Mosaic Magnifique

Guide

Morgan Grundy

February 9, 2021

Contents

Chapter 1

Install Instructions

1.1 Pre-built

Download the pre-built Windows app from: https://github.com/MorganGrundy/MosaicMagnifique/releases
Only use the CUDA version if you have a CUDA-capable GPU or the app will just crash.
You may need to run the included vc_redist executable first.

1.2 Dependencies

Name	Version	Modules
GCC/MinGW or MSVC	>= 8.0.0 >= 2017	
Qt	>= 5.9.5	core, gui, svg, widgets
OpenCV	>= 4.1.1	calib3d, core, features2d, flann, highgui, imgcodecs, imgproc, objdetect

1.3 Optional Dependencies

If you have a CUDA-capable GPU, then you can use the following to generate Photomosaics faster.
Currently only Windows supports CUDA.

Name	Version	Modules
CUDA	>= 10.1	
OpenCV Contrib	>= 4.1.1	cudaarithm, cudafilters, cudaimgproc, cudawarping

CUDA usage controlled by "CONFIG += CUDA" in common.pri file.
OpenCV Contrib usage controlled by "CONFIG += OPENCV_W_CUDA" in common.pri file.
Note: OpenCV Contrib requires CUDA.

1.4 Linux

Linux requires pkg-config for linking OpenCV.

1.4.1 Ubuntu

The provided install-ubuntu.mk makefile can be used to easily install dependencies and build Mosaic Magnifique. Tested on Ubuntu 20.04 + 18.04.

```
make -f install-ubuntu.mk all
```

or instead can install dependencies separately:

```
make -f install-ubuntu.mk gcc
make -f install-ubuntu.mk pkg-config
make -f install-ubuntu.mk qmake
make -f install-ubuntu.mk qt
make -f install-ubuntu.mk opencv
make -f install-ubuntu.mk build
```

1.4.2 Other

Qt

Use installer or build from source: https://doc.qt.io/qt-5/gettingstarted.html

OpenCV

Build from source: https://docs.opencv.org/master/d7/d9f/tutorial_linux_install.html
In configuring step give cmake: -DOPENCV_GENERATE_PKGCONFIG=ON
-DCMAKE_BUILD_TYPE=Release
And for minimal build give cmake module list:
-DBUILD_LIST=calib3d,core,features2d,flann,highgui,imgcodecs,imgproc,objdetect

Mosaic Magnifique

Download source from: https://github.com/MorganGrundy/MosaicMagnifique/releases
Create sub-directory "build"
From build run:

```
qmake ../src/src.pro
make
```

1.5 Windows

1.5.1 Batch script

The provided install-windows.cmd batch script can be used to help install OpenCV and build Mosaic Magnifique, but not MSVC/CUDA/Qt.
It has an additional dependency: wget. Set environment variable %wgetdir% to the directory containing wget.exe.
After installing other dependencies, run the script with admin (Setting OpenCV environment variables requires admin) from command line:

```
set mode=all
install-windows.bat
```

If you have installed OpenCV manually then instead:

```
set mode=build
install-windows.bat
```

1.5.2 Manual

MSVC

Download MSVC installer from: https://visualstudio.microsoft.com/downloads/
Run installer and select Workload "Desktop development with C++", the minimum needed is MSVC C++ x64/x86 build tools and Windows SDK.

Qt

Use installer or build from source: https://doc.qt.io/qt-5/gettingstarted.html
Add Qt bin to %PATH% environment variable.

CUDA (Optional)

Download CUDA installer from: https://developer.nvidia.com/cuda-downloads

OpenCV

Build from source: https://docs.opencv.org/master/d3/d52/tutorial_windows_install.html
In configuring step, give cmake: -DCMAKE_BUILD_TYPE=Release
And for minimal build give cmake module list:
-DBUILD_LIST=calib3d,core,features2d,flann,highgui,imgcodecs,imgproc,objdetect

If you are using CUDA you can give cmake: -DWITH_CUDA:BOOL=ON
-DOPENCV_EXTRA_MODULES_PATH="C:/Path/to/OpenCV/Contrib/modules"
And add the relevant contrib modules to module list:
-DBUILD_LIST=calib3d,core,cudaarithm,cudafilters,cudaimgproc,cudawarping,
features2d,flann,highgui,imgcodecs,imgproc,objdetect

Mosaic Magnifique

Download source from: https://github.com/MorganGrundy/MosaicMagnifique/releases
Create sub-directory "build"
From build run:

```
qmake ../src/src.pro -spec win32-msvc
jom qmake_all
jom
```

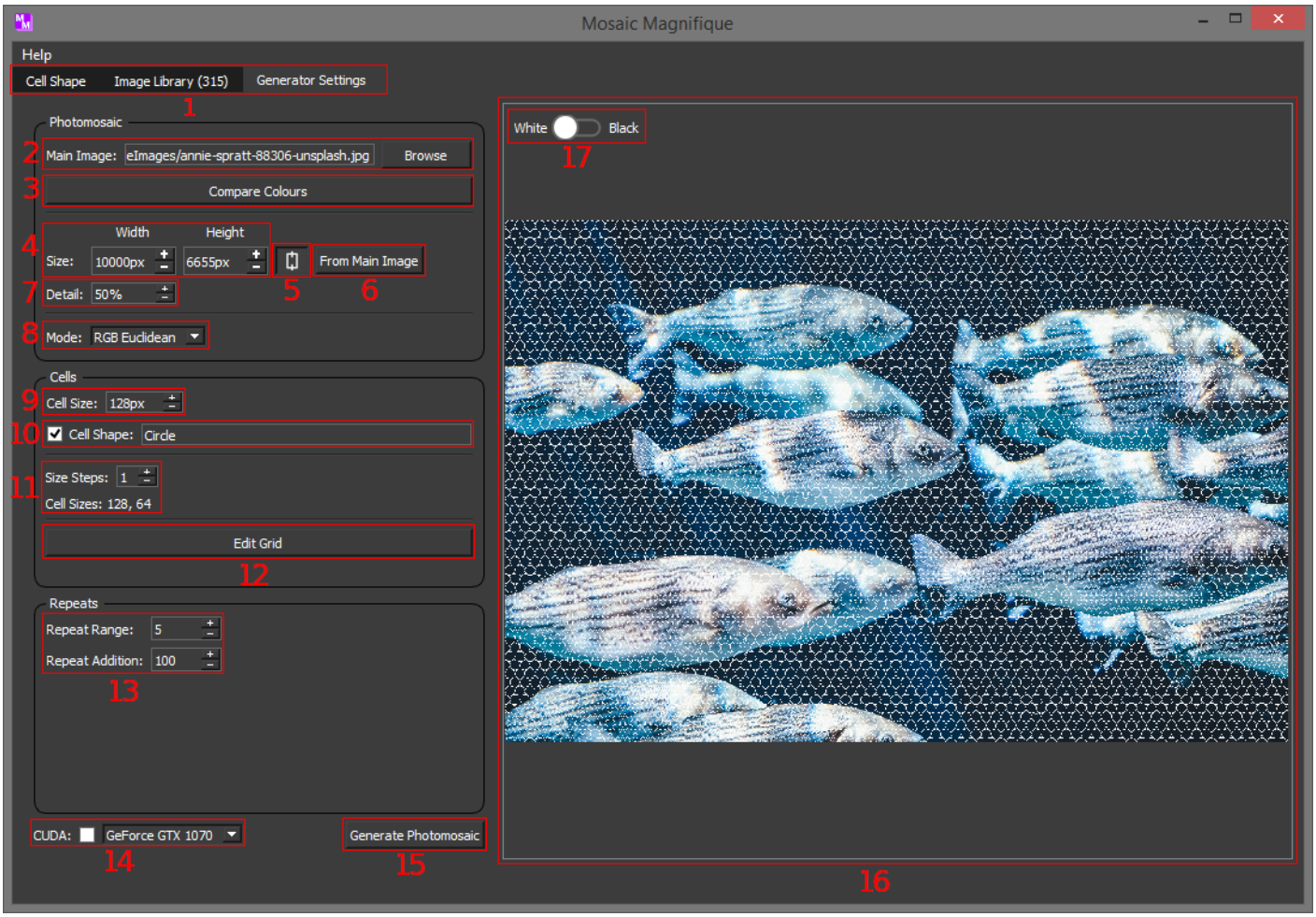
Chapter 2

User Manual

> ***Tip:*** *Press Shift + F1 after clicking a UI element (button, text box, spin box, etc.) to get more info about it.*

Main image used in examples by Annie Spratt on Unsplash : https://unsplash.com/photos/iDkiP2GXlR8

2.1 Generator Settings



1. Generator Settings - Control how the Photomosaic is generated.

2. Main Image - Load an image file for use as main image.
3. Compare Colours - Compare colours in main image against colours in library images.
4. Size - Width and height of Photomosaic.
5. Aspect Ratio - Toggle whether to keep Aspect Ratio when changing size.
6. From Main Image - Sets Photomosaic size to main image size.
7. Detail - Controls percentage of detail to use in image comparison (lower is faster but less accurate).
8. Mode - Which formula to use for comparing colours.
9. Cell Size - Size of cells in Photomosaic.
10. Cell Shape - Shape of cells in Photomosaic, default is square.
11. Size Steps - Controls how many levels of cells to use for Photomosaic. Each level halves cell size.
12. Edit Grid - Modify state of grid.
13. Repeats - Encourages Photomosaic to use a larger variety of library images.
14. CUDA - Toggle CUDA usage when generating.
15. Generate - Generate Photomosaic.
16. Grid Preview - Displays preview of grid overlaid on main image.
17. Grid Preview Colour - Toggle grid preview colour between white and black.

