ModelConverterX

Your flight simulator development toolbox



User Manual - Version 1.7

SceneryDesign.org

Contents

1 Introduction **6**

2 A word about copyrights **8**

3 Installation **10**
3.1 Prerequisites . . . 10
3.2 Installation . . . 10
3.3 Configuration files . . . 10

4 Quick start **11**
4.1 Convert COLLADA to MDL . . . 11
4.2 Add a night texture to an object . . . 14
4.3 Add a special effect to an object . . . 15
4.4 Add a PBR material to an object . . . 18
4.5 Add a landable platform to an object . . . 20

5 User interface **24**
5.1 Preview . . . 25
5.1.1 Visibility condition state . . . 27
5.2 Toolbar . . . 30
5.3 Event log . . . 31
5.4 Status bar . . . 32
5.5 Keyboard shortcuts . . . 33

6 Editors **34**
6.1 Scenery object editor . . . 34
6.2 Object information editor . . . 35
6.3 Object placement editor . . . 36
6.4 Object hierarchy editor . . . 38
6.5 Material editor . . . 43
6.5.1 Properties tab . . . 44
6.5.2 Textures tab . . . 47
6.5.3 Drawcalls tab . . . 50
6.5.4 Optimize tab . . . 51
6.6 Attached object editor . . . 54
6.7 Animation editor . . . 56
6.8 ModelDef.xml editor . . . 57
6.9 Aircraft.cfg editor . . . 58
6.10 Earth curve correction editor . . . 60
6.11 Season editor . . . 60
6.12 Transform object editor . . . 61
6.12.1 Scale object . . . 61
6.12.2 Move object . . . 61
6.12.3 Rotate object . . . 62

6.13 Level of detail creator . . . 62
6.13.1 Vertex clustering . . . 63
6.13.2 Quadratic based error . . . 64
6.14 Merge object editor . . . 66
6.15 Generate object image . . . 66
6.16 Generate object report . . . 67
6.17 Change history . . . 70

7 Wizards **71**
7.1 Convert and place object wizard . . . 71
7.2 Ground polygon wizard . . . 73
7.3 Batch convert wizard . . . 75
7.3.1 Add empty LOD . . . 76
7.3.2 Add placement . . . 76
7.3.3 Assign material template . . . 76
7.3.4 Assign new GUID . . . 76
7.3.5 Center object . . . 77
7.3.6 Convert materials . . . 77
7.3.7 Convert textures . . . 77
7.3.8 Create LOD (quadratic error . . . 78
7.3.9 Create LOD (vertex clustering . . . 78
7.3.10 Fix animations . . . 78
7.3.11 Flat shade . . . 78
7.3.12 Generate object image . . . 79
7.3.13 Match textures . . . 79
7.3.14 Only keep highest LOD . . . 79
7.3.15 Remove UV2 . . . 79
7.3.16 Remove vertex colors . . . 80
7.3.17 Rename object . . . 80
7.3.18 Rename textures . . . 80
7.3.19 Replace double sided materials by triangles . . . 80
7.3.20 Replace effect . . . 80
7.3.21 Scale model . . . 81
7.3.22 Smooth shade . . . 81
7.3.23 Update object placement . . . 81
7.4 Scene builder wizard . . . 81
7.5 Building Creator Wizard . . . 83
7.5.1 Building drawing control . . . 83
7.5.2 Roofs . . . 88
7.5.3 Gables . . . 89
7.5.4 Features . . . 90
7.5.5 Building Texture Configuration Editor . . . 90

8 Special tools **100**
8.1 GUID converter . . . 100
8.2 Coordinate converter . . . 100
8.3 Texture converter . . . 101
8.4 Missing texture finder . . . 103
8.5 MDL tweaker . . . 103
8.6 Animation tweaker . . . 104
8.7 XML Apron to SurfaceType converter . . . 104
8.8 Placement to SHP converter . . . 105
8.9 SCASM Macro to XML placement converter . . . 105
8.10 Replace double sided material by triangles . . . 106
8.11 Import names from XML . . . 106
8.12 Export ground polygons to SHP . . . 106

8.13 Burn material colors into textures . . . 106
8.14 Filter out ground polygons . . . 106
8.15 Generate footprint object . . . 107
8.16 Normals & Shading . . . 107
8.16.1 Flip all triangles . . . 107
8.16.2 Flip triangles with inverse normals . . . 107
8.16.3 Flat shade object . . . 107
8.16.4 Smooth shade object . . . 107
8.17 Optimize scenegraph . . . 108

9 Readers **109**
9.1 Scenery and object readers . . . 109
9.1.1 AC3D . . . 109
9.1.2 Assimp . . . 109
9.1.3 BGL . . . 109
9.1.4 CFG . . . 111
9.1.5 DSF . . . 111
9.1.6 FLT . . . 111
9.1.7 FSC . . . 111
9.1.8 glTF . . . 111
9.1.9 KMZ . . . 111
9.1.10 MDL . . . 111
9.1.11 SCASM . . . 112
9.1.12 Wavefront OBJ . . . 112
9.1.13 X . . . 112
9.1.14 XML (BGLComp) . . . 112
9.1.15 XML (MSFS model) . . . 112
9.1.16 X-Plane OBJ . . . 112
9.1.17 3DS . . . 112
9.1.18 3MF . . . 112
9.2 Texture readers . . . 112

10 Writers **114**
10.1 Object writers . . . 114
10.1.1 AC3D . . . 114
10.1.2 AF2 TGI . . . 114
10.1.3 Assimp . . . 114
10.1.4 FBX (ASCII) . . . 116
10.1.5 FLT . . . 116
10.1.6 FSC . . . 116
10.1.7 glTF . . . 116
10.1.8 MDL . . . 116
10.1.9 Wavefront OBJ (old) . . . 116
10.1.10 X . . . 117
10.1.11 X-Plane OBJ . . . 117
10.1.12 3DS . . . 117
10.2 Scenery writers . . . 117
10.2.1 BGL . . . 117
10.2.2 BGL flatten . . . 117
10.2.3 BGL ground polygon . . . 117
10.2.4 DSF . . . 119
10.2.5 MSFS scenery package . . . 119
10.2.6 SCA ground polygon . . . 119
10.2.7 TSC . . . 119
10.2.8 TXT . . . 119
10.2.9 XML . . . 119

10.3 Texture writers . . . 119
10.3.1 Generic . . . 119
10.3.2 DXT compressed . . . 119
10.3.3 RGB . . . 120

11 Options **121**
11.1 General settings . . . 122
11.2 Renderer settings . . . 123
11.3 Importer settings . . . 124
11.4 Exporter settings . . . 127
11.5 Processor settings . . . 130
11.6 Texture settings . . . 131
11.7 ObjectModel settings . . . 132
11.8 FS related settings . . . 133

12 Command line **135**

13 Background info **137**
13.1 Position control . . . 137
13.2 Rotation order . . . 138
13.3 Animation mapping . . . 139
13.4 Light mapping . . . 140
13.5 Drawcalls . . . 141
13.6 Representations . . . 143
13.7 Level of detail . . . 144
13.8 MSFS texture distortion . . . 146

14 Support **149**
14.1 Support forum . . . 149
14.2 Reporting crashes . . . 149

15 Credits **150**

16 User license **151**

Chapter 1

Introduction

ModelConverterX is a tool whose primary function is, as the name already implies, to convert models. The tool allows you to import models from different formats and can then export them to other formats. For example you can import your old SCASM macros or COLLADA files made with SketchUp and export them into the FSX MDL format. But also exporting to the 3DS or OpenFlight format is possible. A second functionality for ModelConverterX is as a model viewer. You can view a 3D presentation of your object. A third functionality is to make minor changes to your object, like changing a material parameter or an attachpoint.

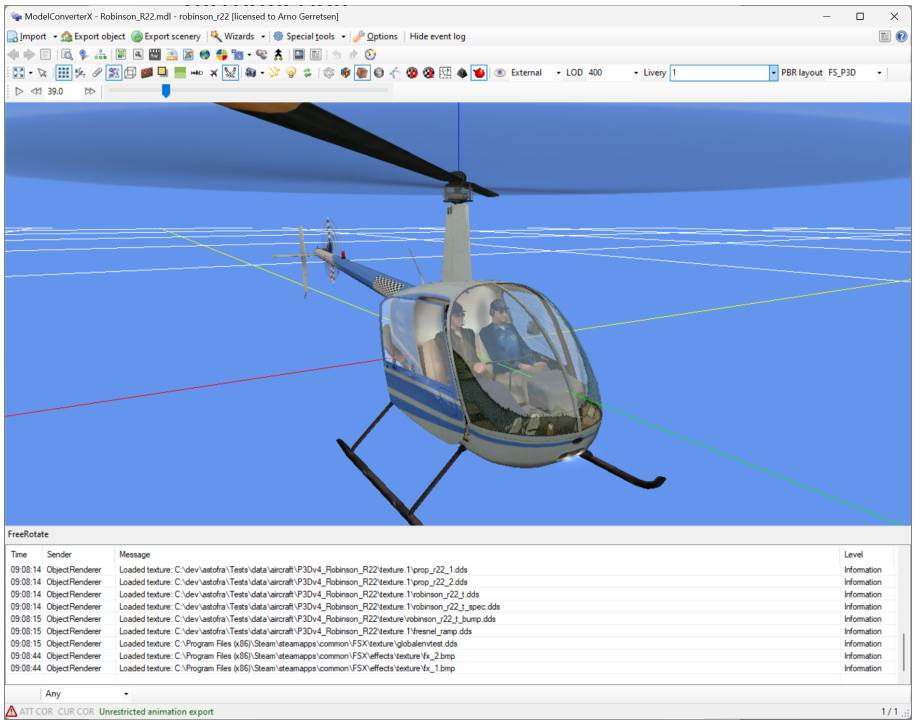


Figure 1.1: A model loaded in ModelConverterX

The layout of the rest of the manual is as follows:

- Chapter 2 discussed copyrights of models in relation to using ModelConverterX.
- Chapter 3 discussed how to install ModelConverterX.
- Chapter 4 gives you a quick start by describing how to do some common tasks using ModelConverterX.

- Chapter 5 describes the user interface and the options.
- Chapter 6 describes the various editors that can be used in the tool.
- Chapter 7 describes the wizards that are available in ModelConverterX.
- Chapter 8describes the special tools that ModelConverterX contains.
- Chapter 9 describes the various file formats that can be read by the tool.
- Chapter 10 describes the various file formats that can be written by the tool.
- Chapter 11 describes the options you can set for ModelConverterX.
- Chapter 13 provides background information that might be useful while using ModelConverterX.
- Chapter 14 describes how to get support for ModelConverterX.

Chapter 2

A word about copyrights

You might wonder why a manual about a tool is started with a chapter about copyrights, but in this chapter I will try to explain why this is important for you as a user of this tool. So please read this chapter as well.



ModelConverterX can convert models between different file formats. It can for example convert old scenery content from FS2004 to the formats used by FSX or Prepar3D, it can convert a COLLADA model made with SketchUp into a MDL file that Flight Simulator can read or it can be used by a developer to convert a MDL file into a format that can be read in a 3D modelling tool because the developer has lost his original sources. These are all conversions that are useful for a developer who wants to modify his content or take it into another flight simulator then it was designed for. These are all examples of genuine developers using ModelConverterX.

But the same conversion capabilities can also be used by "developers" with bad intentions, for example to make derived works of models without permission or to steal the work of others. The simple fact that ModelConverterX can read or convert a model does not mean you are allowed to modify or distribute that model as part of your addon. You might argue that those examples of misusing ModelConverterX illustrate that a tool like this should never have been made available. I don't believe that is true. A hammer can also be used to both build a house and break into one, and nobody would argue that a hammer should not be sold. All the features in ModelConverterX have been added on request of genuine developers with genuine conversion needs and never only to assist the "developers" with bad intentions.

But since I think it is important that as fellow developers we respect the copyright of each other, I have added this chapter to the start of this manual. The general principle is that you should only use or convert work of others after you have received their permission to do so. The simple fact that somebody provided a model for free or does no longer support the model himself does not give you permission to modify or distribute it. So when using ModelConverterX respect the copyrights and intellectual properties of your fellow developers, just as you would like others to respect your rights on the model that you spend a lot of effort and time on.

Below are a few misunderstandings about copyrights that are often heard:

- The fact that a developer gives you a model as freeware, does not mean you have the intellectual property right of the model as well. You have just been given a license to use it for free.

- Some people argue that the Game Content Usage Rules of Microsoft give you permission to make derived works of content in Microsoft games. But these rules clearly state they only cover the content for which Microsoft owns the intellectual property and for most flight simulator addons Microsoft does not own the intellectual property, they remain with the original author. And even for the content that is covered by these rules, restrictions are specified on what kind of derived works you can make.

So to summarize this all, when using ModelConverterX please respect the copyrights of others. And if you don't agree with these, please remove ModelConverterX from your machine and don't use it.

Chapter 3

Installation

3.1 Prerequisites

Before you install ModelConverterX, please make sure that you have the following prerequisites installed on your computer:

- A graphics card capable of supporting OpenGL 3.3
- Microsoft .NET framework 4.7.2
- Microsoft Visual C++ 2015-2019 runtime files. If you run a 64 bit operating system, make sure to get the x64 version, else get the x86 version.
- To be able to export files in the formats used by FS2004, FSX or Prepar3D you need the Software Development Kit (SDK) of the specific flight simulator version you are working with.

3.2 Installation

Installing ModelConverterX is really simple, you can just unpack the ZIP file that you downloaded into a folder of your choice. This ZIP file contains everything that you need.

If you are upgrading from an older version of ModelConverterX, you can overwrite the existing files with the never files from the ZIP.

The first time that you run ModelConverterX, the tool will automatically try to set most options for you. For example, based on the registry entries the paths where flight simulator and the SDK tools are installed will be filled in for you.

3.3 Configuration files

In case you have issues running MCX, it can sometimes help to remove the configuration files. Sometimes they get corrupted. You can find them in the following folder:

`C:\Users\{username}\AppData\Local\SceneryDesign.org`

In that folder you will find a folder who's name starts with ModelConverterX. There is where the configuration files are stored. Deleting the folder will revert ModelConverterX back to the default settings.

Chapter 4

Quick start

In the rest of this manual all the details of the ModelConverterX user interface and the available tools are explained. This chapter is a quick start to get you going even faster. It explains how to perform some common tasks on your models with ModelConverterX.

4.1 Convert COLLADA to MDL

This quick start will show you how you can convert a COLLADA model and its textures to the MDL format so that you can use it in Flight Simulator or Prepar3D. You need to take the following steps:

1. Import the COLLADA (DAE) object into ModelConverterX. You can do this by pressing the **Import** button and selecting the file or by dropping the file onto the ModelConverterX preview control. See Figure 4.1.
2. Open the **Material editor**.
3. Go to the **Textures** tab, see Figure 4.2.
4. Select the folder where the textures should be saved.
5. Select DDS as texture format.
6. Since this model has textures named like texture0.jpg, etc, it is likely that these names are not unique. Therefore press the **Prefix all with model name** button to make sure all texture names are prefixed with the model name.
7. Press the **Save textures** button to convert all textures to DDS. Figure 4.3 shows the Material Editor after converting.
8. Close the **Material editor**.
9. Export the object to MDL. Press the **Export object** button to bring up the dialog. Select the filename you want for the MDL file and also the MDL version you want, e.g. FSX or P3D v4. See Figure 4.4. Once you click the **Save** button the object is written to the selected location.

Now you have your MDL object and the textures in DDS format, you can now use your COLLADA object in your scenery.

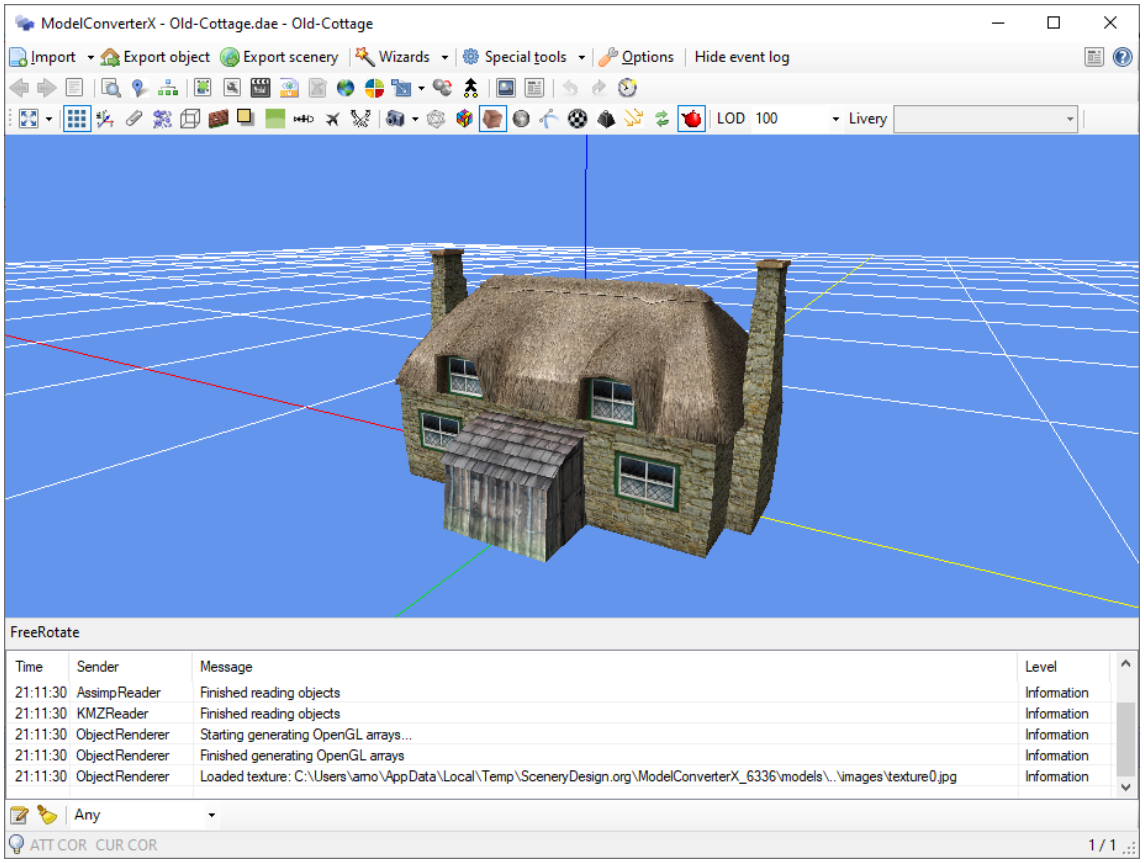


Figure 4.1: After loading the COLLADA object into ModelConverterX

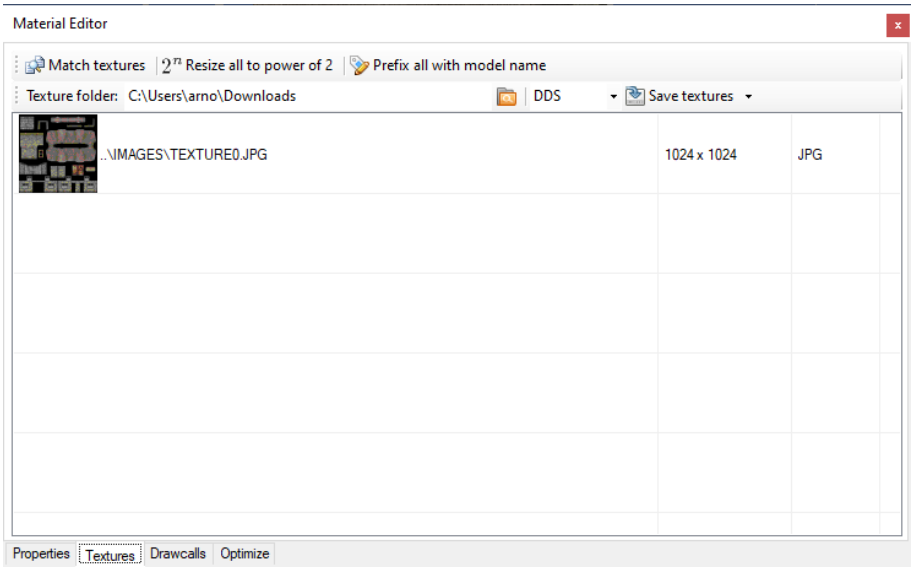


Figure 4.2: The texture tab of the material editor

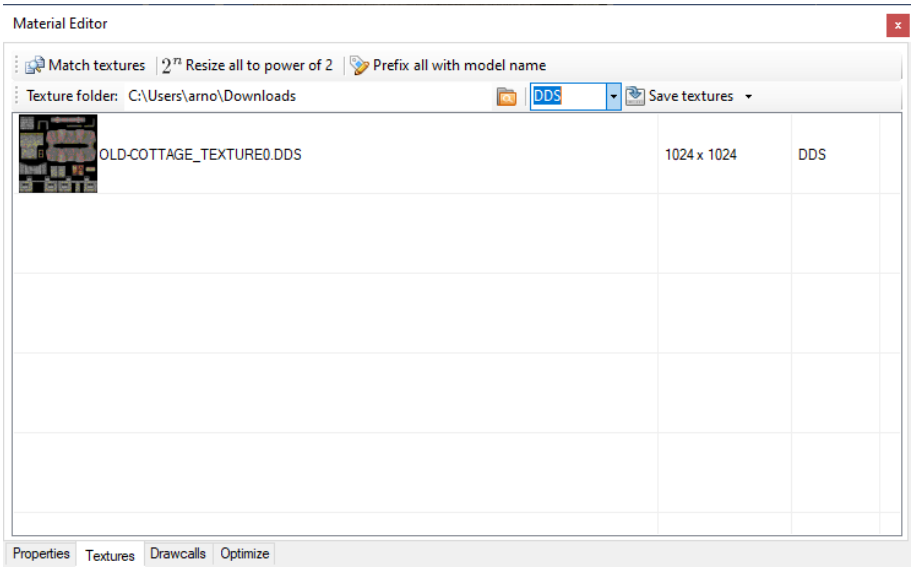


Figure 4.3: After converting textures to DDS

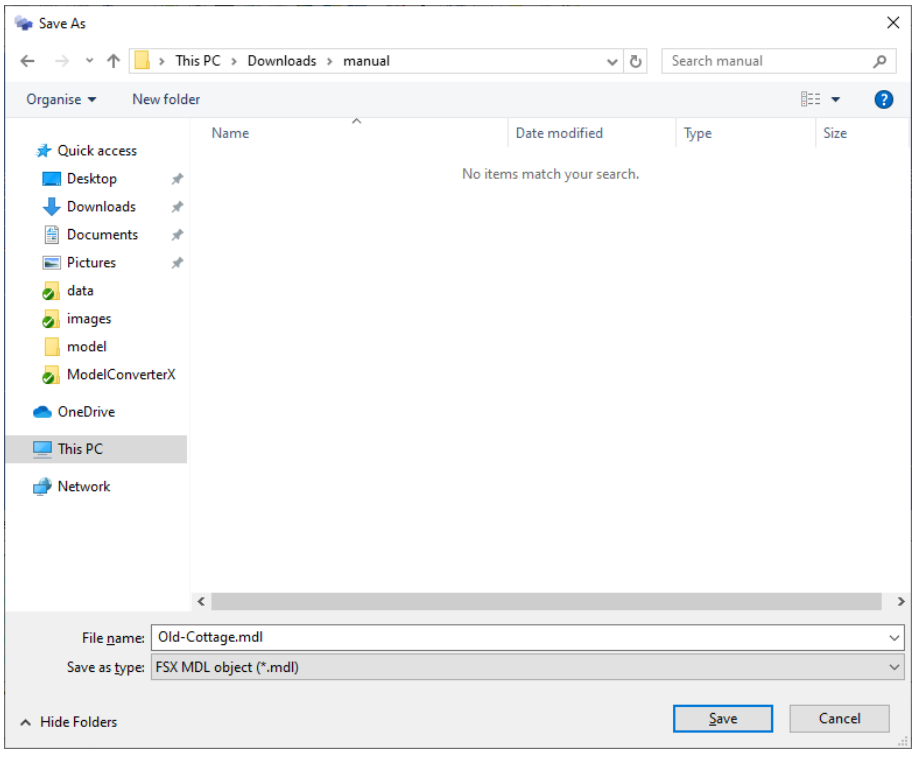


Figure 4.4: The export object dialog

4.2 Add a night texture to an object

This quick start shows you how you can add a night texture to an object that doesn't have one. For this example we will work with the cottage model that was converted to MDL in the previous section. You need to take the following steps:

1. Import the object.
2. Open the **Material editor**.
3. Go to the **Properties** tab.
4. Select the material that should have the night texture.
5. Select the **Add night texture** template, see Figure 4.5.
6. Press the **Apply** button to apply this template to the material.
7. A night texture has now been added to the material, see Figure 4.6.
8. To store the changes you made you need to export your model again to an MDL file. Use the **Export object** button to do so.

Now your model has a night texture added. You will have to create the actual texture file yourself in your favorite graphics program and put it in the texture folder of your scenery.

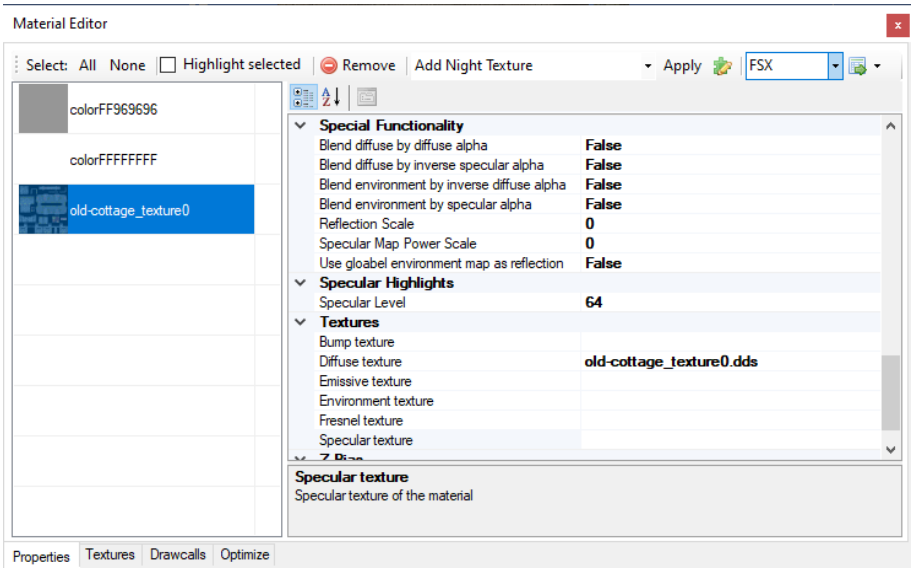


Figure 4.5: The material editor before applying the template

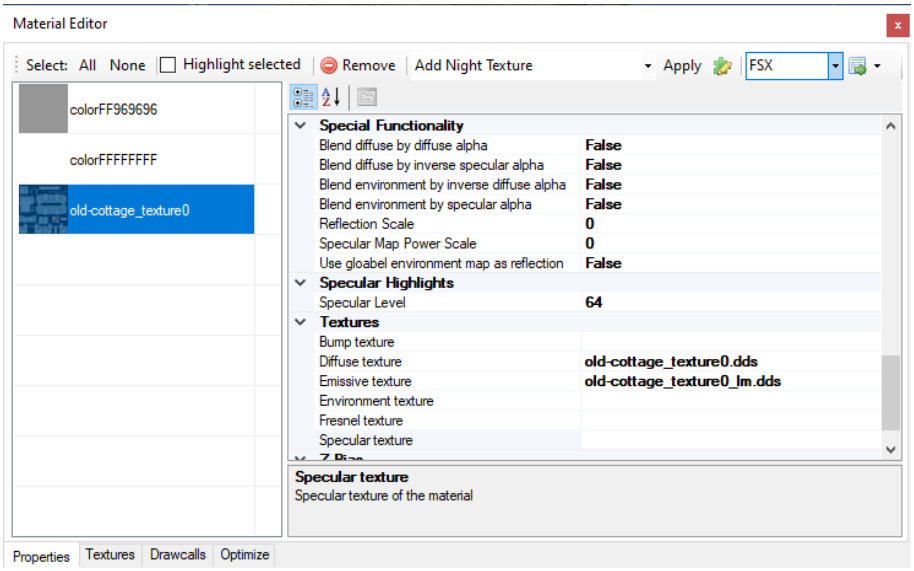


Figure 4.6: The material editor after applying the template

4.3 Add a special effect to an object

This quick start shows you how you can add a special effect to an object. We will add smoke to the chimney of the cottage object that was used in the previous sections. You need to take the following steps:

1. Import the object.
2. Make sure that **Display attached objects** is enabled in the preview. See Figure 4.7.
3. Open the **Attached object editor**.
4. Click on the **Add** button to add a new attached object.
5. Select **Effect** from the dropdown list.
6. In the properties of the attached effect select the filename of the special effect you want to use. For this example we will use fx_SmokeStack.fx, which is a default effect from FS.
7. Adjust the position of the effect so that it shows at the right location on your model. In this case we positioned the object on the chimney of the cottage. See Figure 4.8.
8. Since we want to add another effect on the second chimney, we will duplicate the attached effect with the **Duplicate** button.
9. Adjust the position of the duplicated effect to match the second chimney in the model. See Figure 4.9.
10. Close the **Attached object editor**.
11. Make sure that **Display particle effects** is enabled, so that the effect itself is rendered in the preview. It is better to leave this option disabled while you are positioning the effect, as else it is hard to see the reference cross. You will now see the object with two effects added, see Figure 4.10.
12. To store the changes you made you need to export your model again to MDL file. Use the **Export object** button to do so.

Now your model has a special effect added to it. If you are not using a default FX file, make sure you put the FX file in the effects folder of FS.

