Appendix A

User Guide

A.1 MCEdit Download Instructions

To run the project and recreate its results, MCEdit [5] needs to be installed. First, the platform must be downloaded by using - *git clone –recursive https://github.com/mcgreentn/GDMC*. It can also be found at `https://github.com/mcgreentn/GDMC`. This command will download MCEdit [5]. All the required packages are part of the provided *requirements.txt* file in the main folder of the project described by this paper. They can be installed by following the instructions in the next section. The official guide can be found at `https://github.com/mcgreentn/GDMC/wiki/The-GDMC-Framework` or at `https://gendesignmc.wikidot.com/wiki:submission-mcedit`. Everything needed is covered in the next section, thus, it can be ignored. This is, however, with the exception of if the used operating system is not Windows. If it is MacOS, then the additional provided system-specific steps, under **Mac only section**, need to be followed.

A.2 Project & MCEdit Installation Instructions

All project and MCEdit [5] required libraries can be found under the *"requirements.txt"* file. They can be easily installed by following the described steps. Furthermore, the project has been made on Windows Operating System, thus, the described steps are for that particular system. Nonetheless, links to installation guides for other systems have been made available.

First, **Python 2.7** needs to be present on the machine. If the **virtualenv** package is not present, it is advisable that it is installed from `https://virtualenv.pypa.io/en/latest/installation.html` or by running - *py -2.7 -m pip install virtualenv*, for Windows. Other-

wise, an environment can be created and activated. After navigating to the projects' main folder, an environment can be created by running - *py -2.7 -m virtualenv venv*, for Windows, or check `https://virtualenv.pypa.io/en/latest/user_guide.html#introduction` for more information. After the environment has been created, one can activate it by running the Windows command - *venv\Scripts\activate*, or by looking at the main documentation at `https://virtualenv.pypa.io/en/latest/user_guide.html#activators`. The activated environment should be shown as in figure A.1, depending on the system.

Figure A.1: A Windows Command Line representation after activating the *venv* virtual environment.

The second step, after activating the virtual environment, is to install all required packages. The command - *pip install -r requirements.txt* can be used to easily install all listed libraries. Because the **Tensorflow Library** is deprecated for the current Python version, it must be installed separately from a wheel file. In the main project folder, there are three TensorFlow 1.10 **.whl** files for every type of system. Moreover, they can be downloaded from the following links:

- Tensorflow for Windows - `https://github.com/fo40225/tensorflow-windows-wheel/tree/master/1.10.0/py27/CPU/avx2`.
- Tensorflow for MacOS and Linux - `https://pypi.org/project/tensorflow/1.10.0/#files` and navigate to **Download files**.

To install the package, select the correct file for your system and substitute it in the following command - *pip install "tensorflow .whl filename"*. This should install the library into your virtual environment. An example of these commands can be seen in figure A.2.

(a)

(b)

Figure A.2: This figure shows the two commands needed to install all required project libraries. It must be noted that the second command b) installs the Window's TensorFlow version. On the other hand, command a) installs all listed libraries from the specified file.

Moreover, a complete list of all the used libraries can be found below:

- numpy==1.14.5
- keras==2.2.0
- scipy
- scikit-learn
- PyOpenGL
- pygame==1.9.4
- pyyaml
- pillow
- ftputil==3.4
- pypiwin32
- matplotlib
- opencv-contrib-python
- Augmentor
- Tensorflow==1.10.0

A.3 User Instructions

This section goes over how to run all the files present and how to replicate (relatively) the results covered in a previous chapter. Each of the following subsections describes how to run a specific file.

