Epoch 10/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 632s 6s/step - loss: 0.9669 - ssim_loss: 0.3317 - val_loss: 0.9819 - val_ssim_loss: 0.3481
Epoch 11/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 631s 6s/step - loss: 0.9553 - ssim_loss: 0.3239 - val_loss: 0.9728 - val_ssim_loss: 0.3389
Epoch 12/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 631s 6s/step - loss: 0.9581 - ssim_loss: 0.3228 - val_loss: 0.9811 - val_ssim_loss: 0.3421
Epoch 13/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 633s 6s/step - loss: 0.9555 - ssim_loss: 0.3215 - val_loss: 0.9946 - val_ssim_loss: 0.3468
...
Epoch 24/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 645s 6s/step - loss: 0.9186 - ssim_loss: 0.3023 - val_loss: 0.9462 - val_ssim_loss: 0.3214
Epoch 25/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 640s 6s/step - loss: 0.9174 - ssim_loss: 0.3038 - val_loss: 0.9781 - val_ssim_loss: 0.3467
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

DENSE_U-NET(WITH ATTENTION MECHANISM)



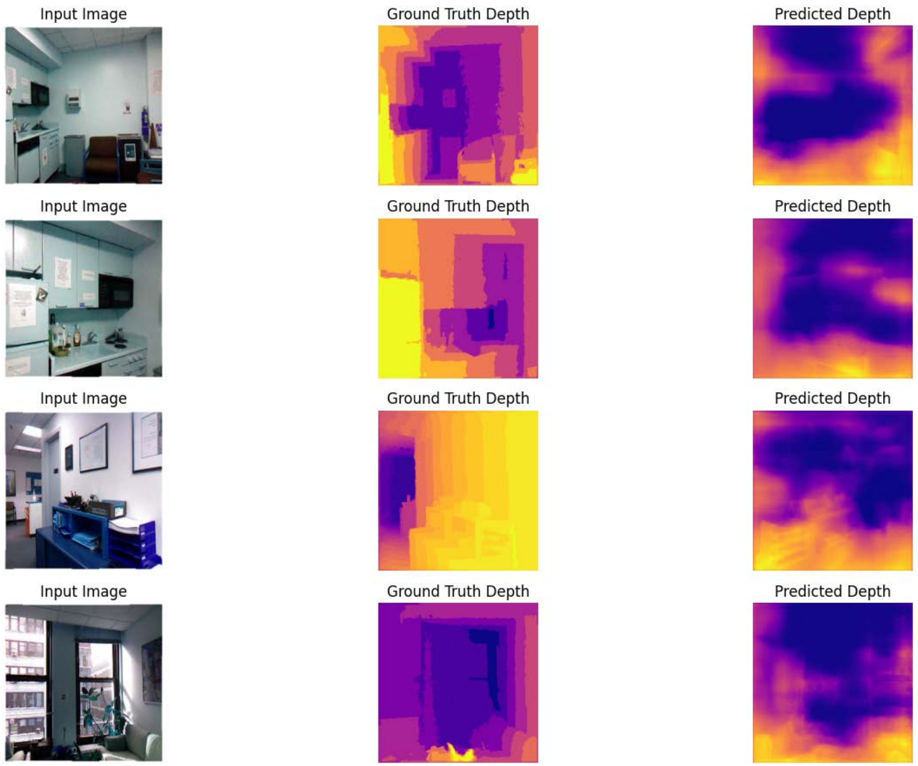