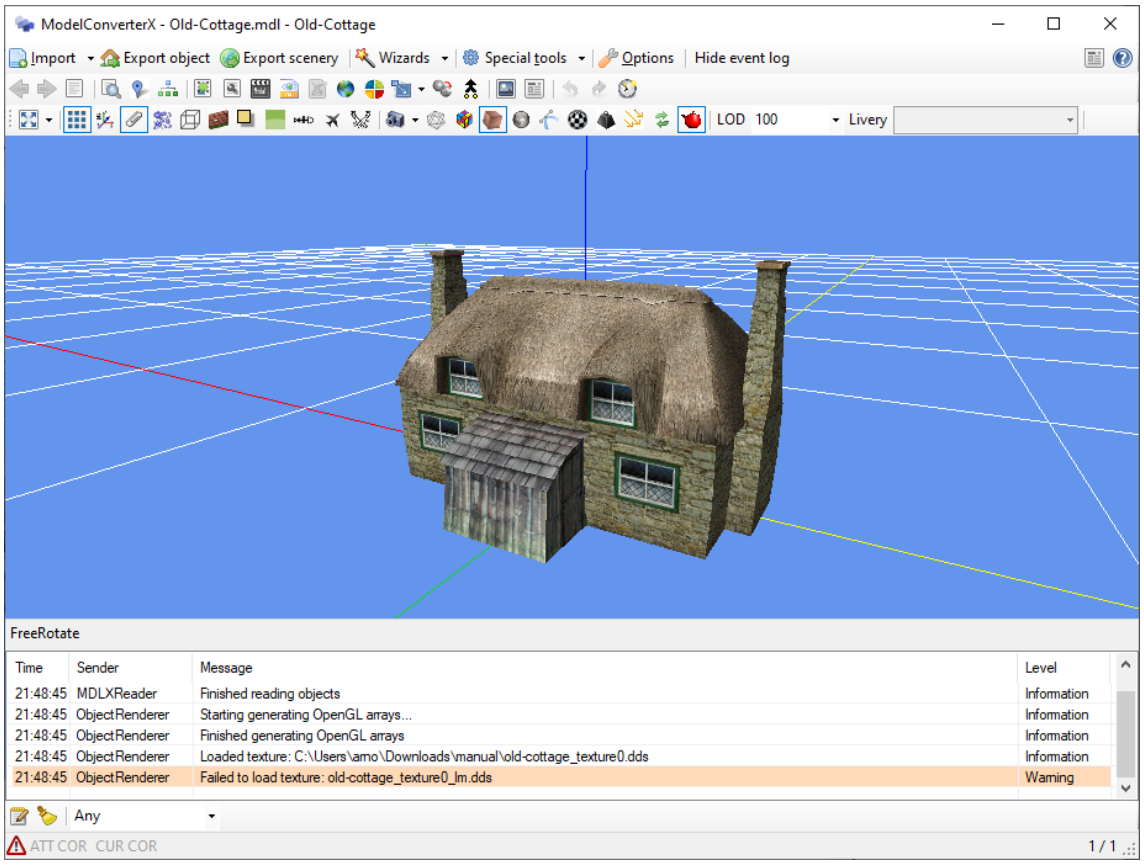


Figure 4.7: After loading the object and selecting the display of attached objects

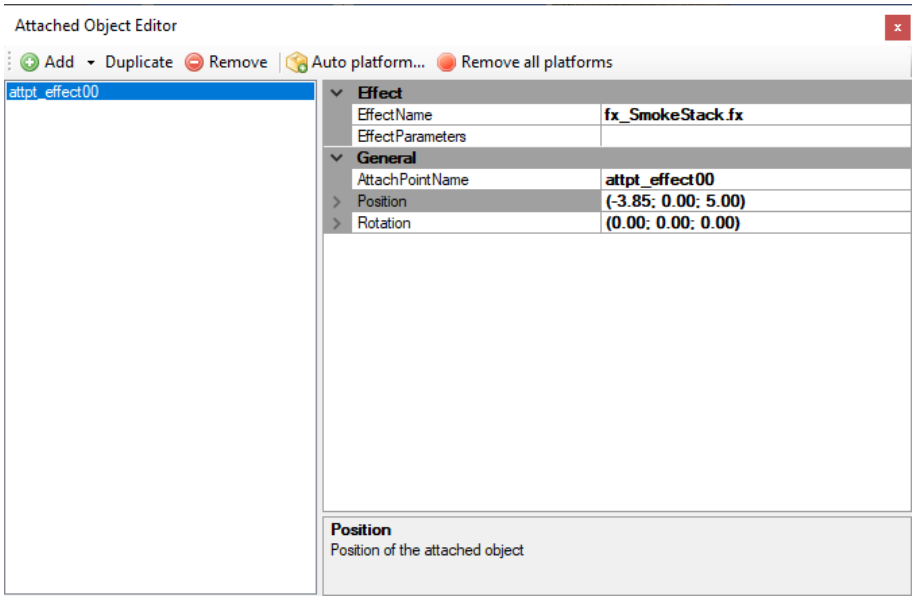


Figure 4.8: After adding the first effect

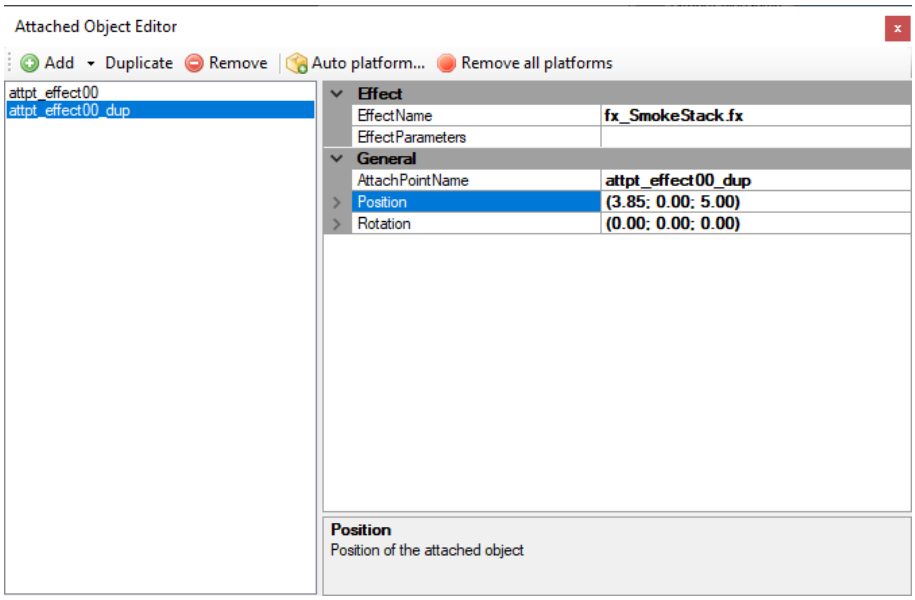


Figure 4.9: After duplicating the effect

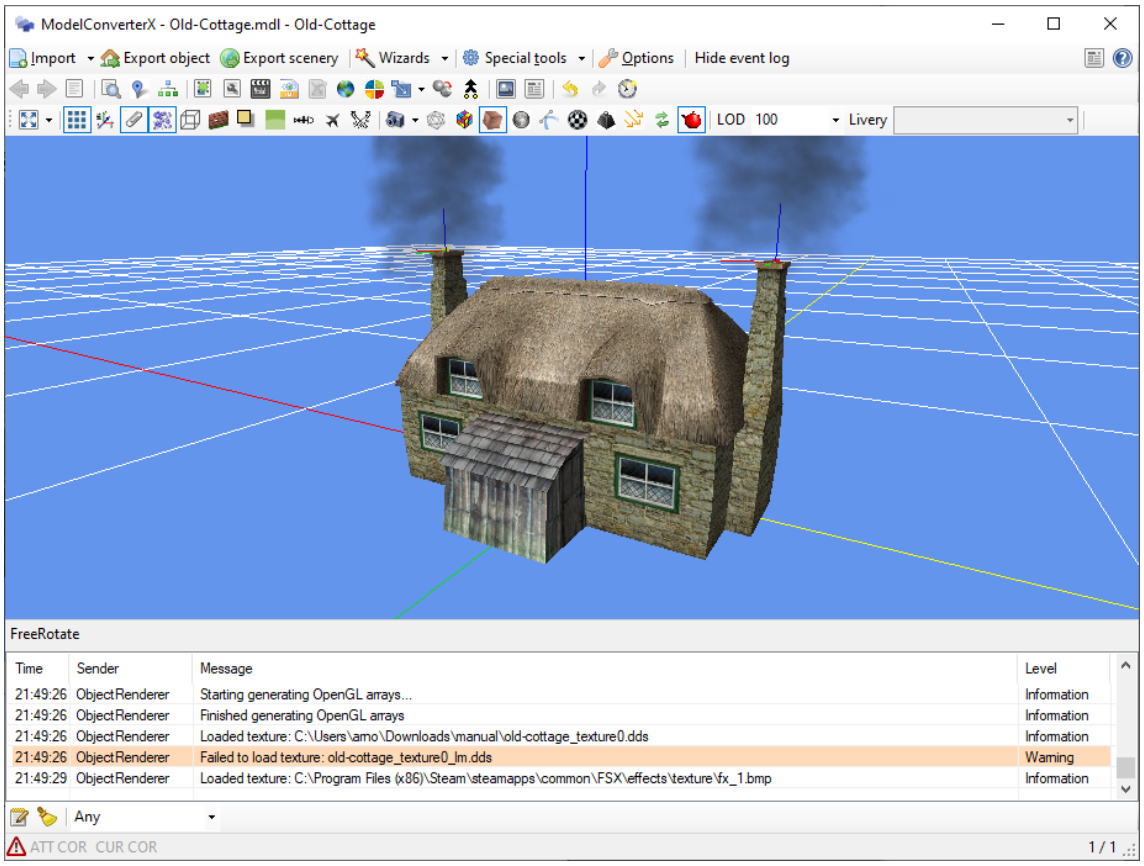


Figure 4.10: The object with the effects after selecting the display of particle effects

4.4 Add a PBR material to an object

This quick start shows you how you can update a model to a PBR material for Prepar3D v4.4. We will use the same cottage model as the previous sections. You need to take the following steps:

1. Import the object.
2. Open the **Material editor**. See Figure 4.11 for the standard properties view in **FSX** mode.
3. Select the **P3Dv44_PBR** mode to see the attributes that apply to PBR materials, see Figure 4.12. You will see that different texture types now so for the PBR material.
4. Make sure that you set the **Is PBR material** attribute to true, see Figure 4.13.
5. Add the smoothness/metallic texture and normal texture that you want to use for the PBR material see Figure 4.14.
6. To store the changes you made you need to export your model again to MDL file. Make sure to export it as a P3D v4.4 MDL. Use the **Export object** button to do so.

Now your model has a PBR material. You will have to create the actual texture files yourself in your paint program and put them into the texture folder of your scenery.

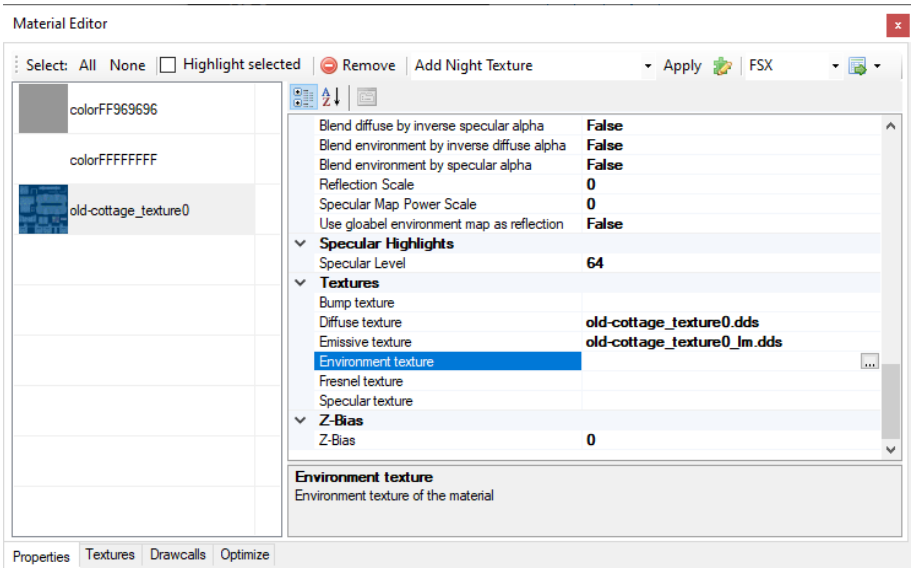


Figure 4.11: The material editor showing the FSX material attributes

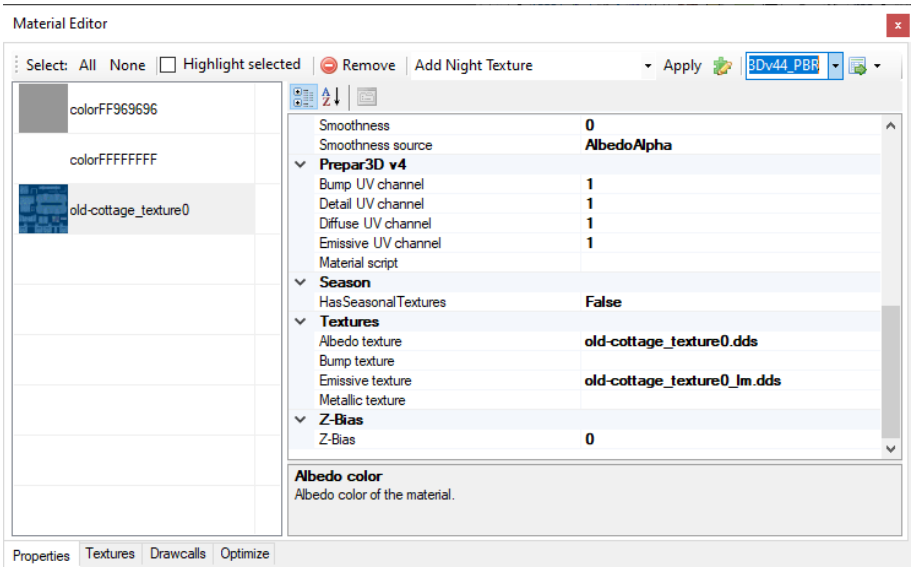


Figure 4.12: The material editor showing the P3D v4.4 PBR material attributes

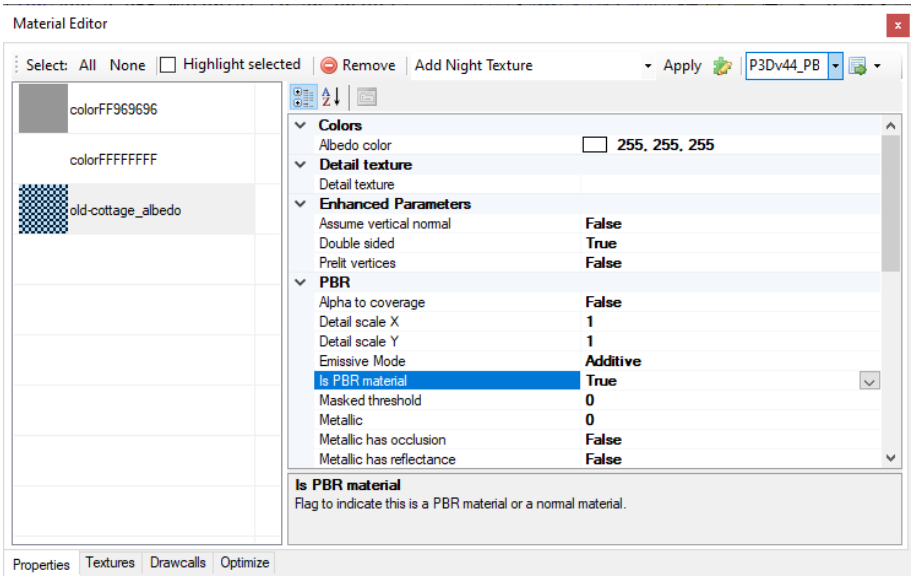


Figure 4.13: Enabling the PBR material

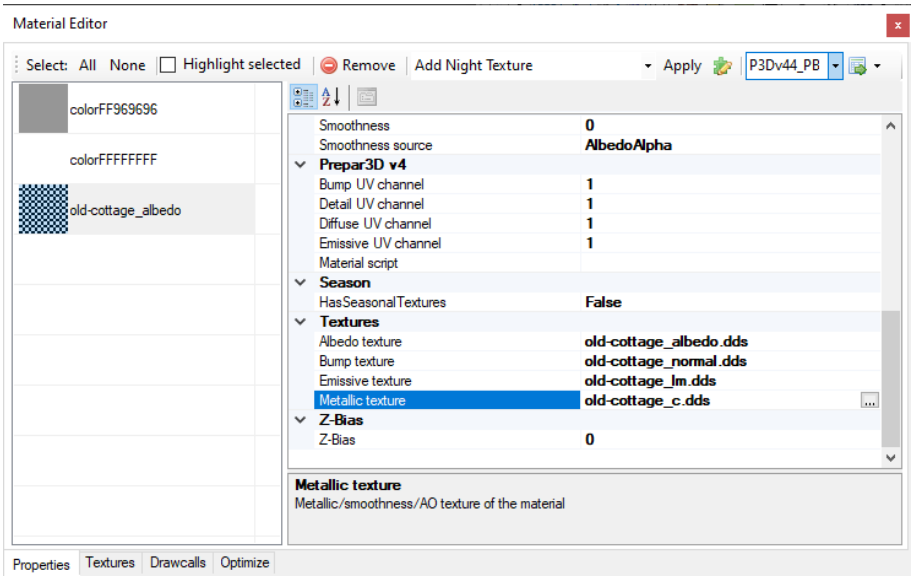


Figure 4.14: Adding the PBR textures

4.5 Add a landable platform to an object

This quick start shows you how you can add a landable platform to your object. You need to take the following steps:

1. Import the object. We will not use the cottage, but an object with 3 apartment buildings instead. See Figure 4.15.
2. Make sure that **Display attached objects** is enabled in the preview. See Figure 4.15.
3. Open the **Attached object editor**.
4. Now there are two approaches you can take, choose the one that best fits your model.
5. Manual platform
   (a) Click on the **Add** button and select **Platform** from the dropdown menu.
   (b) Change the position of the platform so that it is positioned on top and in the middle of the roof of your building, see Figure 4.16.
   (c) Change the length and width attributes so that the size of the platform matches the size of your roof, see Figure 4.16.
   (d) After some trial and error you will have your platform positioned correctly, see Figure 4.17.
6. Automatic platform
   (a) Click on the **Auto platform** button.
   (b) Select the texture that is used on the roof from the list, see Figure 4.18. If your model does not use a different texture for the roof you can skip this step and only filter the polygons of the platform based on their normal vector.
   (c) If your roof is slightly tilted, adjust the normal value used as filter. In this object the roof is flat, so we can leave the value at 1.0.
   (d) Press the **Run** button to calculate the platforms.
   (e) You now have a platform on top of all 3 buildings that exactly match the polygons of the roof texture, see Figure 4.19.

7. To store the changes you made you need to export your model again to MDL file. Use the **Export object** button to do so.

Now your model has a landable platform on top and you can land your helicopter on it.

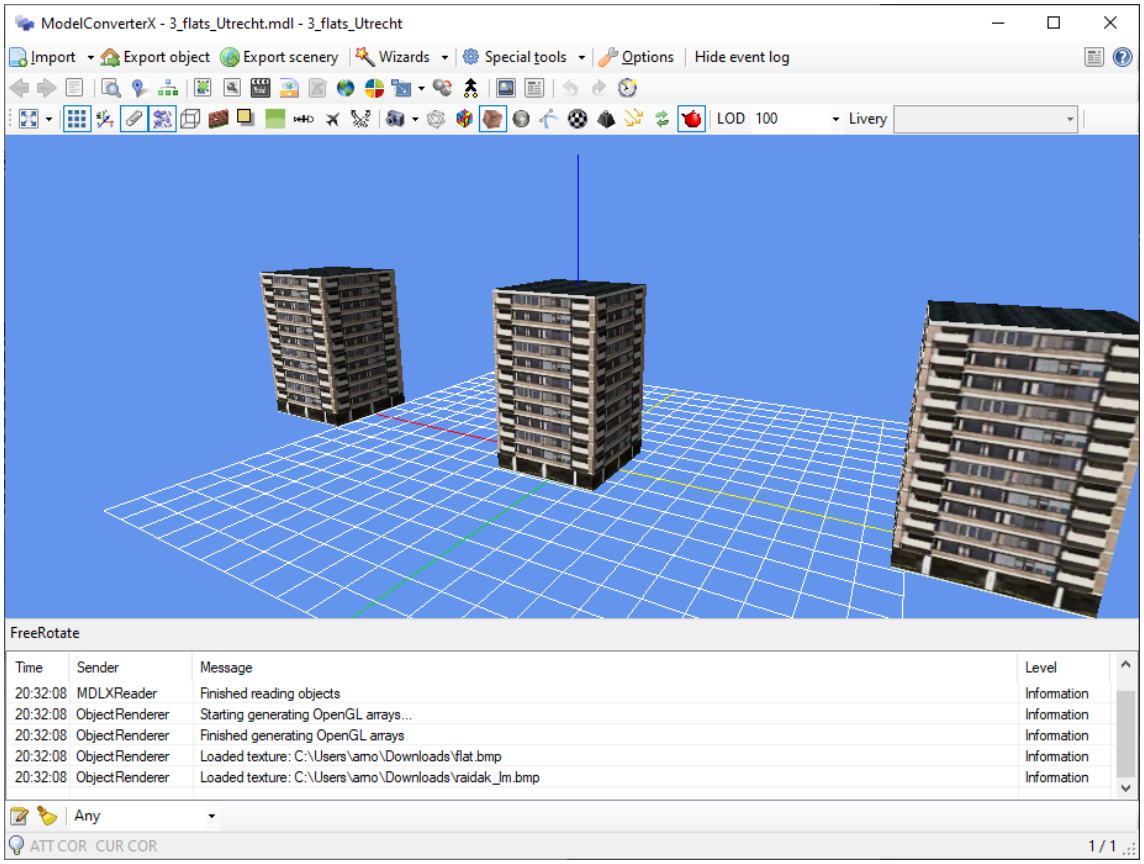


Figure 4.15: The object with 3 buildings

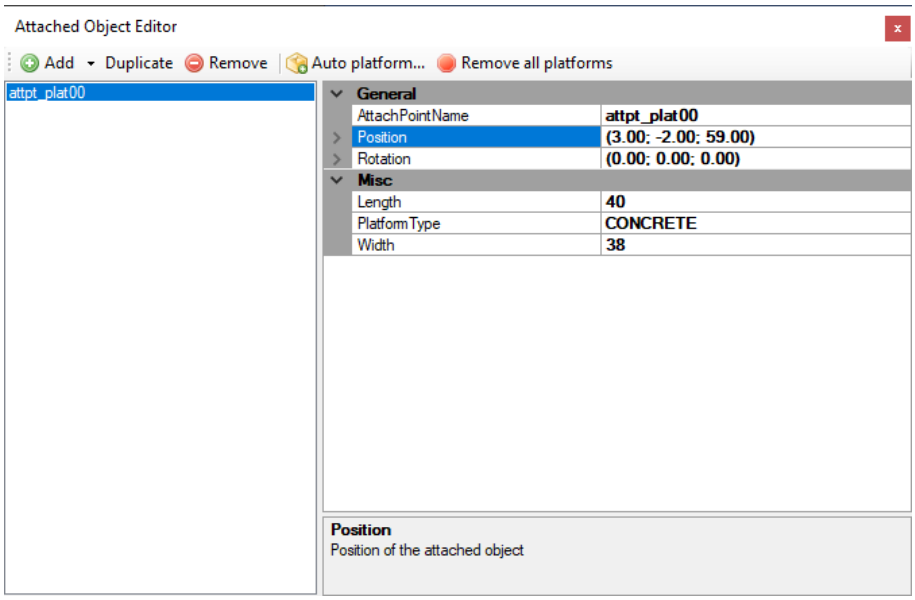


Figure 4.16: The attributes of a manual platform

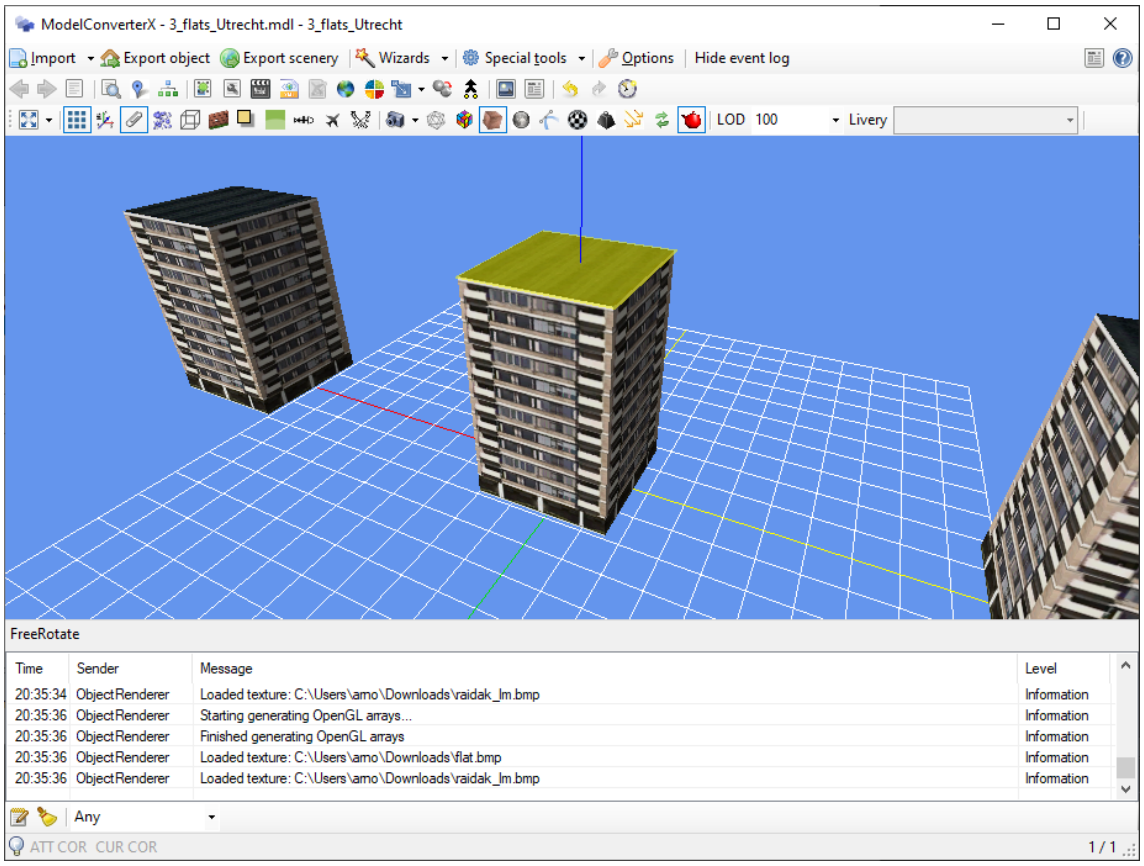


Figure 4.17: The platform on top of the middle building

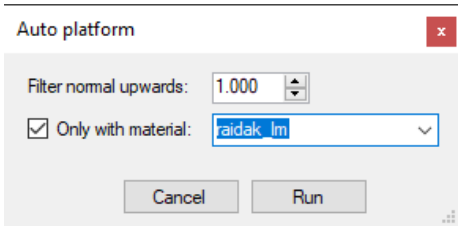


Figure 4.18: The attributes of the automatic platform

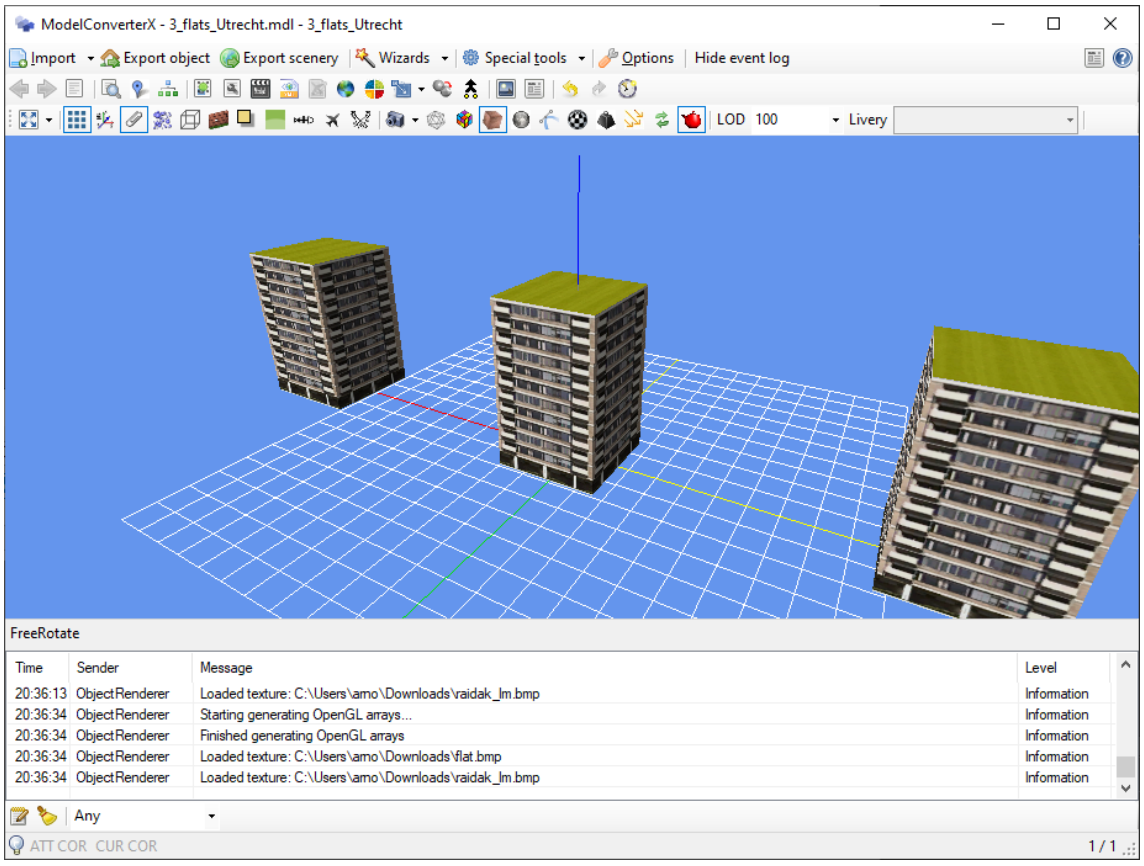


Figure 4.19: Three platforms on top of all buildings

Chapter 5

User interface

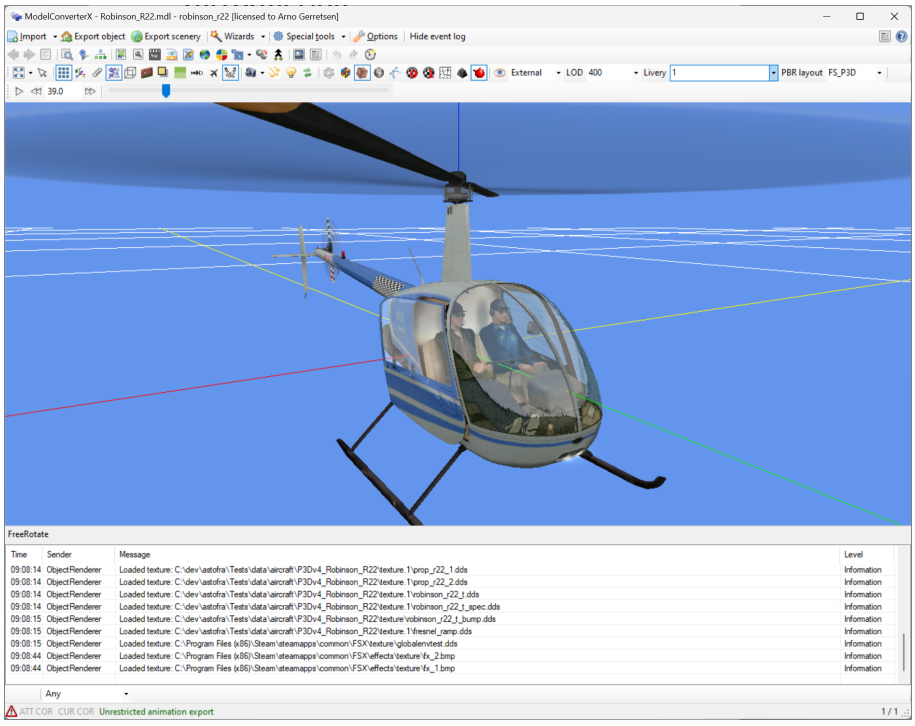


Figure 5.1: Main window of ModelConverterX

All your activities in ModelConverterX are performed from the main screen, see Figure 5.1. In this window you can identify the following elements:

- The 3D preview of the object in the center, see section 5.1 for more information about using the preview.
- The toolbar at the top from where you can perform actions and open editors. See section 5.2 for more information about the buttons on the toolbar.
- The event log with messages, warnings and errors about reading and writing objects at the bottom. See section 5.3 for more information about the event log.

5.1 Preview

The preview displays a three dimensional view of your object, see Figure 5.2. The preview in ModelConverterX tries to mimic how Flight Simulator will render you object as much as possible. Please note that PBR materials are not yet supported in the preview, so although you can set them in the material editor, the effects are not shown in the preview.

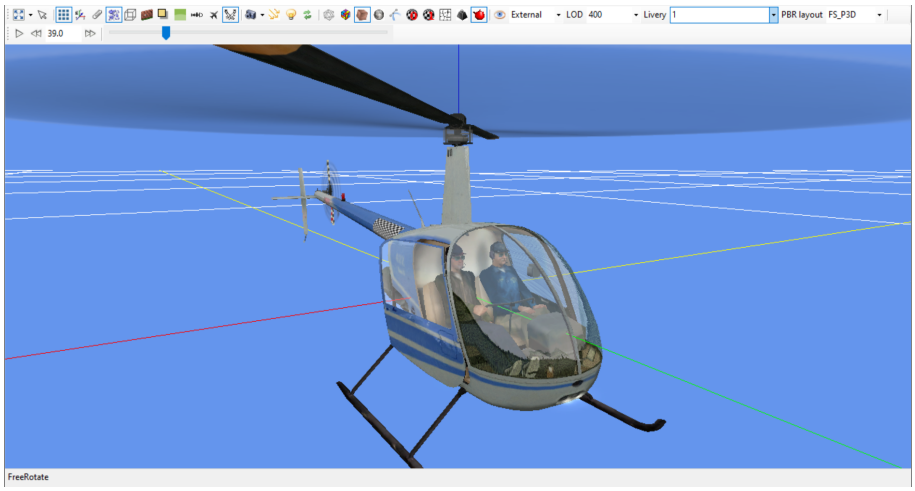


Figure 5.2: Object preview

If you drag the left mouse button you can rotate the preview around the center of the current view. If you drag the right mouse button you can pan the object, so that you can view it from a different position. With the mouse wheel or the + and - keys of the keyboard you can zoom in and out.

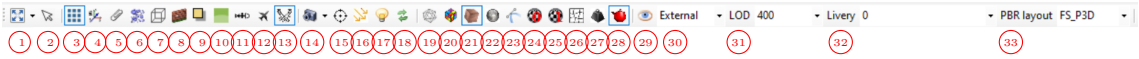


Figure 5.3: Preview toolbar

At the top of the preview is a toolbar, see Figure 5.3, where you can configure how the object in shown in the preview. The following buttons are available in this toolbar:

1. **Reset view** restores the preview to the initial position, this ensure that the entire object is within view. Using the small arrow next to the button you can also bring up a menu where you can select **Default view** as well, this can be used to set a default view position in case the object is too big and therefore zoomed out too much.
2. **Select mode** determines if the mouse can be used to select nodes in the preview. When this button is checked the select mode is active. Various editors, like the hierarchy editor, will respond to the nodes that are selected in the preview.
3. Toggles the display of the **Grid** in the preview.
4. Toggles the display of the **Normals** in the preview.
5. Toggles the display of **Attached objects** in the preview.
6. Toggles the display of **Particle effects** in the preview.
7. Toggles the display of **Bounding boxes** in the preview.
8. Toggles the display of **Crash boxes** in the preview.
9. Toggles the display of **Shadows** in the preview.
10. Toggles the display of the **Ground plane** in the preview.