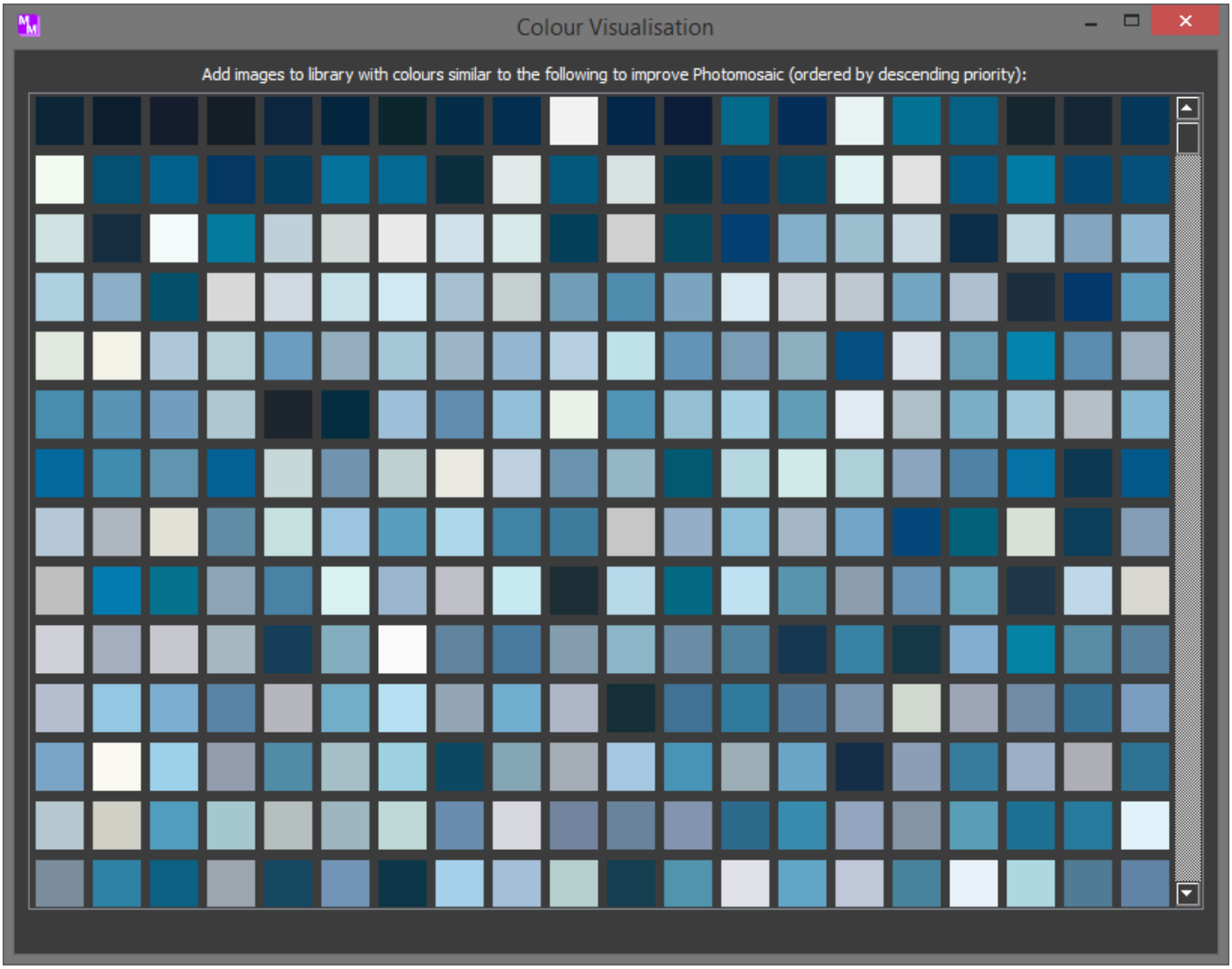
2.1.1 Main Image

The main image is used as the base for the Photomosaic.

Click on the Browse button to open a file browser and select an image file. Supported image types (bmp; dib; jpeg; jpg; jpe; jp2; png; pbm; pgm; ppm; pxm; pnm; sr; ras; tiff; tif; hdr; pic).

The text box shows the filepath of the loaded image.

2.1.2 Colour Visualisation / Compare Colours



Compares the colours of the main image against the colours of the library images, determining which colours are lacking from the image library.
Adding images to the image library that have colours similar to the provided colours are likely to improve the Photomosaic results.

2.1.3 Detail

> ***Tip:*** *Unless you have a very good image library you will likely get very little benefit from 100% detail, so feel free to lower the detail level and save some time.*

Detail level controls how much detail is used when comparing a cell image against a library image. Lowering the detail level will result in faster generation of Photomosaic, but with a loss of accuracy. The loss of accuracy depends on the image library used but in general it is fairly minimal until below 50%.

For example: If cell size is 128px and detail is 50% then the images are resized to 64px before being compared. This would make it 4x faster but slightly less accurate.

2.1.4 Mode

Mode controls which formula is used when comparing images.
Here are the supported modes (in order of increasing result accuracy):

- RGB Euclidean - Images are in the RGB colour space and are compared using a euclidean difference formula.
- CIE76 - Images are in the CIELAB colour space and are compared using a euclidean difference formula.
- CIEDE2000 - Images are in the CIELAB colour space and are compared using a CIEDE2000 difference formula.

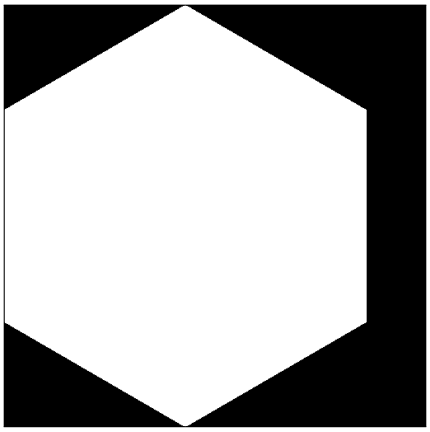
RGB Euclidean and CIE76 take essentially the same time, whereas CIEDE2000 takes significantly longer.
More accurate results are not always "best", Photomosaics are a form of art so experiment.

The CIELAB colour space is designed to be more accurate to how humans perceive colour and the CIEDE2000 formula furthers that goal.

2.1.5 Cell Size

Cell size is the width and height of cells in pixels.

Technically it is the width and height of the cell mask, and the active area can be smaller. For example here is the cell mask used for the hexagon cell:



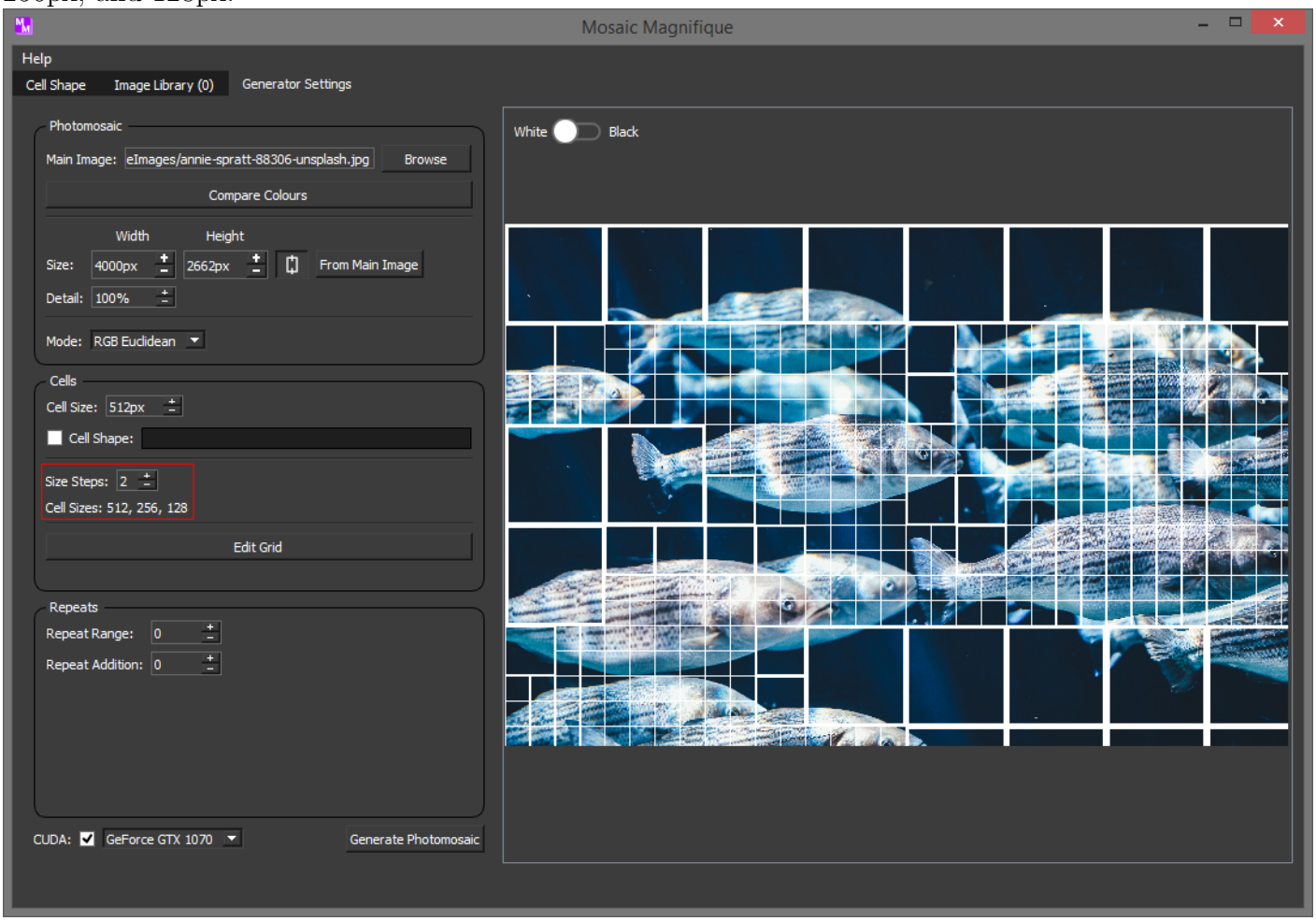
2.1.6 Cell Shape

By default a simple square cell shape is used (when the checkbox is unchecked), else it uses the cell shape currently in the Cell Shape Editor.

2.1.7 Size Steps

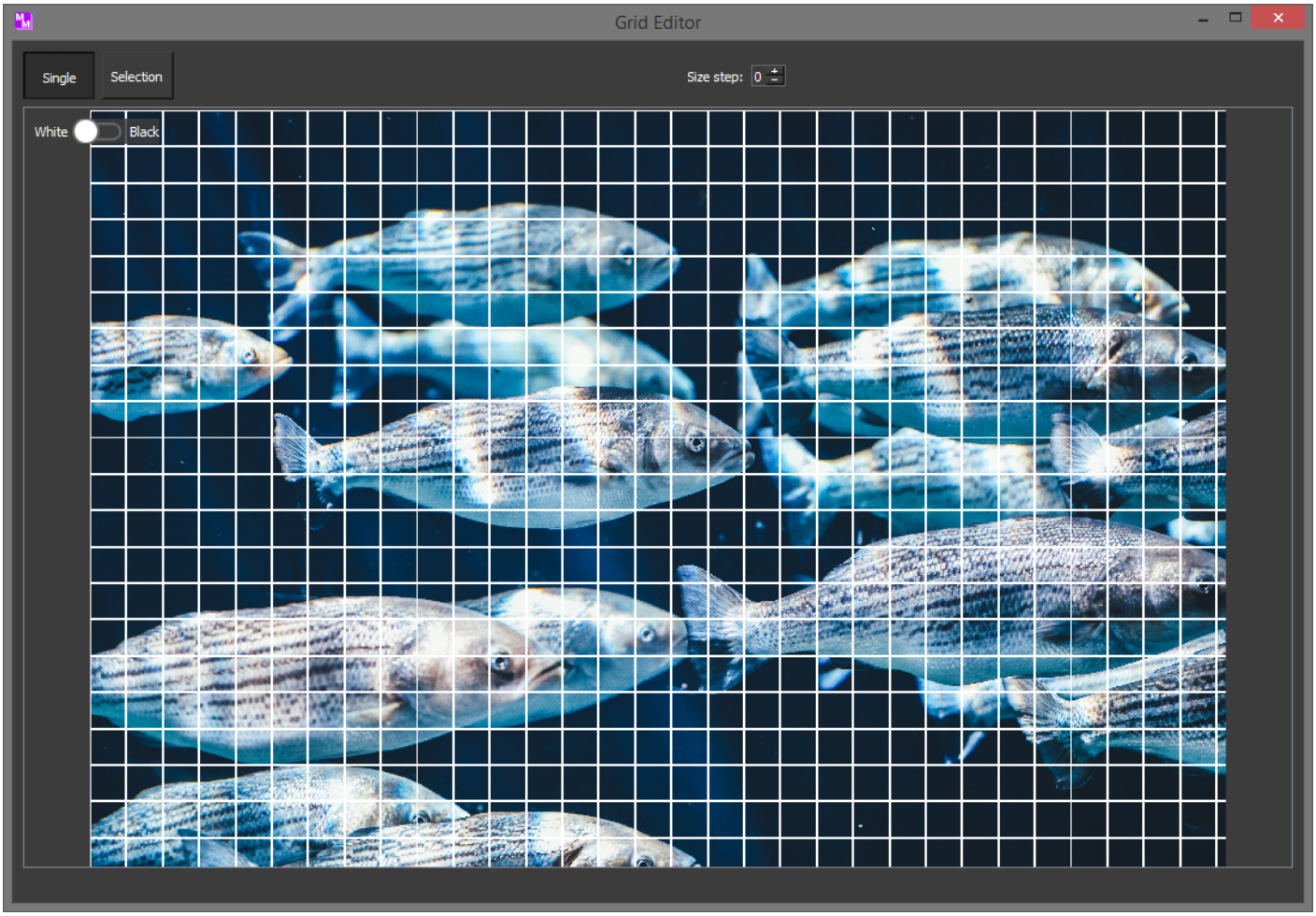
When size steps are 0 there is only a single layer of cells.
For each size step another layer of cells is added with half the cell size of the previous layer.

Entropy (a measure of randomness) is used to determine whether a cell is active. It tends to do a decent job of deciding when a cell should be split up, but if not you can always edit the grid.
For example with a cell size of 512px and 2 size steps there will be three layers with cell sizes 512px, 256px, and 128px.



> ***Tip:*** *If your cell size is not exactly divisible by 2 then your cell layers will not align correctly. Using a power of 2 will guarantee exact division.*

2.1.8 Grid Editor



While the grid is generated automatically you can use the Grid Editor to manually toggle cells.
You can only edit a single layer of cells at a time.
There are two tools:

- Single - Toggle the state of a single cell.
- Selection - Toggle the state of all cells in a selection.

> ***Tip:*** *The Grid Editor allows you to remove areas of the main image from the Photomosaic by disabling the cells in that area.*

2.1.9 Repeats

The repeats options can be used to discourage using the same library images many times.

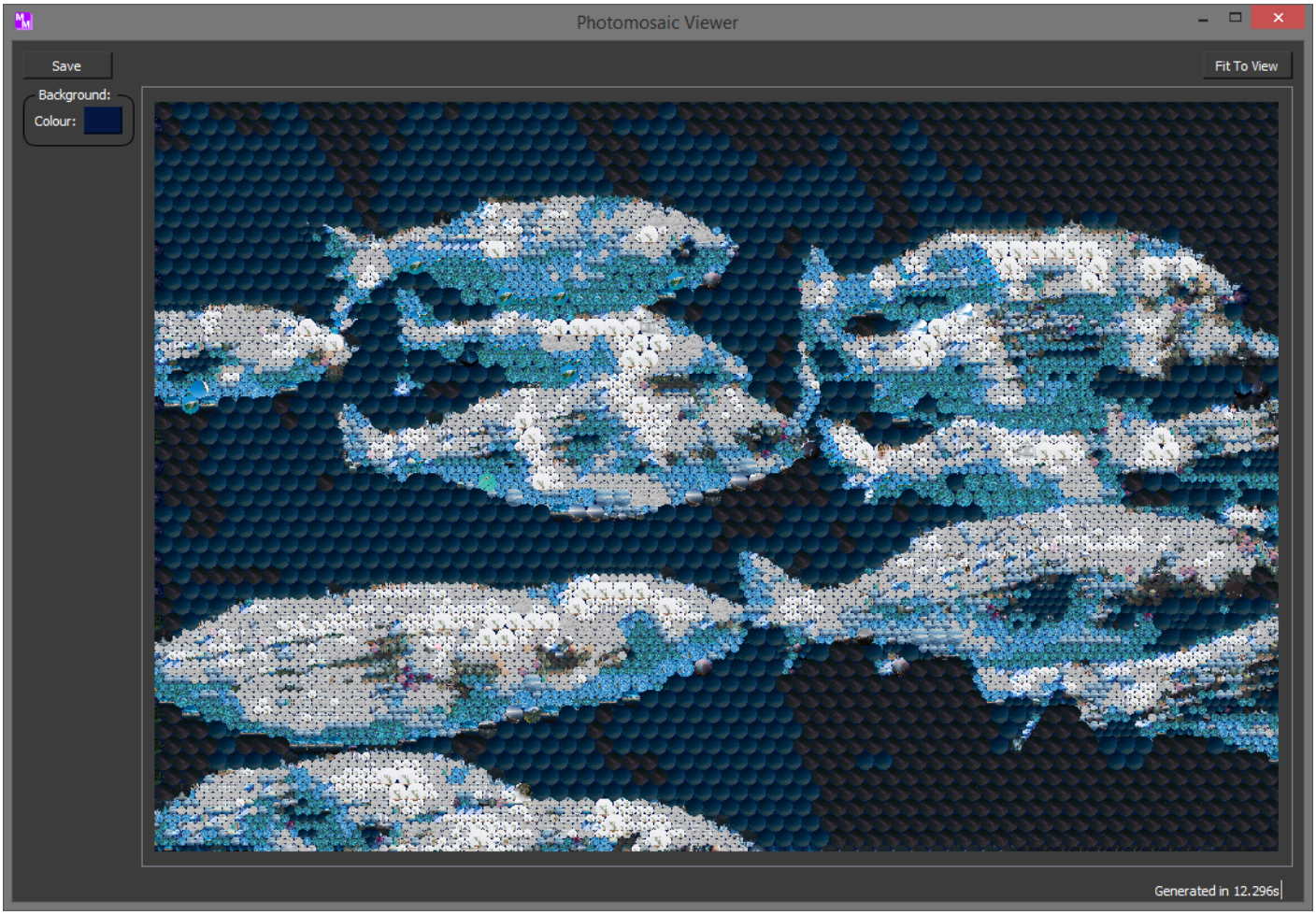
Repeat range represents the range in cells to look for repeating images.
Repeat addition represents the weight added for each time the image is repeated. If you have a good image library you should get lots of variety from just a small repeat addition.

