Instruction for Reproducing the Results of MapMender: *A Multi-Stage Hierarchical Aggregation Model for Localized Damage Restoration of Ancient Maps Using Deep Learning*

ARTICLE HISTORY
Compiled July 23, 2025

This tutorial, provides a detailed step-by-step instruction for replicating the image quality evaluation process described in this study. We will guide you through the process of setting up the environment and running the MapMender code to reproduce the experimental results.

1. Purpose

The provided Python script is designed to systematically evaluate the quality of predicted ancient maps by comparing them with reference (target) maps. The script computes six common image similarity metrics:

(1) Mean Squared Error (MSE)
(2) Root Mean Squared Error (RMSE)
(3) Normalized Mean Squared Error (NMSE)
(4) Mean Absolute Error (MAE)
(5) Peak Signal-to-Noise Ratio (PSNR)
(6) Structural Similarity Index (SSIM)

2. Basic Environment Setup

We'll walk you through every step, from installing dependencies to configuring paper's results.

The hardware used to complete the experiment is as follows:

- **CPU**: Intel Platinum 8358
- **GPU**: NVIDIA $A800$ $80GB$ PCIe

The software environment is configured as follows:

- **Operating System**: Ubuntu 22.04.4 LTS
- **CUDA Version**: 12.4

We utilize Python for programming, along with the PyTorch framework. Anaconda is used to create and manage the Python environment.

To begin, you'll need to create and activate the Python environment. Run the following commands in your terminal:

```
conda create -n MapMender python=3.10.8
conda activate MapMender
```

3. Installing Dependencies

Once the environment is activated, you'll need to install the required dependencies. These include essential libraries such as PyTorch (for deep learning), OpenCV (for image processing), and other supporting packages. Use the following commands to install the dependencies:

```
pip install torch==1.12.1+cu116 torchvision==0.13.1+cu116
torchaudio==0.12.1 --extra-index-url
https://download.pytorch.org/whl/cu116
pip install tqdm==4.67.1
pip install scikit-image==0.25.2
pip install opencv-python==4.11.0.86
pip install pillow==11.1.0
pip install numpy==1.24.0
```

4. Downloading the Code and Dataset

To replicate the experimental results, you'll need to download the code and dataset. The code is available on GitHub, and the dataset can be accessed through Figshare:

- MapMender GitHub Repository: `https://github.com/giserhh/MapMender`
- MapMender Dataset: `https://doi.org/10.6084/m9.figshare.29485238`

After downloading the dataset, place it into the appropriate directories within the project folder: test_data and results. This will complete the initial setup and configuration.

Our MapMender project is structured as follows:

```
MapMender/
   MapMender.py  # Script for evaluating MapMender
   requirements.txt      # Project dependencies
   results/         # MapMender restoration results
   baseline/        # baseline model restoration results
   test_data/       # Test datasets
```

Our experiment explores the localized damage restoration of ancient maps from four perspectives: shape, area, degree, and location. In the following sections, I will demonstrate how to reproduce the results for each of these aspects.

5. Shape(Reproduce the results of Table 2 and Table 3)

5.1. Reproduce the results of Table 2

5.1.1. MapMender's results of shape

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/shape/all"
results_root_folder = r"results/shape/all"
```

After the modification, the corresponding result is shown in the Figure 1:

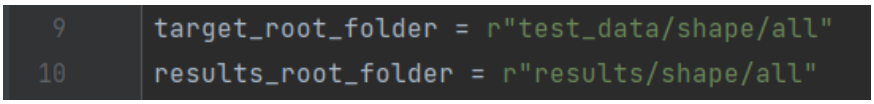


Figure 1. MapMender's path in shape(The green text in the figure indicates the content that needs to be modified.)

After the modification, we run the following command:

```
python MapMender.py
```

The output is shown in the Figure 2.

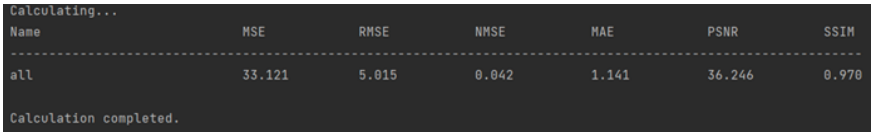


Figure 2. MapMender's output in shape

This is the performance result of MapMender for localized damage restoration on the shape.

5.1.2. Baseline's results of shape

RESCAN's results of shape

We change the path to the RESCAN baseline method's path, which is:

```
'baseline/shape/RESCAN'
```

The modified result is shown in the Figure 3.

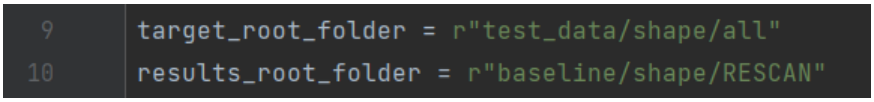


Figure 3. RESCAN's path in shape

After the modification, we run the following command:

```
python MapMender.py
```

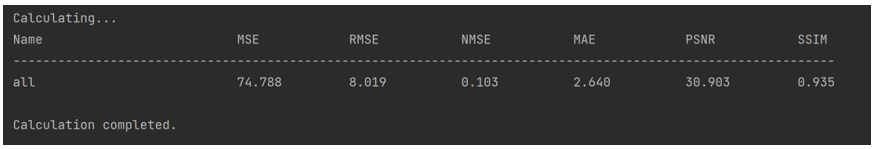


Figure 4. RESCAN's output in shape

The results are shown in the Figure 4, which are the metrics results of the RESCAN baseline model in localized damage restoration on shape.

JORDER-E's results of shape

We change the path to the JORDER-E baseline method's path, which is:

```
'baseline/shape/JORDER-E'
```

The modified result is shown in the Figure 5.

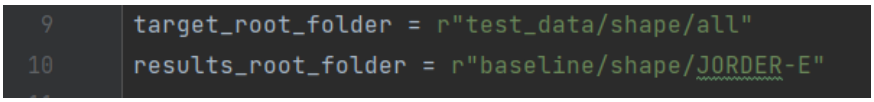


Figure 5. JORDER-E's path in shape

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 6, which are the metrics results of the JORDER-E baseline model in localized damage restoration on shape.

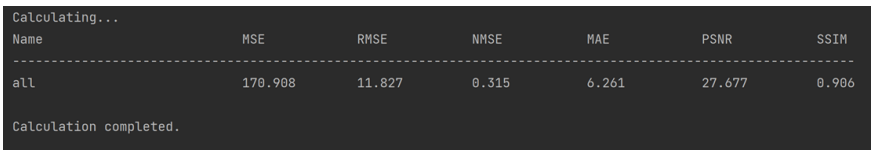


Figure 6. JORDER-E's output in shape

MPRNet's results of shape

We change the path to the MPRNet baseline method's path, which is:

```
'baseline/shape/MPRNet'
```

The modified result is shown in the Figure 7.

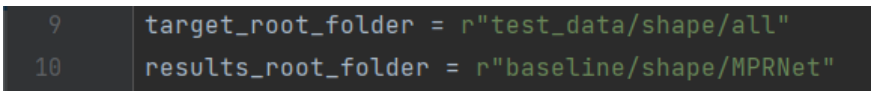


Figure 7. MPRNet's path in shape

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 8, which are the metrics results of the MPRNet baseline model in localized damage restoration on shape.

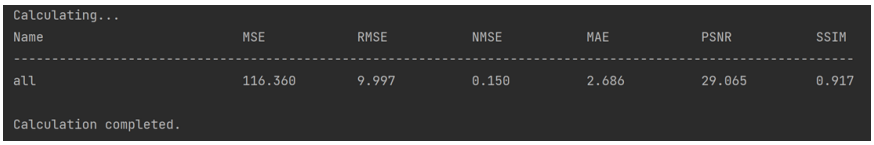


Figure 8. MPRNet's output in shape

DGUNet's results of shape

We change the path to the DGUNet baseline method's path, which is:

```
'baseline/shape/DGUNet'
```

The modified result is shown in the Figure 9.

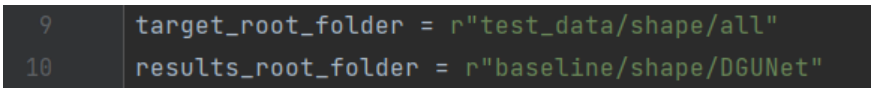


Figure 9. DGUNet's path in shape

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 10, which are the metrics results of the DGUNet baseline model in localized damage restoration on shape.

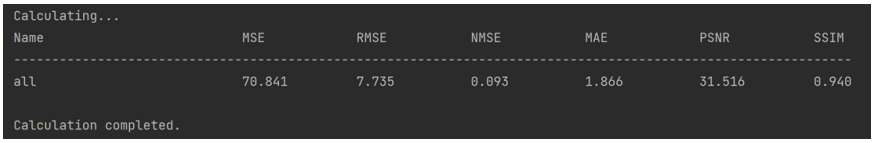


Figure 10. DGUNet's output in shape

NeRD's results of shape

We change the path to the NeRD baseline method's path, which is:

```
'baseline/shape/NeRD'
```

The modified result is shown in the Figure 11.

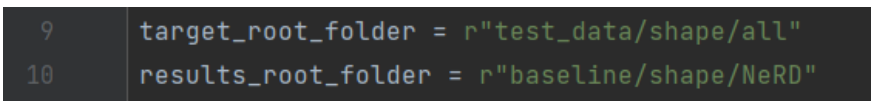


Figure 11. NeRD's path in shape

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 12, which are the metrics results of the NeRD baseline model in localized damage restoration on shape.

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
all                     90.133       8.748        0.148        3.796        30.227       0.937

Calculation completed.
```

Figure 12. NeRD's output in shape

5.1.3. Construct Table 2

By synthesizing the previous experimental results in Figure 13 into a single table, Table 2 can be obtained.

Table 2. Restoration results by damage shape (The best results are bold and the second best results are underlined).

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
RESCAN [a]	74.788	8.019	0.103	2.640	30.903	0.935
JORDER-E [b]	170.908	11.827	0.315	6.261	27.677	0.906
MPRNet [c]	116.360	9.997	0.150	2.686	29.065	0.917
DGUNet [d]	<u>70.841</u>	<u>7.735</u>	<u>0.093</u>	<u>1.866</u>	<u>31.516</u>	<u>0.940</u>
NeRD [e]	90.133	8.748	0.148	3.796	30.227	0.937
MapMender (ours)	**33.121**	**5.015**	**0.042**	**1.141**	**36.246**	**0.970**

Figure 13. Table 2 in the original manuscript

5.2. Reproduce the results of Table 3

5.2.1. MapMender's results of detail shape class

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/shape/detail"
results_root_folder = r"results/shape/detail"
```

After the modification, the corresponding results are shown in the Figure 14.

```
9   target_root_folder = r"test_data/shape/detail"
10  results_root_folder = r"results/shape/detail"
```

Figure 14. MapMender's path in shape

After the modification, we run the following command:

```
python MapMender.py
```

The running results are shown in the Figure 15.

5.2.2. Construct Table 3

By incorporating the previous experimental results into a table in Figure 16, Table 3 can be obtained.

```
Calculating...
Name                    MSE       RMSE      NMSE      MAE       PSNR      SSIM
-------------------------------------------------------------------------------
Concentric annulus      7.870     2.367     0.009     0.306     42.840    0.992
Discrete small area     23.883    4.401     0.030     0.884     36.662    0.978
Irregular annular       15.114    3.598     0.019     0.595     37.866    0.986
Irregular banding       39.086    5.740     0.049     1.305     33.964    0.966
Palm print              53.239    6.806     0.067     1.778     32.227    0.951
Rectangle               60.478    7.264     0.077     2.015     31.662    0.943
Regular banding         20.682    3.957     0.026     0.670     40.060    0.983
Reticulate interlaced   21.762    4.064     0.027     0.851     38.141    0.979
Rounded edge            66.234    7.590     0.085     2.150     31.334    0.942
Scattered droplet       38.246    5.656     0.048     1.252     34.386    0.967
Serrated cluster        54.089    6.787     0.068     1.791     32.390    0.952
Sputtering              38.978    5.741     0.051     1.319     34.030    0.966
```

Figure 15. MapMender's putput in shape

Table 3. MapMender restoration results for various damage shapes

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
Serrated cluster	54.089	6.787	0.068	1.791	32.390	0.952
Rounded edge	66.234	7.590	0.085	2.150	31.334	0.942
Discrete Small Area	23.883	4.401	0.030	0.884	36.662	0.978
Reticulate interlaced	21.762	4.064	0.027	0.851	38.141	0.979
Irregular banding	39.086	5.740	0.049	1.305	33.964	0.966
Regular banding	20.682	3.957	0.026	0.670	40.060	0.983
Scattered Droplet	38.246	5.656	0.048	1.252	34.386	0.967
Sputtering	38.978	5.741	0.051	1.319	34.030	0.966
Irregular Annular	15.114	3.598	0.019	0.595	37.866	0.986
Rectangle	60.478	7.264	0.077	2.015	31.662	0.943
Concentric annulus	7.870	2.367	0.009	0.306	42.840	0.992
Palm print	53.239	6.806	0.067	1.778	32.227	0.951

Figure 16. Table 3 in the original manuscript

6. Area(Reproduce the results of Table 4 and Table 5)

6.1. Reproduce the results of Table 4

6.1.1. MapMender's results of area

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/area/all"
results_root_folder = r"results/area/all"
```

After the modification, the corresponding result is shown in the Figure 17:

```
9   target_root_folder = r"test_data/area/all"
10  results_root_folder = r"results/area/all"
```

Figure 17. MapMender's path in area

After the modification, we run the following command:

```
python MapMender.py
```

The output is shown in the Figure 18.