11. Toggles the display of **Bones** in the preview.
12. Toggles the display of **Aircraft.cfg points** in the preview.
13. Toggles the display of **Spotlights** in the preview.
14. Sets the **Camera direction** of the preview. You can select free rotate mode or one of the 6 sides to see the object in an orthogonal view.
15. Toggles the precise control mode, when this mode is enabled the panning and zooming is done with smaller steps. This makes it easier to view details of the model.
16. Toggles the display of the **Light toolbar**, see Figure 5.4. In this toolbar you can specify the direction where the light comes from using an azimuth and elevation angle. This way you can vary the lighting direction on your object. You can also configure the strength of the diffuse and ambient lights, although it is suggested to keep the default values.
17. Toggles whether the light comes from the camera position or from the position defined in the **Light toolbar**.
18. **Reload textures** will reload all textures used on the object from this. This can be useful if you changed the textures externally from ModelConverterX.
19. Sets the rendering mode of the preview to **Wireframe**.
20. Sets the rendering mode of the preview to **Coloured faces**.
21. Sets the rendering mode of the preview to **Textured faces**.
22. Sets the rendering mode of the preview to **Night textures**.
23. Sets the rendering mode of the preview to **Normals**. In this mode the faces are coloured based on their normals, , see Figure 5.6. This can help you to spot inconsistencies in the normals.
24. Sets the rendering mode of the preview to **Texture coordinates UV0**. In this mode the faces are coloured based on their texture mapping in the first UV channel, see Figure 5.7. This can help you to spot inconsistencies in the texture mapping.
25. Sets the rendering mode of the preview to **Texture coordinates UV1**. In this mode the faces are coloured based on their texture mapping in the second UV channel.
26. Sets the rendering mode of the preview to **MSFS texture distortion**. In this mode the faces are coloured based on the amount of distortion the texture mapping has after running the MSFS package tool, see Figure 5.8. The areas shown in white have no distortion and the more blue is shown the more distortion can be experienced. See section 13.8 for more information on the cause of this distortion.
27. Sets the rendering mode of the preview to **Bone weight**. In this mode the faces are coloured based on weights of the different bones, see Figure 5.9. This can help you to check if the bone weights are assigned in balanced way.
28. Toggles the use of the **Complex shader**. When ModelConverterX uses the complex shader it will use all textures in the preview (bump map, detail map, specular map, etc.). With the simple shader only the diffuse and emissive textures are used. The simple shader can give better performance and on some older graphics cards the complex shader might not work.
29. The **Visibility condition state** button opens a dialog where you can select which visibility conditions are shown in the preview. See section 5.1.1 for more details.
30. The **Representation** dropdown list allows you to select which representation you want to be displayed in the preview.
31. The **LOD** dropdown list allows you to select which level of detail you want to be displayed in the preview.

32. The **Livery** dropdown list allows you to select which livery should be displayed on the model. This function is mainly used for aircraft models.

33. The **Texture layout style** dropdown list allows you to select which style the texture use when rendering the preview. The following styles are supported:

    - **FS_P3D** if the textures use the layout of FS2004, FSX or Prepar3D. This applies both to the non-PBR layouts of these sims and the PBR layout of Prepar3D.
    - **MSFS** if the textures use the layout of MSFS. This is the layout after optimizing by the MSFS packager tool. Compared to the glTF layout the normal texture is organized differently.
    - **glTF** if the textures use the layout according to the glTF standard. This layout is also used by MSFS glTF models before running them through the MSFS packager tool.
    - **X-Plane** if the textures use the layout of X-Plane. This applies to both the non-PBR and PBR layouts of X-Plane.

    Besides the texture layout style, this also affects how the shader renders objects with PBR materials, as ModelConverterX tries to mimic how P3D and MSFS would render such materials. When the style is FS_P3D PBR materials are shown similar to how P3D would do, with the other styles they are shown similar to how MSFS would do.



Figure 5.4: Preview light toolbar

When your object contains animations, the preview will also show an Animation toolbar, see Figure 5.5. Using this toolbar you can control the state of the animation. You can use the mouse to drag the slider to any frame desired. The following buttons are available:

1. **Play/Pause** starts the animation or pauses the animation. When the animation is playing the animation frame will automatically be increased by a timer. When you pause it the animation will stop at the current frame.
2. **Rewind** decreases the animation frame by one.
3. The **Animation frame** textbox shows the value of the current animation frame. You can also type in the number of the frame you want to see in this textbox.
4. **Forward** increases the animation frame by one.
5. With the **Frame slider** you can change the animation frame from the first to the last frame by dragging the pointer on the slides.

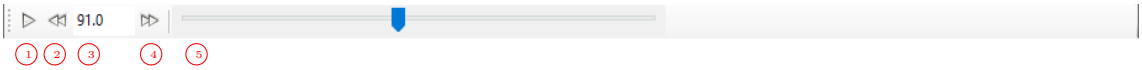


Figure 5.5: Preview animation toolbar

5.1.1 Visibility condition state

With the Visibility condition state dialog you can control the state of the visibility conditions in your model, see Figure 5.10. The preview will only show the geometry for the visibility conditions that are active. For example a propeller typically has different representations for still, slow and blurred and when you do not select with visibility condition is active all representations are shown through each other. So if you want to inspect the model in a specific state this dialog can help to configure that.

The dialog shows a list of all visibility conditions in your model. With the checkboxes in front you can determine if they are currently active or not in the preview. In the toolbar at the top the following controls can be found:

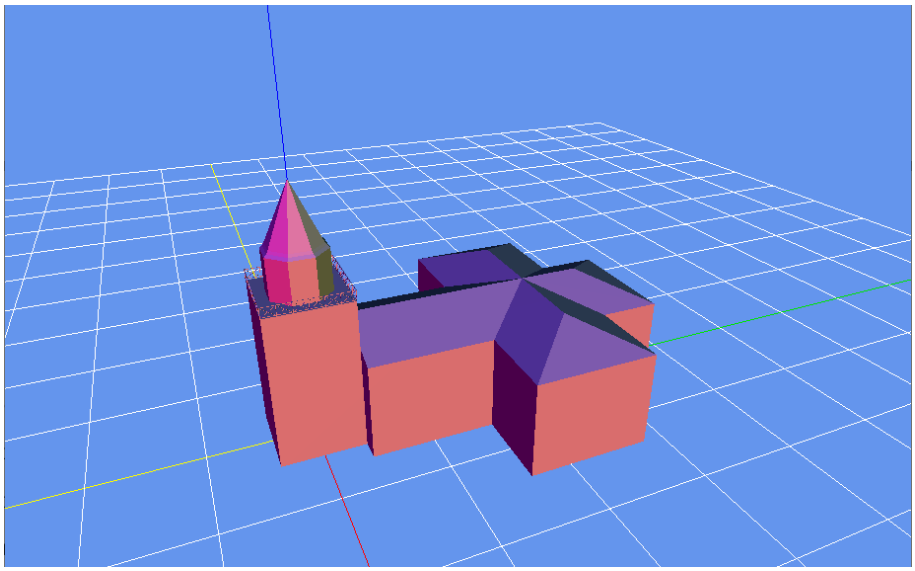


Figure 5.6: Normal rendering mode

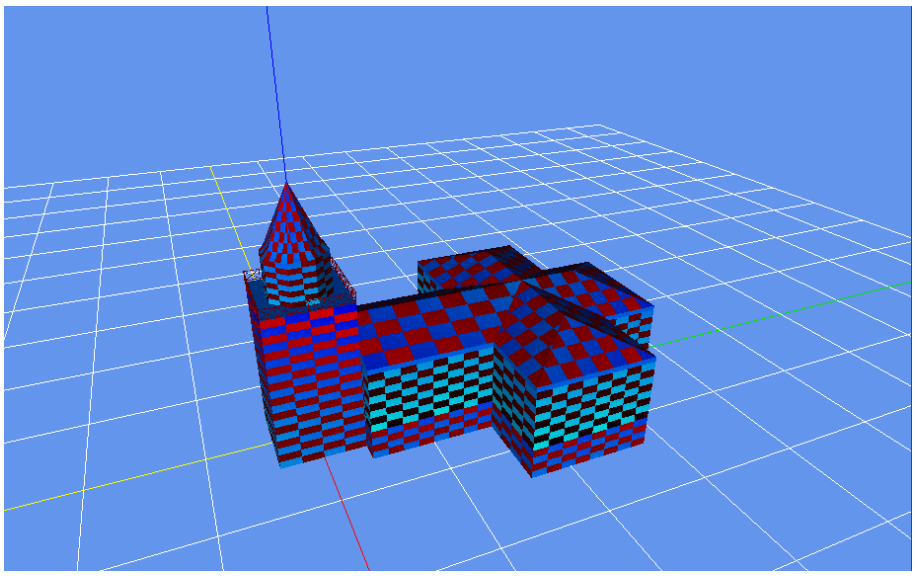


Figure 5.7: Texture coordinate rendering mode

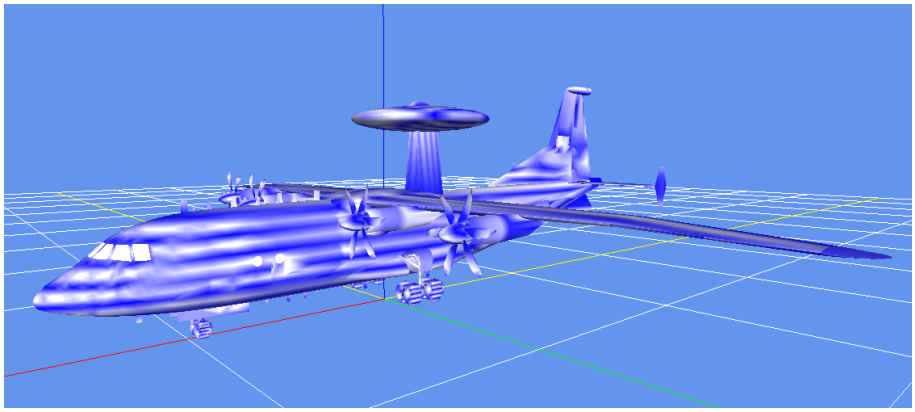


Figure 5.8: MSFS texture distortion rendering mode

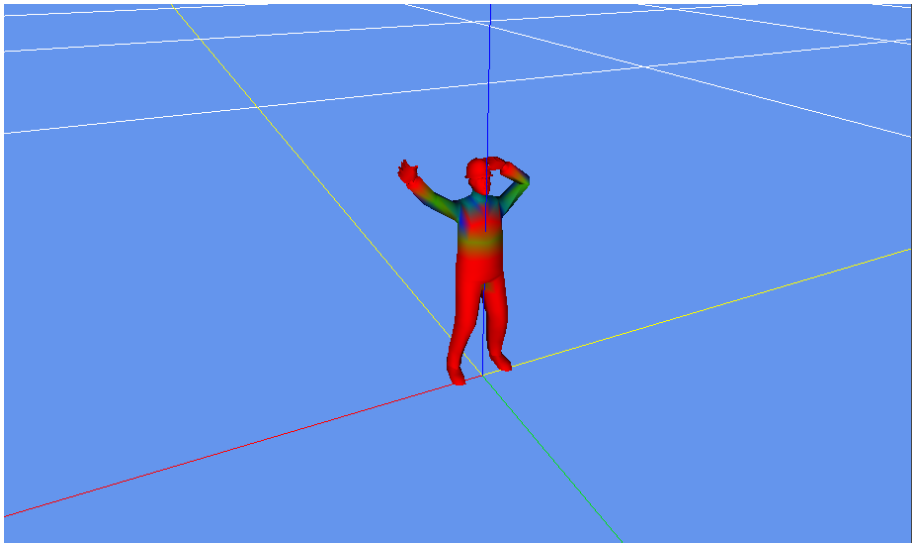


Figure 5.9: Bone weight rendering mode

- Select **All** makes all visibility conditions active in the preview.
- Select **None** makes none of the visibility conditions active in the preview.
- The list normally shows all visibility conditions in the model. If you want to search in them you can use the **Filter** textbox to only show the visibility conditions that contain the text you have entered as a filter. For example if you enter `tire` only the visibility conditions with tire in the name will show. This can make it easier to find a specific visibility condition. if you have a filter applied the select All and select None button work on the filtered list of visibility conditions as well.
- **Clear filter** does clear the currently entered filter, as a result all visibility conditions of the model are shown in the list again.

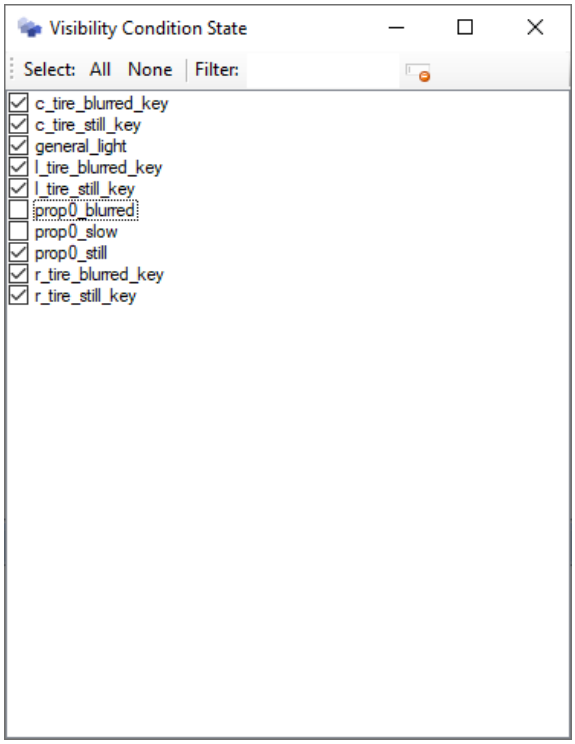


Figure 5.10: Visibility condition state

5.2 Toolbar

The buttons on the toolbar, see Figure 5.11 are used to begin most actions in ModelConverterX. The toolbar consists of two rows. The first row contains buttons to import and export objects and to start wizards. The second row contains the buttons to start specific editors on the loaded object.

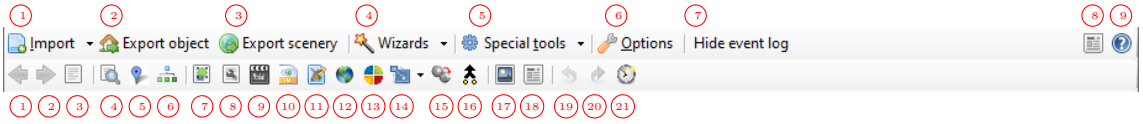


Figure 5.11: Toolbar

In the top row you find the following buttons, they are marked with the red number at the top in Figure 5.11:

1. The **Import** button allows you to select a file that you want to load into the tool. This can be a file that contains a single object, but also a file that contains a whole scenery. See section 9.1 for more information about the supported formats.

2. The **Export object** button allows you to export the currently selected object to various formats. See section 10.1 for more information about the supported formats.

3. The **Export scenery** button allows you to export an entire scenery, e.g. all objects and placement information, to various formats. See section 10.2 for more information about the supported formats.

4. The **Wizards** button opens a menu with the available wizards. The wizards in general provide an easy way to perform certain tasks. See chapter 7 for more information about the available wizards.

5. The **Special tools** button opens a menu where you can select various tools. See chapter 8 for more information about the available tools.
6. The **Options** button opens the options window, where you can set all the options for ModelConverterX. See chapter 11 for more details.
7. The **Hide event log** button hides the event log, so that more screen space is available for the preview.
8. The **Manual** button opens this PDF manual.
9. The **About** button opens the about window that shows information about the version of ModelConverterX that you are using.

In the bottom row you find the following buttons, they are marked with the red number at the bottom in Figure 5.11:

1. **Previous object**, activates the previous object in the scenery, when it contains multiple objects.
2. **Next object**, activates the next object in the scenery, when it contains multiple objects.
3. **Scenery object editor**, see section 6.1.
4. **Object information editor**, see section 6.2.
5. **Object placement editor**, see section 6.3.
6. **Hierarchy editor**, see section 6.4.
7. **Material editor**, see section 6.5.
8. **Attached object editor**, see section 6.6.
9. **Animation editor**, see section 6.7.
10. **ModelDef.xml editor**, see section 6.8.
11. **Aircraft.cfg editor**, see section 6.9.
12. **Earth curve correction editor**, see section 6.10.
13. **Season editor**, see section 6.11.
14. **Transform object editor**, see section 6.12.
15. **Level of detail creator**, see section 6.13.
16. **Merge object editor**, see section 6.14.
17. **Generate object image**, see section 6.15.
18. **Generate object report**, see section 6.16.
19. **Undo**, undos the last change made to the object.
20. **Redo**, redos the last undone change to the object.
21. **Change history**, see section 6.17.

5.3 Event log

In the event log, see Figure 5.12, you can see all the messages that the preview, object reader or object writer modules have generated. There are three types of messages in the event log:

- **Error** messages, which indicates that something really went wrong and ModelConverterX has stopped reading or writing your object. These kind of errors must be fixed before you can continue.

- **Warning** messages, which indicates that something went wrong during reading or writer your object, but that ModelConverterX tried to continue. It might be that the operation can still finish, but you should check if the warning is a problem for you.
- **Information** messages, that just inform you of something that happened. For example that a texture file has been loaded or that an export has started.

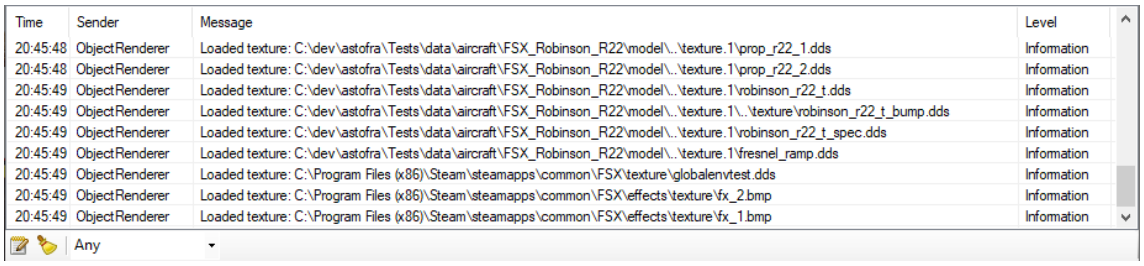


Figure 5.12: Event log

At the bottom of the event log you will find a toolbar with actions for the event log. With the **Save** button you can save the content of the event log to a TXT file. With the **Clear** button you can empty the event log. And using the **Filter** dropdown list you can select which type of messages you want to see. If you select `Any` all messages will show, if you select either `Information`, `Warning` or `Error` only messages with the appropriate event log level will be displayed.

5.4 Status bar

The status bar at the bottom of the ModelConverterX window gives you feedback about the the activities currently taking place, see Figure 5.13. During the import or export you will for example see a progress bar to indicate how far that process is at the moment.

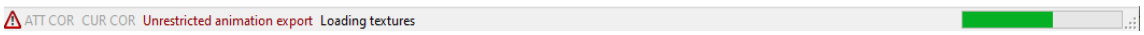


Figure 5.13: Status bar

In addition, in the left corner of the status bar an icon can be displayed to indicate the event log status. If there is no icon the event log is empty, but if there are entries in the log the following icons can be displayed:

- Indicates that there is at least one information message in the event log.
- Indicates that there is at least one warning message in the event log, there can also be information messages present.
- Indicates that there is at least one error message in the event log, there can also be information or warning messages present.

Next to the event log status icon three status labels are displaced:

- **ATT COR** indicates if the attached objects of this object have been correct for the curvation of the earth already. When this text shows in black the attached objects have been corrected already and when it shows in gray they have not.
- **CUR COR** indicates if the object has been correct for the curvation of the earth already. When this text shows in black the object has been corrected already and when it shows in gray it has not.
- **Restricted/Unrestricted animation export** indicates if the animations in this object can be exported to any format. When the text Restricted animation export is shown in red you can only export the animations to X, MDL or BGL files. When the text Unrestricted

animation export is shown in green you can export them to any format (that supports animation export). To prevent piracy ModelConverterX does not allow you to export animations imported from X, MDL or BGL files to open formats, these can only be exported back to the compiled formats of FS.

5.5 Keyboard shortcuts

Most actions in ModelConverterX are performed with the mouse. But to make it easier to perform certain common actions the keyboard shortcuts as shown in Table 5.1 are also available.

Action	**Shortcut**
Alt	
Toggle attached objects display	Alt-A
Toggle grid display	Alt-G
Toggle normals display	Alt-N
Toggle animation play	Alt-P
Reset preview	Alt-R
Ctrl	
Next object	Ctrl-left arrow
Previous object	Ctrl-right arrow
Open attached object editor	Ctrl-E
Open hierarchy editor	Ctrl-H
Open object information editor	Ctrl-I
Open scenery object editor	Ctrl-J
Open animation editor	Ctrl-K
Open level of detail creator	Ctrl-L
Open list of recently opened files	Ctrl-O
Open generate object report	Ctrl-S
Open material editor	Ctrl-T
Open generate object image	Ctrl-W
Redo	Ctrl-Y
Undo	Ctrl-Z
Move object	Shift-M
Rotate object	Shift-R
Scale object	Shift-S
Preview	
Precise (fine) control of pan and rotate in preview	f
Activate select mode in preview	s
Move left in preview	Left arrow
Move right in preview	Right arrow
Move forward in preview	Up arrow
Move backward in preview	Down arrow
Move up in preview	PageUp
Move down in preview	PageDown
Zoom in	=
Zoom out	-

Table 5.1: Keyboard shortcuts

Chapter 6

Editors

This chapter explains the different editors that are available in ModelConverterX. Their functions are described and how you can use them.

6.1 Scenery object editor

The scenery object editor gives an overview of all the objects in the current scenery, see Figure 6.1. They are shown in a list with their name and GUID. If you click on an object that object will be loaded in the preview window of ModelConverterX.

With the **Add object** button you can add an additional object to the scenery. You will be given a file selection dialogue to select the object you want to add to the scenery. With the **Remove object** button you can remove the currently selected object from the scenery. With the (Replace Object) button you can replace the selected object with an updated object. This retains the original GUID for that object.

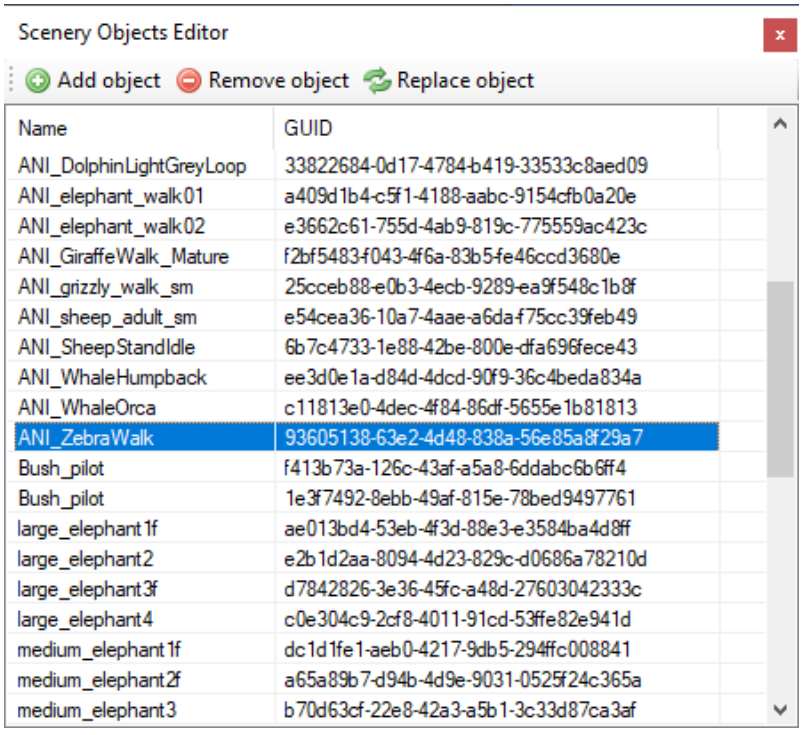


Figure 6.1: Scenery object editor

6.2 Object information editor

The object information editor displays the basic information about the object and allows you to change part of this information, see 6.2. The list below shows which information is available, those marked with a * can also be edited after you press the **Edit** button:

- Name*.
- GUID*. With the **Set new GUID** button a new random GUID is assigned to the object.
- Bounding box minimum and maximum values for each axis. When the override checkbox is selected an alternative bounding box can be provided in the numeric fields below that will be used to override the automatically calculated one. Certain exporters (especially the Prepar3D XtoMDL) support overriding the bounding box in the object on export.
- Radius of the object. When the override checkbox is selected an alternative radius can be in the numeric input field that will be used to override the automatically calculated one. Certain exporters (especially the Prepar3D XtoMDL) support overriding the radius in the object on export.
- The number of animations, mouse rectangles and visibility conditions in the model.
- Drawcall information, the number of drawcalls, triangles and texture vertices for each level of detail of the currently active representation of the object. If the **Show details** checkbox is checked one line of drawcall information will be shown for each material, instead of grouped per level of detail (see Figure 6.3).
- List of all textures in the model.
- List of all attachpoints in the model.

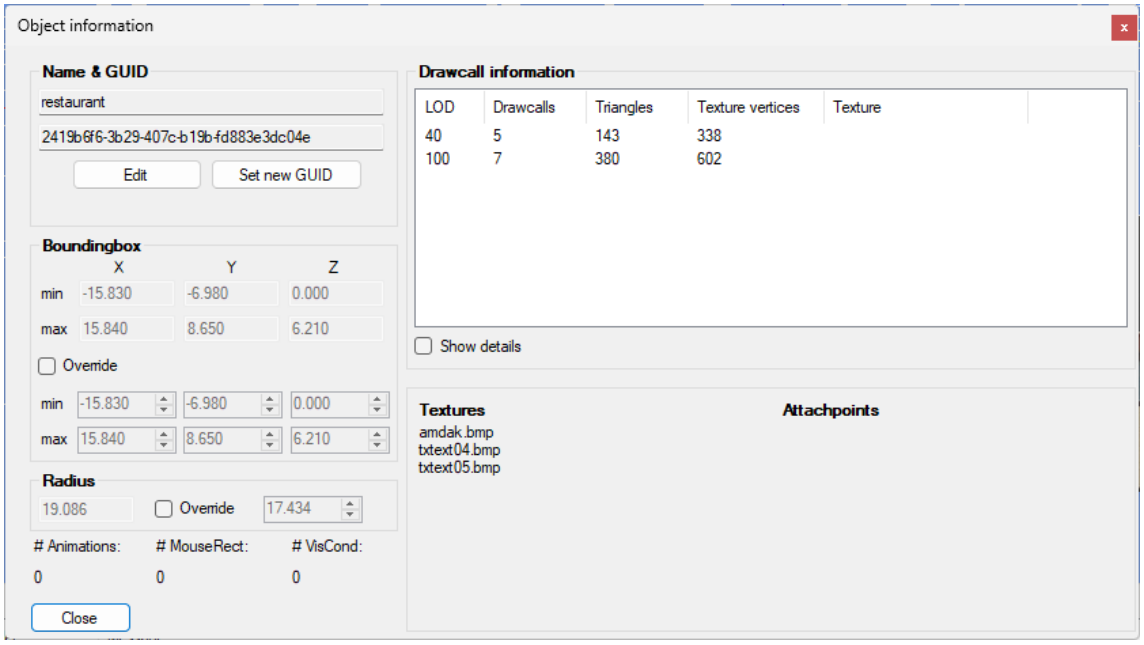


Figure 6.2: Object information editor

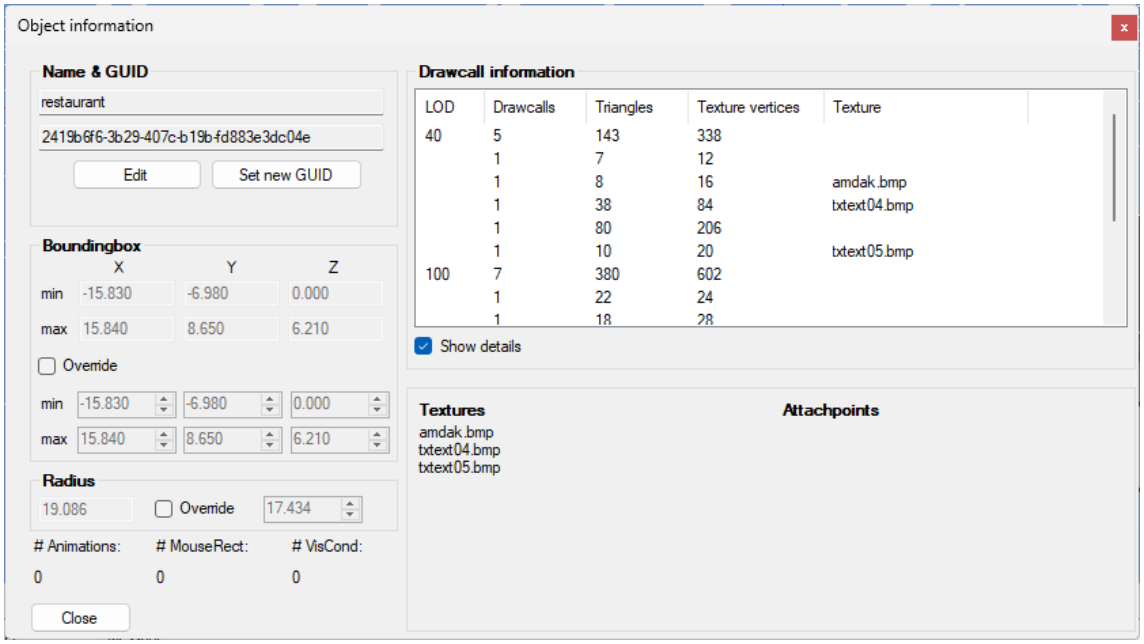


Figure 6.3: Object information editor details

6.3 Object placement editor

The object placement editor allows you to place the selected object at a specific location and also to view all locations that the object has already been placed at, see 6.4. It should be remembered that ModelConverterX is not a full functional object placement tool, this editor has been made for simple use cases where you want to quickly position the objects that are currently loaded in ModelConverterX. Therefore this means that you can only place objects in which the model is contained in the same scenery file.

On the left of the window you see a map with pins showing the locations where the object is placed. Different types of pins are used:

- A red crossmark is used to show the active object placement
- A red placemark is used to show other placements of the same object
- A green placemark is used to show the placement of other objects in the same scenery.

With the selection list at the top of the map, you can select different sources for the map. For example from OpenStreetMap, Google or Bing. You can also select if you want to see a map or imagery in the map view.

If you drag with the left mouse button you can update the location of the active object placement. If you drag with the right mouse button you can pan the map to a different location. With the mouse wheel you can zoom the map in and out.

On the right side of the editor you see a property list where you can edit the other properties of the object placement. This includes the heading, altitude, scenery complexity level and the various placement flags.

With the **Add** button you can add another object placement for the active object, while with the **Remove** button you can remove the currently selected object placement. If the active object has multiple placements, you can use the < and > buttons to cycle through them. If you check the **Random heading** checkbox each new object placement that you add will get a random heading assigned. This can be useful if you are scattering tree objects for example.

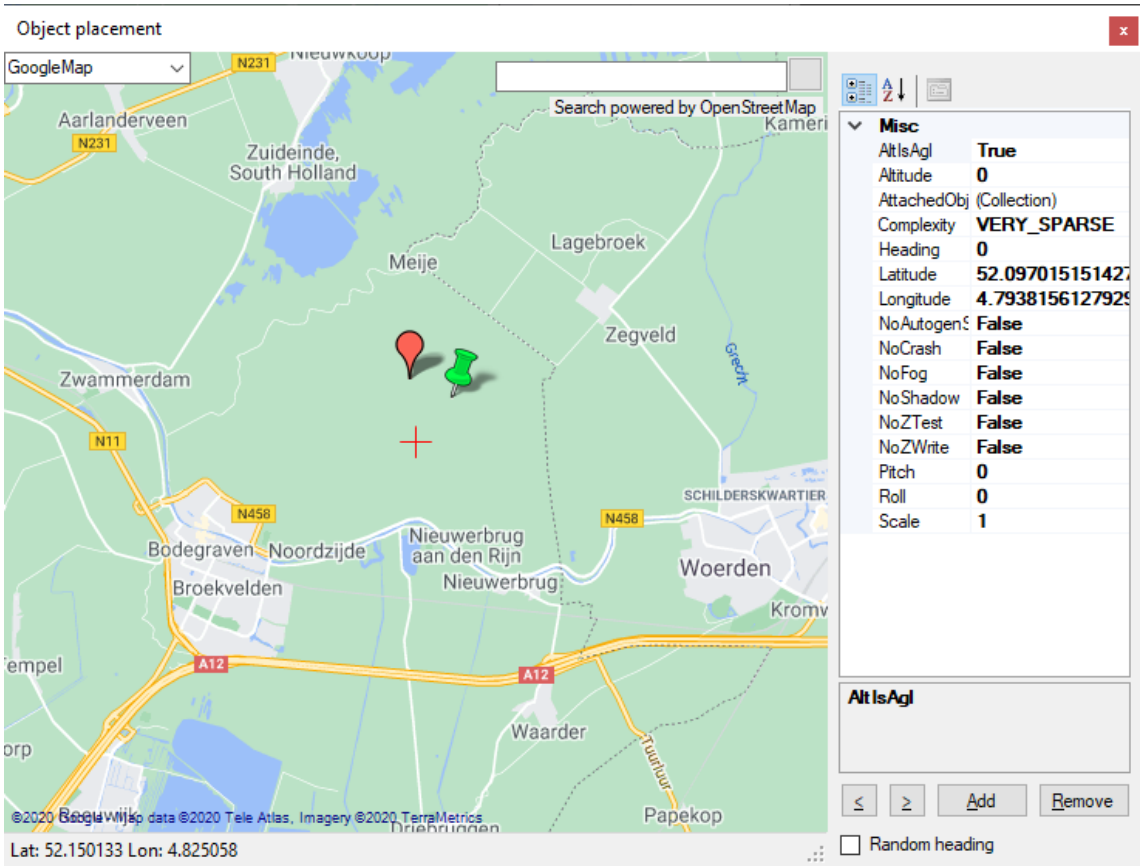


Figure 6.4: Object placement editor

6.4 Object hierarchy editor

The object hierarchy editor shows you the hierarchy of the active representation of your object, see Figure 6.5. Each object is built up from various types of nodes. The following types are supported by ModelConverterX:

SceneGraphNode All node types are also a SceneGraphNode. A basic SceneGraphNode defines the level of detail, can have transformations and animations assigned and can have children nodes to define a hierarchy.