2.1.10 CUDA

If you have the CUDA version of Mosaic Magnifique you can use the checkbox to toggle the use of CUDA when generating a Photomosaic.

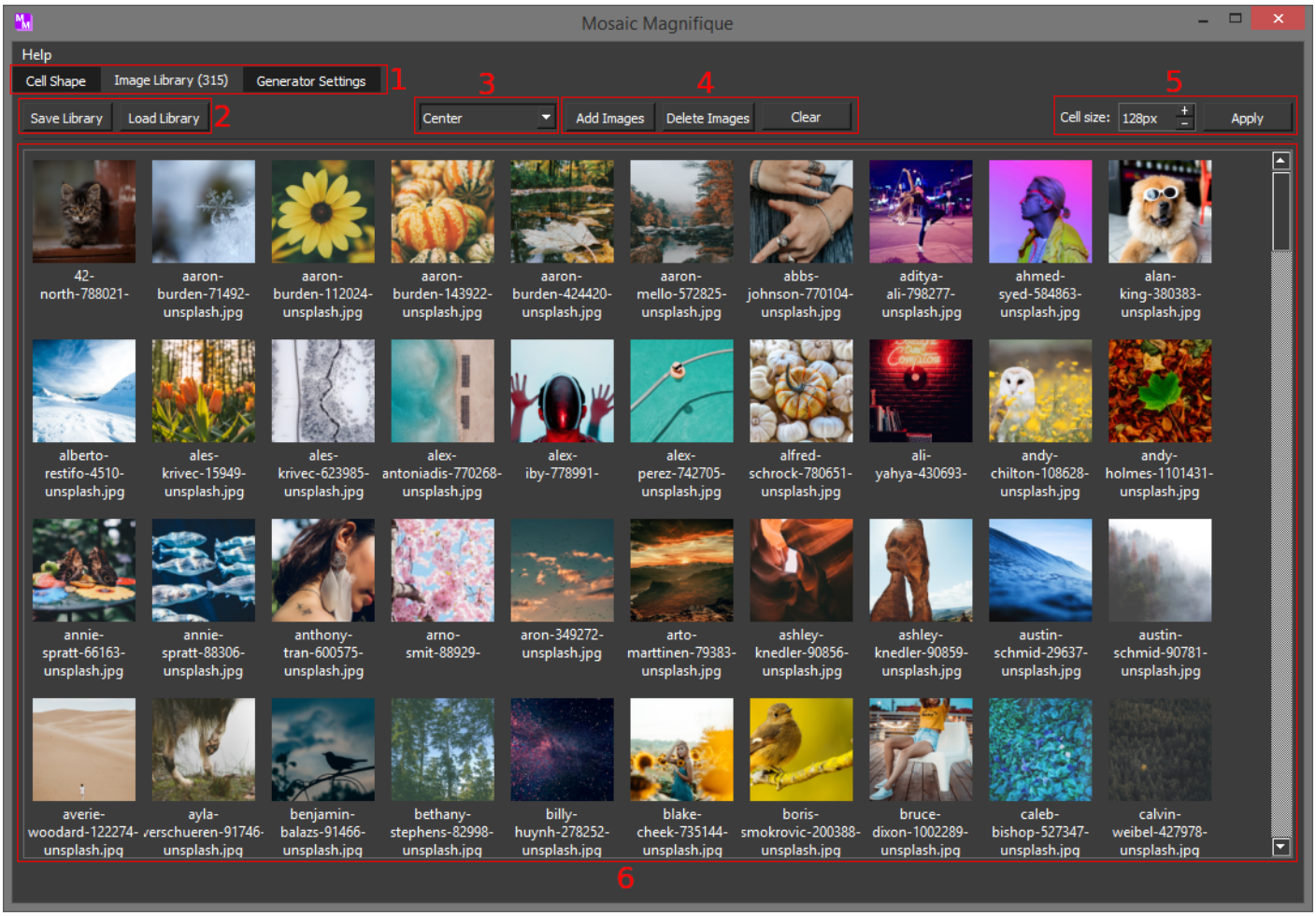
> ***Tip:*** *CUDA really only has a benefit when using mode CIEDE2000. Modes RGB Euclidean and CIE76 are actually faster without CUDA due to how simple the operations are and the high cost of moving the images to the GPU.*

2.1.11 Photomosaic Viewer



After generating a Photomosaic it is displayed in the Photomosaic Viewer.
From here you can set the background colour (including alpha) of the Photomosaic, and save it to an image file.

2.2 Image Library



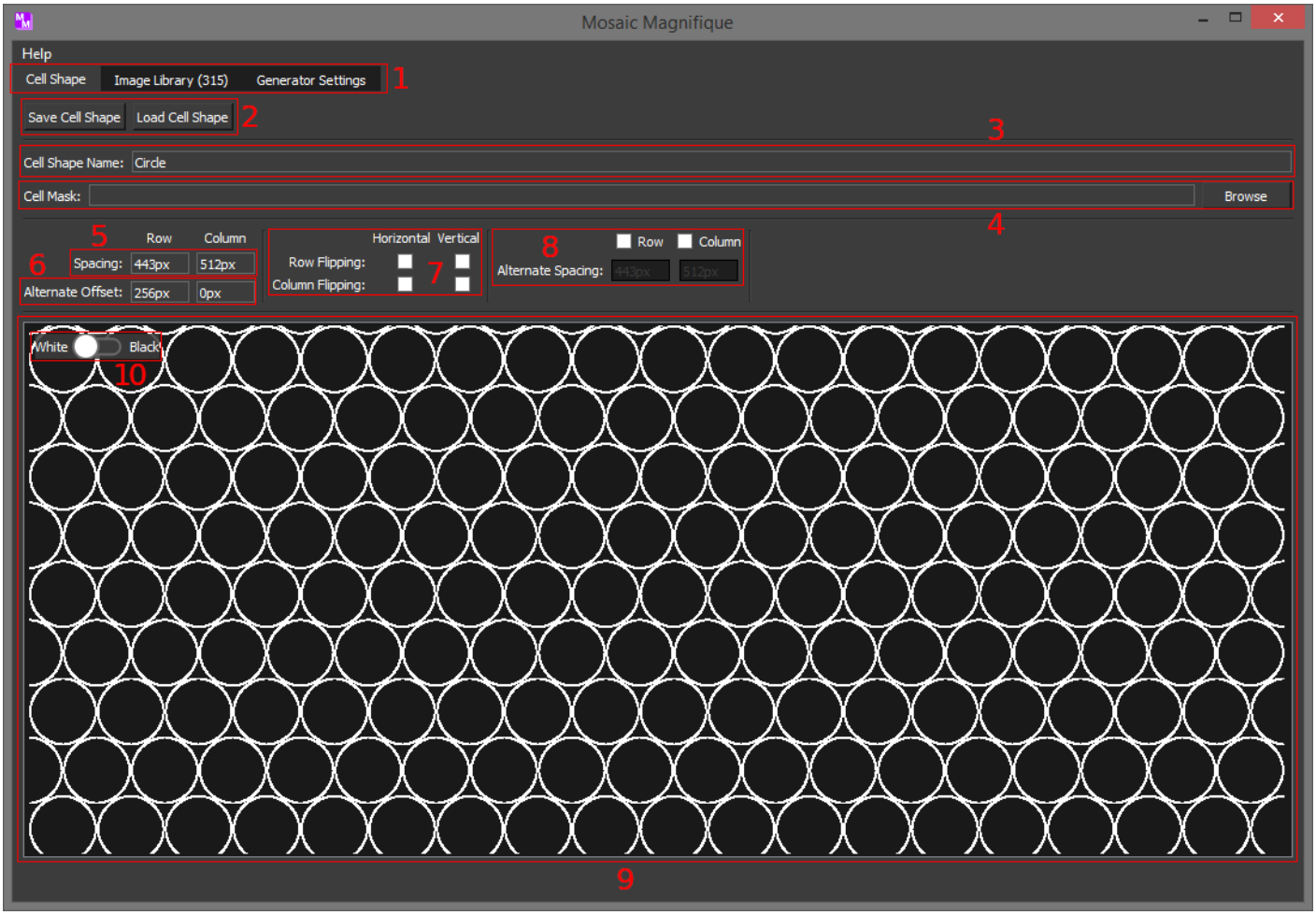
1. Image Library - Create the Image Library to be used in Photomosaic. Shows number of images in library.
2. Save/Load - Save/Load an Image Library to/from a .mil file.
3. Crop Mode - Library images must be square, this determines what method is used for cropping new images.
4. Edit - Add new images, delete selected images, clear all images.
5. Cell Size - All images in the library are resized to this.
6. List - Displays all the images in the library.