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
all                     80.907       7.952        0.103        2.621        32.014       0.930

Calculation completed.
```

Figure 18. MapMender's output in area

This is the performance result of MapMender for localized damage restoration on the area.

6.1.2. Baseline's results of area

RESCAN's results of area

We change the path to the RESCAN baseline method's path, which is:

```
'baseline/area/RESCAN'
```

The modified result is shown in the Figure 19.

```
9   target_root_folder = r"test_data/area/all"
10  results_root_folder = r"baseline/area/RESCAN"
```

Figure 19. RESCAN's path in area

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 20, which are the metrics results of the RESCAN baseline model in localized damage restoration on area.

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
RESCAN                  137.703      10.689       0.186        4.278        28.705       0.898

Calculation completed.
```

Figure 20. RESCAN's output in area

JORDER-E's results of area

We change the path to the JORDER-E baseline method's path, which is:

```
'baseline/area/JORDER-E'
```

The modified result is shown in the Figure 21.

```
9   target_root_folder = r"test_data/area/all"
10  results_root_folder = r"baseline/area/JORDER-E"
```

Figure 21. JORDER-E's path in area

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 22, which are the metrics results of the JORDER-E baseline model in localized damage restoration on area.

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
JORDER-E                241.709      14.248       0.410        7.782        26.097       0.872

Calculation completed.
```

Figure 22. JORDER-E's output in area

MPRNet's results of area

We change the path to the MPRNet baseline method's path, which is:

```
'baseline/area/MPRNet'
```

The modified result is shown in the Figure 23.

```
9   target_root_folder = r"test_data/area/all"
10  results_root_folder = r"baseline/area/MPRNet"
```

Figure 23. MPRNet's path in area

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 24, which are the metrics results of the MPRNet baseline model in localized damage restoration on area.

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
MPRNet                  194.072      12.663       0.243        4.584        27.345       0.870

Calculation completed.
```

Figure 24. MPRNet's output in area

DGUNet's results of area

We change the path to the DGUNet baseline method's path, which is:

```
'baseline/area/DGUNet'
```

The modified result is shown in the Figure 25.

```
9   target_root_folder = r"test_data/area/all"
10  results_root_folder = r"baseline/area/DGUNet"
```

Figure 25. DGUNet's path in area

After the modification, we run the following command:

```
python MapMender.py
```

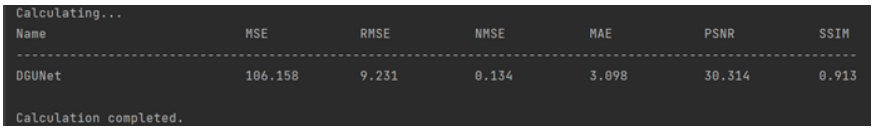


Figure 26. DGUNet's output in area

The results are shown in the Figure 26, which are the metrics results of the DGUNet baseline model in localized damage restoration on area.

NeRD's results of area

We change the path to the NeRD baseline method's path, which is:

```
'baseline/area/NeRD'
```

The modified result is shown in the Figure 27.

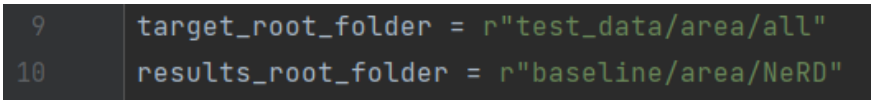


Figure 27. NeRD's path in area

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 28, which are the metrics results of the NeRD baseline model in localized damage restoration on area.

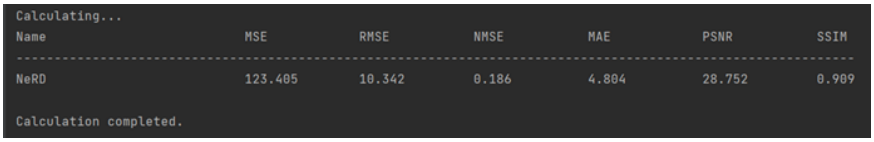


Figure 28. NeRD's output in area

6.1.3. Construct Table 4

By synthesizing the previous experimental results into a single table in Figure 29, Table 4 can be obtained.

Table 4. Restoration results by damage area (The best results are bold and the second best results are underlined).

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
RESCAN	137.703	10.689	0.186	4.278	28.705	0.898
JORDER-E	241.709	14.248	0.410	7.782	26.097	0.872
MPRNet	194.072	12.663	0.243	4.584	27.345	0.870
DGUNet	<u>106.158</u>	<u>9.231</u>	<u>0.134</u>	<u>3.098</u>	<u>30.314</u>	<u>0.913</u>
NeRD	123.405	10.342	0.186	4.804	28.752	0.909
MapMender (ours)	**80.907**	**7.952**	**0.103**	**2.621**	**32.014**	**0.930**

Figure 29. Table 4 in the original manuscript

6.2. Reproduce the results of Table 5

6.2.1. MapMender's results of detail area class

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/area/detail"
results_root_folder = r"results/area/detail"
```

After the modification, the corresponding results are shown in the Figure 30.

```
9   target_root_folder = r"test_data/area/detail"
10  results_root_folder = r"results/area/detail"
```

Figure 30. MapMender's path in area

After the modification, we run the following command:

```
python MapMender.py
```

The running results are shown in the Figure 31.