ModelPart This node defines the actual geometry of the object. It has a collection of triangles and uses a material.

AttachedEffect This node attaches a special effect to the object.

AttachedLight This node attaches a light to the object.

AttachedSpotLight This node attaches a spotlight to the object.

AttachedLibraryObject The node attaches a library object to the object.

AttachedPlatform The node attaches a hard platform to the object.

Bone This node defines a bone that can be used in a skin and bone animation.

CFGPoint This node defines a point as defined in the aircraft.cfg file.

On the left side of the hierarchy editor window you see a treeview of the node hierarchy, so you can see which children nodes a given node has. When you click on a node in the treeview, the panel on the right side will show all kind of details of the selected node. Depending on the node type the following information can be listed:

- Representation. The model representation that the node is part of, this is either External, Internal or Shadow. If you click on the name you get a dropdown list and you can change the representation of the node.
- Level of detail.
- Transformations. If a warning symbol (exclamation mark in a triangle) is shown behind the transformation label this means the transformation might give issues. The tooltip will show you what is wrong. It could be it contains a mirror operation or that it contains non-uniform scaling.
- Animations. If you click on the name of the animation a dropdown list will appear where you can select a different animation type to assign. All animations that are present in the loaded modeldef.xml file are shown.
- Visibility conditions. If you click on the name of the visibility condition a dropdown list will appear where you can select a different visibility condition to assign. All visibility conditions that are present in the loaded modeldef.xml file are shown. Select None to remove the visibility condition.
- Mouse rectangles. If you click on the name of the mouse rectangle a dropdown list will appear where you can select a different mouse rectangle to assign. All mouse rectangles that are present in the loaded modeldef.xml file are shown. Select None to remove the mouse rectangle.
- Bounding box.
- Material. If you click on the name of the material a dropdown list will appear where you can select a different material to assign. All materials present in the current object model are shown. If you hold the Control key down while clicking the new material a clone of the material is made and assigned to the part. This allows you to modify the material without affecting other parts that use the same material. If you hold the Control and Shift keys while

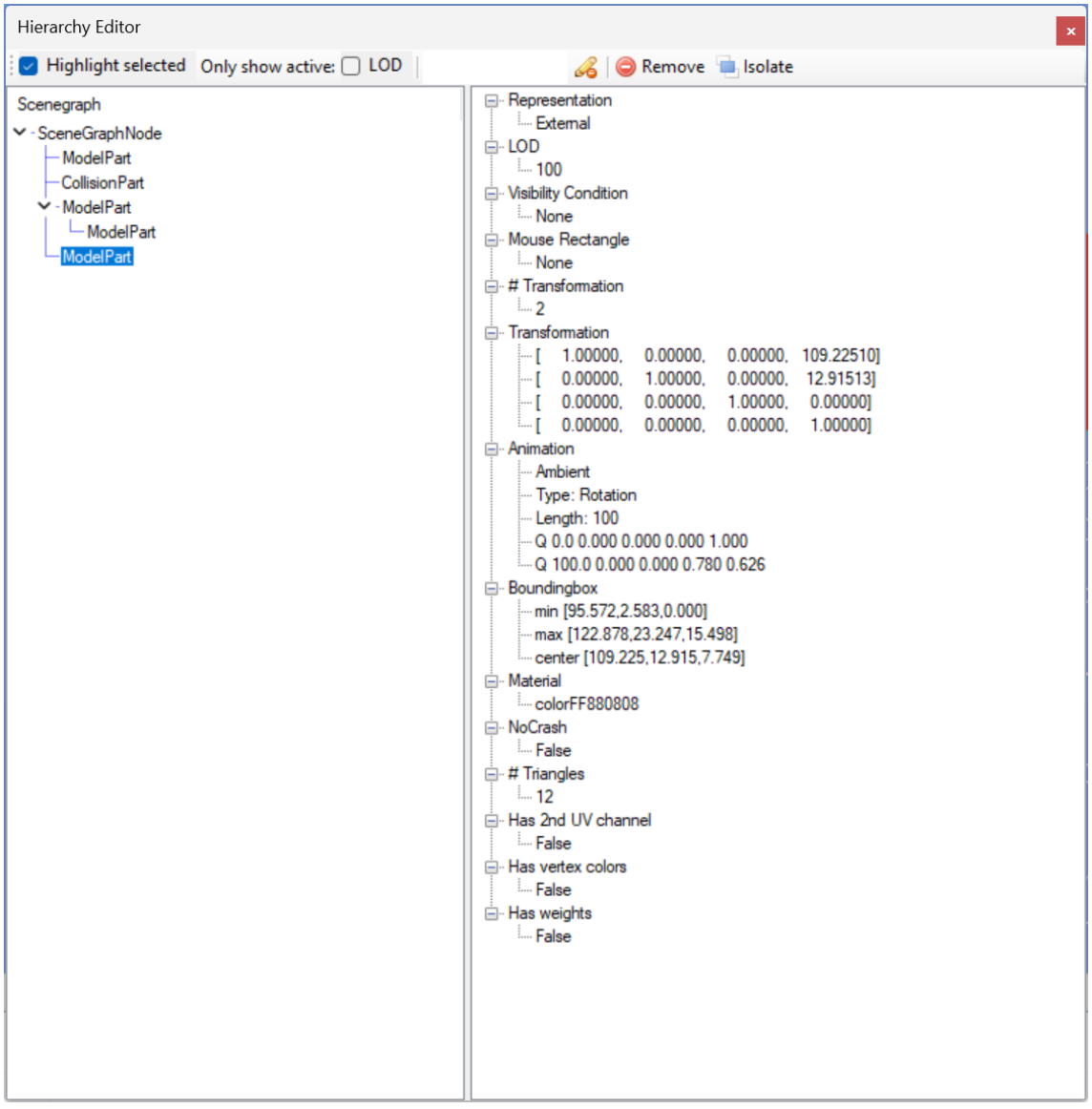


Figure 6.5: Hierarchy editor

clicking the new material the clone of the material is applied to all parts in the LOD that have the same original material.

- Triangle count.
- Bones.
- Vertices.

The **Highlight selected** checkbox in the toolbar determines if the node you select in the hierarchy editor is also highlighted in red in the object preview. When you check the **Only show active LOD** checkbox in the toolbar only the nodes of the currently active level of detail in the object preview will be shown in the hierarchy editor.

Using the search box in the toolbar you can search for a specific node. You can for example type part of a texture, attachpoint or visibility condition name and then only the nodes are shown in the treeview that contain this name. This can be useful to find a specific node quickly. If you enter a word like prop the editor will select only the parts tagged as propeller parts.

When the **Remove** button is pressed you can delete the currently selected node(s) from the object. When the **Isolate** button is pressed only the currently selected objects are retained and those will be used as the new object.

If you select a node in the treeview of the hierarchy editor you can also drag it onto another node. This will make the selected node (and all of its children nodes) a child of the node onto which you drop the node. This can be useful if you want to attach a special effect to an animated node of the object for example. When you drag a node to the top or the bottom of the list, it will automatically scroll. This allows you to drag a node onto a node that is currently not visible in the list.

The following nodes have a context menu to perform additional actions:

- Right clicking on a node in the treeview on the left gives you the following options, see Figure 6.6. Some of these are only available when you click on a modelpart node.
  - **Add child node** adds a new empty scenegraph node as child to the selected node.
  - **Add transformation** adds a new transformation to the selected node.
  - **Add animation** adds a new animation to the selected node.
  - **Flat shade** will flat shade the selected modelpart, see section 8.16.3 for more information on what this does.
  - **Smooth shade** will smooth shade the selected modelpart, see section 8.16.4 for more information on what this does.
  - **Remove vertex colors** removes all vertex colors from the selected modelpart.
  - **Material color to vertex colors** applies the color of the material of the modelpart to the vertex colors and sets the material color to white afterwards.
  - **Normalize texture coordinates** tried to normalize all texture coordinates of the selected modelpart, so that they are in the range of 0 to 1.
  - **Add UV2** adds a secondary UV channel to the modelpart, by making a copy of the texture coordinates in the primary UV channel.
  - **Remove UV2** remove the secondary UV channel from the modelpart.
  - **Flip UV1 vertically** flips the UV1 texture coordinates of the modelpart vertically.
  - **Flip UV2 vertically** flips the UV2 texture coordinates of the modelpart vertically.
  - **Edit vertices...** brings up the vertex editor, where you can modify the data of the vertices of the selected modelpart. The following data can be edited (column name in shown in brackets behind):

* Position (x, y, z)
* Normal (nx, ny, nz)
* Texture coordinates (tx0, ty0)
* Secondary texture coordinates (tx1, ty1)
* Bone indices and weights (bi1, bi2, bi3, bi4, bw1, bw2, bw3, bw4)
* Vertex colors (vcr, vcg, vcb, vca)

Figure 6.7 shows how the editor looks. Values can be modified by typing a new value in the relevant cell. When pressing **Apply** the changes are stored in the model, with **Cancel** the changes are discarded.

– **Paste transformation/animation** pastes the last copy transformation or animation onto the selected scenegrpah node.

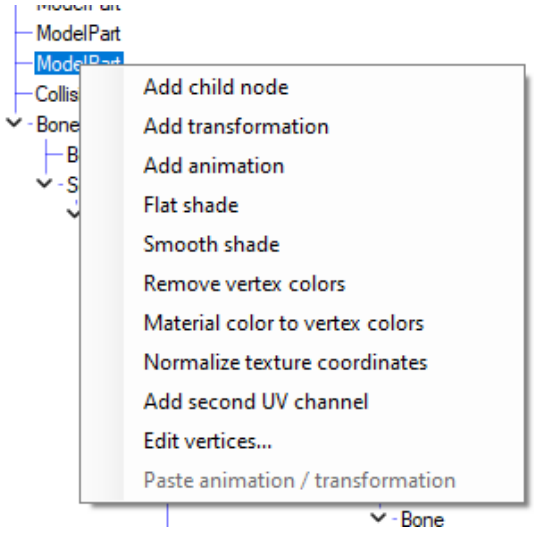
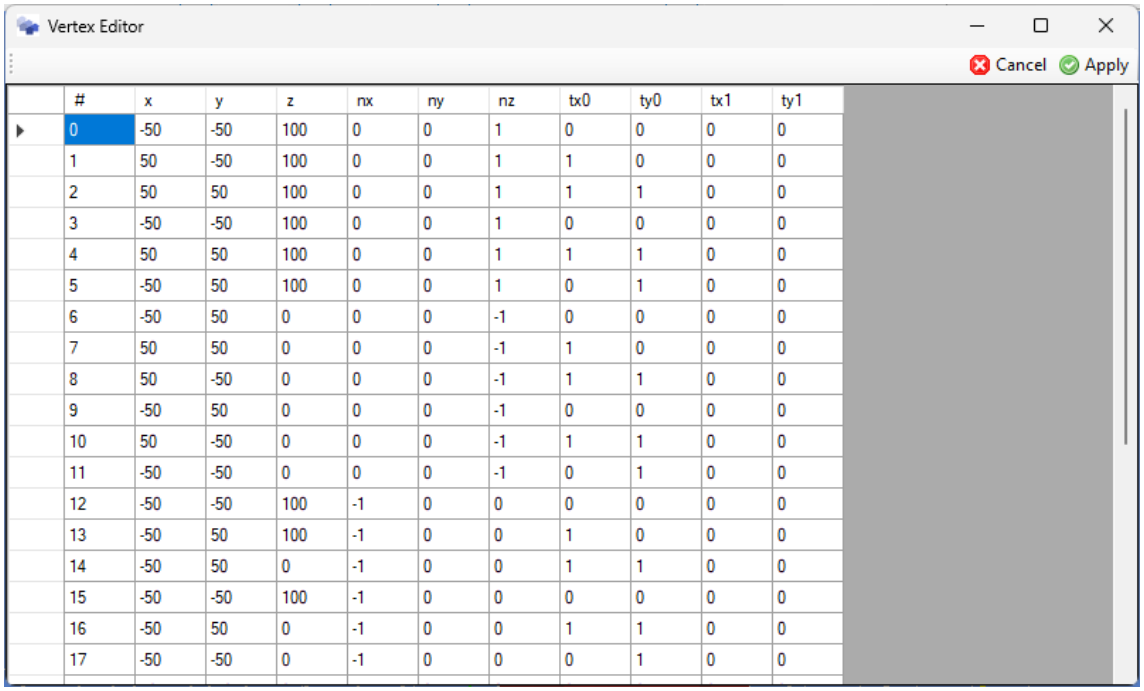


Figure 6.6: Scenegraph node context menu

- Right clicking on the **Transformation** label gives you the following options, see Figure 6.8.
  – **Edit transformation** opens a window where you can manipulate the transformation, see Figure 6.9. In the transformation editor you see the 4x4 transformation matrix at the top and you see a decomposition into translation, rotation and scaling at the bottom. You can edit either representation of the transformation. The dropdown list at the bottom also allows you to select the rotation order that is used when decomposing the rotation, see section 13.2 for more details. Once you click **OK** to close the transformation editor window, the transformation matrix in the hierarchy editor will be updated with your new values.
  – **Delete transformation** removes the selected transformation from the node.
  – **Copy transformation** makes a copy of the selected transformation so that it can be pasted onto another scenepgraph node.
- Right clicking on the **Animation** label gives you the following options, see Figure 6.10.
  – **Reverse animation** reverses the order of the keyframes of the animations.
  – **Edit animation** opens the animation editor window, where you can manipulate the animation, see Figure 6.11. In this editor you see all the animation keys of the animation. You can change the time, translation or rotation values of the keys. You can also add new keys or remove existing keys. To help you modify the rotation quaternion keys,



#	x	y	z	nx	ny	nz	tx0	ty0	tx1	ty1
0	-50	-50	100	0	0	1	0	0	0	0
1	50	-50	100	0	0	1	1	0	0	0
2	50	50	100	0	0	1	1	1	0	0
3	-50	-50	100	0	0	1	0	0	0	0
4	50	50	100	0	0	1	1	1	0	0
5	-50	50	100	0	0	1	0	1	0	0
6	-50	50	0	0	0	-1	0	0	0	0
7	50	50	0	0	0	-1	1	0	0	0
8	50	-50	0	0	0	-1	1	1	0	0
9	-50	50	0	0	0	-1	0	0	0	0
10	50	-50	0	0	0	-1	1	1	0	0
11	-50	-50	0	0	0	-1	0	1	0	0
12	-50	-50	100	-1	0	0	0	0	0	0
13	-50	50	100	-1	0	0	1	0	0	0
14	-50	50	0	-1	0	0	1	1	0	0
15	-50	-50	100	-1	0	0	0	0	0	0
16	-50	50	0	-1	0	0	1	1	0	0
17	-50	-50	0	-1	0	0	0	1	0	0

Figure 6.7: Vertex editor

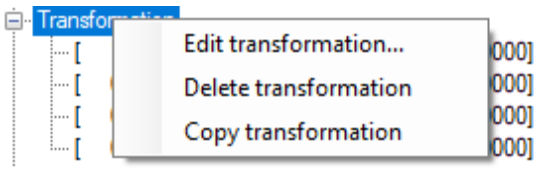


Figure 6.8: Transformation context menu

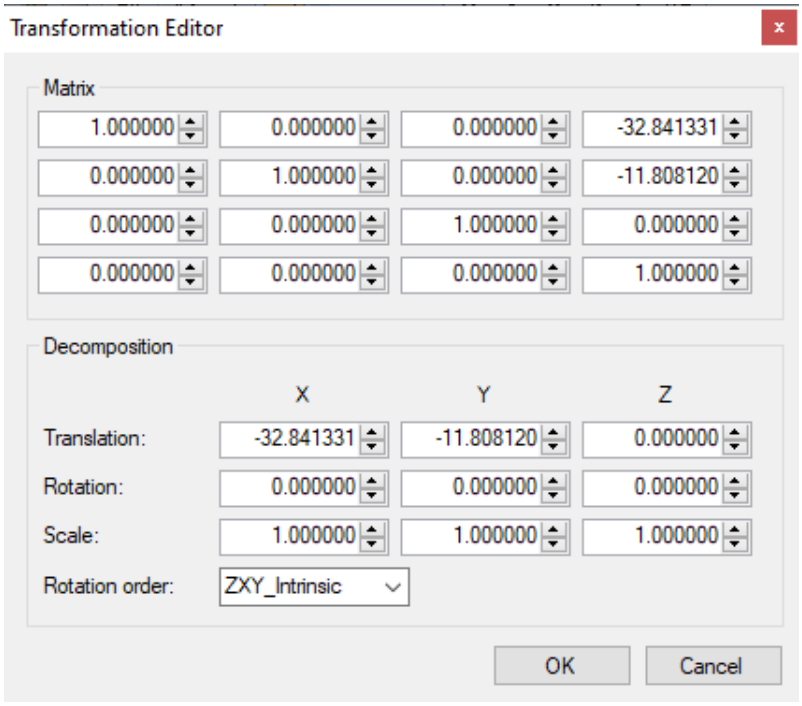


Figure 6.9: Transformation editor window

these are decomposed into rotations along the axes. With the dropdown list you can select the rotation order used for the decomposition, see section 13.2 for more details. Once you click **OK** to close the animation editor window, the animation in the hierarchy editor will be updated with your changes.

– **Delete animation** removes the selected animation from the node.

– **Copy animation** makes a copy of the selected animation so that it can be pasted onto another scenepgraph node.

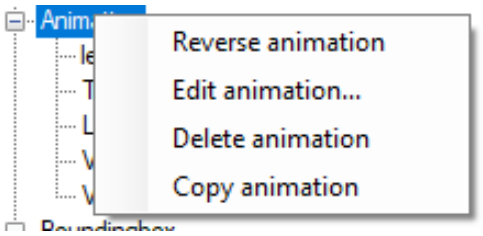


Figure 6.10: Animation context menu

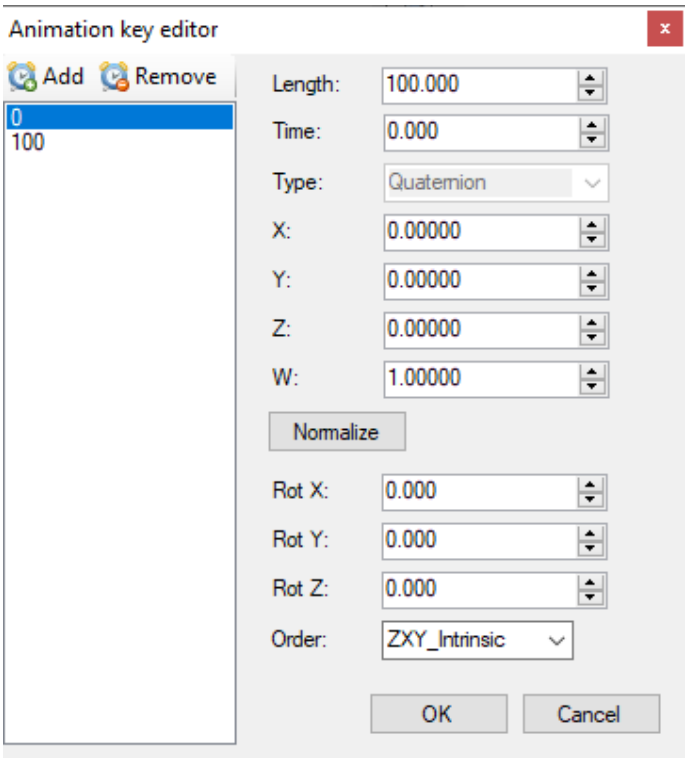


Figure 6.11: Animation editor window

6.5 Material editor

The material editor is the place to look for information and to modify the materials and textures used by the object. It shows all materials and textures of the active representation of the object. This editor has four tabs that allow you to edit different aspects of the material. Each of these tabs is discussed in the sections below.

6.5.1 Properties tab

The properties tab, see Figure 6.12 gives you a list of all materials in the object and allows you to modify the properties of each of them. On the left side of the tab you see the list of materials, while on the right side you see the properties of the selected material(s).

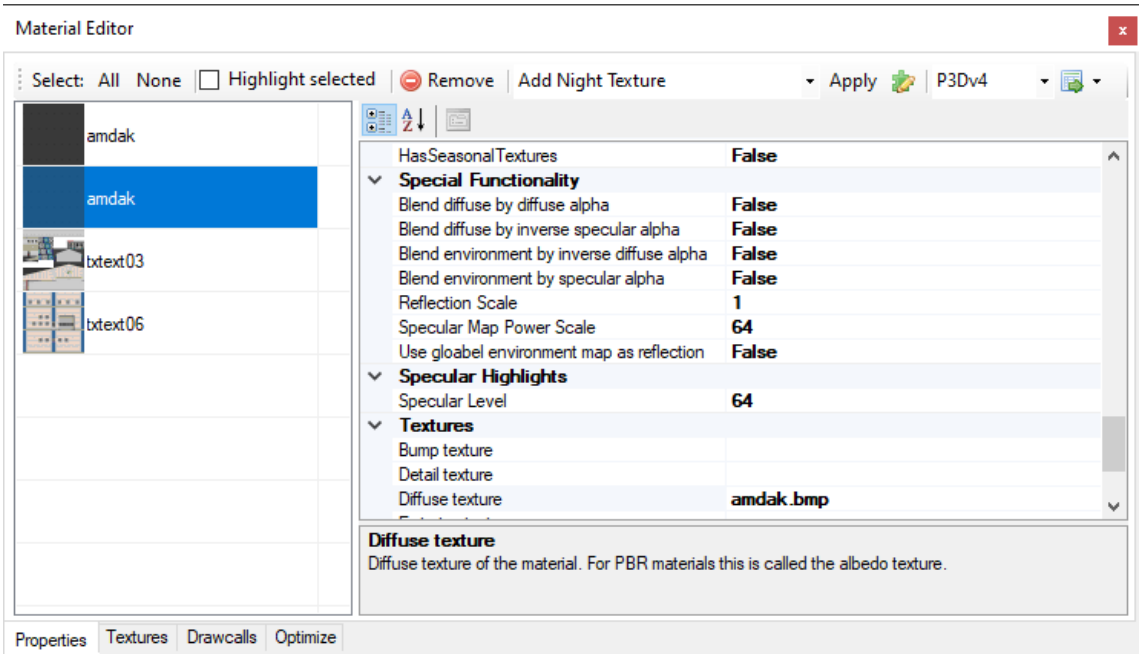


Figure 6.12: Material editor properties tab

If you have the **Highlight selected** checkbox checked the material that you have currently selected in the material editor will be highlighted in red in the preview of the object.

You can select multiple materials at once using the Control key, so that you can give a property the same value quickly. With the **Select All** and **Select None** buttons in the toolbar you can quickly select all materials or none.

With the **Remove** button you can remove the selected materials from the object. Removing the material will also remove the geometry that uses this material from the object.

There are many properties that can be edited for a material, but not all of these properties are applicable to each Flight Simulator version. Therefore there is a filter dropdown list at the right side of the toolbar. If you select a specific filter here, only the material properties that are applicable for that filter are shown. The available filters are:

- **All** shows all material properties.
- **Generic** shows only the basic material properties that are supported by almost any format on export. This includes the standard colours and textures for example.
- **FS2004** shows all properties supported by FS2004 models.
- **FSX** shows all properties supported by FSX models.
- **MSFS** shows all properties supported by MSFS models.
- **P3Dv4** shows all properties supported by P3D v4 models.
- **P3Dv44_PBR** shows all properties supported by P3D v4.4 models that use PBR materials.
- **X-Plane** shows all properties supported by X-Plane models.
- **AeroFly** shows all properties supported by AeroFly FS2 models.

There are common actions that you want to perform on your materials, for example to add an emissive texture to your material. To make such tasks easier the material editor has templates. In the template dropdown list you can select the template you want to apply, see Figure 6.13 and with the **Apply** button you can then apply it to the selected material(s). The following templates are available:

- **Night Texture** will add an emissive texture to your material, using the diffuse texture name and a suffix. The default suffix is LM, but this can be changed by modifying the template in the material template editor.
- **Specular Texture** will add a specular texture to your material, using the diffuse texture name and a suffix. The default suffix is S, but this can be changed by modifying the template in the material template editor.
- **Bump Texture** will add a bump texture to your material, using the diffuse texture name and a suffix. The default suffix is bump, but this can be changed by modifying the template in the material template editor.
- **Detail Texture** will add a detail texture to your material, using the diffuse texture name and a suffix. The default suffix is detail, but this can be changed by modifying the template in the material template editor.
- **Fresnel Texture** will add a fresnel texture to your material, using the diffuse texture name and a suffix. The default suffix is fresnel, but this can be changed by modifying the template in the material template editor.
- **PBR Composite Texture** will add a metallic texture to your material, using the diffuse texture name and a suffix. The default suffix is C, but this can be changed by modifying the template in the material template editor.
- **Set Default Transparent** sets all material properties so that an alpha channel determines the level of transparency. This mimics the set default transparent button in the GMax/3DS Max material editor.
- **Set Default Opaque** sets all material properties so that a material is shown opaque. This mimics the set default opaque button in the GMax/3DS Max material editor.

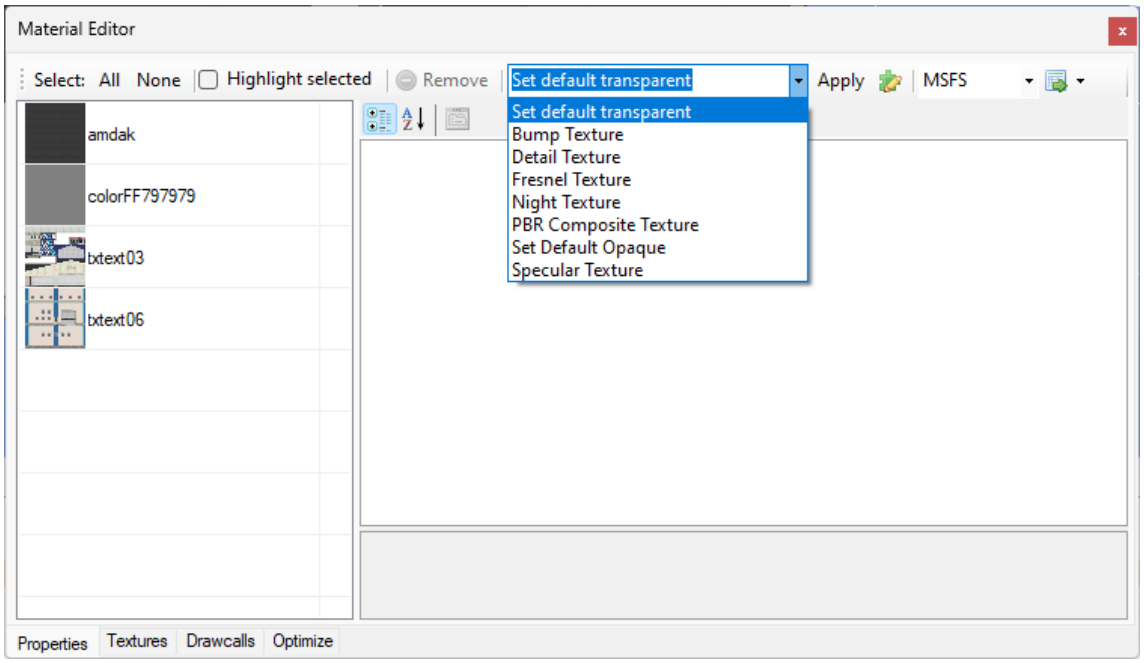


Figure 6.13: Material property templates

Besides these material templates that are available by default, you can also create your own templates. This is done in the material template editor, see Figure 6.14. This template editor has three areas you can work in:

- The panel on the left shows you all templates that exist. Using the **New** button you can add a new template, with the **Remove** button you can remove the selected template, and with the **Copy** button you can make a copy of an existing template. If you click on the small arrow next to the new button you will see a list of all materials in your object and you can make a new template that uses the values of the selected material.
- The panel in the middle shows the selected template. At the top you can edit the name of the template and below is shown a list of the attributes and values that are in the template. With the **Remove property** button you can remove a property from the template.
- The panel on the right shows all material properties that are available for use in the template. You can select a property here and also assign it the value that should be used in the template. Once you press the **Add property** button it is added to your template.

Once you have made your own template, it will be available in the dropdown list of templates and you can use it like any of the default templates are used.

For the textures you can use some special logic in your material template. Instead of typing the new texture name, you can also add some special keywords that MCX will replace dynamically. These keywords are:

- [**filename**] is replaced by the current filename (without extension) of the texture. So this can be used to modify the existing filename, e.g. adding a suffix or prefix to all textures.
- [**diffusetexture**] is replaced by the current diffuse texture filename (without extension) of the material. So if you would write `[diffusetexture]_bump.bmp` for your bump texture in the material template, the new bump texture value would be the diffuse texture with the suffix `_bump`.
- [**ext**] is replaced with the extension of the current or diffuse texture filename.

The default material templates for adding sub textures use this renaming logic, so you can have a lot at these files to see how it works. Be aware that the MaterialTemplateOverwriteTexture option influences if existing textures are overwritten by the template or not.

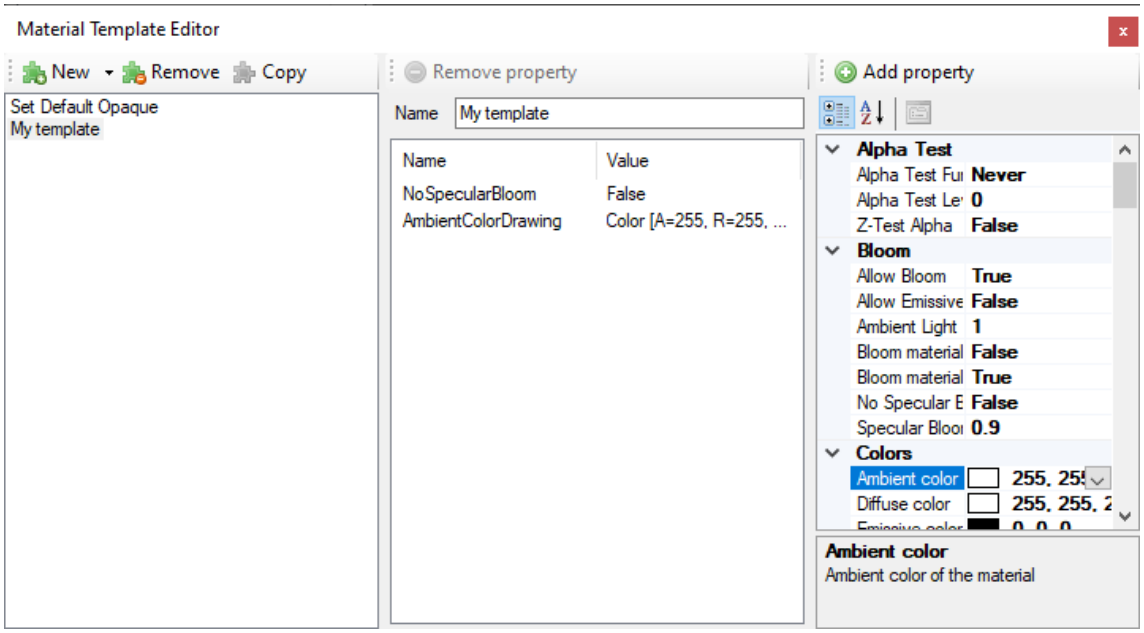


Figure 6.14: Material template editor

The button on the right of the material editor toolbar allows you to convert materials between different flight simulators, see Figure 6.15. This can be useful if you are converting models between different simulators. Table 6.1 shows the different conversions that are done.