2.2.1 Crop Mode

- Manual - Manually crop each image.
- Center - Crops around the center of each image.
- Features - Detects features (specifically corners) in image and crops such that maximum number visible.
- Entropy - Crops image such that entropy (a measure of randomness) is maximised.

- Cascade Classifier - Loads a cascade classifier .xml file for object detection. Crops image such that maximum number of objects visible and closest to center of crop.

2.3 Cell Shape Editor



1. Cell Shape - Create the Cell Shape to be used in Photomosaic.
2. Save/Load - Save/Load a Cell Shape to/from a .mcs file.
3. Name - The name of the Cell Shape (also used for filename).
4. Cell Mask - The filepath of the image used as cell mask.
5. Spacing - Controls spacing between cells in grid.
6. Alternate Offset - Controls alternate offset of rows/columns of cells in grid.
7. Flipping - Controls how alternate row/columns of cells are flipped.
8. Alternate Spacing - Controls spacing of alternate rows/columns of cells in grid.
9. Grid Preview - Displays grid of cell shape outline.
10. Grid Preview Colour - Toggle grid preview colour between white and black.

I recommend experimenting with the Cell Shape Editor yourself to understand how it works but I will try my best to demonstrate here.

2.3.1 Spacing

Figure 2.1: Simple square cells

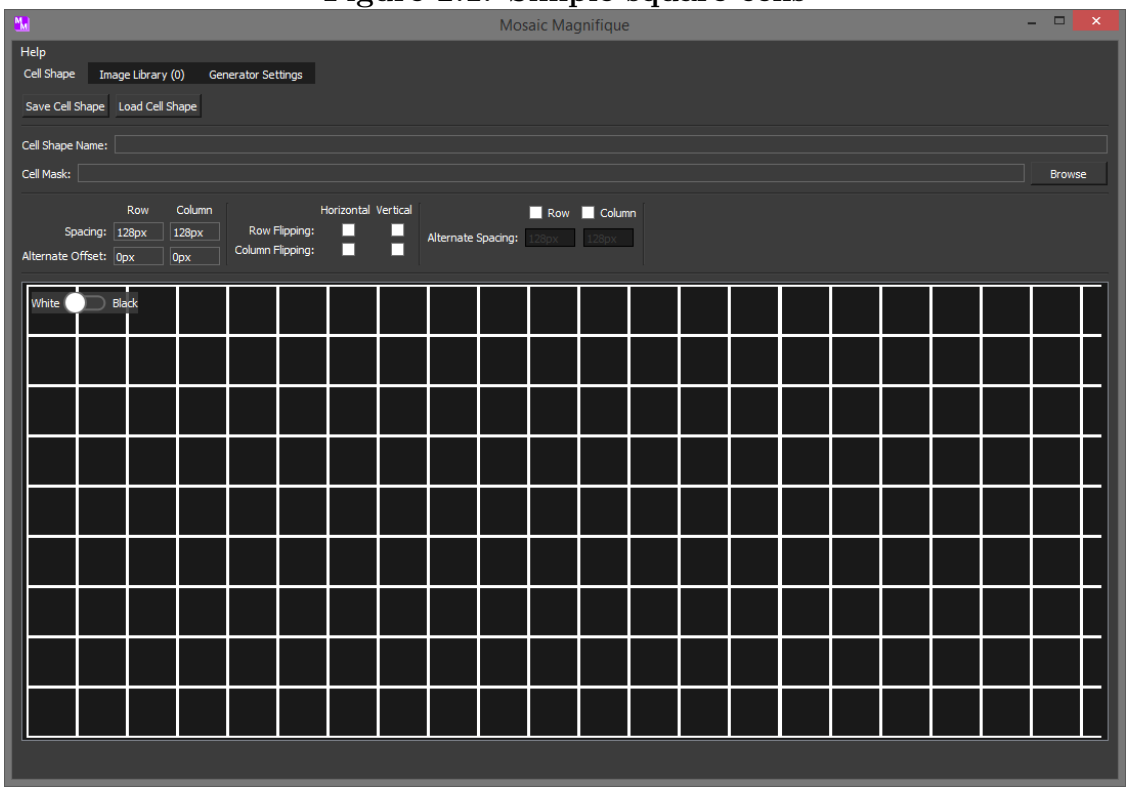


Figure 2.2: Extra row spacing

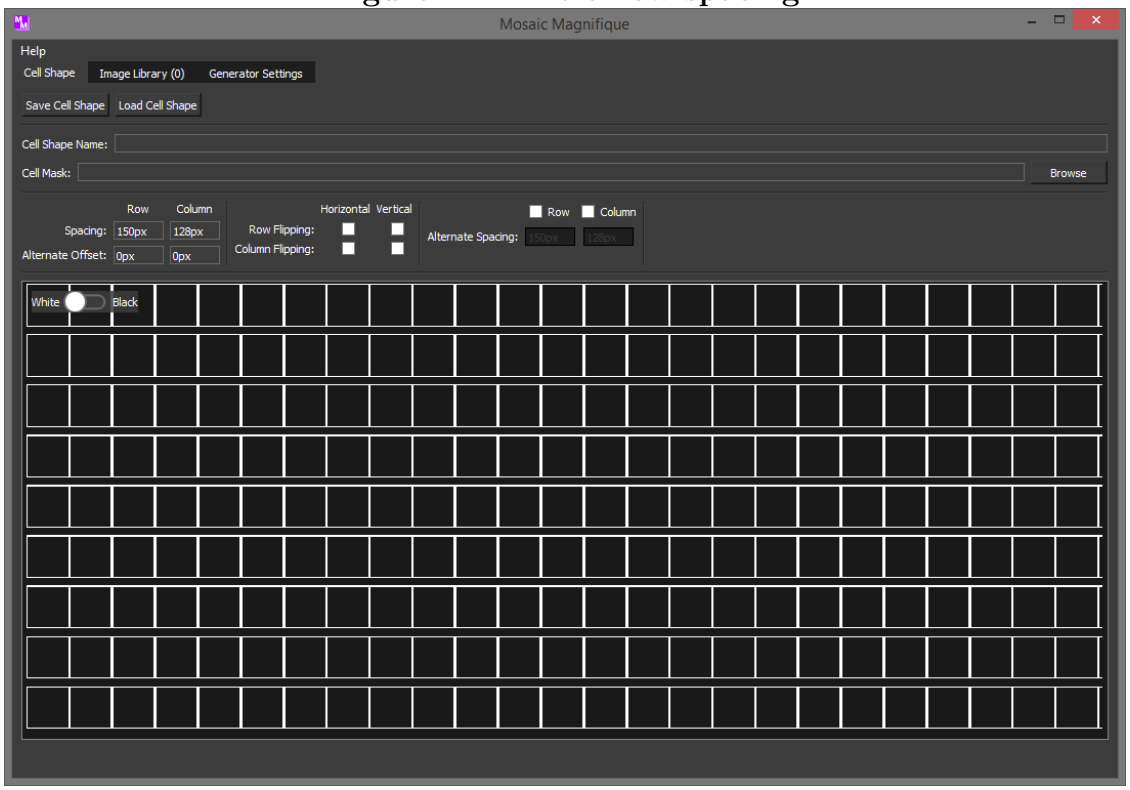
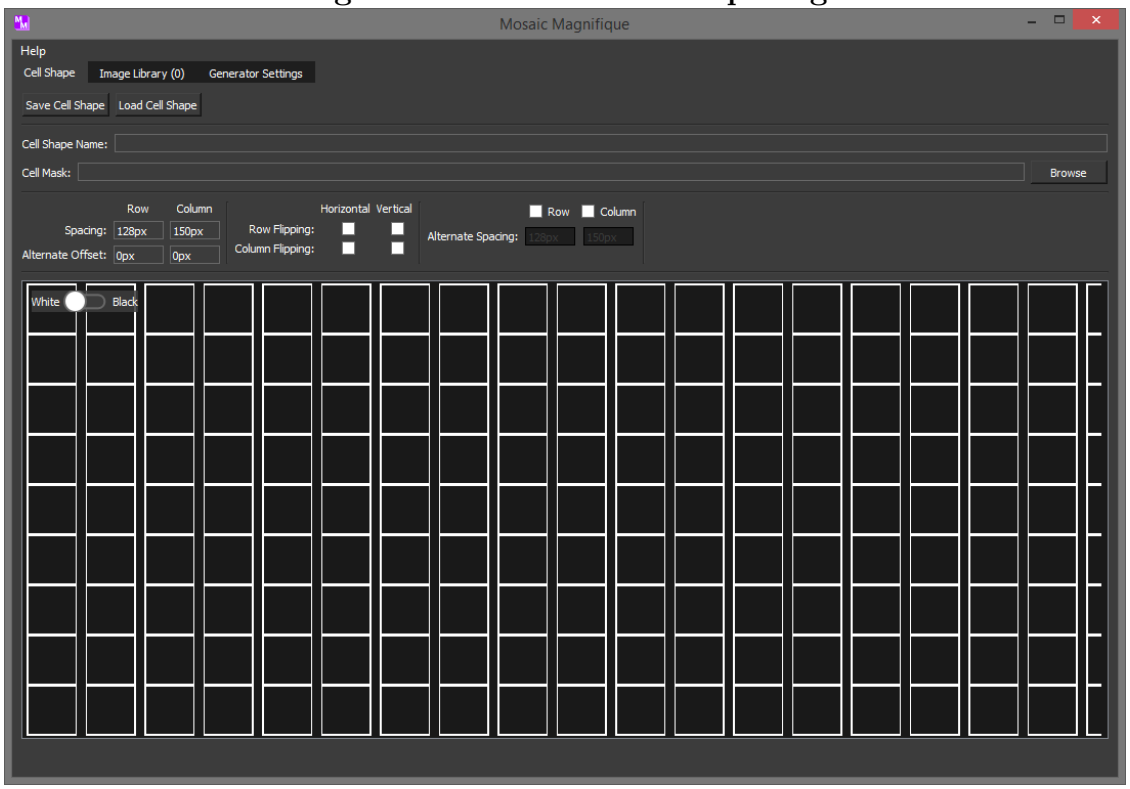