A.3.1 Running finalwgan.py

This file contains the design of the WGAN [8] network and can be used to train the model. This should be done in Google Colab [2] with its GPU accelerator utilised because that is what has been used in this paper. In addition, the previously created virtual environment will not suffice because the model has been trained using **Python 3**. Thus, Google Colab [2] provides all the necessary libraries to train the model. Moreover, the only path that the file needs is

the path to the training data. Currently, the given path is as *"train/"*, which states that the dataset is named as *"train"* and is located at the current root directory. To change the path to the dataset, the line of code in figure A.3 must be changed.

```
204   # Train WGAN
205   model = WGAN()
206   model.train("train/")  # Change the parameter depending on the path
```

Figure A.3: The figure shows the exact line of code that must be changed to accommodate different dataset paths.

The dataset can be acquired from `https://emckclac-my.sharepoint.com/:f:/g/personal/k1763856_kcl_ac_uk/ErB57uMLQTtPrcj7n33KzKEBUraXryiZRG3Kv1BghBKc4g?e=6XUCS4`. Here, it can be only downloaded and immediately used. Alternatively, it can be downloaded from `https://www.kaggle.com/balraj98/deepglobe-road-extraction-dataset?select=train`. However, only the mask images from the **train** folder are taken. Therefore, this method requires further work to be done before usage. The original dataset paper [4] can be found in the references.

At the end of the training process, a new model plot history will be saved, alongside the trained model as *"trained_model.h5"*. In addition, the training loss and accuracy will be shown for every epoch.

A.3.2 Running convert_model.py

In the Convert folder, a *"model.h5"* file can be found. It is an already trained model that is there for testing purposes. This file, as well as any other in the project folder, has to be run with the virtual environment activated. The *"convert_model.py"* file can be executed without changing any parameters. It will convert the aforementioned model to a **Python 2.7** version one. To change the path to the selected model for conversion, the line in figure A.4 must be changed with the new path. Moreover, the same figure A.4 shows which exact line to uncomment if the user would want to generate an image and test the converted model. It must be also noted that an image plot might not be generated, if testing, because of issues with the virtual environment. To fix that, from the original **Python 2.7** folder, the **tcl** folder needs to be copied and placed in the virtual environment directory.

```
78    convert_model("model.h5")
79    # test_converted_model()
```

Figure A.4: The first line in the image shows the position which must be changed if a different model path is to be used. On the other hand, the second line must not be commented if the user would like to test the model.

The converted model has a predefined filename (*"converted_model.h5"*). This newly converted model can be used to generate road images. The file can be run either from IDE or by executing the following command from the project's root directory - *py Convert\convert_model.py*.

A.3.3 Running convert_images.py

This file has been used for preprocessing the dataset before using it for training. The file has two adjustable parameters. The first one is *out_path = "out/"*, which specifies the output folder of the newly converted images. If not changed, the *"out"* folder must be created, as otherwise, the code will throw an error. The second parameter is *in_path = "train/"*, which specifies the path to the dataset. Again, if not changed, the dataset must be present in the current directory under the name *"train"*. The file can be run either from an IDE or from Command Line, using the Windows command from the project's root directory - *py Convert\convert_images.py*.

A.3.4 Running convert_images - Conditional.py

This file has been used when testing a Conditional WGAN [3]. It converts a dataset into a categorical one. There are two predefined classes (small and big roads). The conversion is done based on road coverage, as per chapter 3. In addition, augmentation of the data is present as a possible method to use. It also has the same two parameters as in the previous subsection. The lines that can be changed are shown in figure A.5. Furthermore, the file can be run either from an IDE or by executing the command - *py "Convert\convert_images - Conditional.py"*.

```
# 1st) Augment_data
# augment_data(in_path[:-1])  # Remove last \
# 2nd) Convert data
# Do only after moving output folder to main folder after augmentation
convert_data(in_path, out_path, labels)
```

Figure A.5: The figure shows the code lines that can be changed before running the file. The first non-commented code line can be uncommented to run augmentation on the data. The data will be automatically put into a different folder and must be transferred to the main dataset folder. The last code line can be edited to accommodate for different **"in"** and **"out"** paths.