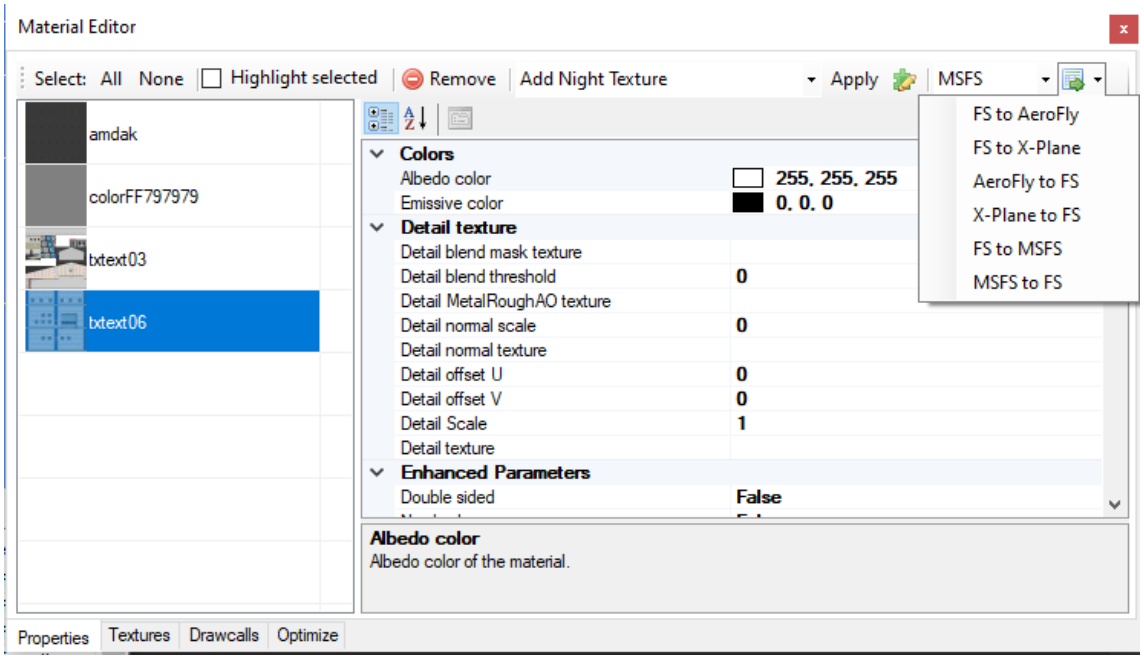


Figure 6.15: Convert materials

Conversion	FS to X-Plane	FS to AeroFly	X-Plane to FS	AeroFly to FS	FS to MSFS	MSFS to FS
Remove diffuse texture alpha when used for reflection	✓	✓			✓	
Convert bump map to standard format	✓	✓			✓	
Combine specular map into bump map	✓					
Remove specular texture	✓				✓	
Remove fresnel texture		✓			✓	
Remove detail texture		✓				
Remove environment texture		✓			✓	
Convert bump map to FS format			✓	✓		✓
Split specular map from bump map			✓			
Convert combined PBR texture to MSFS layout					✓	
Convert combined PBR texture to P3D layout						✓

Table 6.1: Material conversions

6.5.2 Textures tab

The textures tab, see Figure 6.16 allows you to perform various actions on the textures that are used on your object. The list contains all texture types that are used, so not only the diffuse texture, but also the other types like bump textures or emissive textures.

With the toolbar at the top you can perform the following actions on all textures that are used on the object:

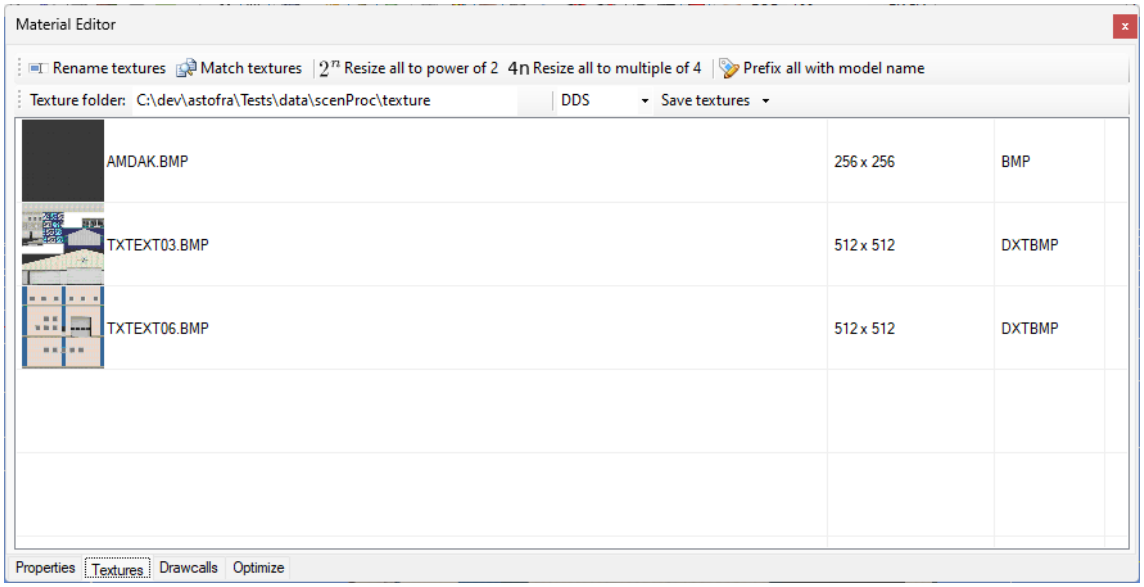


Figure 6.16: Material editor textures tab

- The **Rename textures** function allows you to do find and replace in all texture names and therefore gives you a quick way to rename multiple textures at once. When you click this button a new dialog opens where you can enter the text to find and the text you want to replace it with. Afterwards the replacement is done on all textures in the model.

  For example, let's say you model has textures named `texture1.dds`, `texture2.dds` and `texture3.dds`. If you would enter `texture` as find text and replace it with `my_airport_`, the textures would be named `my_airport_1.dds`, `my_airport_2.dds` and `my_airport_3.dds` afterwards.

- The **Match textures** function searches in the texture folder that is specified in the text box below the toolbar and when a texture is found that is the same as the one used on your model it will be replaced by the texture from the texture folder. This feature can be useful for example if you have converted your textures from JPG to DDS previously and you want to use the matching DDS textures on your model again. Matching the textures is done by checking if the pixels in the two texture images are similar enough to each other.

- The **Resize all to power of 2** function makes sure all your textures have a size that is a power of two. Flight Simulator only supports textures whose size is a power of two, so this function ensures that your textures meet this rule. Examples of texture sizes that are a power of two are: 256x256 pixels, 1024x1024 pixels or 512x1024 pixels. Models made with SketchUp often don't use textures that have this size. After you have resized all your textures to a power of two, you need to make sure that you save them to your texture folder again.

- The **Resize all to multiple of 4** function makes sure all your textures have a size that is a multiple of four. MSFS requires that textures have a size that is a multiple of four, so this function ensures that your textures meet this rule. After you have resized all your textures to a multiple of four, you need to make sure that you save them to your texture folder again.

- The **Prefix all with model name** function will add the name of your model as prefix to all textures. This can be useful if the texture names used by the modeling tool are not unique. For example SketchUp often uses textures that are named texture0.jpg, texture1.jpg, etc. This gives conflicting names if you have multiple models with such names in your scenery. Let's assume your model is called `house`, then this function will name your textures houses_texture0.jpg, house_texture1.jpg, etc. After adding the texture name prefix you need to make sure that you save the textures to the texture folder again.

- The **Save textures** function will save all textures to the texture folder specified in the text box, using the format that has been selected in the format dropdown list. See section 10.3 for more information about the supported texture formats. If you click on the small arrow next to the save button you get a menu where you can select settings used while saving the textures, see Figure 6.17. The possible settings are:
  - **Ensure size power of two** will make sure all textures have a size that is a power of two before saving them.
  - **Ensure size multiple of four** will make sure all textures have a size that is a multiple of four before saving them.
  - **Overwrite existing textures** allows textures that already exist in the texture folder to be overwritten.

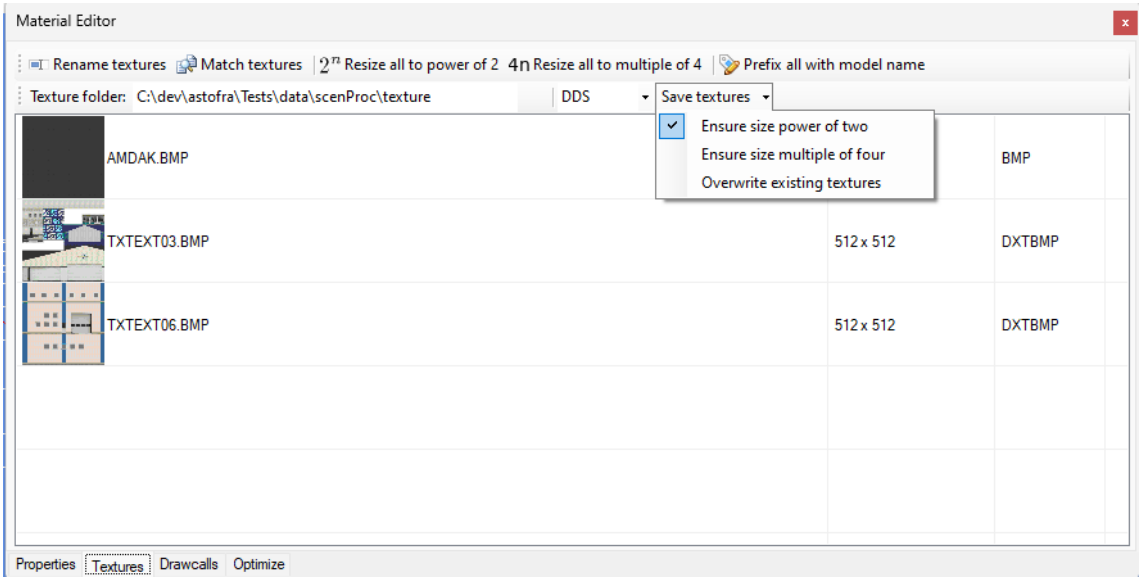


Figure 6.17: Texture save settings

Besides the actions described above that apply to all textures at once, there are also a number of actions you can perform on individual textures. Most of these are accessible from the context menu that is shown when you right click on the row in the list of textures, see Figure 6.18. The following actions can be performed:

- When you double click on a texture in the texture list, the texture is opened in the texture converter window, see Figure 8.3. For more details on the texture converter see section 8.3.
- When you click on the name of the texture in the list, you will be able to change the name by typing in a new name. You need to save your texture to the texture folder again to make sure the texture with the new name also exists.
- The **Resize power of two** context menu option allows you to resize the texture, the sub menu will show a number of power of two textures sizes that are near the current texture size.
- The **Resize multiple of four** context menu option allows you to resize the texture to the nearest multiple of four size.
- The **Format convert** context menu options allows you to convert this specific texture to another format, you can select the format to use in the sub menu. See section 10.3 for more information about the available formats.
- The **Flip vertical** context menu option flips the selected texture vertically. This is sometimes needed because not all modeling tools use the same origin for DDS textures.

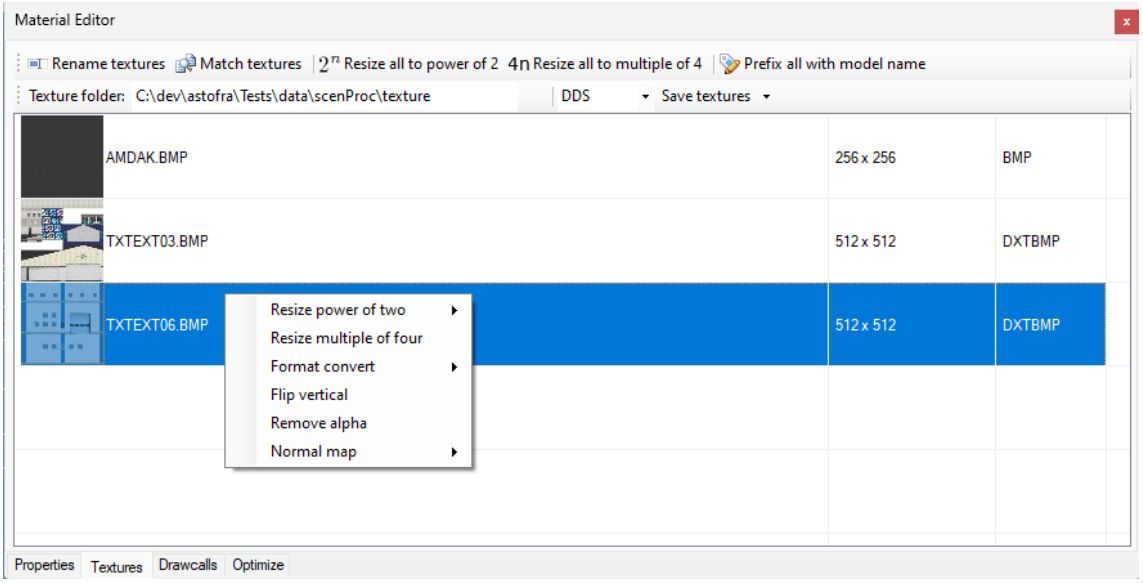


Figure 6.18: Texture context menu

- The **Remove alpha** context menu option removes the alpha channel from the selected texture.
- The **Normal map** context menu option allows you to convert a texture between the normal map structure that is used by tools like Photoshop and the normal map structure has used by Flight Simulator. The difference between those two representations are that some channels are shifted. You can convert a normal map in both directions using the sub menu options. This menu also gives you the option to make the blue channel of the normal map white, which can be useful to prevent incorrect display of the normal map in certain tools.

6.5.3 Drawcalls tab

The drawcalls tab, see Figure 6.19 gives you the drawcall minimizer functionality. See section 13.5 for more background information on drawcalls. What the drawcall minimizer does is reduce the number of drawcalls in your object and thereby improves the performance.

The following settings for the drawcall minimizer are available:

- The **maximum texture size** of the new texture that is generated. ModelConverterX will use the smallest texture sheet that can contain all your textures, but this setting defines the maximum size that can never be exceeded.
- With the **Skip texture with alpha channel** checkbox you can determine if textures that have an alpha channel should be excluded from the combined texture sheet or not.
- With the **Skip specific textures** checkbox you can determine if specific textures should be excluded from the combined texture sheet or not. If you click the **Select textures to skip** button, you will get a dialog to select which textures should be excluded. See Figure 6.20. The label will show the number of textures you have selected to be excluded.
- With the **Include repeating textures** checkbox you can determine if textures that are tiled on your object should be included in the new texture sheet as well. If enabled the drawcall minimizer will repeat the texture as many times as needed in the new texture sheet.
- With the **Material color to vertex colors** checkbox you can determine how material colors are processed. When this box is checked the material color is copied to the vertex colors, if it is not checked the material color is turned into a texture. The option to use vertex color is only supported by MSFS at the moment.

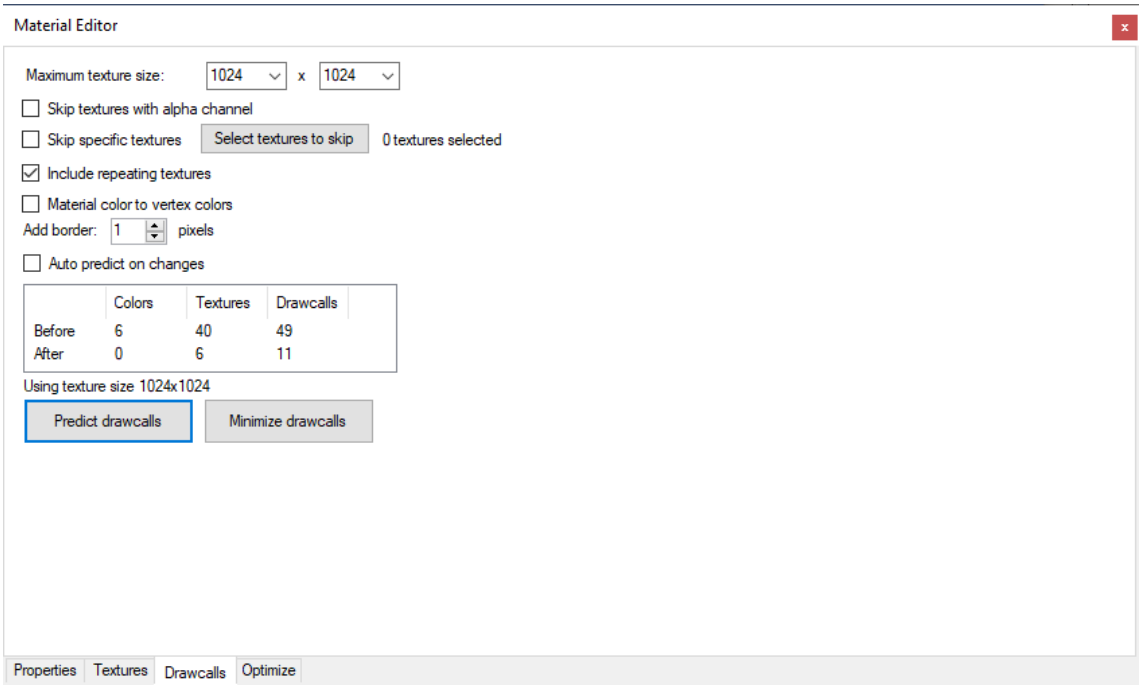


Figure 6.19: Material editor drawcalls tab

- With the **Add border** checkbox you can determine if the drawcall minimizer should add a border around the texture before adding it to the new texture sheet. Sometimes adding a border can result in a better visual quality, but given that it alters the size of the textures it might also make it harder to fit them nicely on a new texture sheet. You can select the size of the border that is added as well.

When you have the **Auto predict on changes** checkbox selected, the number of drawcalls will be automatically predicted once you change a setting. But for complex objects it is better to keep this unchecked and manually start the prediction once you are happy with all objects, as the prediction can take some time.

With the **Predict drawcalls** button you can start a calculation of how much reduction the current settings will give you. The results will be shown in the box in the middle once the calculation is done. When you press the **Minimize drawcalls** button the reduction of drawcalls is actually applied to your object. Don't forget to save your textures after minimizing the drawcalls, the drawcall minimizer only updates the textures in memory, but does not save them to disk. You can do this from the textures tab.

And now that I have explained how you can use the drawcall minimizer, let's also give some background information about what this feature actually does. The first step of the drawcall minimizer is to replace all colours in your object with a small texture of the same color. Afterwards the drawcall minimizer will try to combine all the different textures used in the object on a new texture sheet. This is like trying to make a collage of your photos on a page, where the drawcall minimizer tries to organize them as tightly packed as possible to use as few texture sheets as possible. Figure 6.21 shows an example of a combined texture sheet made by the drawcall minimizer.

6.5.4 Optimize tab

The optimize tab, see Figure 6.22 shows you a list of materials that are very similar to each other, but have a few attributes that differ. Sometimes these differences are not intentional and by making sure all attributes are the same you can reduce the amount of drawcalls (see section 13.5 for more information on drawcalls).

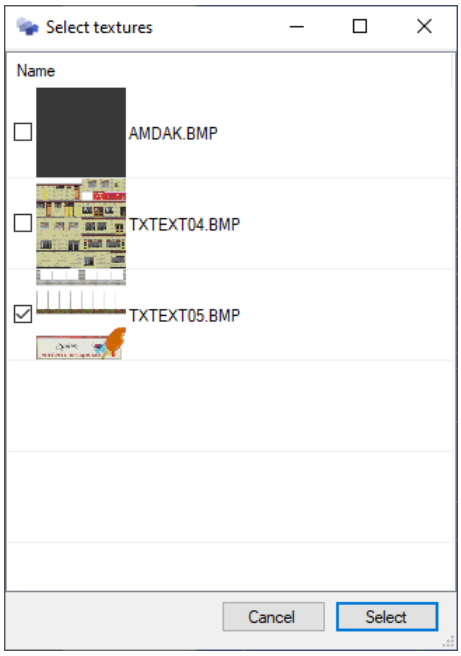


Figure 6.20: Dialog to select the textures to skip



Figure 6.21: Example texture sheet of drawcall minimizer

If you right click on a material attribute in the optimize tab you get a context menu where you can select if the left or the right value should be used in both materials, see Figure 6.23.

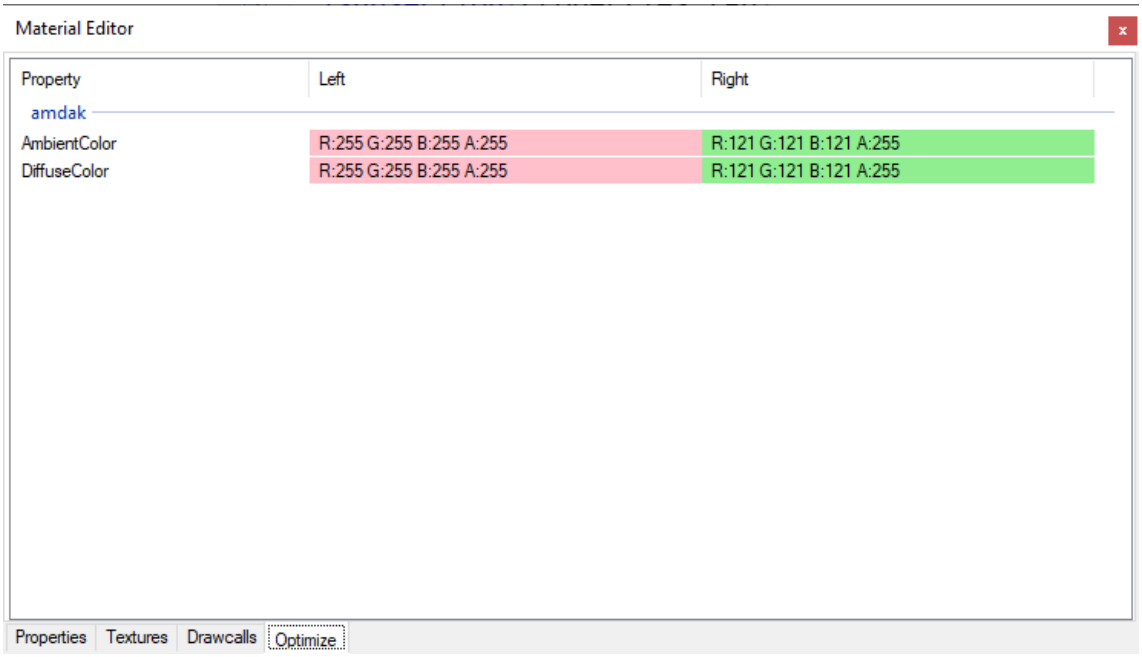


Figure 6.22: Material editor optimize tab

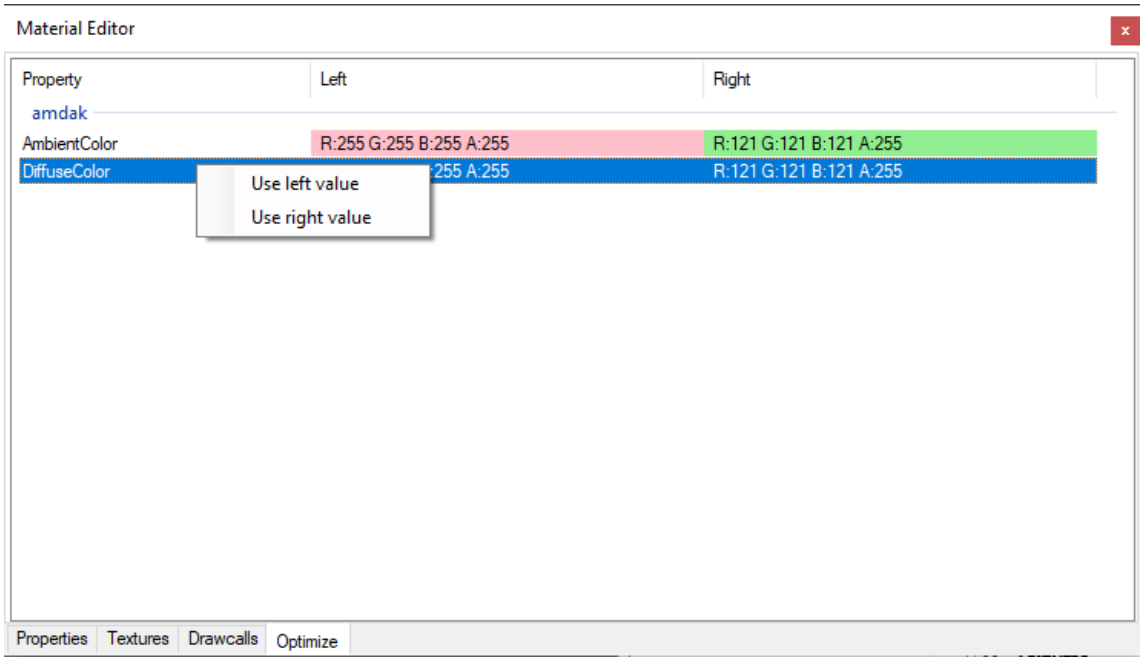


Figure 6.23: Material editor optimize context menu

6.6 Attached object editor

The Attached object editor lists all attachpoints of the active representation of your object, see Figure 6.24. You can view or edit them using this editor. The following attachpoint types are supported:

- Effects
- Lights. These can be either point lights or spot lights. On export to FS2004, FSX or Prepar3D ModelConverterX will make effect files for these automatically. When exporting to MSFS they become light sources.
- Library objects
- Platforms
- Empty attachpoints

On the left side of the window there is a list of the attachpoints in your object. If you click on an attachpoint, the properties will be shown in the right side of the window.

Using the **Add** button you can add a new attachpoint; clicking this button will bring up a menu where you can select the type of attachpoint you want to add. Using the **Duplicate** button you can also make a clone of an existing attachpoint, which is handy if you want to add another attachpoint that is very similar to an existing one. With the **Remove** button you can remove the currently selected attachpoint.

When editing an effect attachpoint, clicking the **...** button for the effect parameters, will bring up the effect parameter editor, see Figure 6.25. In this editor you can select the parameter that you want to use and also the value or value range that it should have. Using the editor can be more efficient than typing all parameters by hand.

With the **Preset DAY/NIGHT** and **Preset DUSK/DAWN** buttons you can quickly add the effect parameters that are needed for showing an object only during dusk, dawn and night. Due to an issue in Flight Simulator, for an effect to appear in all four time periods you need to use two attachpoints with the parameters as set by these buttions.

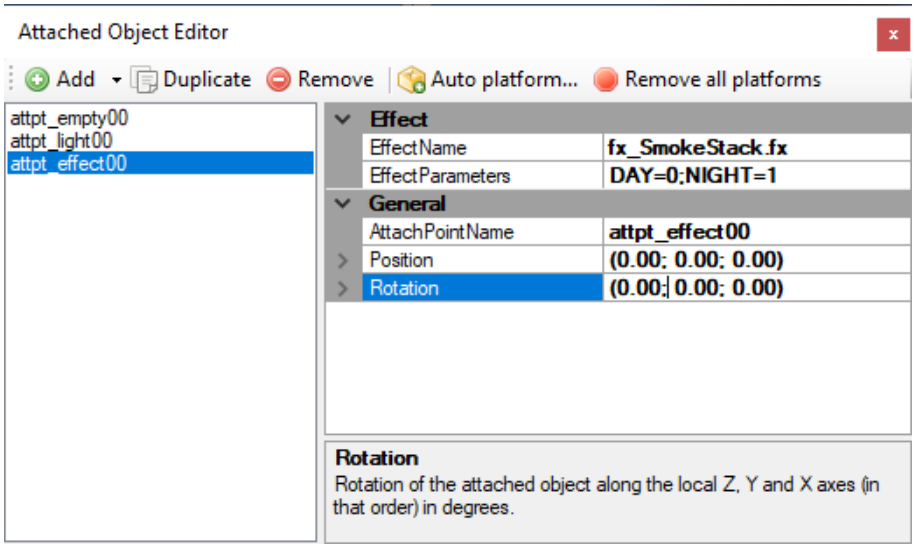


Figure 6.24: Attached object editor

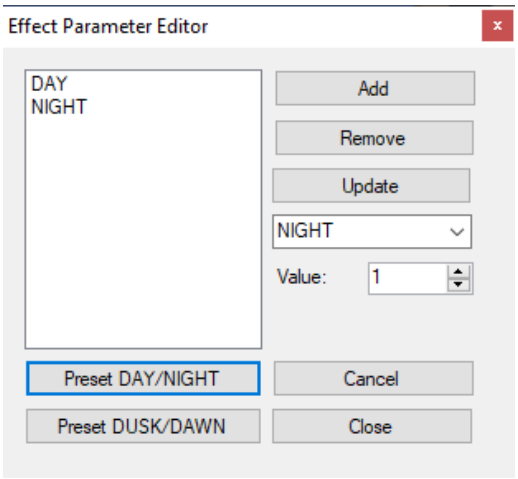


Figure 6.25: Effect parameter editor

When editing platform attachpoints you have two options on how to specify them. You can add a platform attachedobject via the Add menu and then specify the position (altitude) and length and width of the platform in the properties. When making a simple rectangular platform, MCX will create the platform object for you when you export your model.

When attaching effects to an attachpoint the Z axis (blue line) is in general the up direction of the effect. E.g. smoke will appear in that direction. Some effect work along a different axis, for example the landing light effect shines along the -Y axis (inverse of green line) and also spot lights shine along that axis.

When you want to add a platform that should follow the shape of your object there is another way to add them however, using the **Auto platform** function. Pressing this button on the toolbar brings up the Auto platform window, see Figure 6.26. In this editor you can select how much the normal of the triangle should be pointing upwards[1] and optionally you can also specify that only triangles with a specific texture should be considered. ModelConverterX will then make a platform of all triangles that meet these filter conditions. So if for example you select your roof texture and an upwards value of 0.7 you can quickly make a platform of your roof.

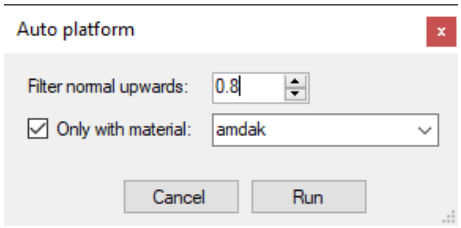


Figure 6.26: Auto platform window

Since platforms for more complexly shaped objects can result in many platform attachpoints, the **Remove all platforms** button in the toolbar of the attached object editor allows you to remove all existing platforms at once.

6.7 Animation editor

The animation editor displays information about the animations in the active representation of your object and allows you to edit these animations, see Figure 6.27. The editor lists all animations in the object at the top and has a number of buttons to edit and control the animations at the bottom. When an animation is shown with a pink background color, it means this animation was not found in the modeldef.xml file. Such animations will not be exported correctly to Flight Simulator and you should fix them before exporting your object.

With the checkboxes in the list of animations you can select which animations are currently selected. Using the **Select all** button you can select all animations. Using the **Select none** button you can unselect all animations. Using the **Select invert** button you can invert the current selection. Using the **Select name containing** button you can select all animations that contain the text value entered in the textbox to the right of the button. This way you can for example select all gear animations quickly.

Only the selected animations will be played in the object preview window. Using the **Play** button you can start and pauze the animations. Next to this button the current animation frame is displayed in the textbox. If you specify a value in the min and max textboxes, the animation will only play between these selected frames. By default the max value is the length of the animation and the min value is zero.

[1]The value you enter here is the dot product of the triangle normal and the normal pointing upwards. A vlaue of 1.0 means the normal points upwards, while a value of 0.0 is a normal that points horizontal, and a value of -1.0 is a normal that points downwards. So a value of 1.0 selects only triangles that point exactly upwards. If you want to select the triangles of a gabled roof you would have to use a lower value, for example around 0.7. If you select a value that is too low, ModelConverterX might also select your walls as platform.

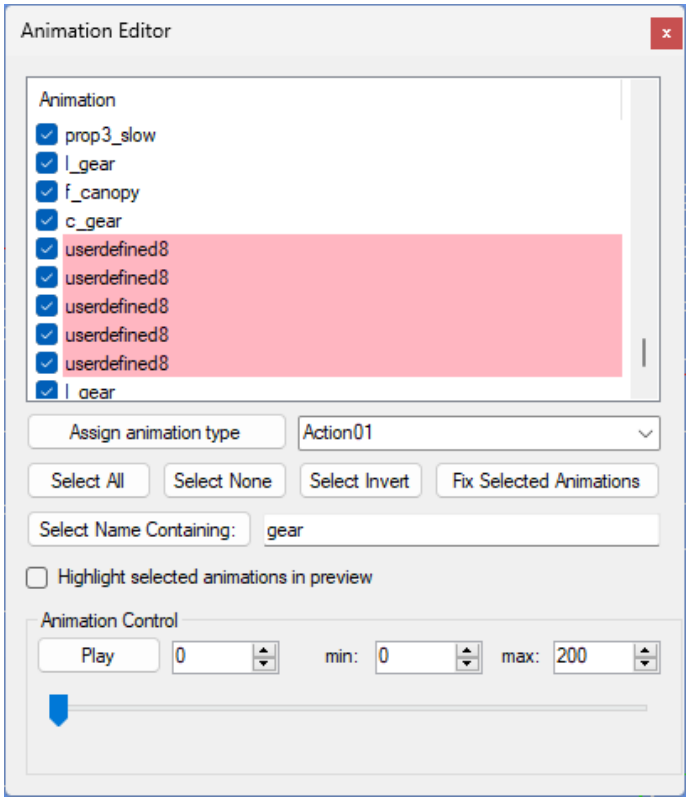


Figure 6.27: Animation editor

Using the **Assign animation type** button you can change the animation type of the selected animations. The drop down box next to the button allows you to select which animation type should be used. The available types are read from the modeldef.xml file specified in Options.

Sometimes you want to remove an animation from your model and replace it with the static geometry at a specific frame of the animation. For example you need the landing gear of an aircraft to be always extended when you want to use the model as a static aircraft. With the **Fix selected animations** button you can do this. Make sure you select the animations you want to remove, move the animation slider to the desired value, and then press this button. The animation will then be removed and the static geometry will display at the selected animation frame.

When the **Highlight selected animation in preview** checkbox is selected the parts that have the animations that you have active in the animation editor will be highlighted in the preview of the object. This can make it easier to spot where a certain animation is used.

6.8 ModelDef.xml editor

With the ModelDef.xml editor you can see the model definitions that are used by your object, see Figure 6.28. This editor can be used to inspect the definitions of a specific object and to make modifications to them.

The toolbar offers you the following options:

- A textbox to enter text to search for in the modeldef.xml file.
- **Find previous** button to jump to the previous occurrence of the search text.
- **Find next** button to jump to the next occurrence of the search text.
- **Only show used** checkbox to determine if all entries from the modeldef.xml file are shown or only those used by the object.