Figure 2.3: Extra column spacing



2.3.2 Alternate Offset

Figure 2.4: Alternate row offset

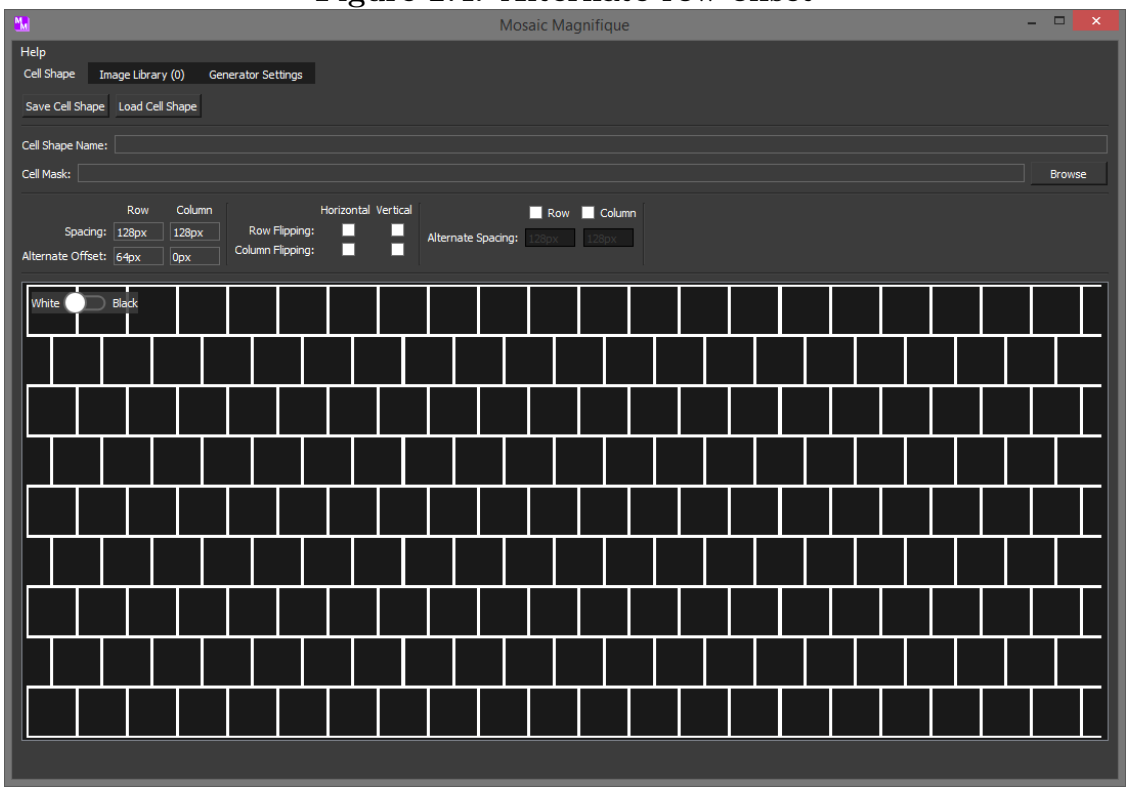
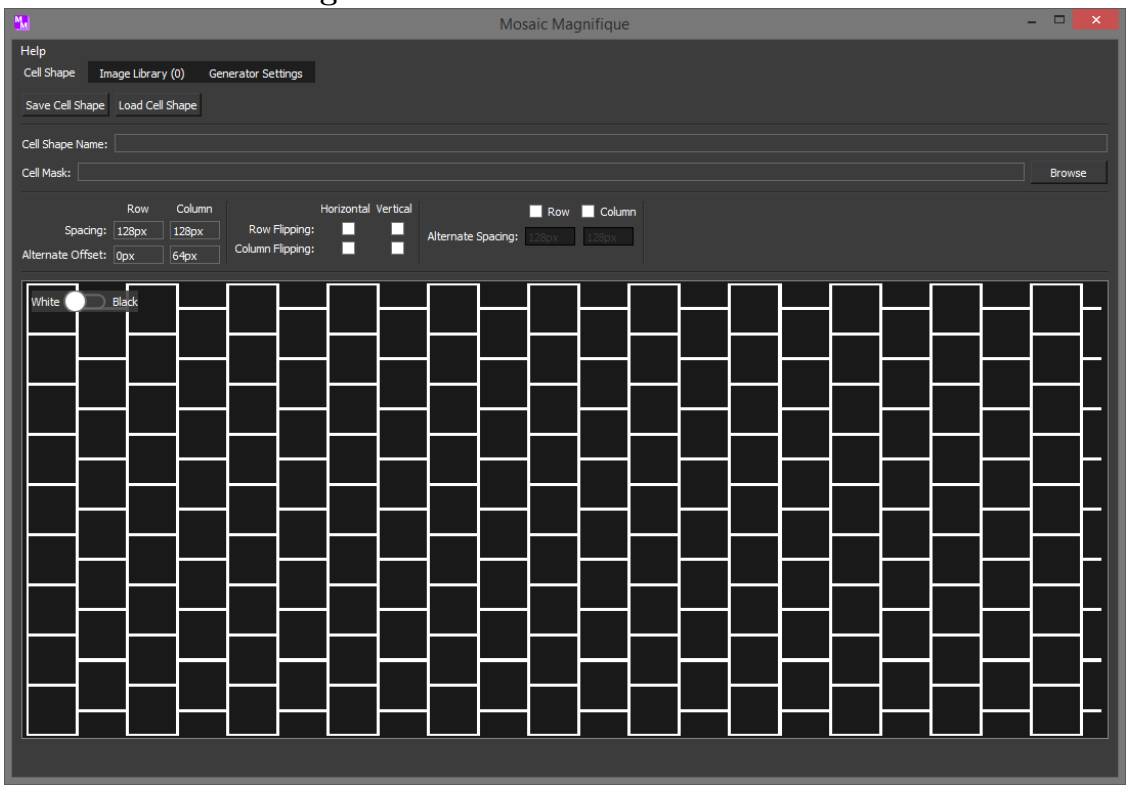