A.3.5 Running MCEdit

After downloading and installing MCEdit [5], as per previous sections, one must know how to navigate and run filters on the platform. As it is a 3^{rd} party platform, the following guidelines can be followed - `https://github.com/mcgreentn/GDMC/wiki/Using-MCEdit`. Moreover, the GDMC wiki [6] can be referenced for more information - `https://gendesignmc.wikidot.com/wiki:submission-mcedit`.

A.3.6 Running Filters

This section shows the reader how to run the main portion of the project. First and foremost, the **"Filters"** folder must be put inside the newly downloaded MCEdit [5] directory. This should be done under **"stock-filters"**, where all filters are originally located. It must be noted that the original David Mason filter [7] is not present, as for plagiarism purposes. It can be downloaded and put inside the **"stock-filters"** folder from `https://drive.google.com/file/d/1lVUHBibnWSWRdyLsIgqpO_rcrB6et5hn/view`.

To begin with, the **"Fitlers"** folder contains four separate files. These files can be seen in the tree below:

```
Filters
├── A_star.py
├── RoadFilter_WGAN.py
├── David Mason RF.py
└── RoadWGAN.h5
```

The first file is the A-star [10] algorithm file. It is not an original codebase. It was taken from `https://github.com/BaijayantaRoy/Medium-Article` [1] under a free licence. Moreover, it does not require any separate execution. The code is automatically run from the main **RoadFilter_WGAN.py** file.

The **RoadWGAN.h5** file is the trained model and it does not require any separate execution either. It is also automatically run by **RoadFilter_WGAN.py**.

The first filter file is the **RoadFilter_WGAN.py** and it contains the main code base of this project. It generates a road system and edits the environment respectively. To be executed as an individual filter, it can be simply called from MCEdit's [5] filter menu on a selected region, by choosing the filter's name. Although it is an autonomous filter, there are two parameters that can be adjusted. They both affect the generated image and different values can be tested. The first parameter is the *n_maps=2*, which decides how many images to combine, as per chapter

3. The second one is *n_imgs=25*, and it specifies the number of generated road images.

The second filter file, **David Mason RF.py**, is the combination of the proposed filter and David Mason's original filter [7]. It can be executed in the same way as the aforementioned road filter, from MCEdit's [5] filter menu. It generates a road system and builds houses on top of it. Figure A.6 shows how exactly the proposed road filter is incorporated in the original David Mason filter.

```
2314    # Added code to connect the RoadFilter_WGAN Filter.
2315    block = choice([(98,0),(98,0),(98,0),(98,0),(1,5),(4,0),(13,0),(98,1),(98,2)])
2316    road = rf.perform(level, box, options, block, True)
```

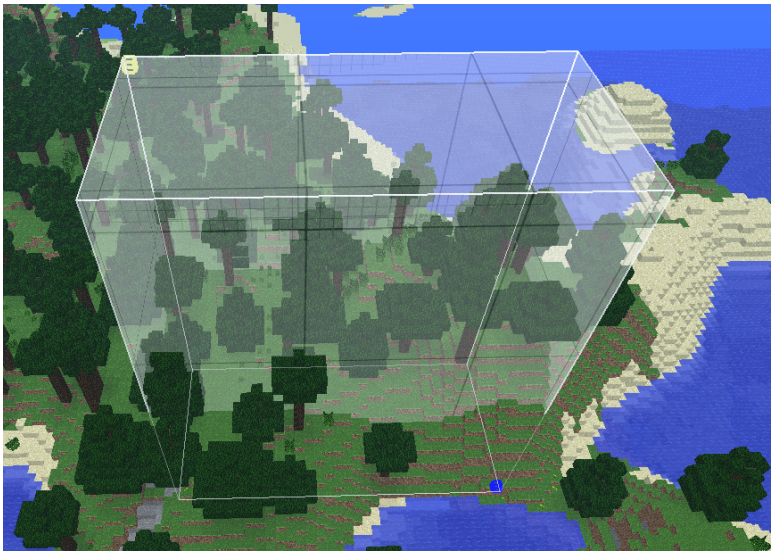
(a)

```
2679        # Added code to connect the RoadFilter_WGAN Filter.
2680        rf.connect_houses_to_roads(level, house.door_pos, road, [(1,6),(4,0),(43,0),(43,5)])
```