```
Calculating...
Name                MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------
Colossal            146.219      11.608       0.187        4.675        27.252       0.878
Giant               108.369      9.965        0.140        3.465        28.628       0.907
Huge                84.679       8.781        0.108        2.817        29.768       0.923
Large               65.701       7.676        0.083        2.193        31.040       0.940
Massive             124.925      10.649       0.157        4.019        28.105       0.893
Medium              52.876       6.895        0.067        1.720        31.980       0.952
Mega                167.232      12.467       0.218        5.333        26.585       0.863
Mini                19.274       4.040        0.024        0.642        37.267       0.982
Small               34.127       5.483        0.043        1.149        34.156       0.968
Tiny                5.672        1.952        0.007        0.194        45.362       0.995

Calculation completed.
```

Figure 31. MapMender's putput in area

6.2.2. Construct Table 5

By incorporating the previous experimental results into a table in Figure 32, Table5 can be obtained.

7. Degree(Reproduce the results of Table 6 and Table 7)

7.1. Reproduce the results of Table 6

7.1.1. MapMender's results of degree

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/degree/all"
results_root_folder = r"results/degree/all"
```

After the modification, the corresponding result is shown in the Figure 33:
After the modification, we run the following command:

Table 5. MapMender restoration results for various damage areas.

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
Tiny	5.672	1.952	0.007	0.194	45.362	0.995
Mini	19.274	4.040	0.024	0.642	37.267	0.982
Small	34.127	5.483	0.043	1.149	34.156	0.968
Medium	52.876	6.895	0.067	1.720	31.980	0.952
Large	65.701	7.676	0.083	2.193	31.040	0.940
Huge	84.679	8.781	0.108	2.817	29.768	0.923
Giant	108.369	9.965	0.140	3.465	28.628	0.907
Massive	124.925	10.649	0.157	4.019	28.105	0.893
Colossal	146.219	11.608	0.187	4.675	27.252	0.878
Mega	167.232	12.467	0.218	5.333	26.585	0.863

Figure 32. Table 5 in the original manuscript

```
9   target_root_folder = r"test_data/degree/all"
10  results_root_folder = r"results/degree/all"
```

Figure 33. MapMender's path in degree(The green text in the figure indicates the content that needs to be modified.)

```
python MapMender.py
```

The output is shown in the Figure 34.

```
Calculating...
Name                    MSE          RMSE          NMSE          MAE          PSNR          SSIM
-------------------------------------------------------------------------------------------------
all                     47.548        6.231         0.060         1.602        33.260        0.957

Calculation completed.
```

Figure 34. MapMender's output in degree

This is the performance result of MapMender for localized damage restoration on the degree.

7.1.2. Baseline's results of degree

RESCAN's results of degree

We change the path to the RESCAN baseline method's path, which is:

```
'baseline/degree/RESCAN'
```

The modified result is shown in the Figure 35.

```
9   target_root_folder = r"test_data/degree/all"
10  results_root_folder = r"baseline/degree/RESCAN"
```

Figure 35. RESCAN's path in degree

After the modification, we run the following command:

```
python MapMender.py
```

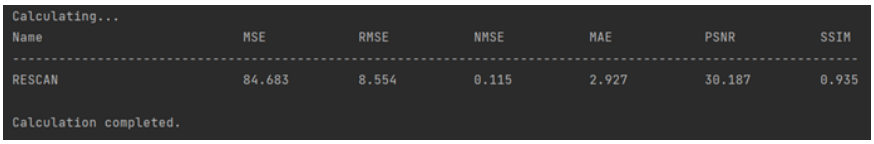


Figure 36. RESCAN's output in degree

The results are shown in the Figure 36, which are the metrics results of the RESCAN baseline model in localized damage restoration on degree.

JORDER-E's results of degree

We change the path to the JORDER-E baseline method's path, which is:

```
'baseline/degree/JORDER-E'
```

The modified result is shown in the Figure 37.

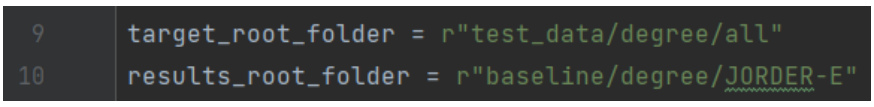


Figure 37. JORDER-E's path in degree

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 38, which are the metrics results of the JORDER-E baseline model in localized damage restoration on degree.

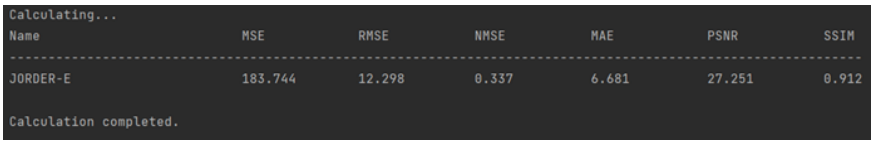


Figure 38. JORDER-E's output in degree

MPRNet's results of degree

We change the path to the MPRNet baseline method's path, which is:

```
'baseline/degree/MPRNet'
```

The modified result is shown in the Figure 39.

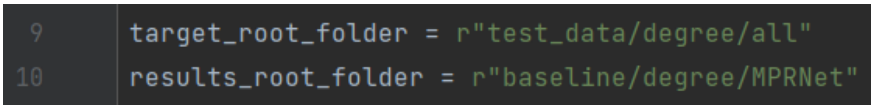


Figure 39. MPRNet's path in degree

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 40, which are the metrics results of the MPRNet baseline model in localized damage restoration on degree.

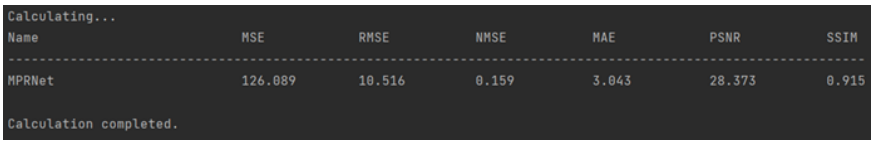


Figure 40. MPRNet's output in degree

DGUNet's results of degree

We change the path to the DGUNet baseline method's path, which is:

```
'baseline/degree/DGUNet'
```

The modified result is shown in the Figure 41.

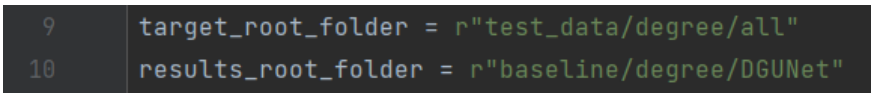


Figure 41. DGUNet's path in degree

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 42, which are the metrics results of the DGUNet baseline model in localized damage restoration on degree.

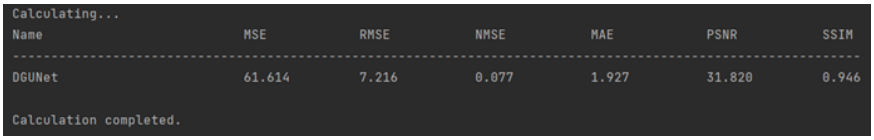


Figure 42. DGUNet's output in degree

NeRD's results of degree

We change the path to the NeRD baseline method's path, which is:

```
'baseline/degree/NeRD'
```

The modified result is shown in the Figure 43.