RES_UNET(WITH ATTENTION MECHANISM)



RES_UNET(WITH ATTENTION MECHANISM)

```
c:\Users\User\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\layers\activations\leaky_relu.py:41: UserWarning: Argument `alpha` is deprecated. Use `ne
  warnings.warn(
Epoch 1/25
c:\Users\User\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\trainers\data_adapters\py_dataset_adapter.py:120: UserWarning: Your `PyDataset` class sho
  self._warn_if_super_not_called()
100/100 ━━━━━━━━━━━━━━━━━━━━ 308s 3s/step - loss: 1.1928 - ssim_loss: 0.5086 - val_loss: 1.0501 - val_ssim_loss: 0.3655
Epoch 2/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 289s 3s/step - loss: 1.0543 - ssim_loss: 0.3746 - val_loss: 1.0432 - val_ssim_loss: 0.3628
Epoch 3/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 286s 3s/step - loss: 1.0560 - ssim_loss: 0.3828 - val_loss: 1.0295 - val_ssim_loss: 0.3590
Epoch 4/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 285s 3s/step - loss: 1.0203 - ssim_loss: 0.3672 - val_loss: 1.0044 - val_ssim_loss: 0.3472
Epoch 5/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 282s 3s/step - loss: 0.9975 - ssim_loss: 0.3473 - val_loss: 0.9718 - val_ssim_loss: 0.3295
Epoch 6/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 281s 3s/step - loss: 1.0013 - ssim_loss: 0.3535 - val_loss: 0.9718 - val_ssim_loss: 0.3299
Epoch 7/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 279s 3s/step - loss: 0.9835 - ssim_loss: 0.3439 - val_loss: 0.9643 - val_ssim_loss: 0.3233
Epoch 8/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 279s 3s/step - loss: 0.9903 - ssim_loss: 0.3455 - val_loss: 0.9616 - val_ssim_loss: 0.3204
Epoch 9/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 284s 3s/step - loss: 0.9797 - ssim_loss: 0.3413 - val_loss: 0.9955 - val_ssim_loss: 0.3379
Epoch 10/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 283s 3s/step - loss: 0.9800 - ssim_loss: 0.3404 - val_loss: 0.9622 - val_ssim_loss: 0.3242
Epoch 11/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 282s 3s/step - loss: 0.9758 - ssim_loss: 0.3373 - val_loss: 0.9851 - val_ssim_loss: 0.3279