(b)

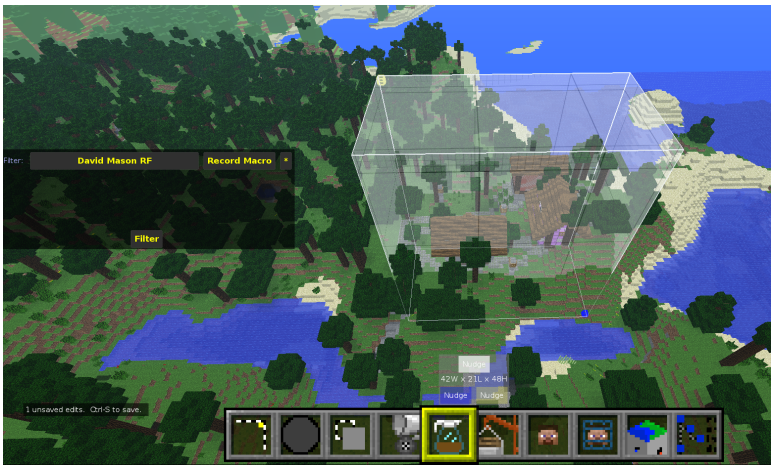
Figure A.6: The figure shows the exact and only lines of code that were added to combine both filters. In addition, on line 16, the filter has been imported via - *import RoadFilter_WGAN as rf*. Image a) shows the code that was added to generate the road. Image b) shows the code that was added for connecting disconnected houses. Moreover, lines 2263 - 2313 have been commented. This is the code where David Mason's filter [7] uses to generate its roads.

In addition, a step-by-step tutorial is shown in figure A.7. It must be also noted that the selected regions in MCEdit [5] need to cover perfectly the bottom floor, without any air blocks present. This is a limitation of David Mason's filter [7], where it does not recognise air blocks and the code breaks. Furthermore, because of the same reasons, the height of the selected region must be made considerably tall.



(a)

(b)



(c)

Figure A.7: This is a figure that serves as a guide to executing a filter in MCEdit [5]. Image a) shows that the first step is selecting a region. The region specifications are covered in the paragraph before the image. Image b) has 3 highlighted steps (in red). The first step is selecting the filter menu, then selecting the filter to run, and finally, executing the filter. The name of the filter may differ from the one shown in the image, depending on the directory. The final image c) shows the results of running the selected filter.

A.3.7 Reconstructing Results

By running the previously mentioned filters, in addition to the original David Mason filter [7], the results in chapter 4 can be reconstructed. Moreover, these filters should be executed on the provided **"TestWorld"** Minecraft [9] world. This exact world has been used for getting all results in chapter 4. It can be loaded from the main menu in MCEdit [5], as it can be seen in figure A.8. Nonetheless, the shown results cannot be fully recreated because of the specificity of the selected regions. There is no way to transfer the exact positions of the selected regions, thus, this must be done by observation.