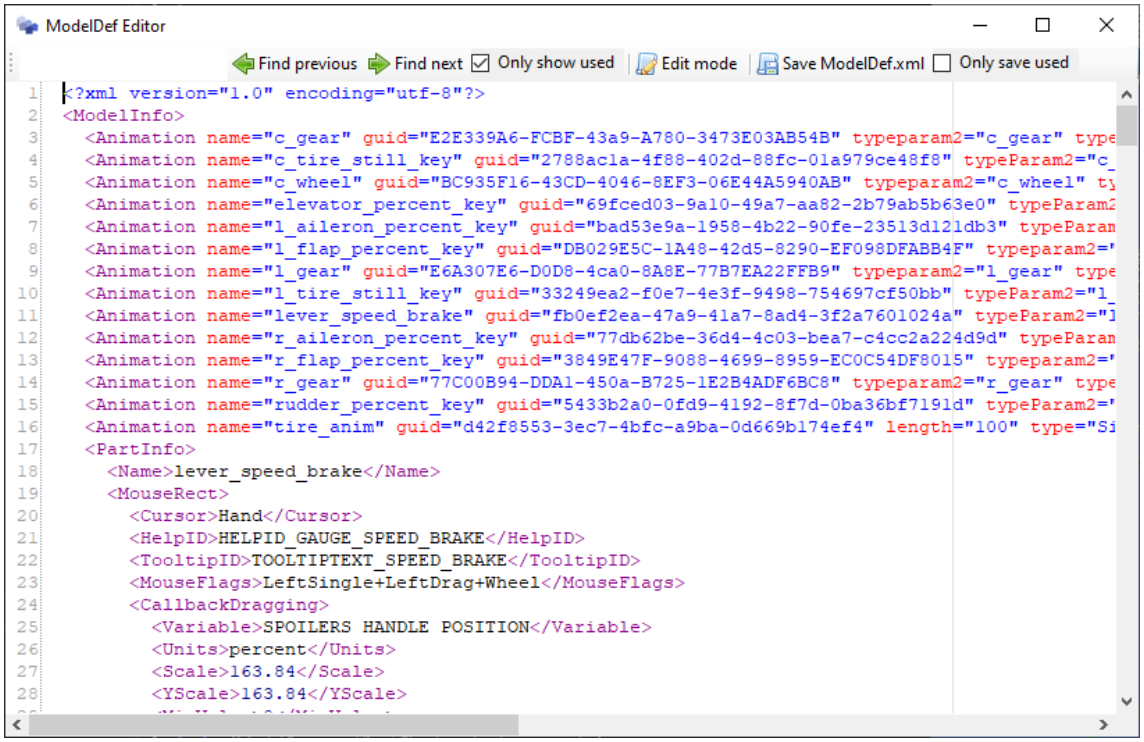


Figure 6.28: ModelDef.xml editor

- **Edit mode** button to toggle edit mode. When in edit mode you can modify the definitions in the XML file. Be aware that when you use the edit more while you have the Only show used option enabled that all definitions that are not used will be lost when you exit edit more, so sometimes it is better to edit while all definitions are shown.
- **Save ModelDef.xml** button to save the modeldef.xml file as shown in the editor to disk. When the **Only save used** checkbox is enabled only the definitions used in the objects are saved to file.

6.9 Aircraft.cfg editor

The aircraft.cfg editor, see Figure 6.29, is a text editor used to modify the aircraft.cfg file of aircraft models. The advantage over using a normal text editor is that the aircraft.cfg editor is linked to the preview in ModelConverterX so it can highlight the CFG points you are working on, and any changes you make to their positions are shown interactively in the preview. To be able to see the CFG points you need to make sure their display is enabled in the preview.

After you have made changes to the aircraft.cfg file you can use the **Save** button to save these changes to disk. With the **Undo** and **Redo** buttons you can undo and redo changes that you have made to the file.

Using the search textbox and the **Find previous** and **Find next** buttons you can search for specific text in the aircraft.cfg file. This way you can quickly locate the section you are looking for.

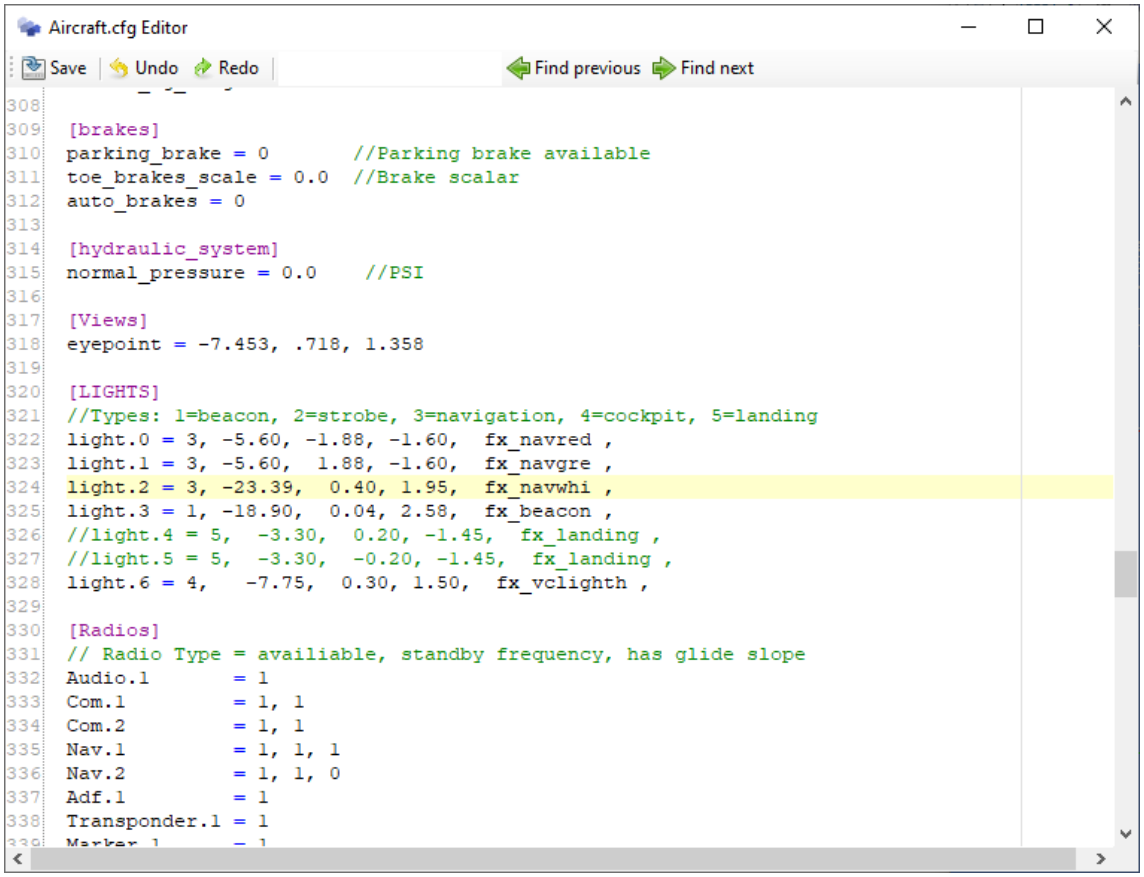


Figure 6.29: Aircraft.cfg editor

6.10 Earth curve correction editor

MDL objects in Flight Simulator use flat earth coordinates. However they are placed in an earth that is curved. This means that for big objects you can see that they start to float the further you go from the reference point. Besides that there is an issue in Flight Simulator that causes attached objects that are further away from the reference point to appear at the wrong position. The earth curve editor, see Figure 6.30 can be used to correct both of these effects.

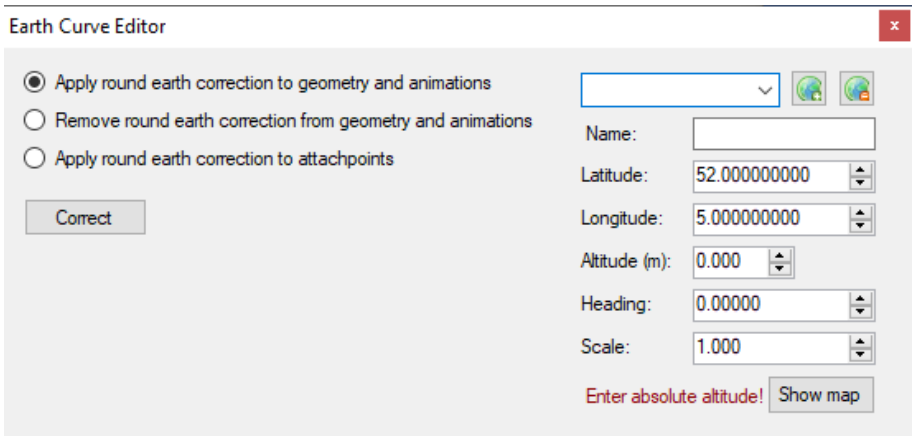


Figure 6.30: Earth curve editor

On the left side of the editor you can select one of the following options:

- **Apply earth curve correction to geometry and animations** is used if you have have an object that should follow the curve of the earth. For example a big building or a scene with multiple buildings in it. Based on the reference point you specify the editor will calculate the displacement that is needed to have all geometry and animations follow the curve of the earth exactly. The editor assume that your input model uses the FS2004 flat earth coordinates.
- **Remove earth curve correction from geometry and animations** is the reverse of the option described above and can be used to undo an earth curve correction that was applied to a model.
- **Apply earth curve correction to attachpoints** can be used if you don't want to correct your geometry, but still want to make sure that attachpoints line up with your geometry.

On the right side of the editor you can use the position control to specify the reference point that should be used for the earth curve correction. See section 13.1 for more information about using the position control. Once you have applied the earth curve correction, using the **Correct** button, you need to make sure that the object is placed at this reference coordinate in your scenery as well.

6.11 Season editor

When exporting FS2002 style ground polygons or Prepar3D v4 (and above) MDL files Model-ConverterX can support seasonal texture changes. This is controlled by the HasSeasonalTextures attribute in the material. The season editor, see Figure 6.31 can be used to define which seasons should be used in that case.

Using the checkboxes you can specify which season is needed for the area you are making your scenery for and for each active season you need to specify the start and end day (these are counted from the start of the year, so day 1 is January 1st. The editor will automatically ensure that the seasons you have activated connect with each other.

The winter2 season is to define an additional winter season. In some areas of the world you need

to define a period of winter without snow (winter), followed by a winter with snow (heavy winter), followed by a winter without snow again (winter2).

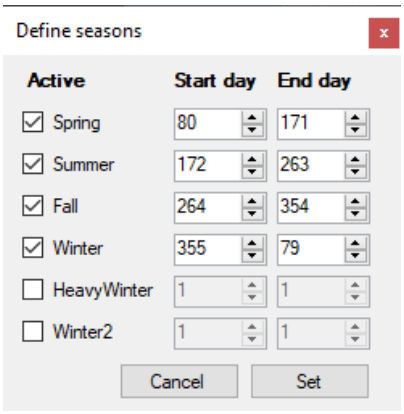


Figure 6.31: Season editor

6.12 Transform object editor

ModelConverterX offers three different editors to transform the entire object. With these editors you can scale, move and rotate the object. Each of them is discussed in the following sections.

6.12.1 Scale object

With the Scale object function, see Figure 6.32, you can make the object larger or smaller.

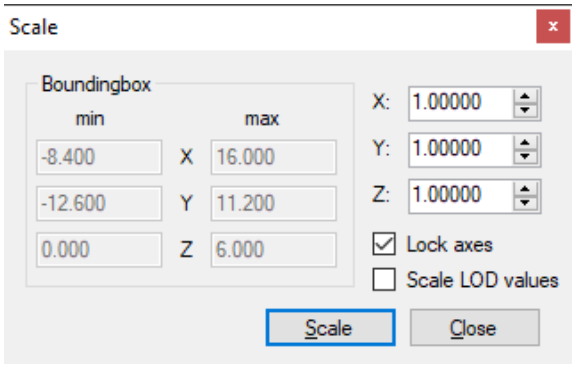


Figure 6.32: Scale object

On the right side of the window you can enter the scale value for the X, Y and Z axis. If the **Lock axes** checkbox is selected the same scaling is applied to all three axes automatically. If the **Scale LOD values** checkbox is selected MCX will not only scale the object itself, but also adjust the level of detail (LOD) values so that the LOD switching still happens at the same distance.

On the left side you see the minimum and maximum values of the object bounding box. While you change the scale values, these bounding box values are updated so that you can see what the size of the object will be after scaling.

Only when you press the **Scale** button the object is actually scaled with the entered scale values. When you press the **Close** button the scale object window is closed without applying a scale.

6.12.2 Move object

With the Move object function, see Figure 6.33, you can move the object.

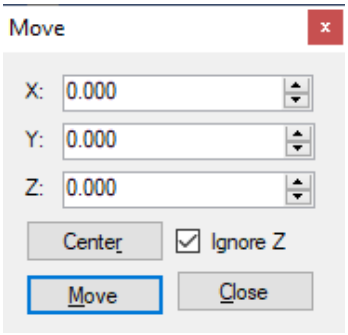


Figure 6.33: Move object

At the top of the window you can enter the offset along the X, Y and Z axes that should be applied to the object.

If you click the **Center** button ModelConverterX will calculate the X, Y and Z values that are needed to center the object. When the **Ignore Z** checkbox is selected the Z axis is ignored when calculating the offset needed to center the object, so if you want to keep your building on ground level it's better to leave this option checked.

Only when you press the **Move** button the object is actually moved with the entered offset values. When you press the **Close** button the move object window is closed without applying a movement.

6.12.3 Rotate object

With the Rotate object function, see Figure 6.34, you can rotate the object.

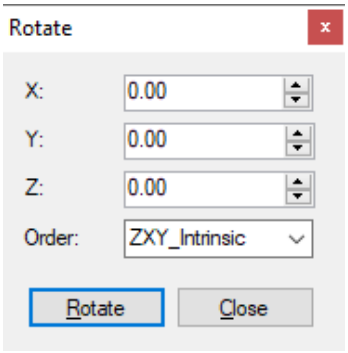


Figure 6.34: Rotate object

At the top of the window you can enter the rotation along the X, Y and Z axes that should be applied to the object. With the dropdown list below you can select the order in which the rotation is applied. See section 13.2 for more details about the rotation order.

Only when you press the **Rotate** button the object is actually rotated with the entered rotation values. When you press the **Close** button the rotate object window is closed without applying a rotation.

6.13 Level of detail creator

With the level of detail creator, see Figure 6.35 you can automatically generate a simplified version of your object that can be used as a lower level of detail version. See section 13.7 for more background information about levels of detail.

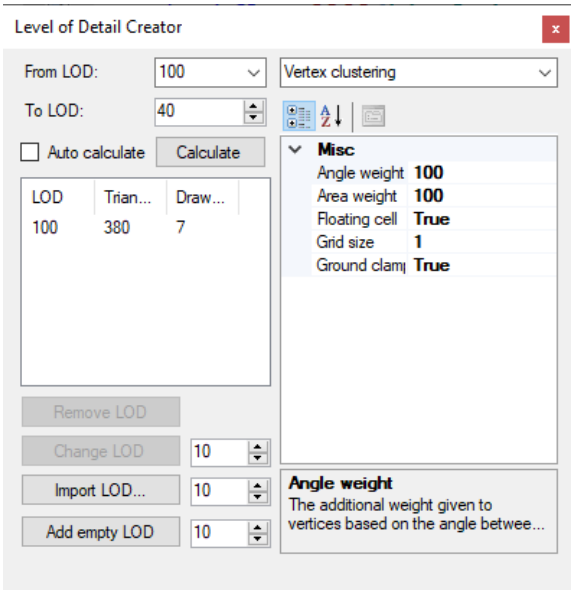


Figure 6.35: Level of detail creator

On the left side of the window you configure which level of detail you want to generate. **From LOD** specifies which level of detail should be used as input for the calcuation, while **To LOD** specifies which for which level of detail the generated object should be used.

On the right side of the window you can select the algorithm that should be used and below that the parameters for that algorithm are shown. In section 6.13.1 and section 6.13.2 the details of the supported algorithms are discussed.

Once you have configured all parameters correctly, you can press the **Calculate** button to start the generation. If you select the **Auto calculate** checkbox the level of detail is automatically generated once you change one of the parameters. For complex models it is better to leave this unchecked as the calculation might take quite a long time.

In the box below the calculate button you can see all levels of detail that the object contains and the number of triangles and drawcalls for each of them is shown. If you click on a level of detail in this list, that level of detail will be selected for display in the preview. With the **Remove LOD** button you can remove the currently selected level of detail from the object. With the **Change LOD** button you can change the level of detail number; the textbox next to the button specifies the new level of detail value to use. This allows you to change the level of detail from 20 to 30 for example. With the **Import LOD** button you can select another file that will be loaded and imported as the level of detail value specified in the textbox next to the button. Thus this can be used to combine different models that each represent a different level of detail version of the object. With the **Add empty LOD** button you can add an empty level of detail with the specified value to the object

6.13.1 Vertex clustering

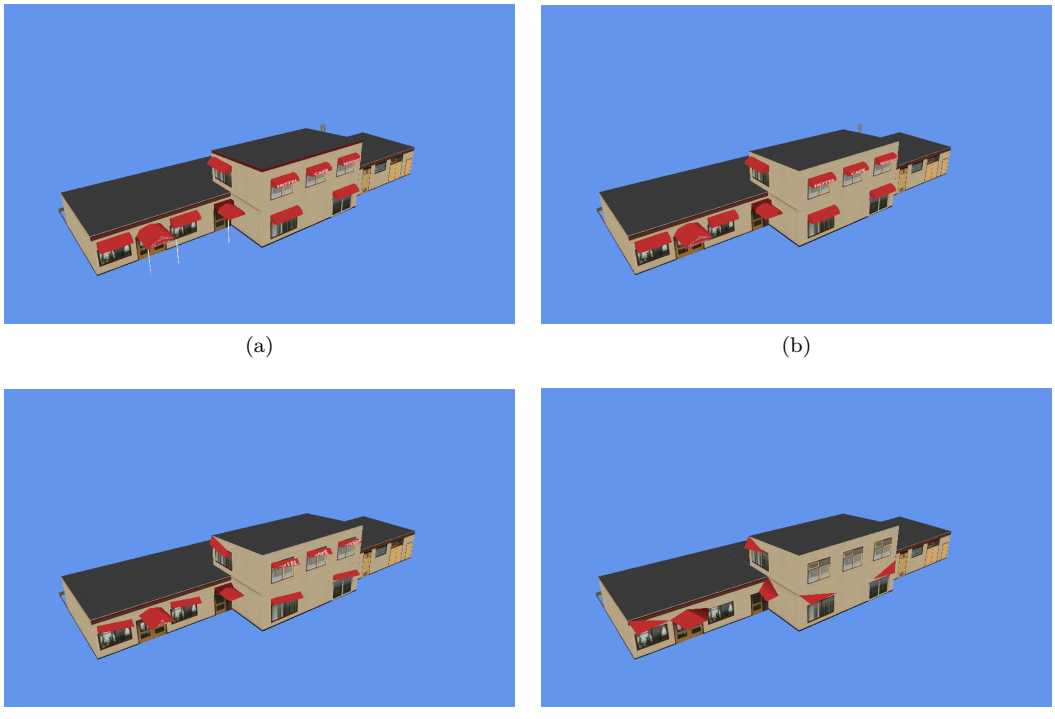
This section discusses the details of the vertex clustering algorithm. The basic principle of this algorithm is that a 3D grid with a specific grid size is calculated and all vertices that fall within the same cell of the grid are combined to one vertex. So if your object has many vertices close together they will be combined into one and this results in a reduction of the complexity of the model.

An additional weight can be assigned to the vertices. Vertices with a higher weight have more influence on the combined vertex when different vertices in one grid cell are combined.

The following parameters can be set for the algorithm:

- **Angle weight** specifies the additional weight given to vertices based on the angle between the triangles they are part of. This gives vertices that are part of triangles with different normals more weight. This means that hard edges of the object have more impact on the clustered result.
- **Area weight** specifies the additional weight that should be given to the vertices based on the area of the triangle that the vertex is part of. This means that vertices that are part of big triangles, which contribute more to the visual appearance, get more influence on the clustered result.
- **Floating cell** determines if the vertex clustering algorithm uses floating cells or not. Floating cells are placed more optimally and therefore give better results in general.
- **Grid size** specifies the size in meters of the grid that is used for clustering the vertices.
- **Ground clamp** determines if vertices that are at or below ground level should get a big additional weight. This prevent them from floating after the vertex clustering.

Figure 6.36 shows an object and three different levels of details generated by varying the grid size of the vertex clustering algorithm.



(a) (b) (c) (d)

Figure 6.36: Example of vertex clustering object simplification. (a) original model with 380 triangles. (b) clustering with 0.5 meter grid resulting in 315 triangles. (c) clustering with 1 meter grid resulting in 143 triangles. (d) clustering with 2 meter grid resulting in 72 triangles.

6.13.2 Quadratic based error

This section discusses the details of the quadratic based error algortihm. The basic idea of this algorithm is that each edge (line connecting two vertices) of the object is given a score of how much it contributes to the appearance of the object. In the parameters, as discussed below, you will see that different aspects can contribute to this score.

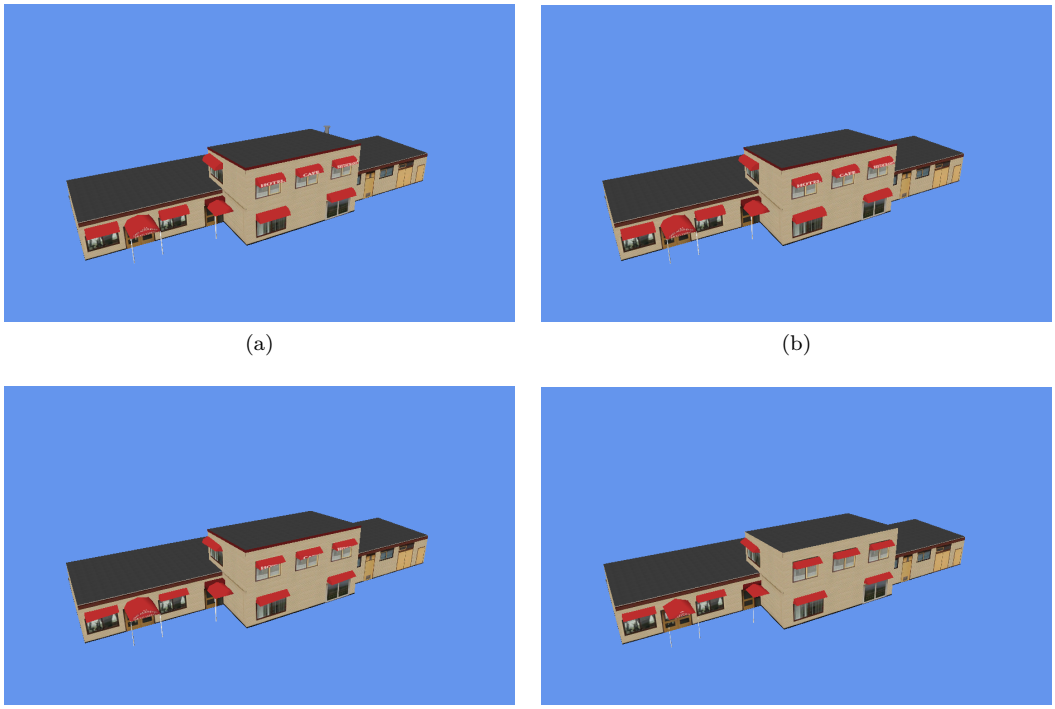
To simplify the object edges are being collapsed, starting with the edge that has the lowest score. This is continued until the target for the amount of triangles is reached or until the error in the

appearance of the object is bigger than a specified threshold.

The following parameters can be set for the algorithm:

- **Area penalty** specifies an additional penalty based on the triangle area when determining the edge score. This results in edge of big triangles being less likely to be affected by the simplification algorithm.
- **Ground clamp** determines if vertices that are at or below ground level should stick to the ground or not. This is to prevent buildings from floating in the air.
- When **Max error** is set to true, the optimization will stop once the specified max error value has been reached.
- **Max error value** specifies the maximum error value when the optimization should stop. This value is only used when Max error is set to true.
- **Preserve material** specifies that an extra penalty should be given to edges where two different materials meet. This results in these edges being less likely to be affected by the simplification algorithm.
- **Triangle target** specifies the target of the amount of triangles that the simplified model should have. It is given as a percentage of the amount of triangles in the object you start from.

Figure 6.37 shows an object and three different levels of details generated by varying the triangle target in the quadratic based error algorithm.



(a) (b) (c) (d)

Figure 6.37: Example of quadratic based error object simplification. (a) original model with 380 triangles. (b) simplifiied model with 80% triangle target resulting in 304 triangles. (c) simplifiied model with 60% triangle target resulting in 226 triangles. (d) simplifiied model with 40% triangle target resulting in 152 triangles.

6.14 Merge object editor

Using the merge object editor, see Figure 6.38 you can combine multiple objects into one object. You can use this to combine multiple buildings into one object for example.

At the top of the editor you can specify which level of detail (LOD) should be used for the object that is inserted and which offset from the reference point of the currently selected object should be used. By specifying different levels of detail you can also use this editor to add simpler versions of your model as a LOD.

At the bottom of the window you can select which object should be merged; there are two options:

1. Using the **Load object** window you can load the object that should be merged from file.
2. If you have a scenery with multiple objects loaded, the list below the load object button will show the other objects in your scenery. You can select an object to be merged from this list as well.

Once you press the **Merge** button the selected objects will be merged into the currently active object with the offset you have specified.

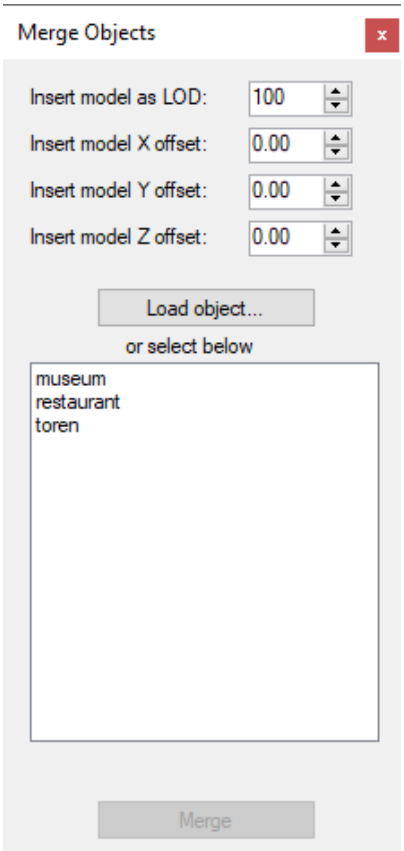


Figure 6.38: Merge object editor

6.15 Generate object image

Using the generate object image window you can create preview images of your object(s), see Figure 6.39.

To generate preview images you need to:

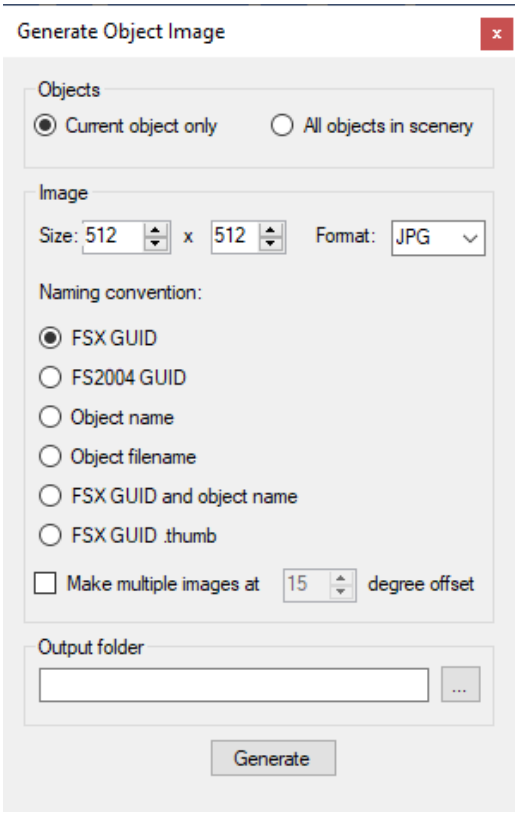


Figure 6.39: Save preview image

1. Select if you want to generate an image of the current object or all loaded objects.
2. Specify the size of the preview image and the format it should be saved in.
3. Select the naming convention of the generated file(s), the name can be based on the GUID, name or filename.
4. Specify if you want to make different images where the object is rotated by a specified angle between each image, this can be useful if you want preview images from all sides of your object.
5. Select the folder where the images should be stored.
6. Press the **Generate** button to make the images.

The images are captures of the object preview window, so any options like the grid or attachpoints that you have visible there will be part of your preview image as well.

6.16 Generate object report

Using the generate object report window you can create HTML reports of your objects, see Figure 6.40. You can configure which information is included for the objects.

To generate an object report you would take the following steps:

1. Select if you want a report for the current object or all loaded objects.
2. Specify which information should be included in the report, you can choose from:
   - Preview image of the object
   - Information about the level of details and drawcalls

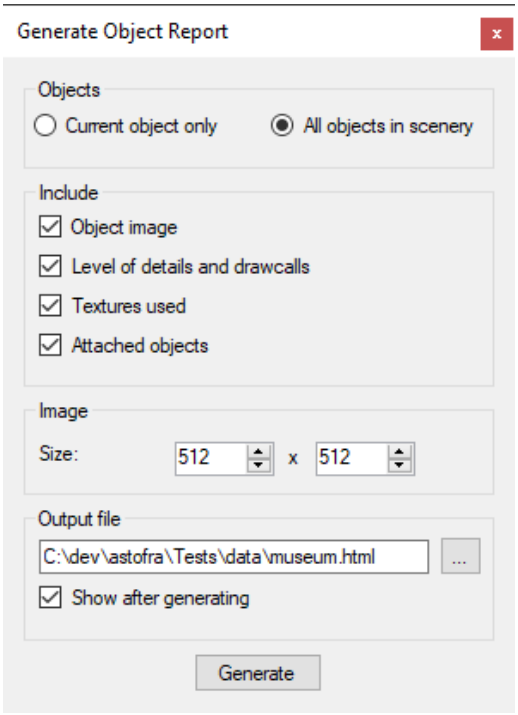


Figure 6.40: Generate object report

- Information about textures used
- Information about attachpoints

3. Select the size of the preview images.
4. Select the folder where the object report should be saved.
5. Press **Generate** to start generating the report.

Figure 6.41 shows an example of the object report that is generated.

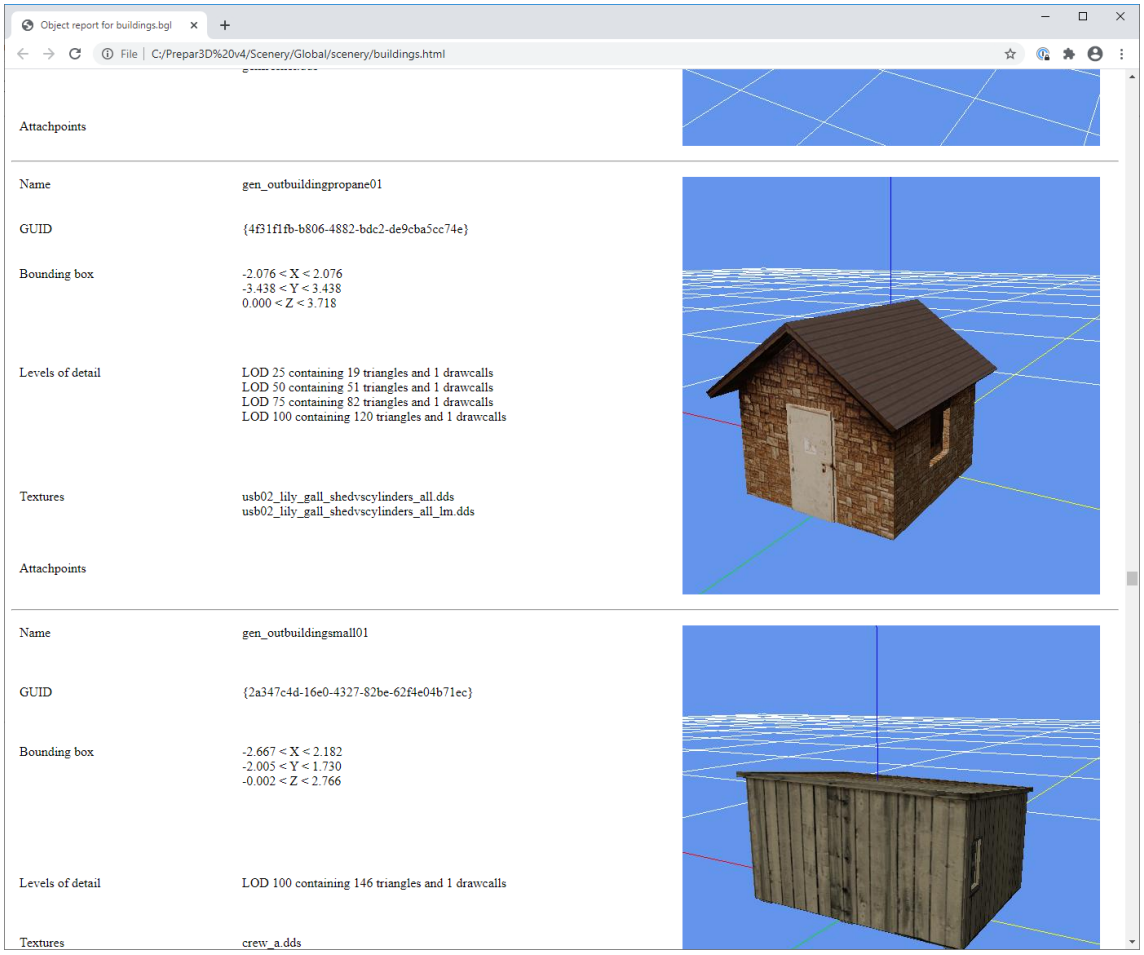


Figure 6.41: Example object report

6.17 Change history

In the change history window you can see an overview of the changes and edits that have been made to the object, see Figure 6.42. The undo button will undo the last change shown in this list from your object.

