

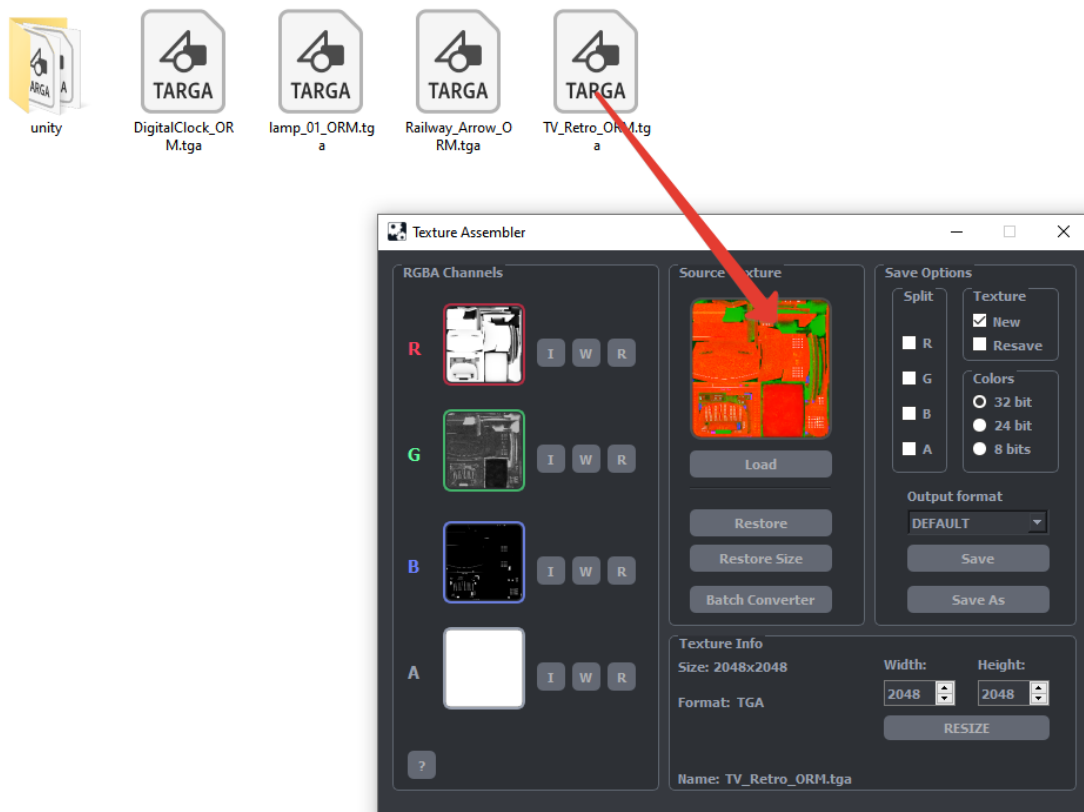
Texture Assembler

Is a tool for quick and easy manipulation of RGBA image channels. For example, it can be used to convert textures between different engines. It can:

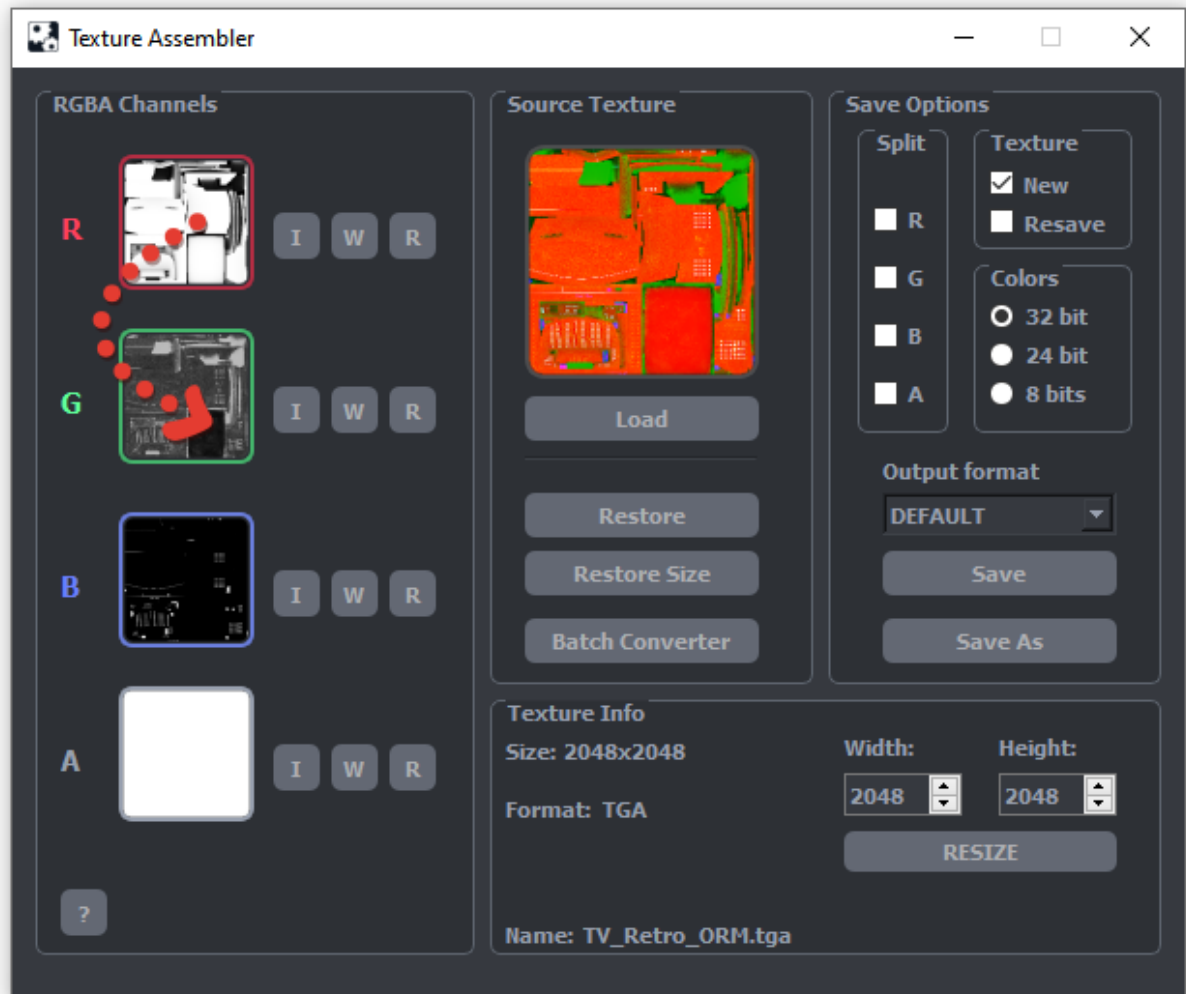
- Pack different images into single image channels.
- Split image channels into separate images.

It can also process a large batch of images with a single click.