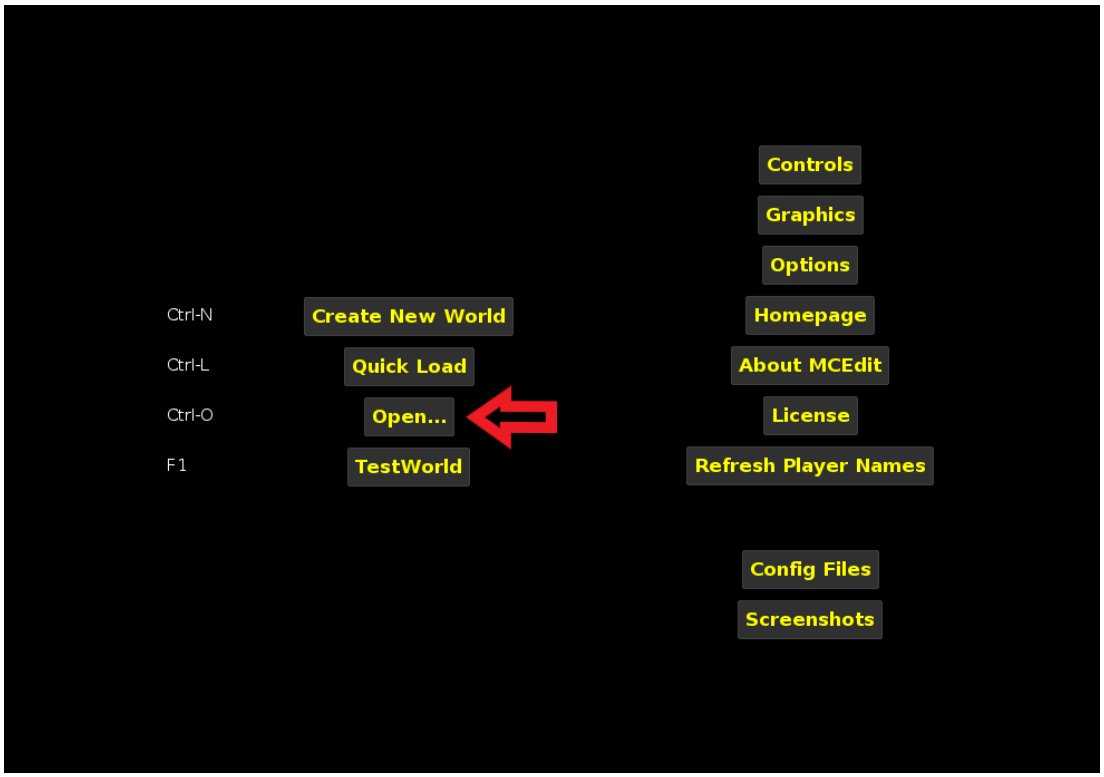


Figure A.8: This image shows the MCEdit [5] Open menu, which loads worlds from a folder.

References

[1] Baijayanta Roy at Medium-Article. A star algorithm code. `https://github.com/BaijayantaRoy/Medium-Article`, last checked on 10-06-2021.

[2] Ekaba Bisong. Google colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 59–64. Springer, 2019.

[3] Yue Cao, Bin Liu, Mingsheng Long, and Jianmin Wang. Hashgan: Deep learning to hash with pair conditional wasserstein gan. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1287–1296, 2018.

[4] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. Deepglobe 2018: A challenge to parse the earth through satellite images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[5] MCEdit. Mcedit: World editor for minecraft. `https://www.mcedit.net/`, last checked on 11-06-2021.

[6] Rodrigo Canaan Christian Guckelsberger Julian Togelius Michael Cerny Green, Christoph Salge. Generative design in minecraft (gdmc) settlement generation. `https://gendesignmc.wikidot.com/wiki:settlement-generation-competition`, last checked on 11-06-2021.

[7] Christoph Salge, Claus Aranha, Adrian Brightmoore, Sean Butler, Rodrigo Canaan, Michael Cook, Michael Cerny Green, Hagen Fischer, Christian Guckelsberger, Jupiter Hadley, et al. Impressions of the gdmc ai settlement generation challenge in minecraft. *arXiv preprint arXiv:2108.02955*, 2021.

[8] Lilian Weng. From gan to wgan. *arXiv preprint arXiv:1904.08994*, 2019.

[9] Wikipedia. Minecraft, sandbox video game developed by mojang.

https://en.wikipedia.org/wiki/Minecraft, last checked on 11-06-2021.

[10] Wikipedia. A star search algorithm.

https://en.wikipedia.org/wiki/A*_search_algorithm, last checked on 11-06-2021.