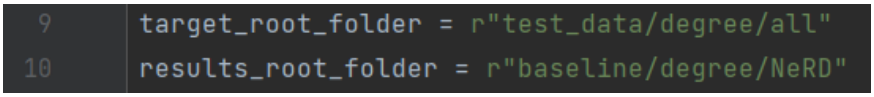


Figure 43. NeRD's path in degree

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 44, which are the metrics results of the NeRD baseline model in localized damage restoration on degree.

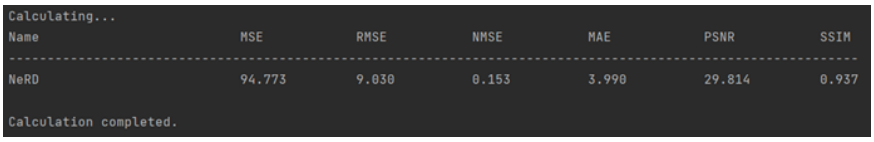


Figure 44. NeRD's output in degree

7.1.3. Construct Table 6

By synthesizing the previous experimental results into a single table in Figure 45, Table 6 can be obtained.

Table 6. Restoration results by damage degree (The best results are bold and the second best results are underlined).

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
RESCAN	84.683	8.554	0.115	2.927	30.187	0.935
JORDER-E	183.744	12.298	0.337	6.681	27.251	0.912
MPRNet	126.089	10.516	0.159	3.043	28.373	0.915
DGUNet	<u>61.614</u>	<u>7.216</u>	<u>0.077</u>	<u>1.927</u>	<u>31.820</u>	<u>0.946</u>
NeRD	94.773	9.030	0.153	3.990	29.814	0.937
MapMender(ours)	**47.548**	**6.231**	**0.060**	**1.602**	**33.260**	**0.957**

Figure 45. Table 6 in the original manuscript

7.2. Reproduce the results of Table 7

7.2.1. MapMender's results of detail degree class

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/degree/detail"
results_root_folder = r"results/degree/detail"
```

After the modification, the corresponding results are shown in the Figure 46.

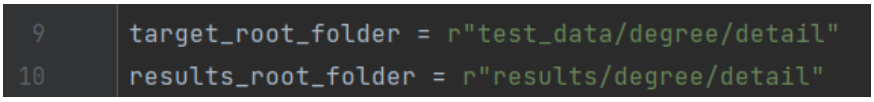


Figure 46. MapMender's path in degree

After the modification, we run the following command:

```
python MapMender.py
```

The running results are shown in the Figure 47.

7.2.2. Construct Table 7

By incorporating the previous experimental results into a table in Figure 48, Table 7 can be obtained.

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
Catastrophic            102.202      9.468        0.125        2.324        29.494       0.927
Destructive             52.876       6.895        0.067        1.720        31.980       0.952
Moderate                26.082       4.863        0.034        1.311        34.900       0.969
Severe                  38.102       5.871        0.049        1.511        33.333       0.960
Slight                  18.476       4.057        0.024        1.145        36.595       0.975

Calculation completed.
```

Figure 47. MapMender's putput in degree

Table 7. MapMender restoration results for various damage degrees.

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
Slight	18.476	4.057	0.024	1.145	36.595	0.975
Moderate	26.082	4.863	0.034	1.311	34.900	0.969
Severe	38.102	5.871	0.049	1.511	33.333	0.960
Destructive	52.876	6.895	0.067	1.720	31.980	0.952
Catastrophic	102.202	9.468	0.125	2.324	29.494	0.927

Figure 48. Table 7 in the original manuscript

8. Area-Degree(Replicate Figure 10)

8.1. MapMender's Results of Area-Degree

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/area-degree"
results_root_folder = r"results/area-degree"
```

After the modification, the corresponding results are shown in the Figure 49.

```
9   target_root_folder = r"test_data/area-degree"
10  results_root_folder = r"results/area-degree"
```

Figure 49. MapMender's path in area-degree

After the modification, we run the following command:

```
python MapMender.py
```

The output is shown in the Figure 50.

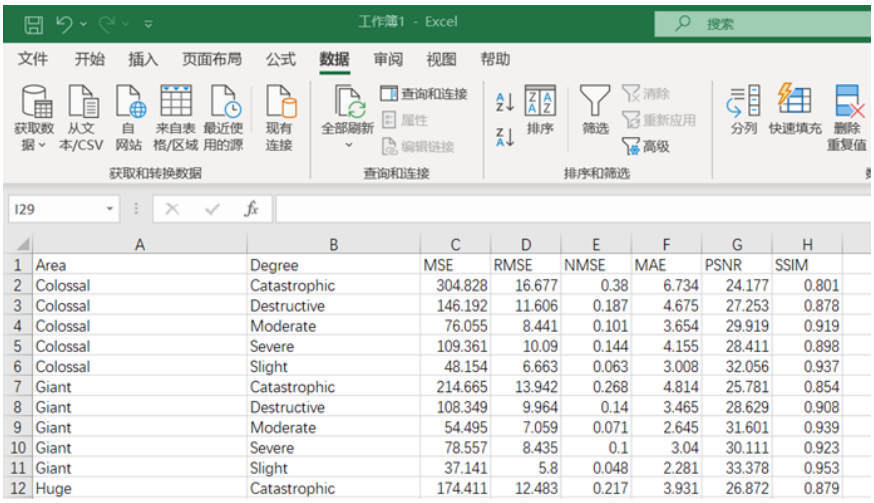
8.2. Excel

Import the metrics from this result into Excel for column-wise organization to obtain the table shown in the Figure 51:

Replace the texts of "area" and "degree" with the corresponding damaged area class and damage degree class in Figure 52.