Epoch 12/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 283s 3s/step - loss: 0.9785 - ssim_loss: 0.3399 - val_loss: 0.9563 - val_ssim_loss: 0.3210
Epoch 13/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 281s 3s/step - loss: 0.9705 - ssim_loss: 0.3385 - val_loss: 0.9668 - val_ssim_loss: 0.3283
...
Epoch 24/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 279s 3s/step - loss: 0.9520 - ssim_loss: 0.3218 - val_loss: 0.9306 - val_ssim_loss: 0.2999
Epoch 25/25
100/100 ━━━━━━━━━━━━━━━━━━━━ 280s 3s/step - loss: 0.9466 - ssim_loss: 0.3188 - val_loss: 0.9368 - val_ssim_loss: 0.3052
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

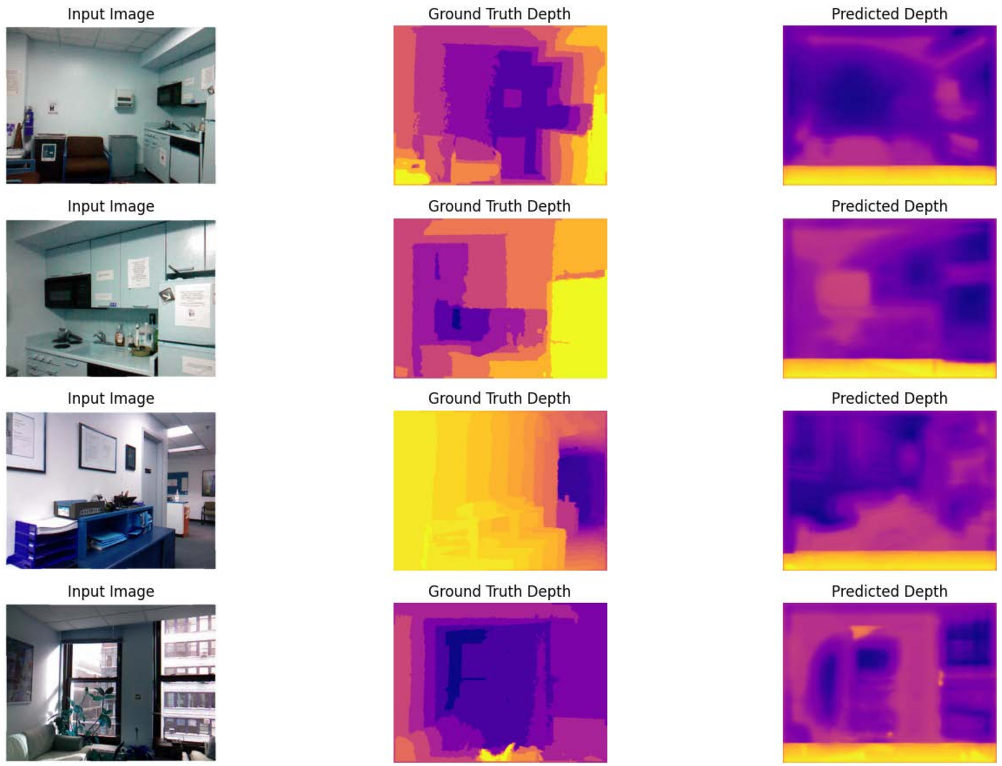
RES_UNET(WITH ATTENTION MECHANISM)



U-MODEL

```
c:\Users\User\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\layers\activations\leaky_relu.py:41: UserWarning: Argument `alpha` is deprecated. Use ne
  warnings.warn(
Epoch 1/10
c:\Users\User\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\trainers\data_adapters\py_dataset_adapter.py:120: UserWarning: Your `PyDataset` class sho