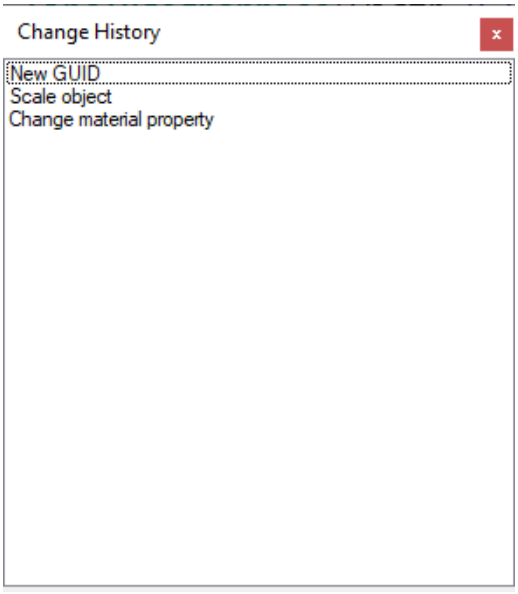


Figure 6.42: Change history

Chapter 7

Wizards

7.1 Convert and place object wizard

The convert and place object wizard, see Figure 7.1, is an easy way to convert an object to the format required by Flight Simulator and position it at a specific location in one process. This will also convert the textures of the object to the right formats. This can for example be used to quickly create a BGL file from a SketchUp object and have all textures converted on the go for you. It is primarily meant for users that quickly want to convert a single object. If you have many objects to place, it is often more efficient to make a library BGL of these objects.

When using the wizard you will fill in all details from top to bottom, the steps are:

1. Select the object that should be converted using the **...** button.
2. Select the FS version that you want to convert the object to using the radio buttons.
3. Select the output scenery folder where the BGL file should be placed using the **...** button. It will default to the Addon scenery folder of your selected FS version. Be aware that you need to select the base folder of your scenery, not the scenery subfolder that contains the BGL files.
4. Specify the output name, this will become the name of the BGL file. The default value is the name of the object that you loaded.
5. Specify the position where the object should be placed. See section 13.1 for more information about how to specify the position.
6. If you want existing textures to be overwritten select the **Overwrite existing textures** checkbox.
7. If you want all texture names to start with the output name, select the **Prefix texture names** checkbox.
8. If you want the number of textures to be minimized, select the **Minize number of textures** checkbox. This will run the drawcall minimizer, as described in section 6.5.3, on the object. The dropdown list next to the checkbox is used to specify the maximum texture size that the drawcall minimizer will use.
9. If you don't want the object to supress autogen objects, check the **No autogen suppression** checkbox.
10. If you want to disable the crashbox of the object, so that your aircraft can't crash into the object, check the **No crash** checkbox.
11. Press the **Convert** button once all settings are correct. This will create the BGL file and textures.

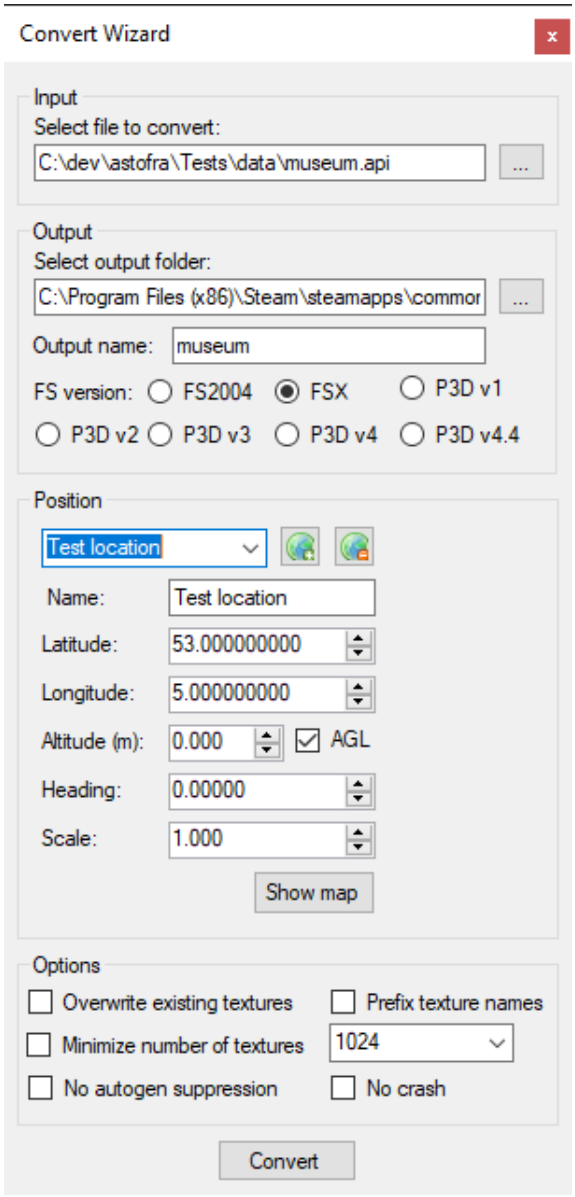


Figure 7.1: Convert and place object wizard

7.2 Ground polygon wizard

If you use normal scenery objects for your custom ground polygons they will typically flicker, because the different textures have not been layered correctly. And for FSX and Prepar3D you will also run into the issue of floating polygons due to the curve of the earth. The ground polygon wizard, see Figure 7.2, is a tool that will help you to create a ground polygon BGL file without these issues.

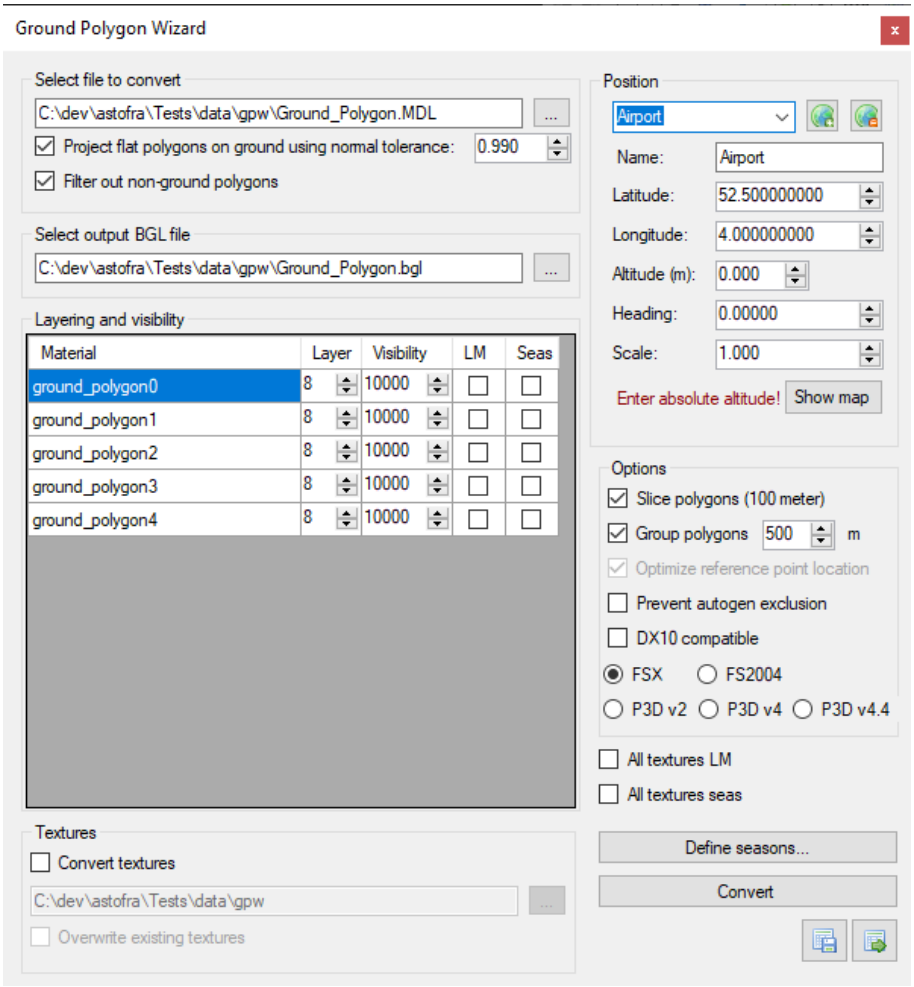


Figure 7.2: Ground polygon wizard

When using the wizard you will fill in all details of the window, the steps are:

1. Select the object that contains the ground polygons using the **...** button. The checkboxes to **Project flat polygons** and **Filter out non-ground polygons** affect how your object is loaded from the specified file. By default both these options are enabled and that will give the best results in general.

   Projecting polygons on the ground means that all polygons for which the dot product of the normal vector and the upwards normal is above the specified threshold value will be projected onto ground level. This lowers them to the ground. So for example the roof of a house will then become a ground plane.

   Filtering out non ground polygons means that all polygons whose Z coordinate values are not zero will be removed from the object.

2. Select the output BGL file for the ground polygon using the **...** button.

3. After selecting the object to convert, the list of materials in the object will be filled. In this grid you see the material name, layer, viability, night texture and seasonal texture status. You can alter any of these columns to specify how the ground polygon should be created.

   For FS2004 and FSX the layer values are positive and typically an increment of 4 is used between the layers The higher the layer number the more priority the texture has. For Prepar3D the layer values are negative, the more negative the number the higher the priority of the texture is. The wizard will make sure that the numbering convention of the selected output simulator is used, so if you enter positive layers, but export to Prepar3D that will be fixed.

   The visibility value indicates at which distance the polygon should become visible. This is only supported for FS2004 and FSX output, else this column will be disabled.

   The night texture (LM) checkbox indicates if this material has a night texture assigned. You will have to ensure yourself that the corresponding texture file also exists.

   The seasonal texture checkbox indicates if this material has seasonal textures assigned. This is only supported for FS2004, FSX and Prepar3D v4 output, else this column is disabled.

4. With the **Convert textures** checkbox you can indicate if the textures of the object should be converted as well. If enabled, the output folder is specified in the textbox below (you can use the **...** button to select the folder). With the **Overwrite existing textures** checkbox you can indicate if existing textures should be overwritten during the conversion.

5. Specify the position where the ground polygons should be placed. See section 13.1 for more information about how to specify the position.

6. In the options section select for which FS version you want to create ground polygons. Certain options might be disabled based on the version you select.

7. The **Slice polygons** checkbox determines if polygons should be sliced into pieces of 100 meter. To be able to follow the curve of the earth correctly your polygons need to be small enough, so when you disable this option make sure you slice them in your modelling tool.

8. The **Group polygons** checkbox determines if polygons are grouped together. These grouped polygons than become one object or reference point in the output BGL file. In general grouping the polygons together for a given area improves the performance in the simulator. With the textbox you can specify the size of the grid that is used for grouping.

9. The **Optimize reference point** checkbox determines if the reference point is placed optimally in the center of the object. When grouping polygons this option is always enabled.

10. The **Prevent autogen exclusion** checkbox determines if your ground polygons should prevent autogen object suppression or not. For FS2004 and FSX output a hack is used to prevent the suppression, for other versions the BGLComp placement flag is used.

11. The **DX10 compatibility** checkbox determines if the FSX output is written in such a way that it also works in DX10 mode of FSX. This however disables the use of night textures.

12. With the **All textures LM** checkbox you can quickly enable or disable the night texture status on all materials.

13. With the **All textures seas** checkbox you can quickly enable or disable the seasonal texture status on all materials.

14. With the **Define seasons** button you open the editor to define the seasons, see section 6.11 for more details.

15. Once all settings are set, you can use the **Convert** button to start the creation of the ground polygon BGL file.

With the **Save settings** button you can save all the settings that you made for the ground polygons to a file, so that you can reuse them later when you want to convert the same object again. With the **Load settings** button you can load them back in again.

7.3 Batch convert wizard

With the batch convert wizard, see Figure 7.3 you can apply the same operation on multiple files. This for example allows you to quickly convert a number of objects from one format to another, but you can also use it to modify objects quickly or to generate screenshots.

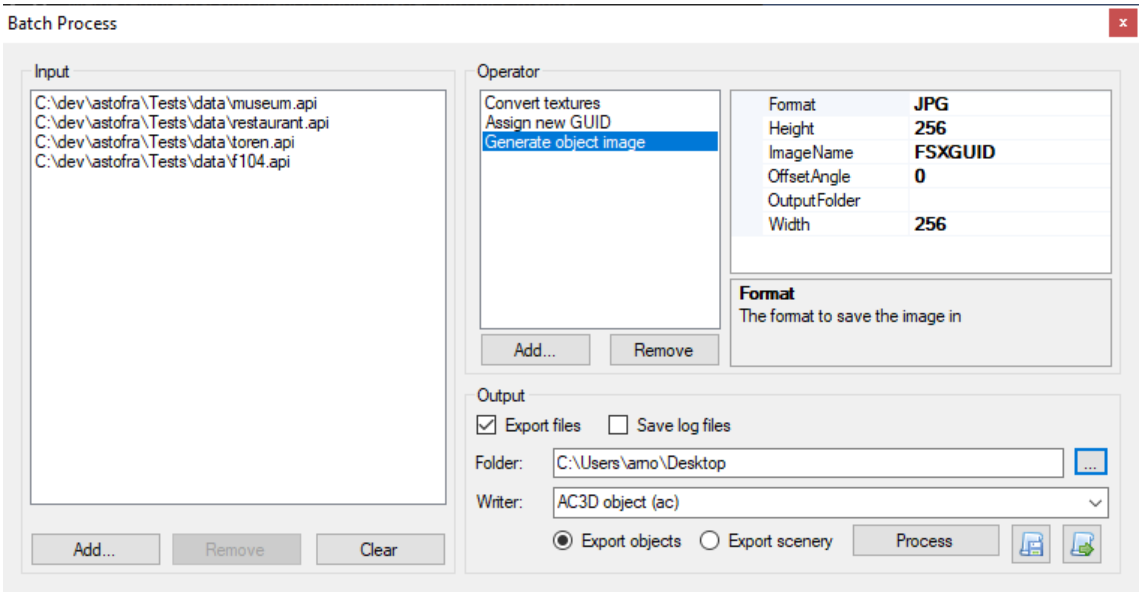


Figure 7.3: Batch convert wizard

On the left side of the wizard window you see a list of all files that should be processed. With the **Add** button you can add more files to the list and with the **Remove** button you can remove the selected file from the list. With the **Clear** button you can remove all files at once from the list.

On the top right of the wizard window you specify which operation should be performed on the files. With the **Add** button you can add a new operator to this list. When you click this button a menu displays a list where you can select the desired operator. See section 7.3.1 for an overview of the available operators. With the **Remove** button you can remove the currently selected operator from the list.

It is not required to select operators for the batch conversion. If you just want to convert from one format to the other, you can do so without using any operators.

The bottom right of the wizard is where you specify what kind of output should be generated. With the **Export files** checkbox you can specify if the wizard should do any export. For example if you want to make screenshots of objects, you will not want to export any files. With the **Save log files** checkbox you can specify if the log files from the export should automatically be saved as well. In the **Folder** textbox you specify in which folder the exported files should be generated With the radio button you can select if you want to **Export objects** or **Export scenery**. Based on this selection different writers can be selected in the **Writers** dropdown list.

Once you have all settings correct, you can press the **Process** button to start the batch operation. With the **Save batch settings** button you can store all settings you made to a file, so that you can run them again later on. With the **Load batch settings** button you can load such a saved configuration.

The sections below describe the operators that are available in the batch wizard and their parameters.

7.3.1 Add empty LOD

Description

This operator adds an empty level of detail to each object.

Parameters

LOD The level of detail value that should be assigned to the empty LOD.

7.3.2 Add placement

Description

This operator adds an object placement for each object.

Parameters

Altitude The altitude value of the object placement.

Heading The heading value of the object placement.

Latitude The latitude value of the object placement.

Longitude The longitude value of the object placement.

NoAutogenSuppression Whether the placement should have no autogen suppression enabled or not.

NoCrash Whether the placement should have no crash enabled or not.

NoFog Whether the placement should have no fogenabled or not.

NoShadow Whether the placement should have no shadow enabled or not.

NoZTest Whether the placement should have no Z test enabled or not.

NoZWrite Whether the placement should have no Z write enabled or not.

7.3.3 Assign material template

Description

Assigns a material template to each material in the object. This can for example be used to assign a night texture to the object. Some material templates are provided by default by ModelConverterX, but you can also make your own using the material template editor. See section 6.5.1 for more information about making material templates.

Parameters

Selected template The material template that should be applied.

7.3.4 Assign new GUID

Description

Assigns a new GUID to each object. If the scenery contains object placement for the object, that GUID will be changed to the new GUID as well.

Parameters

None

7.3.5 Center object

Description

This operator will center the object. If no placement information is included in the scenery, the object will just be centered based on the bounding box, where the Z value is ignored.

If there is placement information then the placement information of the object will also be updated, so that the object remains positioned the same after the loading origin of the object has been centered based on the bounding box.

It is also possible to specify that the input object uses a different projection when centering the object, this can be useful if they have been created from GIS data with a given projection.

Parameters

FSX Flag to indicate if the moved object should be corrected for the FSX earth curve or not.

LocalOrigin Flag to indicate if the projection uses a zero position.

Projected Flag to indicate of the input model uses a projection. When true the projection is used, else FS2004 flat earth coordinates are assumed.

Projection The proj4 string that defines the projection used.

ZeroLatitude Zero position latitude.

ZeroLongitude Zero position longitude.

7.3.6 Convert materials

Description

The convert materials operator changes materials to conform to the specific requirements of different simulators. This is explained in more detail in the section about the material editor, see section 6.5.1. Table 6.1 shows the various conversions that are performed.

Parameters

FromVersion The flight simulator for which the model materials have been made.

ToVersion The flight simulator to which the materials should be converted.

7.3.7 Convert textures

Description

This operator converts the textures of the object to the specified format and saves them in the specified folder. It is also possible to minimize the number of drawcalls in this operator, like the drawcall minimizer in the material editor, see section 6.5.3 for more details.

Parameters

BorderSize Border size that is used by the drawcall minimizer.

MinimizeAmountTextures Flag to indicate that the number of drawcalls should be minimized.

OutputPath The folder where the textures should be saved.

OverwriteExistingTextures Flag to indicate that existing textures should be overwritten.

PrefixWithModelName Flag to indicate that all textures should be prefixed with the name of the object.

RequireTextureSize Specifies if the texture is required to have a specific size. Possible values are None, PowerOfTwo and MultipleOfFour. Textures that do not meet the size requirement are resized during the conversion.

SkipAlpha Flag to indicate if textures with an alpha channel should be skipped when minimizing textures.

TextureFormat The format in which the textures are saved, see section 10.3 for more information about the supported formats.

TextureOrigin The location of the texture origin (not used for all formats, but FSX/P3D use top for DDS, while MSFS and X-Plane use bottom.

TextureSize The maximum texture size as used by the drawcall minimizer.

7.3.8 Create LOD (quadratic error

Description

Add a level of detail to the model using the quadratic error algorithm.

Parameters

Ground clamp Flag to indicate that vertex on the ground should get a higher error.

Triange target Target for the triangle target the algorithm should aim for (percentage between 0 and 100).

Target LOD The LOD number of the new LOD being created.

7.3.9 Create LOD (vertex clustering

Description

Add a level of detail to the model using the vertex clustering algorithm.

Parameters

Floating cells Flag to indicate that floating cells should be used.

Grid size The size in meters of the grid that is used for clustering the vertices.

Target LOD The LOD number of the new LOD being created.

7.3.10 Fix animations

Description

This operator fixes all animations of the objects at the specified frame. This means that the animated part is replaced by a static part with the state of that frame.

Parameters

AnimationFrame The frame number of the frame at which the animation should be fixed.

7.3.11 Flat shade

Description

Flat shades the object. See section 8.16.3 for more details.

Parameters

None

7.3.12 Generate object image

Description

Saves an image of the object to the specified folder. This functionality is similar to the editor that is available, see section 6.15 for more details.

Parameters

FilenameSubFolder When set the true the image is put in a subfolder with the name of the scenery object.

Format The image format in which the image should be saved.

Height The height of the image in pixels.

ImageName The naming convention that should be used for the image.

OffsetAngle The offset angle. When a value other than 0 is entered here, multiple images will be generated where the heading of the object is changed by offset between each image.

OutputFolder The folder where the image should be saved.

Width The width of the image in pixels.

7.3.13 Match textures

Description

This operator is similar to the match textures function in the material editor, see section 6.5.2. It tries to find textures that are the same in the given folder and will update the object to use these textures.

Parameters

MinimumMatch The minimum match level required to match a texture. A value of 1.0 means that all pixels should match between the two textures.

TextureFolder The folder to search for the matching textures.

Tolerance The tolerance when comparing pixel values, when the difference between the pixels in the different textures is within this tolerance the pixels are considered to match.

7.3.14 Only keep highest LOD

Description

This operator will only keep the highest level of detail of the object and remove all other levels of detail.

Parameters

None

7.3.15 Remove UV2

Description

Remove the secondary texture coordinates (UV2) from all parts in the object.

Parameters

None

7.3.16 Remove vertex colors

Description

Remove the vertex colors from all parts in the object.

Parameters

None

7.3.17 Rename object

Description

This operator renames the object. When the scenery contains multiple objects the name if suffixed with an index number.

Parameters

Name The new name of the object.

RenameMode The mode to use when renaming, either Prefix or Rename. When Prefix is used Name is added in front of the existing name. When Rename is used the object name is changed to Name, when the scenery contains multiple objects they get a number assigned in Rename mode.

7.3.18 Rename textures

Description

This operator will rename all textures in the object. The new name for all textures is the object name followed by an increasing number. Warning: this operator only changes the name of the diffuse textures.

Parameters

None

7.3.19 Replace double sided materials by triangles

Description

This operator will replace all double sided materials by a set of additional triangles. See section 8.10 for more information.

Parameters

None

7.3.20 Replace effect

Description

This operator will replace effects and their parameters by the new values provided.

Parameters

FromEffect The effect name to replace.

NewEffectParameters The new value of the effect parameters when they are replaced.

ReplaceAllEffects Replace all effects, if true all effects are replace by ToEffect, if false only those for which the name matches FromEffect.

ReplaceEffectParameters Replace the effect parameters for the replaced effects as well. The value is taken from NewEffectParameters.

ToEffect The effect name to replace with.

7.3.21 Scale model

Description

This operator scales all models with the specified scale factor.

Parameters

ScaleLOD Flag to indicate if the LOD values should be adjusted when scaling the model as well.

ScaleX Scale factor for the X axis.

ScaleY Scale factor for the X axis.

ScaleZ Scale factor for the X axis.

7.3.22 Smooth shade

Description

Smooth shades the object. See section 8.16.4 for more details.

Parameters

None

7.3.23 Update object placement

Description

This operator updates attributes of the object placements in a scenery.

Parameters

NoAutogenSuppression Whether the placements should have no autogen suppression enabled or not.

NoCrash Whether the placements should have no crash enabled or not.

NoFog Whether the placements should have no fogenabled or not.

NoShadow Whether the placements should have no shadow enabled or not.

NoZTest Whether the placements should have no Z test enabled or not.

NoZWrite Whether the placements should have no Z write enabled or not.

ProcessAllPlacements When true all placements in the scenery are processed, when false only the placements for which the object model is contained in the scenery as well are processed.

7.4 Scene builder wizard

With the scene builder wizard, see Figure 7.4, you can construct one single object based on a scenery with multiple objects and object placements. This can be useful if you want your entire scenery to consist of a single object, for example if you want to convert to it another system that prefers to have an airport as one object, while Flight Simulator works more efficiently if you have it split into multiple objects. What the scene builder does is use the placement information to determine the offset of each object and in that way combine all objects into a single scene.

On the right side of the wizard you see a list of scenery files that are selected to be processed. You can add new scenery files with the **Add Scenery File** button, while you can remove the selected item from the list with the **Remove Scenery File** button. If you want to clear the entire list you can use the **Clear Files** button.

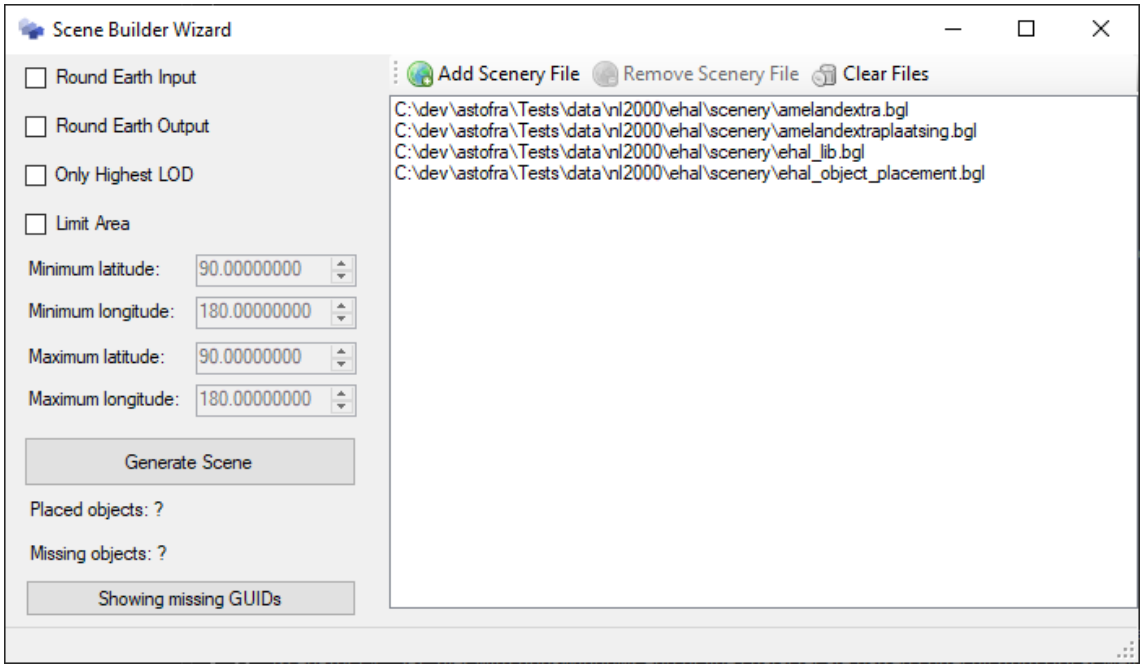


Figure 7.4: Scene builder wizard

On the left side of the wizard you can set options for the wizard. The following items can be set:

- With the **Round Earth Input** checkbox you can indicate if the input scenery has been corrected for the round earth or not. If so, the tool will first reverse this correction before building the combined scene from the objects.
- With the **Round Earth Output** checkbox you can specify if the combined scene that the wizard generates should also be corrected for the curve of the earth.
- When the **Only Highest LOD** checkbox is checked, the wizard will only use the highest LOD from each object in the scene. For sceneries where objects have different levels of detail you might otherwise get an unexpected result with certain objects only visible at specific level of detail values.
- When the **Limit Area** checkbox is checked, the scene builder wizard will only include object placements that are within the specified area. You can specify the minimum and maximum latitude and longitude values of this area using the text boxes below.

Once you are happy with all the settings you can press the **Generate Scene** button to start the generation. For a complex scenery with many objects and placements this might take a while. Once the wizard is done the combined scene object is shown in the preview window. The **Placed objects** and **Missing objects** labels show how many objects have been included in the combined scene and how many placements have been skipped because the referenced object could not be found. With the **Show missing GUIDs** button you can get a list of the GUIDs of the objects that the scene builder wizard could not include. Figure 7.5 shows an example of a combined object of an airport that was made with the scene builder wizard.

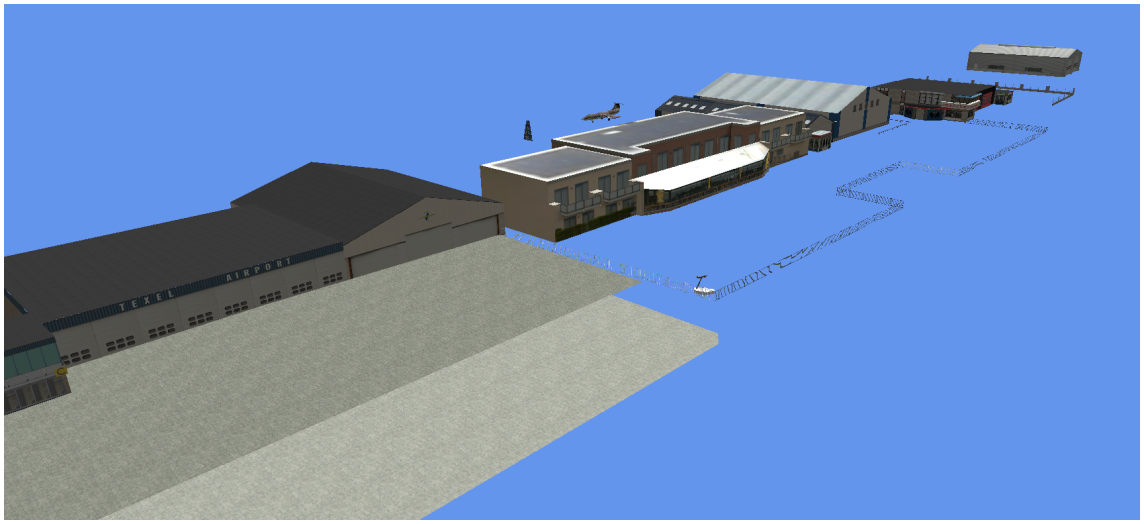


Figure 7.5: Merged scene made with scene builder

7.5 Building Creator Wizard

This section describes the Building Creator Wizard. With this wizard you can draw a footprint of a building and the wizard will generate a building model for you. In the sections below the drawing control and the algorithm that generates the building are described in more detail. Once you have create a building with the wizard, you can export it like any other model from ModelConverterX.

7.5.1 Building drawing control

When you open the Building Creator Wizard, the building drawing control as shown in Figure 7.6 is opened. In the top part of the this control you can draw the footprint of your building and in the bottom part you can modify the properties of the selected building element.

Toolbars

At the top of the window there are two toolbars. The first toolbar provides the following functions:

- **Save** saves the building to a Building Definition (BD2) file.
- **Load** loads a building from a Building Definition (BD2) file.
- **New** creates a new building and clears the currently loaded building. After pressing this button a new polygon is added automatically and you are in edit mode, so you can start drawing the vertices of the polygon.
- **Add polygon** adds an additional building to the building. After pressing this button you are in edit mode, so you can start drawing the vertices of the polygon.
- **Add chimney** adds a chimney to the building. If you have a polygon selected the chimney will be place inside that polygon.
- **Add dormer** adds a dormer to the building. If you have a polygon selected the dormer will be place inside that polygon.
- **Remove polygon** removes the currently selected polygon from the building.
- **Remove edge** removes the currently selected edge from the polygon.
- **Remove vertex** removes the currently selected vertex from the polygon.

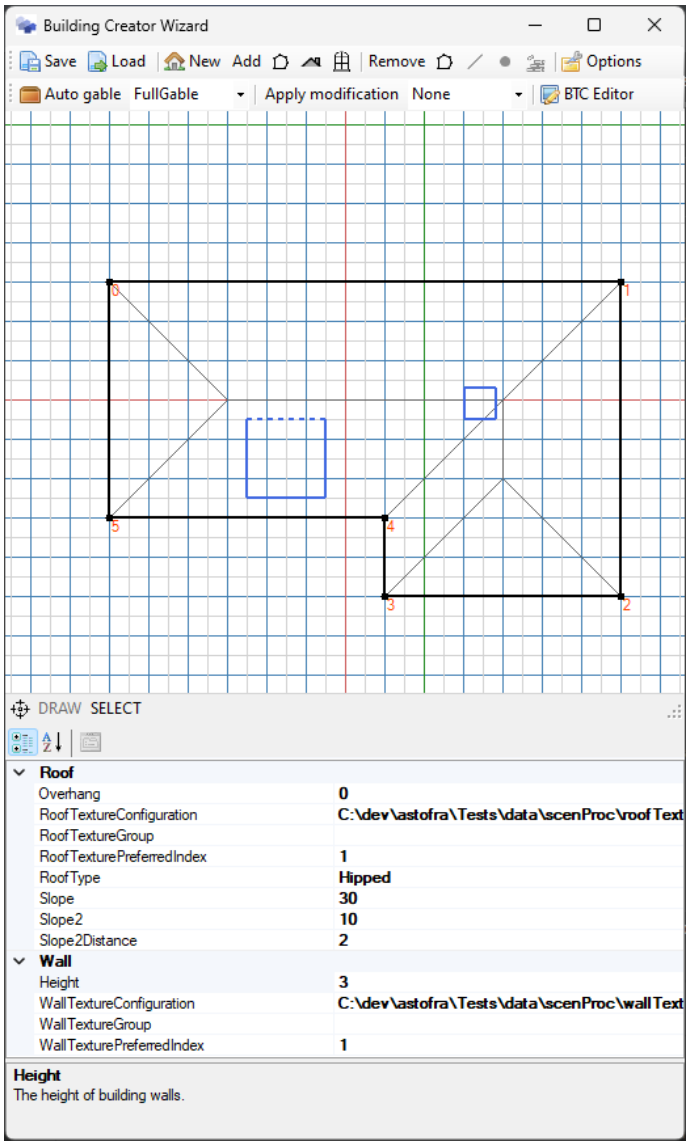


Figure 7.6: The building drawing control of the Building Creator Wizard

- **Remove feature** removes the currently selected feature (chimney or dormer) from the building.
- **Options** opens the options form, where you can specify the default texture configuration file, the default texture group and the default texture index for both the wall and roof texture. See Figure 7.7 for the options form that will open.