```
Calculating...
Name                          MSE        RMSE       NMSE       MAE        PSNR       SSIM
--------------------------------------------------------------------------------------------
Colossal_Catastrophic         304.828    16.677     0.380      6.734      24.177     0.801
Colossal_Destructive          146.192    11.606     0.187      4.675      27.253     0.878
Colossal_Moderate             76.055     8.441      0.101      3.654      29.919     0.919
Colossal_Severe               109.361    10.090     0.144      4.155      28.411     0.898
Colossal_Slight               48.154     6.663      0.063      3.008      32.056     0.937
Giant_Catastrophic            214.665    13.942     0.268      4.814      25.781     0.854
Giant_Destructive             108.349    9.964      0.140      3.465      28.629     0.908
Giant_Moderate                54.495     7.059      0.071      2.645      31.601     0.939
Giant_Severe                  78.557     8.435      0.100      3.040      30.111     0.923
Giant_Slight                  37.141     5.800      0.048      2.281      33.378     0.953
Huge_Catastrophic             174.411    12.483     0.217      3.931      26.872     0.879
Huge_Destructive              84.668     8.780      0.108      2.817      29.768     0.923
Huge_Moderate                 43.703     6.317      0.057      2.163      32.601     0.948
Huge_Severe                   64.958     7.695      0.084      2.531      30.902     0.934
Huge_Slight                   30.451     5.233      0.039      1.886      34.301     0.959
Large_Catastrophic            141.906    11.219     0.175      3.197      27.903     0.901
Large_Destructive             65.692     7.676      0.083      2.193      31.041     0.940
Large_Moderate                35.187     5.634      0.045      1.746      33.665     0.958
Large_Severe                  53.899     6.979      0.069      2.049      31.848     0.946
Large_Slight                  23.404     4.608      0.030      1.486      35.358     0.968
Massive_Catastrophic          256.091    15.212     0.317      5.729      25.063     0.830
```

Figure 50. MapMender's output in area-degree



	A	B	C	D	E	F	G	H
1	Area	Degree	MSE	RMSE	NMSE	MAE	PSNR	SSIM
2	Colossal	Catastrophic	304.828	16.677	0.38	6.734	24.177	0.801
3	Colossal	Destructive	146.192	11.606	0.187	4.675	27.253	0.878
4	Colossal	Moderate	76.055	8.441	0.101	3.654	29.919	0.919
5	Colossal	Severe	109.361	10.09	0.144	4.155	28.411	0.898
6	Colossal	Slight	48.154	6.663	0.063	3.008	32.056	0.937
7	Giant	Catastrophic	214.665	13.942	0.268	4.814	25.781	0.854
8	Giant	Destructive	108.349	9.964	0.14	3.465	28.629	0.908
9	Giant	Moderate	54.495	7.059	0.071	2.645	31.601	0.939
10	Giant	Severe	78.557	8.435	0.1	3.04	30.111	0.923
11	Giant	Slight	37.141	5.8	0.048	2.281	33.378	0.953
12	Huge	Catastrophic	174.411	12.483	0.217	3.931	26.872	0.879

Figure 51. Area-degree's excel

8.3. Visualization

Import the modified Excel table into Origin for visualization. As shown in the Figure 53, perform 3D visualization for each metric to obtain paper's Figure 10.

9. Location(Reproduce the results of Table 8 and Table 9)

9.1. Reproduce the results of Table 8

9.1.1. MapMender's results of location

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/location/all"
```

序号	area	name
1	0< ≤0.05	Tiny
2	0.05< ≤0.10	Mini
3	0.10< ≤0.15	Small
4	0.15< ≤0.20	Medium
5	0.20< ≤0.25	Large
6	0.25< ≤0.30	Huge
7	0.30< ≤0.35	Giant
8	0.35< ≤0.40	Massive
9	0.40< ≤0.45	Colossal
10	0.45< ≤0.50	Mega

(a) Area class.

序号	degree	name
1	0.05	Catastrophic
2	0.20	Destructive
3	0.25	Severe
4	0.35	Moderate
5	0.50	Slight

(b) Degree class.

Figure 52. Damaged area class and damage degree class.

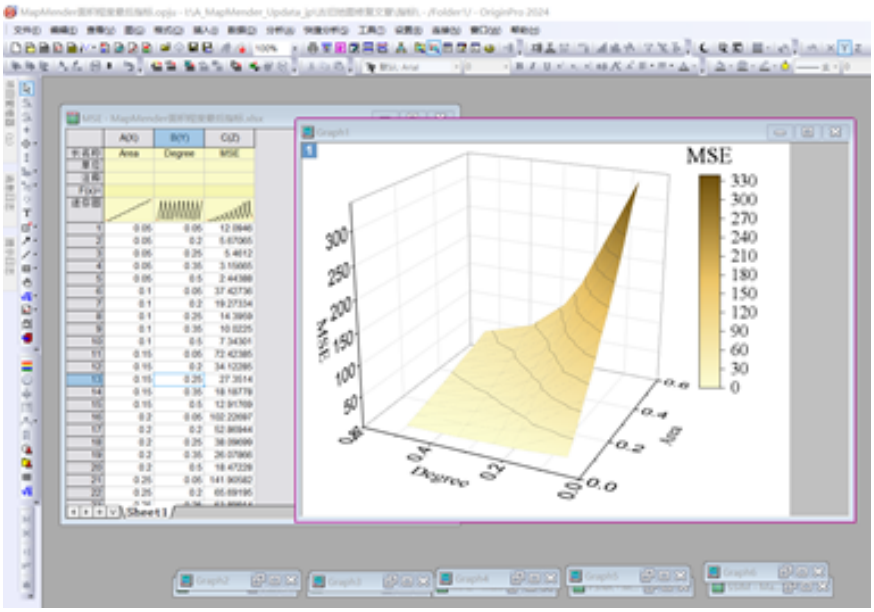


Figure 53. 3D metrics' visualization

```
results_root_folder = r"results/location/all"
```

After the modification, the corresponding result is shown in the Figure 54:

```
9   target_root_folder = r"test_data/location/all"
10  results_root_folder = r"results/location/all"
```

Figure 54. MapMender's path in location(The green text in the figure indicates the content that needs to be modified.)

After the modification, we run the following command:

```
python MapMender.py
```

The output is shown in the Figure 55.

```
Calculating...
Name                    MSE         RMSE        NMSE        MAE         PSNR        SSIM
-------------------------------------------------------------------------------------------
all                     57.084      7.181       0.072       1.802       31.679      0.955

Calculation completed.
```

Figure 55. MapMender's output in location

This is the performance result of MapMender for localized damage restoration on the location.

9.1.2. Baseline's results of location

RESCAN's results of location

We change the path to the RESCAN baseline method's path, which is:

```
'baseline/location/RESCAN'
```

The modified result is shown in the Figure 56.

```
9   target_root_folder = r"test_data/location/all"
10  results_root_folder = r"baseline/location/RESCAN"
```

Figure 56. RESCAN's path in location

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 57, which are the metrics results of the RESCAN baseline model in localized damage restoration on location.

```
Calculating...
Name                    MSE         RMSE        NMSE        MAE         PSNR        SSIM
-------------------------------------------------------------------------------------------
RESCAN                  92.492      9.227       0.120       2.894       29.262      0.930

Calculation completed.
```

Figure 57. RESCAN's output in location

JORDER-E's results of location

We change the path to the JORDER-E baseline method's path, which is:

```
'baseline/location/JORDER-E'
```

The modified result is shown in the Figure 58.