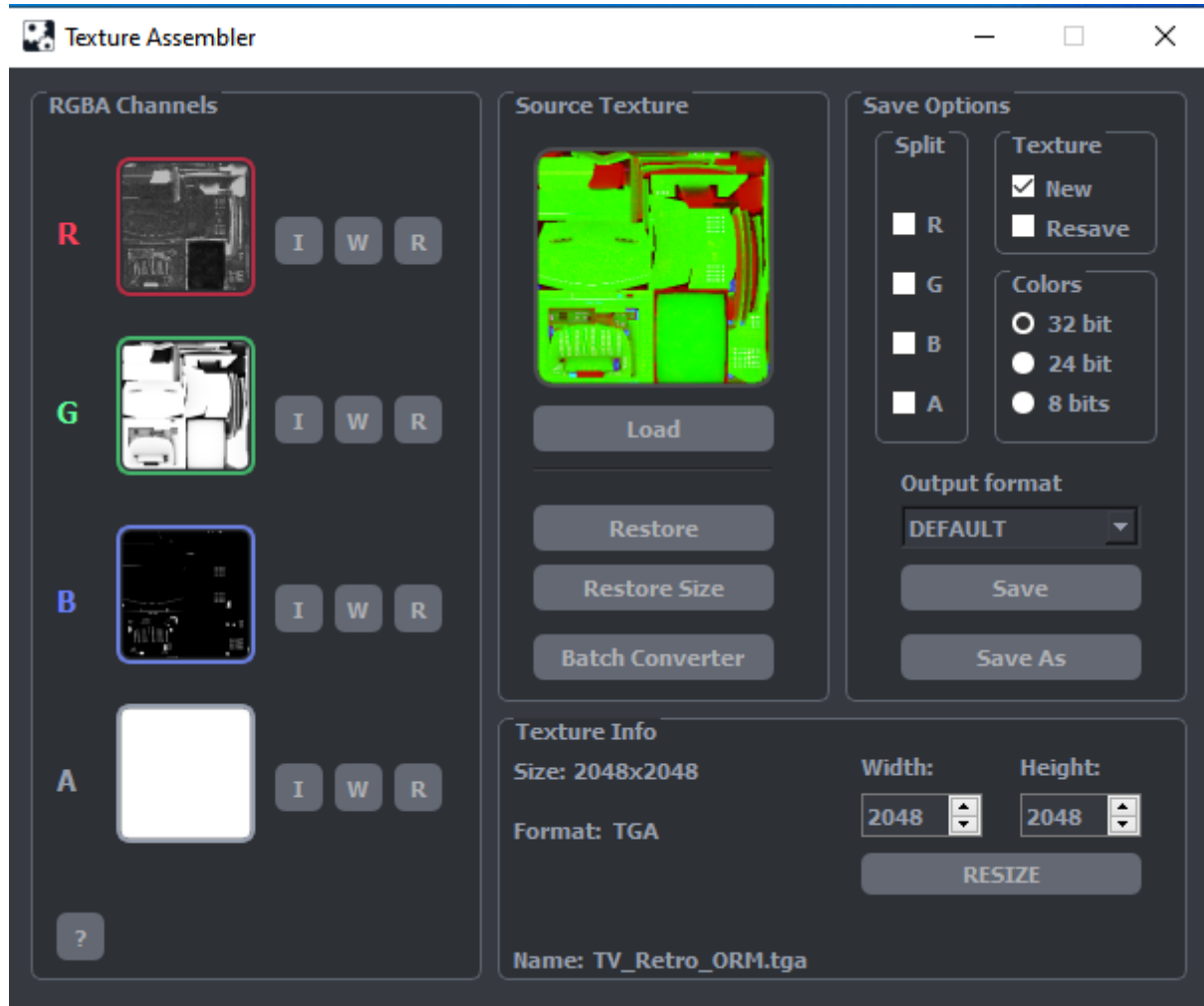
To start working with Texture Assembler, just drag'n'drop an image file into texture input slot as shown below:



As you can see, the texture was loaded and now you can start working with it. To change it, just drag the image from one channel to another



As you can see on the image below, it now looks different:



You can also just load the texture and then drag it into the channel slot. Keep in mind that it'll become monochrome when inserted to the channel. Holding Shift while dragging from channel to channel will copy image contents instead of moving.,

Functions:

Restore: restores original channel order.

Restore Size: restores texture size.

I: inverts the channel (useful for e.g. normal maps).

W: fills the channel with white.

R: restores original channel contents.

RESIZE: resizes the image.

Save options:

R,G,B,A split - Save texture by channels to separate files.

New - This option creates a new texture in the folder.

Option colours:

32 - RGBA

24 - RGB

8 - Gray

Resave - Is current texture overwriting in file:

Note:

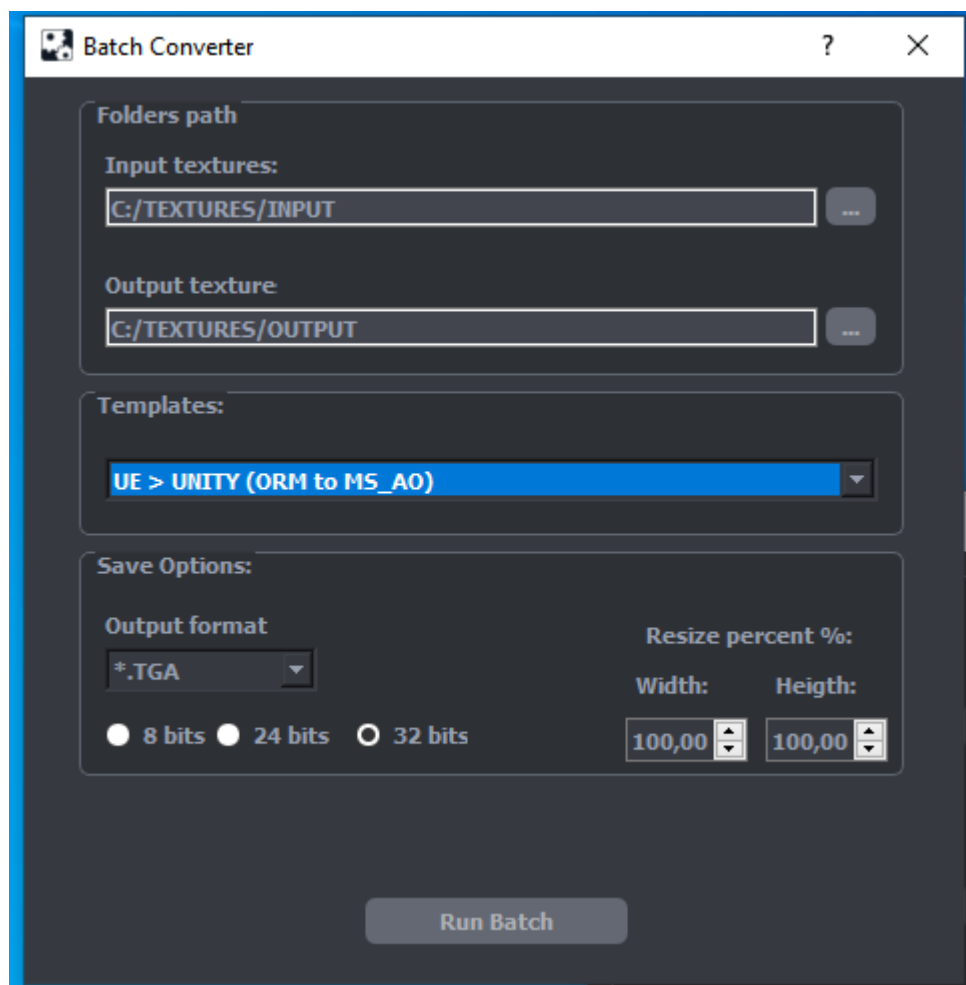
BMP, TGA, PNG, TIFF, JPEG are available for export.

You can also import PIC, PSD, GIF, HDR as optional, without the ability to export.

Use manual work with RGBA channels if you have a small number of textures or different tasks for each texture.

2. Batch Converter.

Click on the “**Batch Converter**” button to open a pop-up window



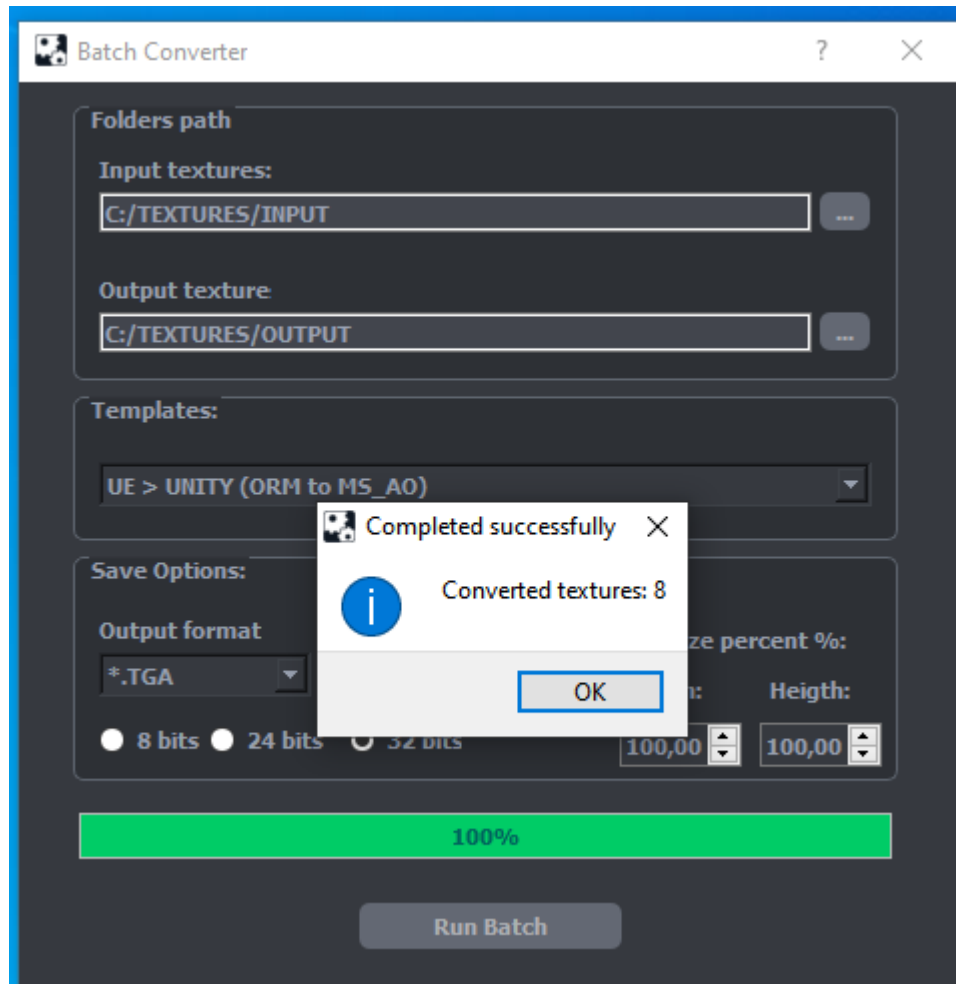
Input textures : the path to the folder where all your textures are located.