  self._warn_if_super_not_called()
100/100 ━━━━━━━━━━━━━━━━━━━━ 1607s 16s/step - loss: 0.4450 - val_loss: 0.4206
Epoch 2/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1577s 16s/step - loss: 0.3627 - val_loss: 0.3731
Epoch 3/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1571s 16s/step - loss: 0.3659 - val_loss: 0.3808
Epoch 4/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1562s 16s/step - loss: 0.3468 - val_loss: 0.3751
Epoch 5/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1553s 16s/step - loss: 0.3445 - val_loss: 0.3709
Epoch 6/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1570s 16s/step - loss: 0.3497 - val_loss: 0.3621
Epoch 7/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1573s 16s/step - loss: 0.3422 - val_loss: 0.3763
Epoch 8/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1572s 16s/step - loss: 0.3409 - val_loss: 0.3741
Epoch 9/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1579s 16s/step - loss: 0.3445 - val_loss: 0.3650
Epoch 10/10
100/100 ━━━━━━━━━━━━━━━━━━━━ 1575s 16s/step - loss: 0.3442 - val_loss: 0.3680
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using ir
```

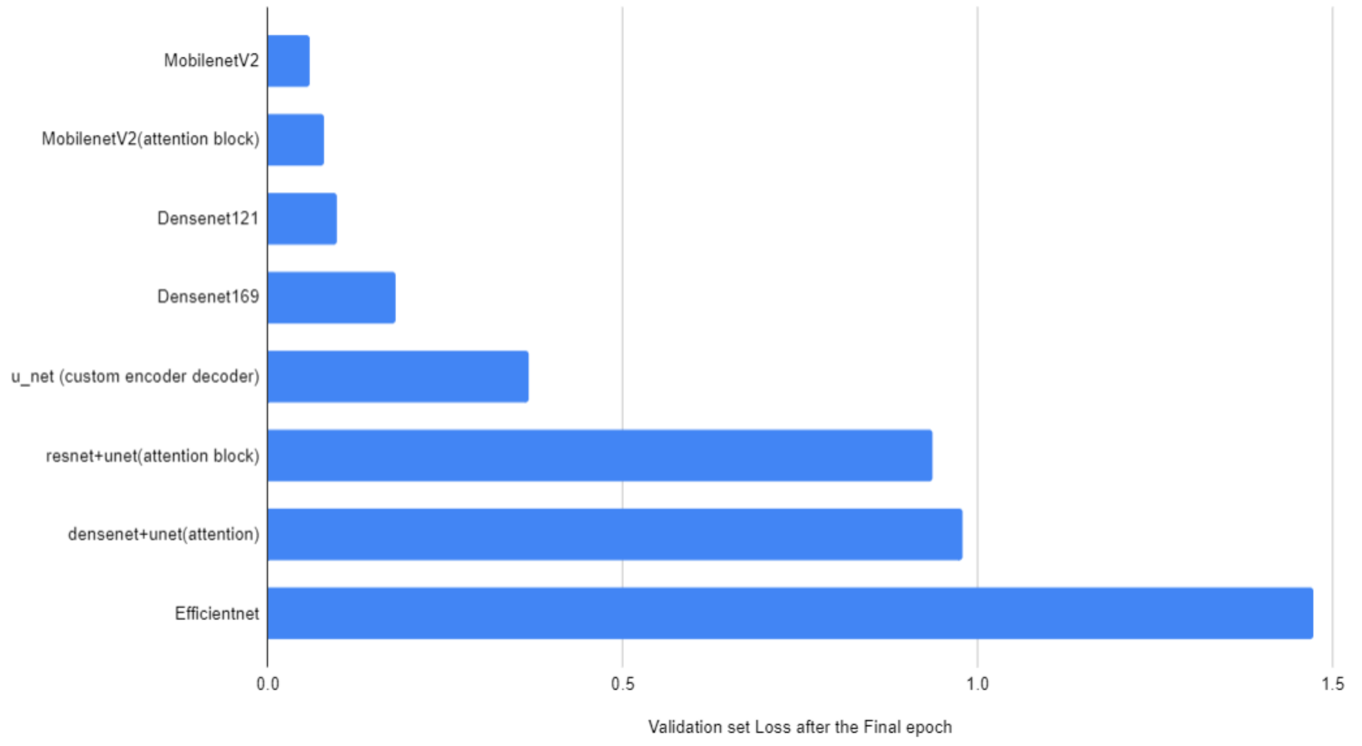
U-MODEL


Input Image
Ground Truth Depth
Predicted Depth
Input Image
Ground Truth Depth
Predicted Depth
Input Image
Ground Truth Depth
Predicted Depth
Input Image
Ground Truth Depth
Predicted Depth

VALIDATION LOSS COMPARISON TABLE

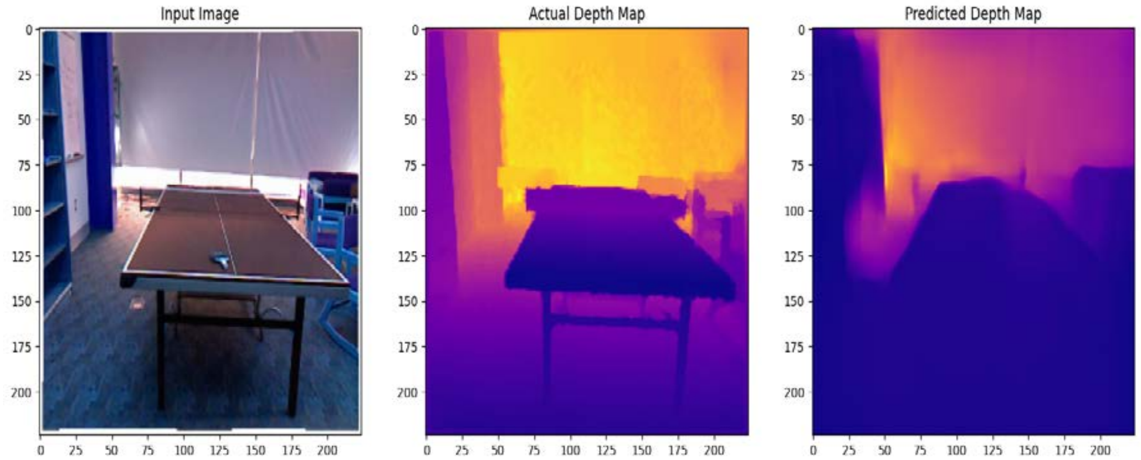
	MobileNetV2	MobileNetV2(attention mechanism)	DenseNet121	DenseNet169	EfficientNet	U-Net	Res_Unet(with attention mechanism)	Dense_Unet(with attention mechanism)
Validation set Loss after the Final epoch	0.0585	0.0779	0.0972	0.179	1.4721	0.3680	0.9368	0.9781

VALIDATION COMPARISON GRAPH

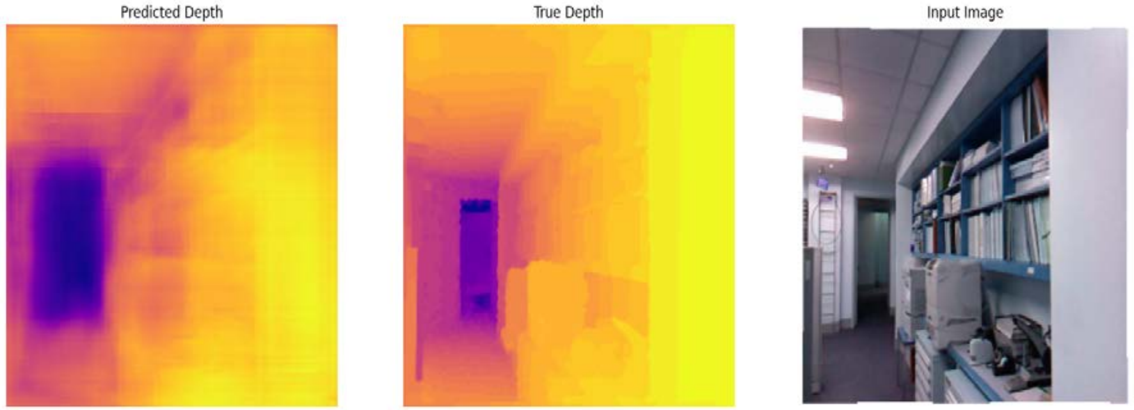


COMPARE WITH PRE-TRAINED MODEL OUTPUT

- Pre-trained model: monodepthV2

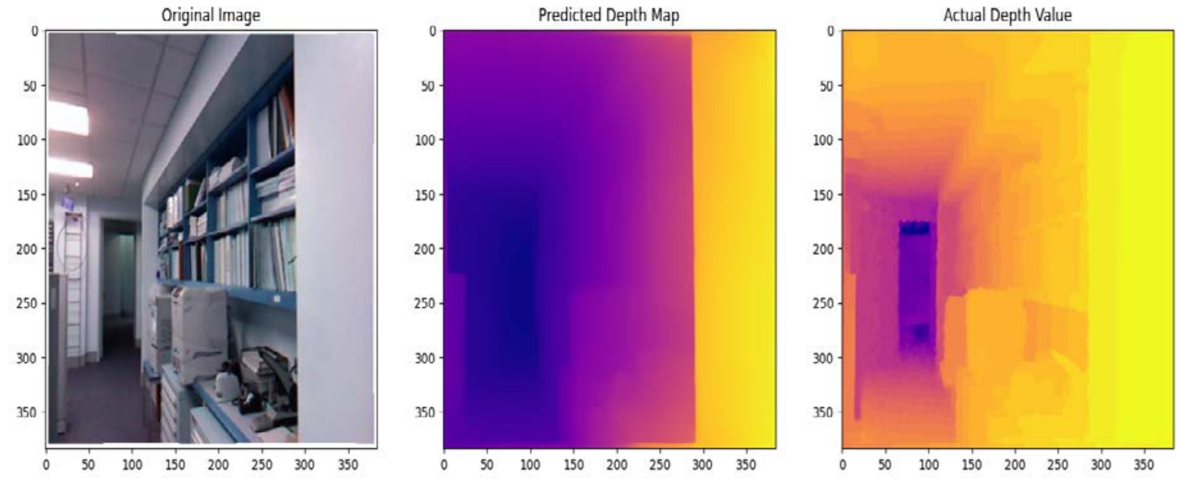


- mobilnetV2

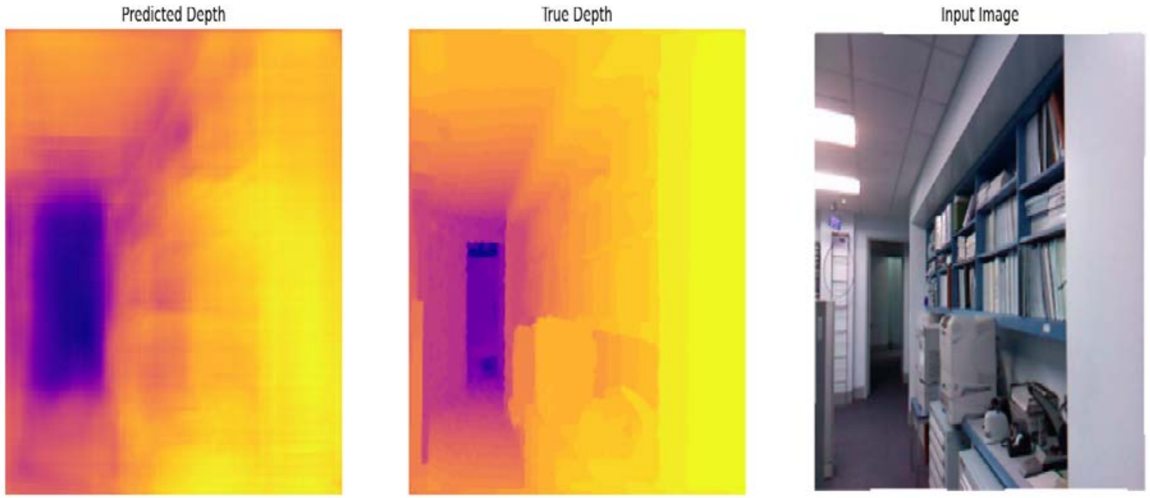


COMPARE WITH PRE-TRAINED MODEL OUTPUT