Figure 2.5: Alternate column offset



2.3.3 Alternate Spacing

Figure 2.6: Alternate row spacing

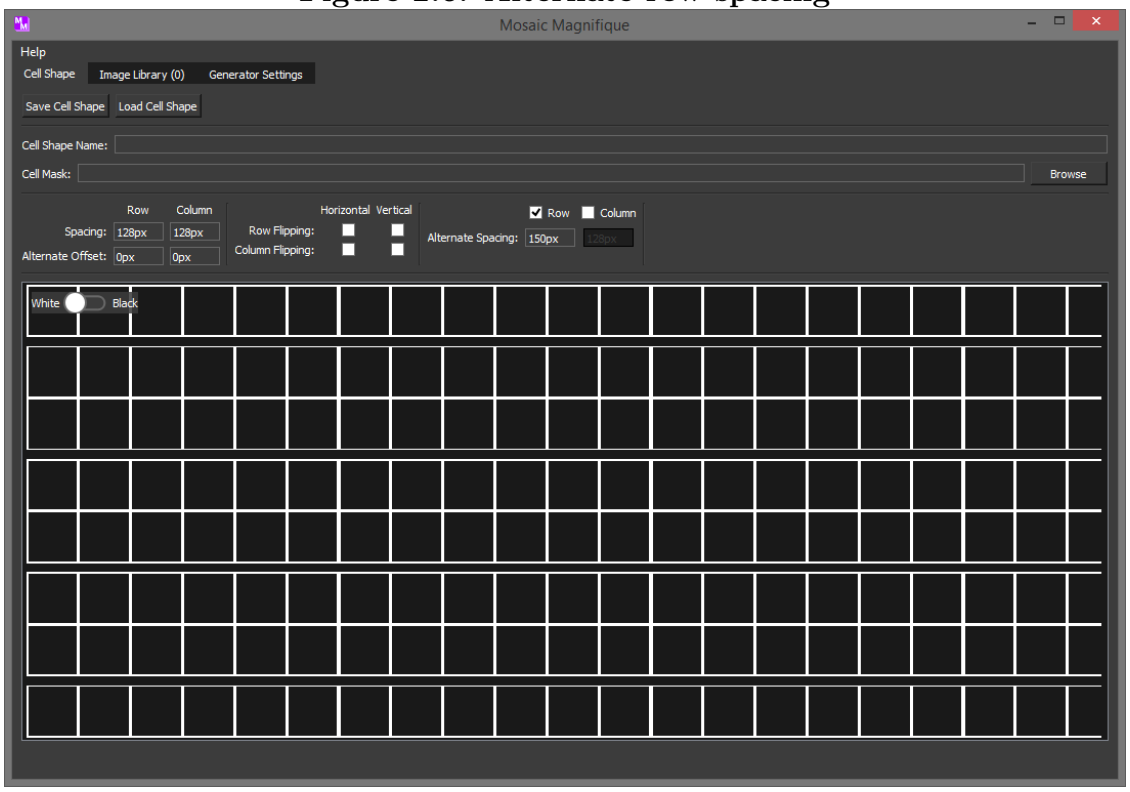
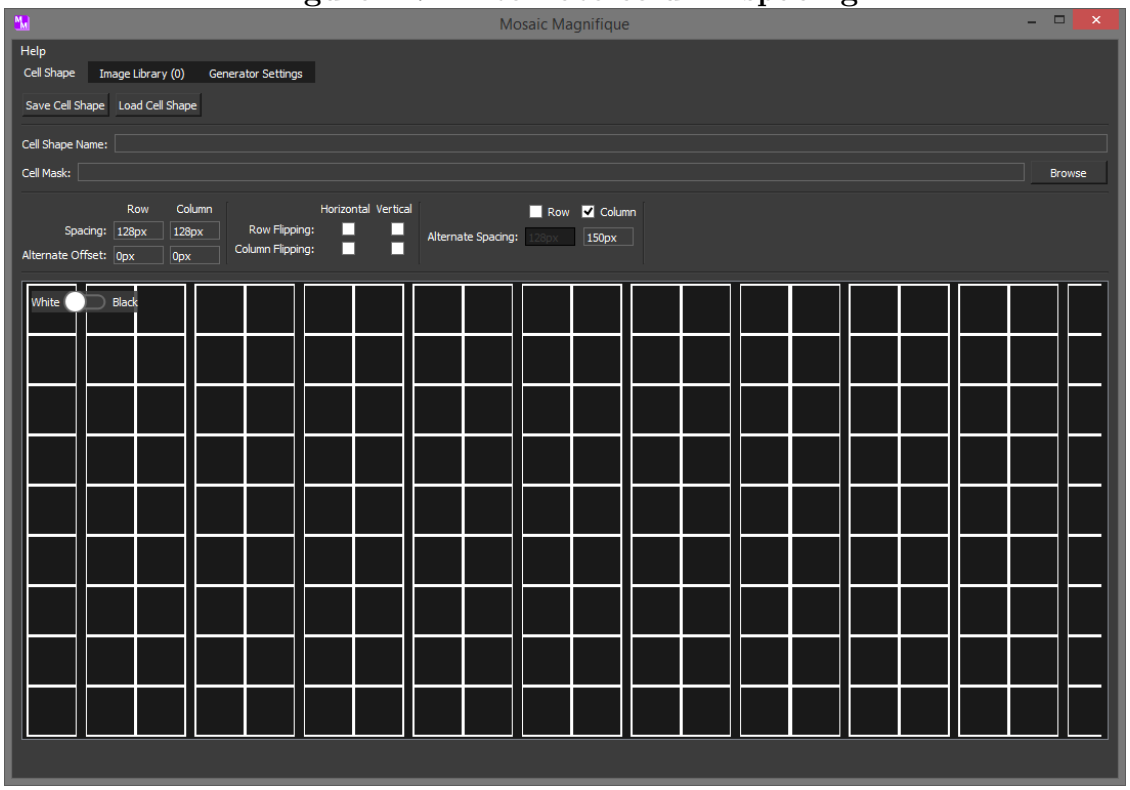