Output texture : the path where all processed textures will be saved.

Templates: Drop-down list of the templates for the converter process.

Save Options : The Output file format and image depth (bits) in which all textures will be saved. Resize the image as a percentage (the default value is 100, the size will not be changed, the value can be smaller to reduce the size of the image, but it can not be more than 100).

When all the settings are set, click on the “**Run Batch**” button to start the converter process:



Congratulations, the batch converter process has been successfully completed!

3. Template system.

You can create your own templates for Batch Converter. The template allows you to set your own custom texture packaging, which makes the Batch Converter a universal tool for your own pipeline.

To create your own template In the program folder “**Texture Assembler\Templates**” create a new file with the “.ini” extension and open it in any text editor.

Template Commands:

[TEMPLATE_NAME]

A header to indicate the description of the template.

Variables	Description
NAME = (NAME); Example NAME = UNITY > UE (MS_AO to ORM);	Sets the name for the template. This name will be visible in the template selection list.

[TEXTURES_COUNT]

The header to indicate the number of input and output textures.

Variables	Description
IN_TEXTURES = (1..N);	number of input textures
OUT_TEXTURES = (1..N);	number of output textures

[INPUT-TEXTURE_(ID)]

The header for the input texture. Where the texture ID is its ordinal number (1..N). If ID = 1, then it is the main input texture for groups.

Example:

[INPUT-TEXTURE_1];

Variables	Description
DEFAULT_PREFIX = (ANY, PREFIX_NAME); Example: //ANY ignores any prefixes. DEFAULT_PREFIX = ANY; DEFAULT_PREFIX = ORM;	Input prefix name. The name of the prefix, so that files containing it in the name are found. If the value of the variable is "ANY", then the prefixes will be ignored, all textures will be used without a filter by prefix name.
DEFAULT_COLOR_(R,G,B,A) = (0..255); Example DEFAULT_COLOR_R = 255; DEFAULT_COLOR_G = 255; DEFAULT_COLOR_B = 255; DEFAULT_COLOR_A = 0;	Set the default colors in case the texture was not found.

[OUT-TEXTURE_(ID)]

The header for the output texture. Where the texture ID is its ordinal number (1..N). If ID = 1, then it is the main output texture.

Example:

[OUT-TEXTURE_1];

Variables	Description
<pre>OUT_(R,G,B,A) = IN_(R,G,B,A)@(ID); OUT_(R,G,B,A) = IN_(COLOR)@(0..255);</pre> Example: <pre>OUT_R = IN_B@1; OUT_G = IN_G@1; OUT_B = IN_R@2; OUT_A = COLOR@255;</pre>	Sets the texture output channels to any image Input channel or a numeric brightness value (0..255). ID - id texture input (1..N). COLOR- sets the brightness for the channel.
<pre>OUT_(R,G,B,A)_INVERT = true;</pre> Example: <pre>OUT_A_INVERT = true; OUT_G_INVERT = true;</pre>	Inverts the channel.
<pre>OUT_PREFIX = (PREFIX_NAME);</pre> Example: <pre>OUT_PREFIX = SH;</pre> The saved textures will look like this: Texture Name_SH.tga	Output suffix name.

If you have any questions, please contact me:

vintprog@gmail.com.