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 59, which are the metrics results of the JORDER-E baseline model in localized damage restoration on location.

MPRNet's results of location

We change the path to the MPRNet baseline method's path, which is:

```
target_root_folder = r"test_data/location/all"
results_root_folder = r"baseline/location/JORDER-E"
```

Figure 58. JORDER-E's path in location

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
JORDER-E                224.013      14.202       0.378        7.430        25.547       0.898

Calculation completed.
```

Figure 59. JORDER-E's output in location

`'baseline/location/MPRNet'`

The modified result is shown in the Figure 60.

```
target_root_folder = r"test_data/location/all"
results_root_folder = r"baseline/location/MPRNet"
```

Figure 60. MPRNet's path in location

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 61, which are the metrics results of the MPRNet baseline model in localized damage restoration on location.

```
Calculating...
Name                    MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------------
MPRNet                  145.602      11.529       0.182        3.193        27.506       0.908

Calculation completed.
```

Figure 61. MPRNet's output in location

DGUNet's results of location

We change the path to the DGUNet baseline method's path, which is:

`'baseline/location/DGUNet'`

The modified result is shown in the Figure 62.

```
target_root_folder = r"test_data/location/all"
results_root_folder = r"baseline/location/DGUNet"
```

Figure 62. DGUNet's path in location

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 63, which are the metrics results of the DGUNet baseline model in localized damage restoration on location.

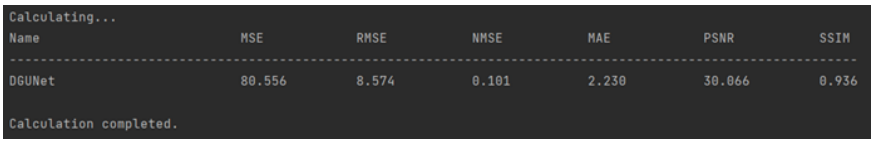


Figure 63. DGUNet's output in location

NeRD's results of location

We change the path to the NeRD baseline method's path, which is:

```
'baseline/location/NeRD'
```

The modified result is shown in the Figure 64.

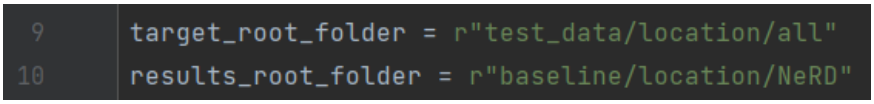


Figure 64. NeRD's path in location

After the modification, we run the following command:

```
python MapMender.py
```

The results are shown in the Figure 65, which are the metrics results of the NeRD baseline model in localized damage restoration on location.

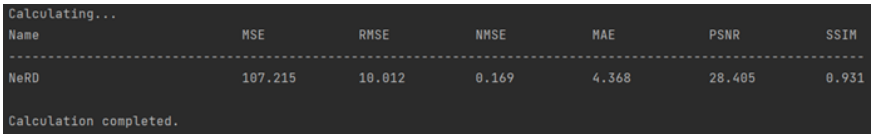


Figure 65. NeRD's output in location

9.1.3. Construct Table 8

By synthesizing the previous experimental results into a single table in Figure 66, Table 8 can be obtained.

Table 8. Restoration results by damage location (The best results are bold and the second best results are underlined).

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
RESCAN	92.492	9.227	0.120	2.894	29.262	0.930
JORDER-E	224.013	14.202	0.378	7.430	25.547	0.898
MPRNet	145.602	11.529	0.182	3.193	27.506	0.908
DGUNet	<u>80.556</u>	<u>8.574</u>	<u>0.101</u>	<u>2.230</u>	<u>30.066</u>	<u>0.936</u>
NeRD	107.215	10.012	0.169	4.368	28.405	0.931
MapMender(ours)	**57.084**	**7.181**	**0.072**	**1.802**	**31.679**	**0.955**

Figure 66. Table 8 in the original manuscript

9.2. Reproduce the results of Table 9

9.2.1. MapMender's results of detail location class

First, we need to modify the file paths. In the MapMender.py file, modify lines 9 and 10 as follows:

```
target_root_folder = r"test_data/location/detail"
results_root_folder = r"results/location/detail"
```

After the modification, the corresponding results are shown in the Figure 67.

```
9   target_root_folder = r"test_data/location/detail"
10  results_root_folder = r"results/location/detail"
```

Figure 67. MapMender's path in location

After the modification, we run the following command:

```
python MapMender.py
```

The running results are shown in the Figure 68.

```
Calculating...
Name                MSE          RMSE         NMSE         MAE          PSNR         SSIM
-------------------------------------------------------------------------------------------
building            54.799       7.246        0.086        1.707        31.119       0.959
greenland           58.754       7.512        0.066        2.021        30.798       0.944
river               70.979       7.596        0.074        1.880        32.584       0.956
road                39.878       6.157        0.062        1.474        32.613       0.958
text                61.010       7.391        0.071        1.929        31.280       0.957

Calculation completed.
```

Figure 68. MapMender's putput in location

9.2.2. Construct Table 9

By incorporating the previous experimental results into a table in Figure 69, Table 9 can be obtained.

Table 9. MapMender restoration results for various damage locations.

	MSE ↓	RMSE ↓	NMSE ↓	MAE ↓	PSNR ↑	SSIM ↑
River	70.979	7.596	0.074	1.880	32.584	0.956
Road	39.878	6.157	0.062	1.474	32.613	0.958
Greenland	58.754	7.512	0.066	2.021	30.798	0.944
Text	61.010	7.391	0.071	1.929	31.280	0.957
Building	54.799	7.246	0.086	1.707	31.119	0.959

Figure 69. Table 9 in the original manuscript

10. Other figures

The original images of Figure 6, Figure 7, Figure 8, Figure 9 and Figure 11 in the article can all be viewed and downloaded on the Figshare website:`https://doi.org/10.6084/m9.figshare.29485238`.

The storage path of Figure 6 is:

```
MapMender\test_data
```

```
MapMender\results\shape\detail\Palm print
MapMender\baseline\shape\RESCAN
# For different baseline models, simply replace RESCAN with them.
```

Find all the original images of Figure 7 under this path, as shown in the Figure 70.

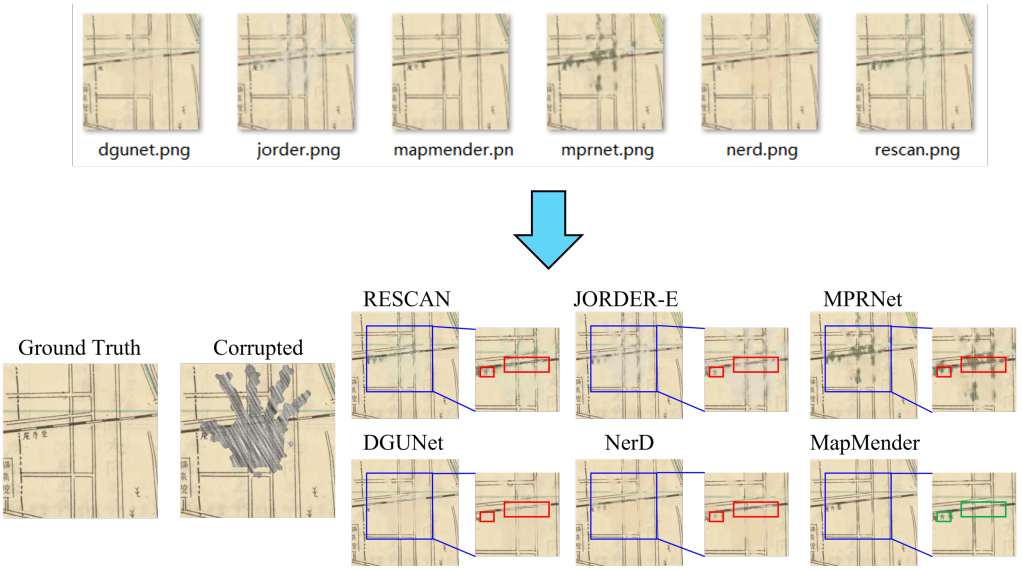


Figure 70. Figure 7 in the original manuscript

```
MapMender\results\area\detail
MapMender\baseline\area\RESCAN
# For different baseline models, simply replace RESCAN with them.
```

Find all the original images of Figure 8 under this path, as shown in the Figure 71

```
MapMender\results\degree\detail
MapMender\baseline\degree\RESCAN
# For different baseline models, simply replace RESCAN with them.
```

Find all the original images of Figure 9 under this path, as shown in the Figure 72

```
MapMender\results\location\detail
MapMender\baseline\location\RESCAN
# For different baseline models, simply replace RESCAN with them.
```

Find all the original images of Figure 11 under this path, as shown in the Figure 73

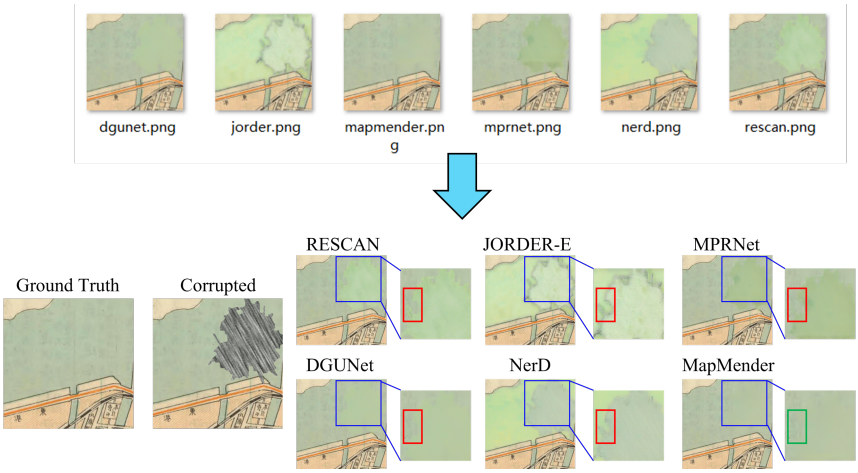


Figure 71. Figure 8 in the original manuscript

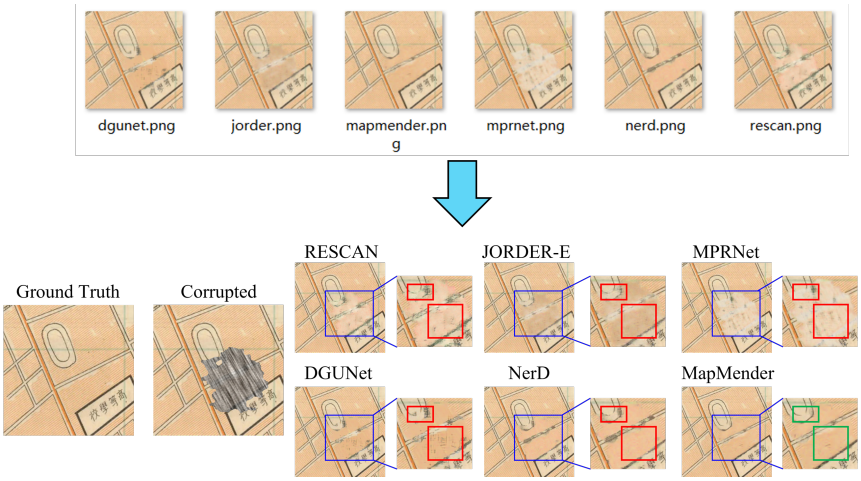


Figure 72. Figure 9 in the original manuscript

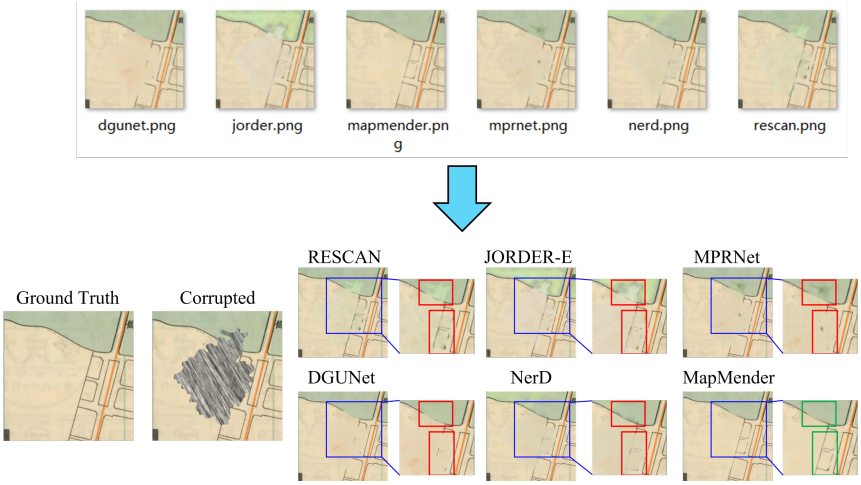


Figure 73. Figure 11 in the original manuscript