Figure 2.7: Alternate column spacing



2.3.4 Flipping

Figure 2.8: Isosceles triangles rotated 45 degrees

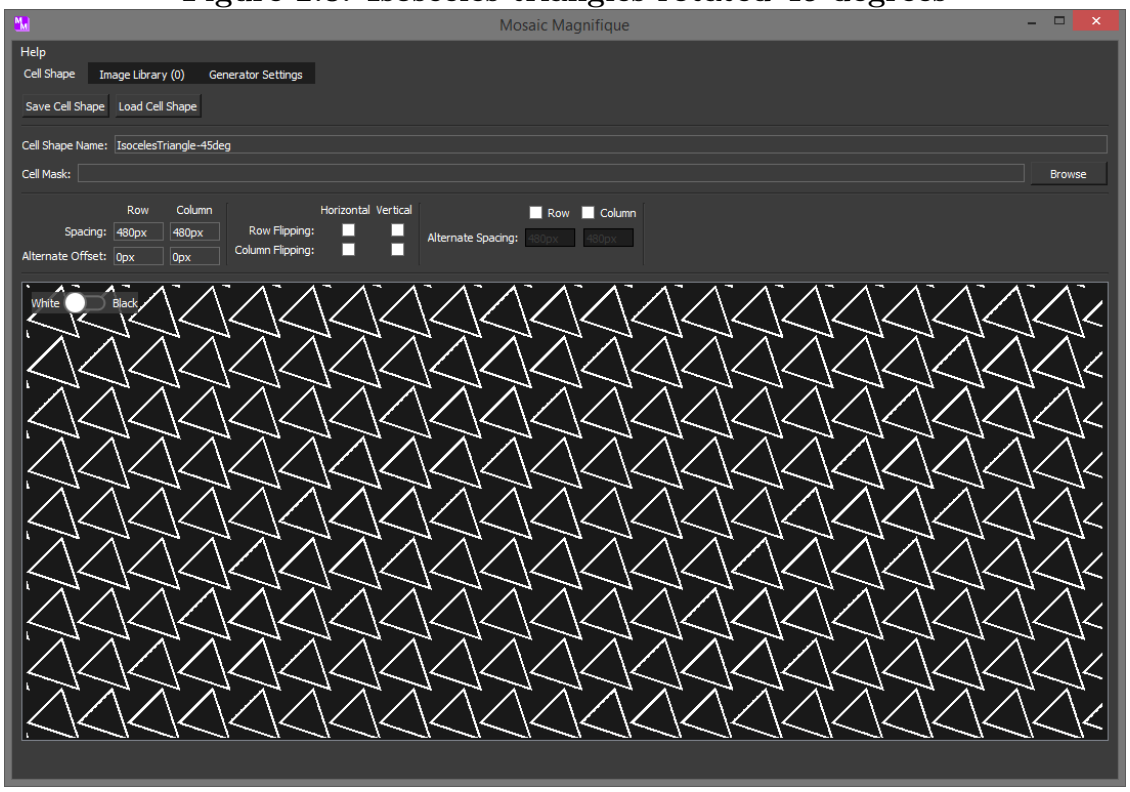


Figure 2.9: Alternate rows use horizontally flipped cells

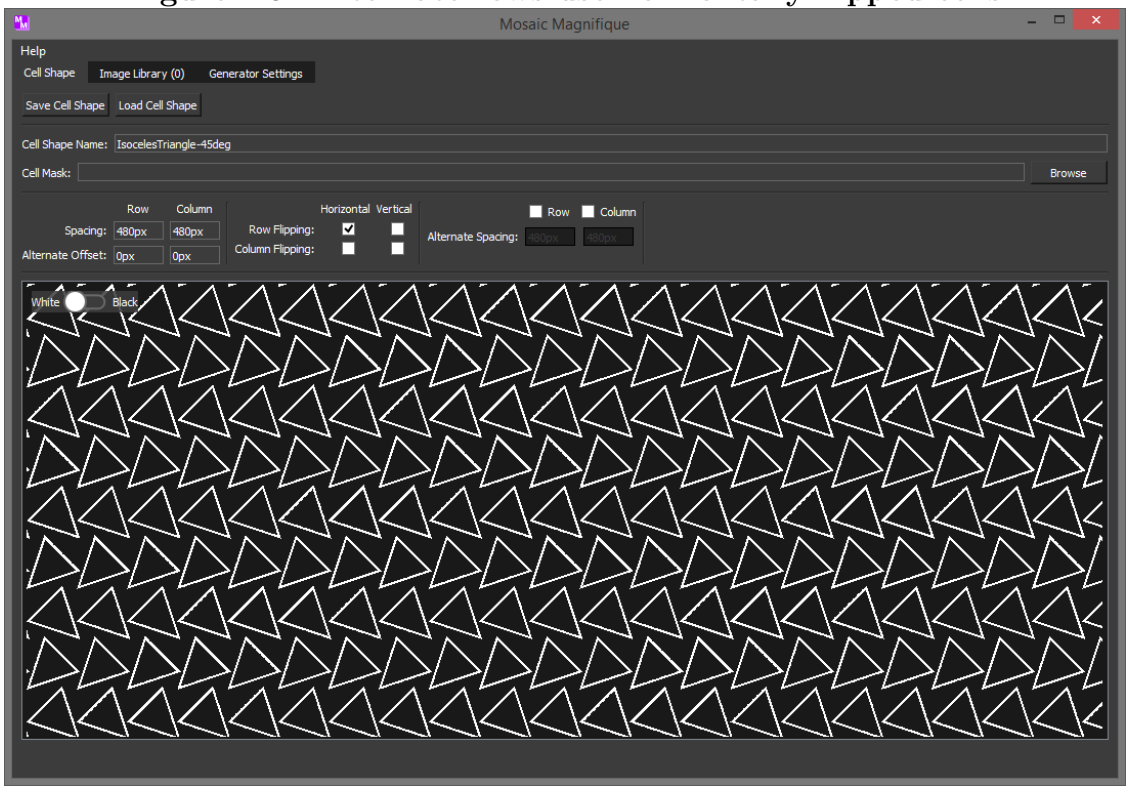


Figure 2.10: Alternate rows use vertically flipped cells

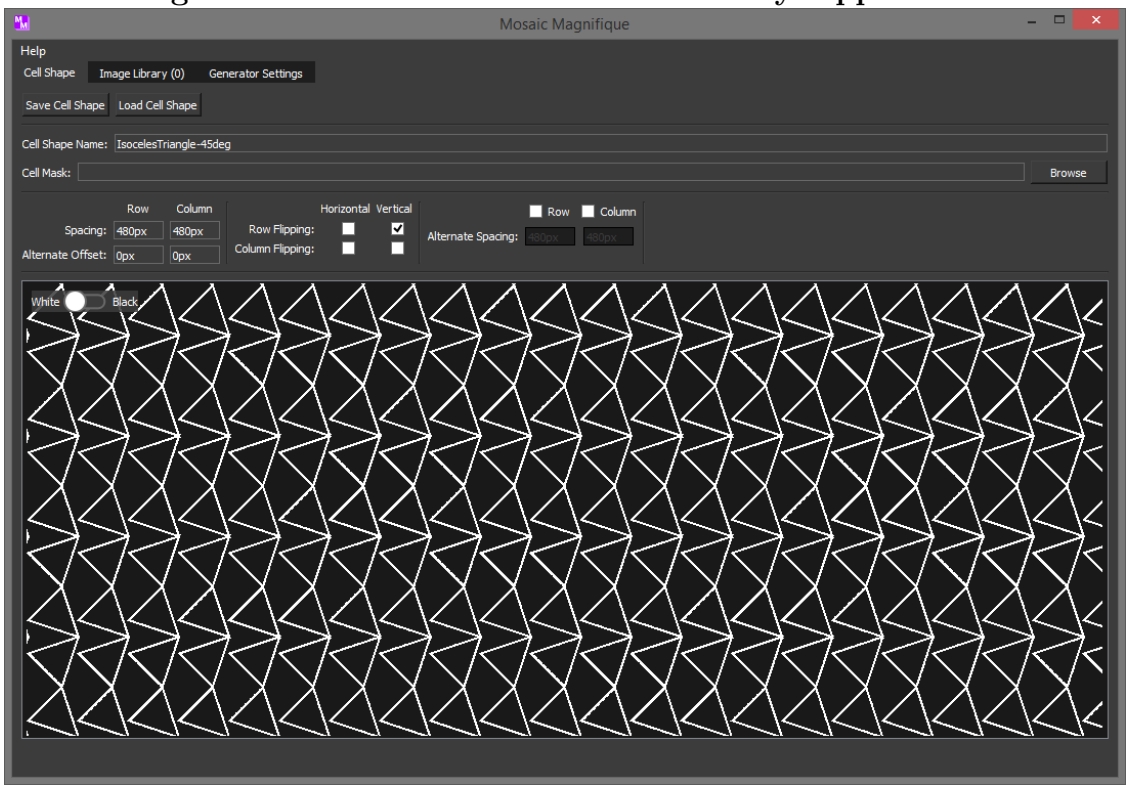


Figure 2.11: Alternate columns use horizontally flipped cells

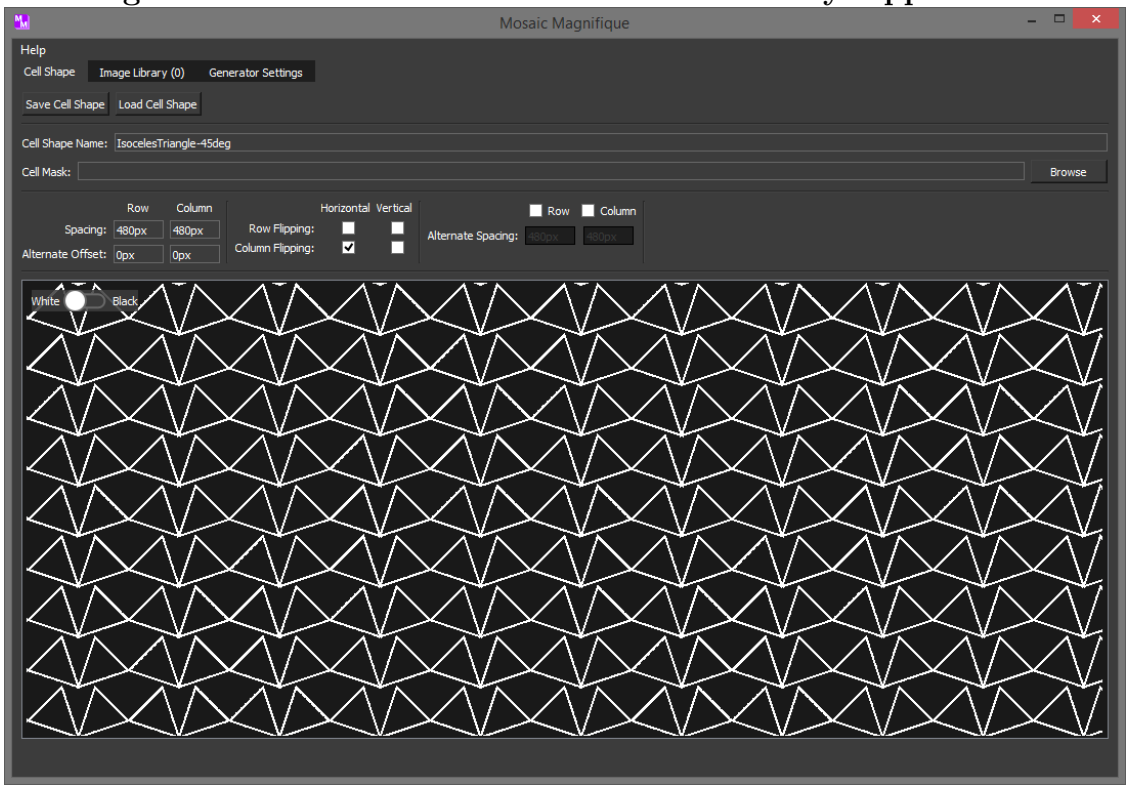


Figure 2.12: Alternate columns use vertically flipped cells