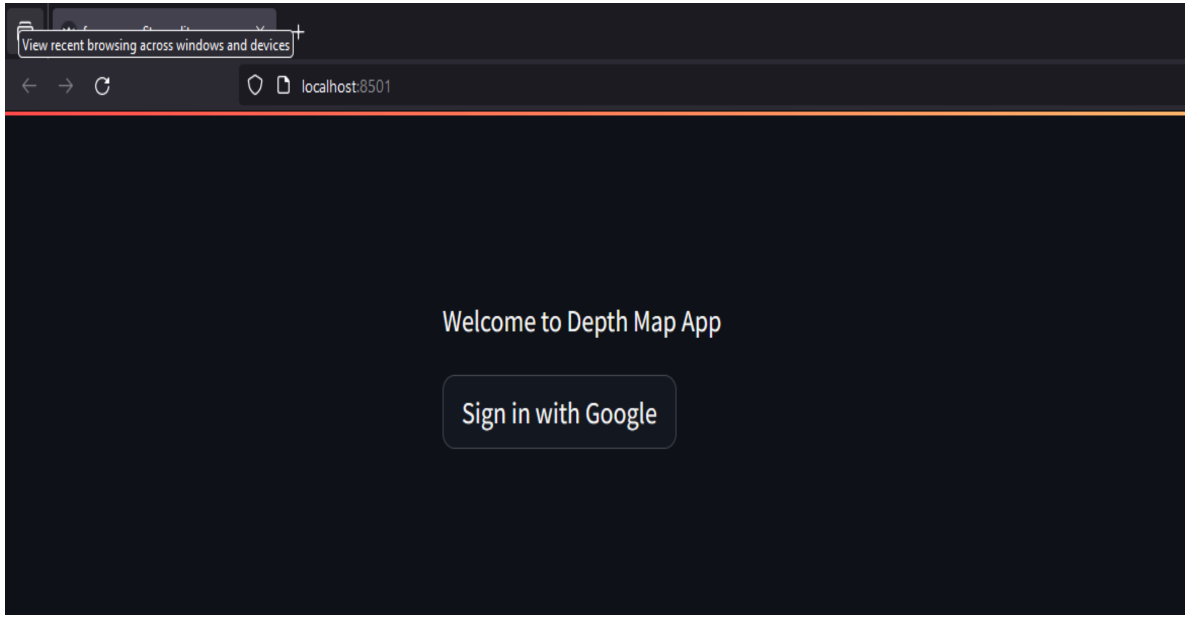
- Pre-trained model: midas

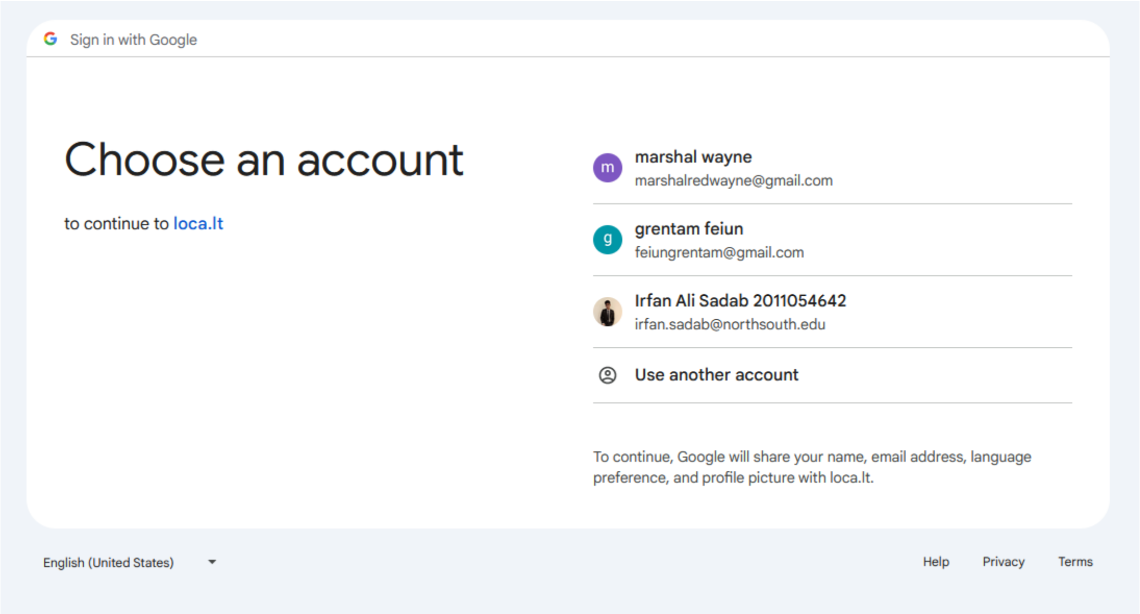


- mobilnetV2

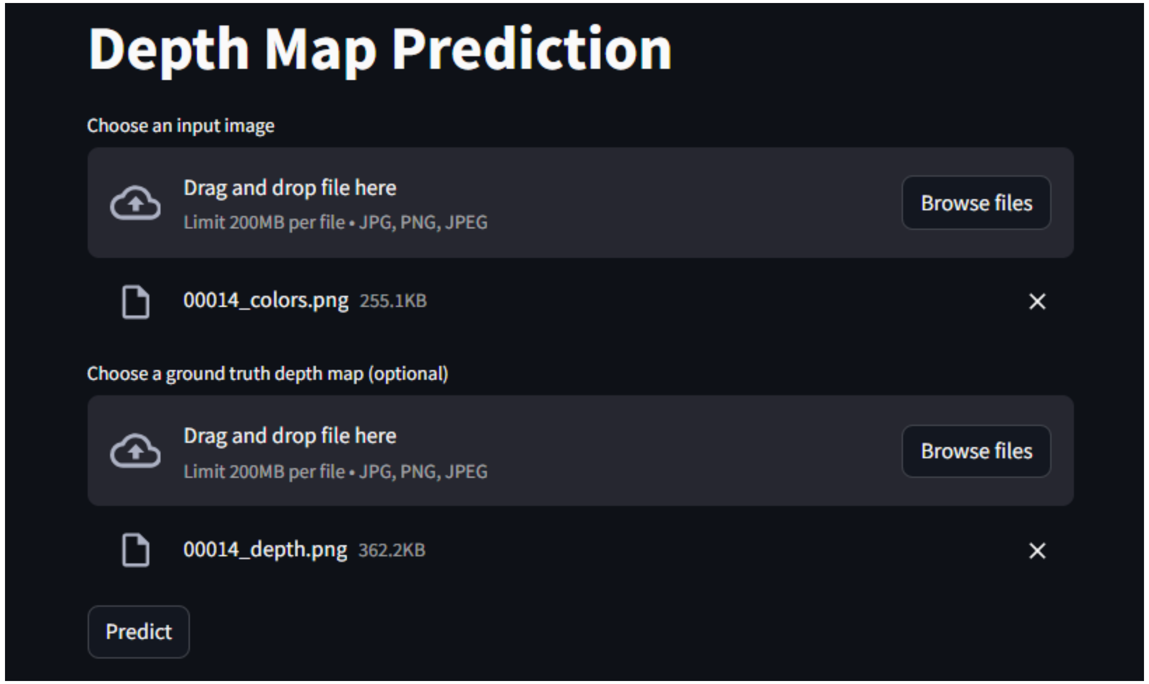


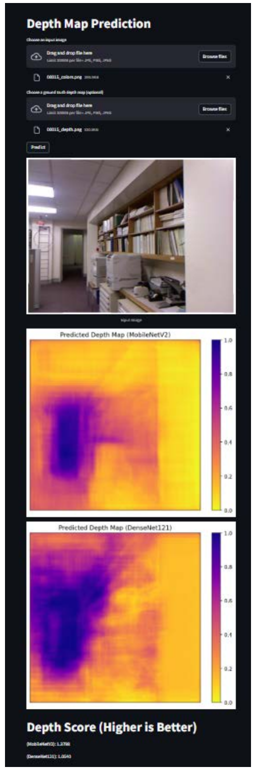
WEBSITE


View recent browsing across windows and devices
localhost:8501
Welcome to Depth Map App
Sign in with Google

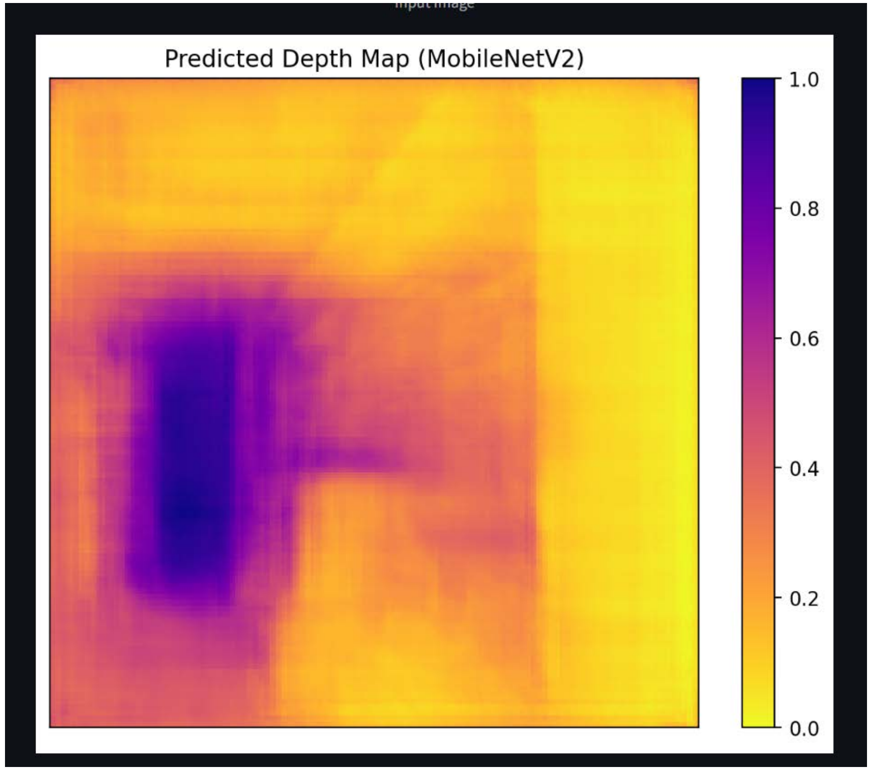

Sign in with Google
Choose an account
to continue to loca.lt
marshal wayne
marshalredwayne@gmail.com
grentam feiun
feiungrentam@gmail.com
Irfan Ali Sadab 2011054642
irfan.sadab@northsouth.edu
Use another account
To continue, Google will share your name, email address, language preference, and profile picture with loca.lt.
English (United States)
Help
Privacy
Terms

WEBSITE


Depth Map Prediction
Choose an input image
Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG
Browse files
00014_colors.png 255.1KB
Choose a ground truth depth map (optional)
Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG
Browse files
00014_depth.png 362.2KB
Predict


Depth Map Prediction
Predicted Depth Map (MobileNetV2)
Predicted Depth Map (DenseNet121)
Depth Score (Higher is Better)

WEBSITTE



Depth Score (Higher is Better)

(MobileNetV2): 1.3798

(MovileNetV2(Attention Mechanism): 1.0540

FUTURE PLAN

- future plans for monocular depth estimation:
  - Model Architecture Enhancements.
  - Loss Function Exploration..
  - Advanced Data Augmentation.
  - Evaluation on Diverse Datasets.
  - Integration with Other Sensor Modalities.
  - Integration into Practical Applications.
  - Integration with Other Sensor Modalities.
  - Deployment on Edge Devices.

CONCLUSION:

- While our current implementation is not perfect for commercial use yet, but demonstrates the feasibility and potential of future in monocular depth estimation, there are numerous avenues for future research and improvement. As outlined in our future plans, we aim to explore more advanced neural network architectures, alternative loss functions, and techniques to incorporate temporal information and unsupervised learning.

THANK YOU FOR YOUR PRECIOUS TIME, FROM ENIGMA