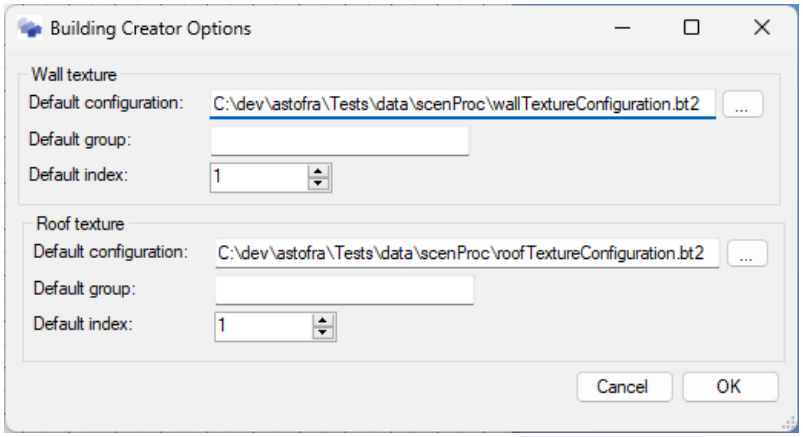


Figure 7.7: The options of the Building Creator Wizard

The second toolbar provides the following functions:

- The **Auto gable** button automatically assigns gabled edges to your footprint. With the drop down list at the right of the button you can select the type of gable (NoGable, FullGable or HalfGable). See section 7.5.3 for more information on the different types of gables.
- The **Apply modification** button applies the modification that is selected in the drop down list next to it. See section 7.5.3 for more details about these modifications.
- **BTC Editor** opens the Building Texture Configuration Editor. In this editor you can define the texture that is used to generate the building. See section 7.5.5 for more details about using the editor.

At the bottom of the drawing control there is a status bar which provides the following information:

- The **Center** button will center the drawing area on the origin again.
- When the **DRAW** label is shown in black the control is in drawing mode.
- When the **SELECT** label is shown in black the control is in selection mode.
- When an error occurs while generating the building a red label **Generation error** will show up. This typically happens while moving vertices, try to move the vertex further and the building should generate again.

Drawing a footprint

You can pan around in the drawing area by dragging with the right mouse button pressed. With the zoom wheel you can zoom the drawing area in and out. If you want to reset the panning back to the center you can press the **center** button in the status bar.

When the control is in the drawing mode you can add new vertices to the current polygon by clicking with the left mouse button. While you move the mouse you will dynamically see how the polygon would become when you add a vertex at the current mouse location. The mouse will snap to the grid lines that are shown in the control, this makes it easier to make straight lines. When you double click the location of the double click is added as last vertex to the polygon and the drawing mode is exited.

When the control is in selection mode you can click on polygons, edges, vertices or features to select them. This will display the properties of the selected element. You can also remove the selected element using the remove buttons in the toolbar.

If you have selected a vertex, you can move it by holding the Control button while moving the mouse. The vertex will then be moved to the current cursor position.

If you have selected an edge, you can split the edge by pressing the Control button and clicking on the location where a new vertex should be added. After you have done this the new vertex is selected automatically so that you can move it to the desired location if you keep the Control button pressed.

If you have selected a feature you will see three control dots rendered at the feature. Dragging these dots allows you to manipulate the feature.

- With the **red** control point you can move the feature to a different location.
- With the **green** control point you change the size of the feature. The size if specified as a length and width for the current direction of the feature.
- With the **blue** control point you can change the direction of the feature.

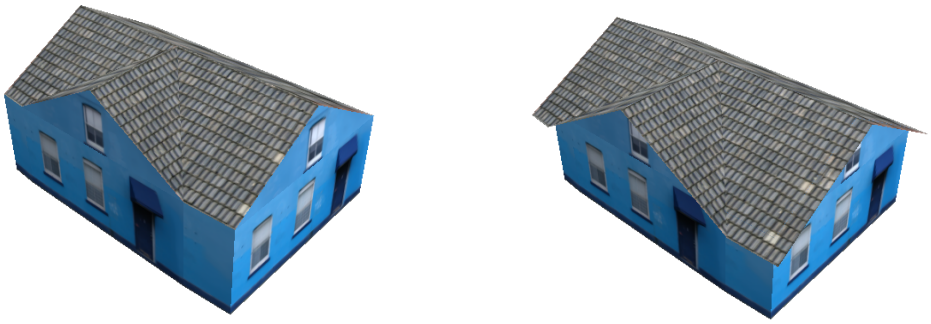
Editing properties

In the property grid you can edit the attributes of the selected object. You can either select a polygon, edge or a feature (chimney or dormer). Vertices can also be selected, but these have no properties. Below for each type of object the properties are described.

Polygon properties

Height The height of the walls for this polygon in meters.

Overhang The overhang of the roof in meters. When you specify an overhang the roof is made larger than the walls of the building. This can generate more realistic buildings. To keep the polygon count of the buildings low, the underside of the overhanging roof is not included in the model, as you would not see this from the air generally. See Figure 7.8 for an example of a roof with overhang.



(a) (b)

Figure 7.8: Example of a building (a) without and (b) with an overhanging roof of 0.5 meter

RoofTextureConfiguration The texture configuration file that is used for the roof texture.

RoofTextureGroup The group in the texture configuration file that is used for the roof texture. If left empty a texture is selected from any group.

RoofTexturePreferredIndex The index of the roof texture that is selected. Leave at the default value of -1 to randomly select a texture.

RoofType The roof type to use on this polygon. See section 7.5.2 for more details. Possible values are:

- Flat
- Hipped
- Mansard

Slope The slope of the roof in degrees. This is not used for flat roofs.

Slope2 The secondary slope of the roof in degrees. This is only used for mansard roofs.

Slope2Distance The distance along the roof edge when the slope changes to the secondary slope. This is only used for mansard roofs.

WallTextureConfiguration The texture configuration file that is used for the wall texture.

WallTextureGroup The group in the texture configuration file that is used for the wall texture. If left empty a texture is selected from any group.

WallTexturePreferredIndex The index of the wall texture that is selected. Leave at the default value of -1 to randomly select a texture.

Edge properties

Gable The type of gable that should be generated for this edge. See section 7.5.3 for more information on gabled roofs. Possible values are:

- NoGable: no gable is generated for this roof, the roof will be sloped.
- FullGable: a gable is generated for this roof.
- HalfGable: a gable is generated for the first part of the mansard roof and a sloped roof for the second part. For non-mansard roofs this gives the same result as FullGable.

TextureMappingPreference This specifies if building generation algorithm should try to map specific texture content on this edge. Possible values are:

- NoPreference: there is no preferred texture content, any piece of the texture that fits can be mapped.
- Window: the algorithm should try to map only windows on this wall.
- Door: the algorithm should try to map only doors on this wall.
- DoorWindow: the algorithm should try to map doors and windows on this wall.
- NoSemantics: the algorithm should try to map a wall without semantics (e.g. no windows or doors) on this wall.

Chimney properties

Height The height of the chimney. For a slope roof the height is measured from the vertex of the chimney point that has the highest roof height. See section 7.5.4 for more details on chimneys.

RoofTextureConfiguration The texture configuration file that is used for the roof texture.

RoofTextureGroup The group in the texture configuration file that is used for the roof texture. If left empty a texture is selected from any group.

RoofTexturePreferredIndex The index of the roof texture that is selected. Leave at the default value of -1 to randomly select a texture.

WallTextureConfiguration The texture configuration file that is used for the wall texture.

WallTextureGroup The group in the texture configuration file that is used for the wall texture. If left empty a texture is selected from any group.

WallTexturePreferredIndex The index of the wall texture that is selected. Leave at the default value of -1 to randomly select a texture.

Dormer properties

RoofTextureConfiguration The texture configuration file that is used for the roof texture.

RoofTextureGroup The group in the texture configuration file that is used for the roof texture. If left empty a texture is selected from any group.

RoofTexturePreferredIndex The index of the roof texture that is selected. Leave at the default value of -1 to randomly select a texture.

RoofType The type of roof used on the dormer. See section 7.5.4 for more details. Possible values are:

- Flat
- Sloped
- Hipped
- GabledHipped

Slope The slope of the dormer roof in degrees. The slope is not used for flat roofs.

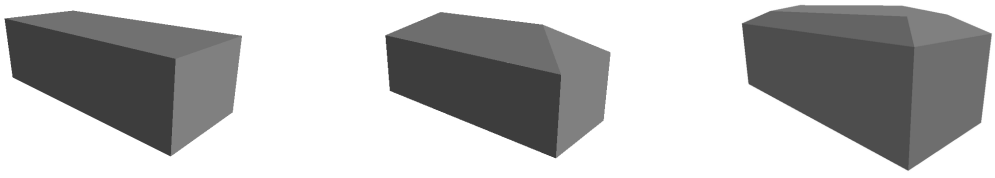
WallTextureConfiguration The texture configuration file that is used for the wall texture.

WallTextureGroup The group in the texture configuration file that is used for the wall texture. If left empty a texture is selected from any group.

WallTexturePreferredIndex The index of the wall texture that is selected. Leave at the default value of -1 to randomly select a texture.

7.5.2 Roofs

The building generation algorithm does support three basic types of roofs, Figure 7.9 gives a graphical representation of these roofs.



(a) (b) (c)

Figure 7.9: The basic roof types: (a) flat, (b) hipped and (c) mansard

Flat roofs are the most simple roof type and they can be applied to any building footprint shape.

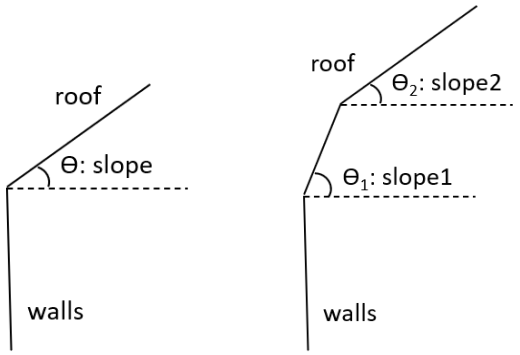
Hipped roofs are created by continuously shrinking the building footprint so that the skeleton of the roof is defined. This is called a straight skeleton and the edges of the skeleton are then given a slope to define the shape of the hipped roof. The definition of the slope angle for hipped roofs is given in Figure 7.10.

In general hipped roofs can be applied to any footprint shape. But if the building is very wide, it is best to keep the slope angle of the roof low, else you will get very tall roofs that look unrealistic.

Mansard roofs are similar to hipped roofs, but instead of using a constant slope for the edges of the skeleton, for a mansard roof two slope angles are used. Once the distance along the

edge is over a certain threshold value a secondary slope angle is used. This gives roofs with two different slopes. The definition of the slope angles is given in Figure 7.10.

In general mansard roofs can be applied to any footprint shape. But if the building is very wide, it is best to keep the slope of the roof low, else you will get very tall roofs that look unrealistic.

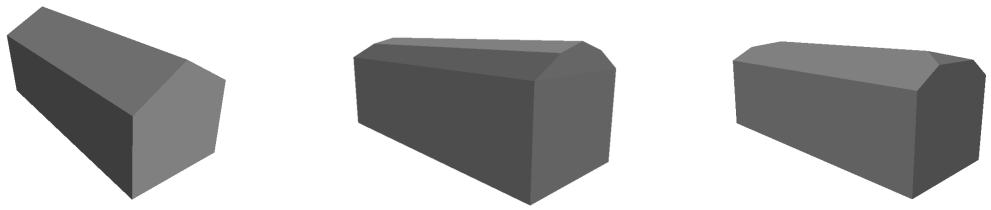


(a) (b)

Figure 7.10: Definition of the slope angle for (a) hipped and (b) mansard roofs

7.5.3 Gables

More various of the three basic roof types, as described in section 7.5.2, can be added by using gabled roofs. Gabled roof various from the previous roofs in the sense that certain edges of the skeleton will be turned into vertical walls, instead of slope roof segments. Figure 7.11 gives a graphical representation of different gabled roofs. The semi-gabled mansard roof is created by creating a wall from the segment of the roof that has the primary slope, while keeping the sloped roof for the segment with the secondary slope angle.



(a) (b) (c)

Figure 7.11: Gabled roofs: (a) gabled hipped, (b) gabled mansard (c) semi-gabled mansard

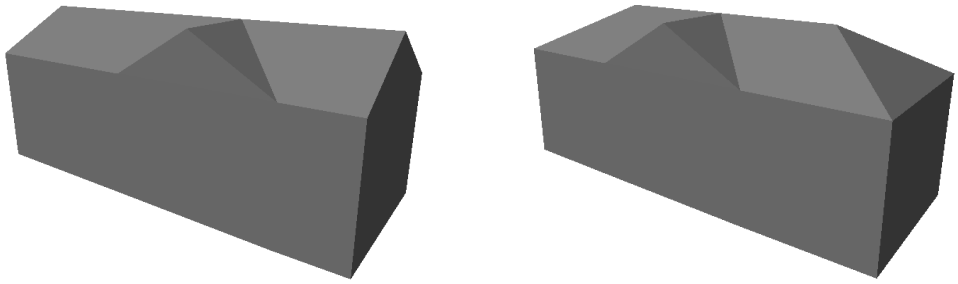
The building algorithm can automatically assign which edges should become gabled. This is done by evaluating the angles between the edges in the footprint and will assign gables to the shorter sides of a rectangle. Since this algorithm works best on footprints that are not too complex, it is advised to use gables only on footprints with not too many vertices and that contain only perpendicular angles.

When using the interactive mode to define a building footprint it is also possible to manually assign which edges should become gabled.

Another way to add gables to buildings is by applying a footprint modification. This will add an additional gable on top of the one that is assigned automatically for rectangular and L-shaped footprints.

The footprint modification for rectangles does modify one of the longest edges of the footprint so

that an additional gable is added. Figure 7.12 shows how this looks for a gabled and a hipped roof. You need to make sure that the polygon you run the modification on is rectangular, which means it has 4 vertices and 4 perpendicular angles.



(a) (b)

Figure 7.12: Rectangular buildings with an extra gable (a) gabled roof (b) hipped roof

The footprint modification for L-shaped does modify one of the edges of the footprint so that an additional gable is added. This additional edge is placed opposing to one of the legs of the L-shape. Figure 7.13 shows how this looks for a gabled and a hipped roof. You need to make sure that the polygon you run the modification on is L-shaped, which means it has 6 vertices and 6 perpendicular angles.

7.5.4 Features

The building algorithm can also generate additional features for the building. It is possible to add chmineys and dormers. These are explained in the sections below.

For each feature you can set a position, a direction and a size. In the interactive editing mode you can set these yourself, if you use the automatic placement logic for the features they are calculated based on the shape of the footprint.

Chimney

A chimney can be added on any roof type, see Figure 7.14 for a graphical representation. If the roof texture configuration file that you use contains a group called ___CHIMNEY this texture is automatically selected for the chimney. It is adviced to have a relatively dark roof texure in that group, as that will look best on a chimney.

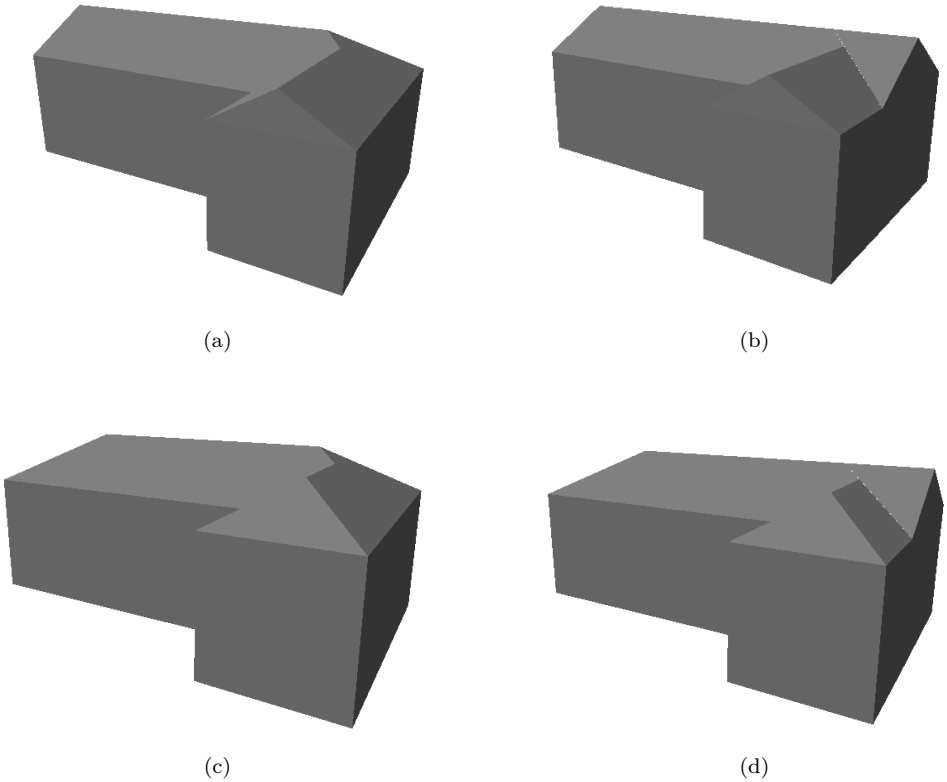
Dormer

A dormer can only be added to a building that does not have a flat roof. Dormers can have different roof types, see Figure 7.15 for a graphical representation. The possible roof types are:

- Flat roof
- Sloped roof
- Hipped roof
- Gabled hipped roof

7.5.5 Building Texture Configuration Editor

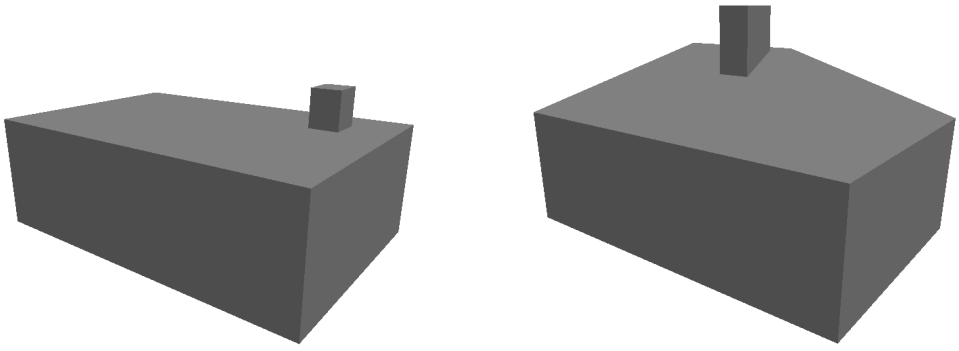
Until now only the geometry of the building has been discussed, but for the final appearance of the building in your scenery also the texture that is applied to the building is very important, see Figure 7.16 for an example of a textured building made by the building algorithm.



(a) (b)

(c) (d)

Figure 7.13: (a) hipped L-shaped building (b) hipped L-schaped building with extra gable (c) gabled L-shaped building (d) gabled L-shaped building with extra gable



(a) (b)

Figure 7.14: Buildings with a chimney (a) flat roof (b) hipped roof

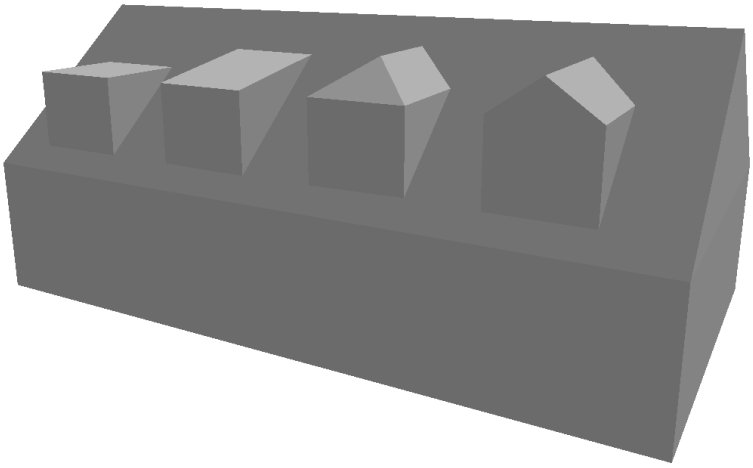


Figure 7.15: Building with dormers. From left to right (1) flat roof, (2) sloped roof, (3) hipped roof and (4) gabled hipped roof



Figure 7.16: Example of a building with a texture applied

For autogen buildings Flight Simulator has a fixed layout of the texture that is used and therefore any custom textures you make need to follow that exact layout. The building generation algorithm does not work like that, you can use any texture layout that you want. But to allow the algorithm to map the texture realistically on the building you have to provide additional information to be able to define the optimal mapping. For example which part of the texture contains the walls and the roofs. And where the windows and doors are located, so that the algorithm can ensure that they are not split over different wall segments. That is defined in the Building Texture Configuration file. When creating a building you can specify a configuration file for the walls and another one for the roofs.

When the algorithm makes a gabled roof it tries to prevent that windows are clipped by the slope of the gable as well. Two approaches can be used for this:

1. Try to find a window in the wall texture segment that has enough empty space around it to fit on the gable wall. See Figure 7.17 for an illustration of this strategy. The yellow rectangle should contain a window, while the other parts of the gable triangle should not contain a window.

2. Try to find a area in the wall texture segment without any windows and use that on the gable wall.

In the Building Texture Configuration file you can specify what is the preferred approach of these two is. If the preferred approach fails, the algorithm will automatically try the other as fall-back.



Figure 7.17: Mapping strategy for gable with window

When you are placing multiple buildings in your scenery, you also don't want that they all look the same. Therefore you can specify multiple wall and roof textures in your Building Texture Configuration file and the algorithm can randomly pick one for each building that is created. You can also create different groups within the building texture configuration file and then you can instruct the algorithm to pick a texture for a specific group. You can for example create groups for different colors of roofs (dark grey, light gray, red, etc).

All of the topics mentioned above are configured in the Building Texture Configuration file and you can use the Building Texture Configuration Editor, see Figure 7.18, to make and edit this configuration file. The texture shown in Figure 7.19 is used as an example in this section to explain how that is done. You can see that this building texture configuration file contains 3 different wall segments.

In the Building Texture Configuration Editor you see a list of texture segments on the left side. This list contains either wall or roof segments, depending on whether this configuration file is used for walls or roofs. On the right side you see the properties of the selected texture segment and graphical quality indicators of your texture. The part in the middle is used to show the texture segment you are configuring. Each of these areas of the editor will be described below.

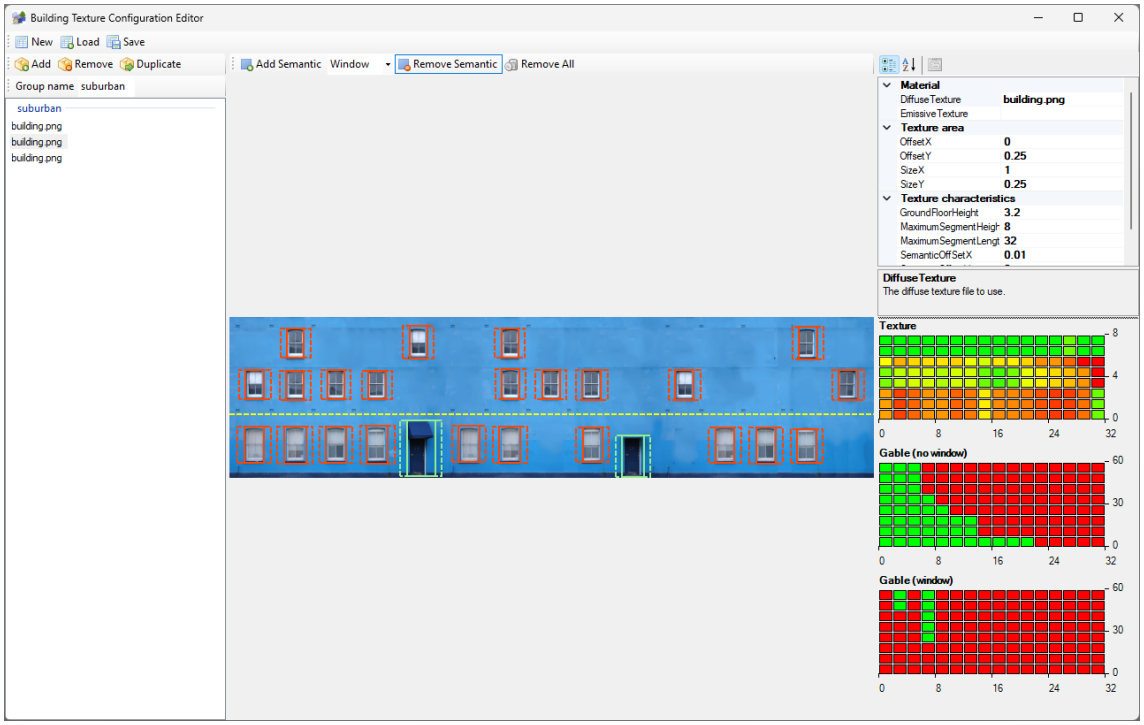


Figure 7.18: Building Texture Configuration Editor



Figure 7.19: Example building texture

Main toolbar

The toolbar at the top of the Building Texture Configuration Editor gives you access to the following functions:

- **New** creates a new empty configuration file.
- **Load** reads a saved BT2 configuration file from disk.
- **Save** writes the configuration file to disk as a BT2 file.

Texture segment list

The list of texture segments, see Figure 7.20, lists all the texture segments of this building texture configuration file. If you click on a texture segment from the list, this segment will be shown in the central picture and the properties are shown in the property dialog. The texture list has the following buttons in the toolbar:

- **Add** will add a new texture segment to the list. After clicking Add you will get a file selection dialog to select which texture file should be used for this segment.
- **Remove** will remove the currently selected texture segment from the Building Texture Configuration file.
- **Duplicate** will make a copy of the currently selected texture segment and add it to the list. This can be useful if you want to add another segment that only differs slightly from an already existing one.
- **Group name** specifies the group that the currently selected roof texture segment belongs to. By changing this name you can assign it to a different group. When making the building you can specify that the algorithm should only select wall or roof texture segments from a specific group.

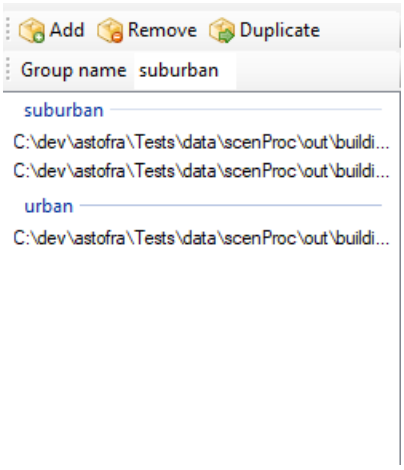


Figure 7.20: Texture segment list

Property list

The property list shows all the properties of the selected texture segment. The following properties are available:

- **Diffuse texture** specifies the diffuse texture that is used for this texture segment (When using a PBR material this texture is used as albedo texture).
- **Emissive texture** specifies the emissive texture that is used for this texture segment. When no name is specified the building will not have an emissive texture.

- **Ground floor height** specifies the height of the ground floor in meters for wall texture segments. When the wall texture segment has to be repeated vertically the ground floor is excluded, to prevent that you get doors floating in the air, see Figure 7.21. In the texture preview the ground floor height is shown as a dashed yellow line. When the entire texture can be repeated vertically you can enter a ground floor height of zero.



Figure 7.21: Example of building where the ground floor is not repeated vertically

- **Maximum segment length** specifies what the length of the texture segment is in meters. This information is used to scale the texture correctly on the building. When the building is longer than this length the polygons are split into multiple polygons to make sure that the texture can be applied correctly.
- **Metallic** specifies the metalic level of the material. This is only used for PBR materials.
- **MetalSmoothAo texture** specifies the metallic/smoothness/AO texture that is used for this texture segment. When no name is specified the building will not have a metallic/smoothness/AO texture. This texture is only used for PBR materials.
- **Normal texture** specifies the normal texture that is used for this texture segment. When no name is specified the building will not have an emissive texture. This texture is only used for PBR materials.
- **Offset X** and **Offset Y** specify the offset of this texture segment in the texture sheet as a percentage of the size of the texture sheet. This way you can use different segments from the same texture sheet on your building. Table 7.1 shows the offset values for the sample texture sheet shown in Figure 7.19.
- **Semantic offset X** and **Semantic offset Y** specify if an additional offset should be added in the X and Y direction around semantics (windows, doors). Such an offset can be used to prevent that a door or window is mapped directly at the corner of two walls. The offset is specified relative to the size of the texture sheet. For example a value of 0.01 means that the offset is 1% of the size of the texture. In the texture preview the window offset is shown as a red dashed rectangle.
- **Size X** and **Size Y** specify the size of this texture segment as a percentage of the size of the texture sheet. This way you can use different segments from the same texture sheet on your building. Table 7.1 shows the size values for the sample texture sheet shown in Figure 7.19.
- **Smoothness** specifies the smoothness of the material. This is only used for PBR materials.
- **UsePBRMaterial** specifies if the material is a PBR material or a standard FSX material.
- **Window gable percentage** specifies which percentage of the gables should get a window.

Texture segment	Offset X	Offset Y	Size X	Size Y
wall 1	0.00	0.00	1.00	0.25
wall 2	0.00	0.25	1.00	0.25
wall 3	0.00	0.50	1.00	0.25
roof 1	0.00	0.75	0.50	0.25
roof 2	0.50	0.75	0.50	0.25

Table 7.1: Offset and size values for the sample texture

A value of 1.0 means that the algorithm tries to place a window on all gables, while a value of 0.0 means that none of the gables should get a window.

Texture preview & Semantic editing

The texture preview shows you the currently selected texture segment, see Figure 7.22. This way you can easily verify that you have entered the correct offset and size values for your segment. Besides that you can also edit the semantic definitions in the preview. These semantics define where windows, door and other elements are that the texture mapping should take into account, as you don't want such semantic features to be split on your building. To edit the semantics you use the buttons shown in the toolbar:

- **Add semantic** enables the mode where you can add new semantics. By clicking and dragging with the mouse you can draw new semantics on top of the texture. The semantic you are currently adding is shown as a green rectangle. Clicking again on the button will disable the add mode again. In the drop down list next to the button you can select the type of semantic you want to add, the following types are available:
  - Window, shown in red on the texture segment.
  - Door, shown in light green on the texture segment.
  - Other, shown in dark pink on the texture segment. Other is an element on the texture that is not a door or a window, but still should not be split when mapping the texture. For example a drainage or a mailbox attached on the wall.
- **Remove semantic** enables the mode where you can remove semantics. When this mode is enable a click inside a semantic will remove that semantic from the Building Texture Configuration. Clicking again on the button will disable to remove mode.
- **Remove all** removes all semantics from the Building Texture Configuration so that you can start with a clean sheet.

Texture quality indicators

Depending on the layout of your texture segment it might be easier or harder for the algorithm to map it on a building. For example if you have a lot of windows in the segment, it might be hard to map it on a wall without putting any window on the edge. How hard it is to map the texture also depends on the length of the wall segment that is being mapped of course. The texture quality indicators, see Figure 7.23 are a visual indication of how suitable your texture segment is to map on walls or different lengths. This should help you in designing a texture segment that is well balances, so that it can be applied on buildings with different shapes and sizes. Three different indicators are available:

- The **Texture** indicator shows how easy it is to map the texture on a wall or roof of a given length. The horizontal axis indicates the length of the wall, between 0 and the maximum segment length. The vertical axis indicates the vertical areas in your segment. The quality indicator shown in Figure 7.23 is for the texture segment shown in 7.22. As the ground floor of that texture segment has more windows, you can see that vertically the bottom of the texture segment is harder to map (more orange or red squares), while the top part is easier



Figure 7.22: Semantic editing

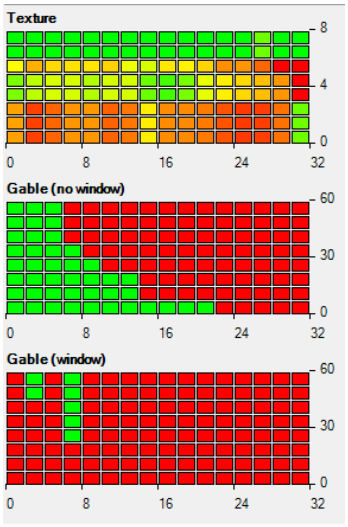


Figure 7.23: Texture quality indicators

to map. Horizontally you can see that a wall of 4 meters in length is harder to map than a wall of 16 meters.

- The **Gable (no window)** indicator shows you how suitable the texture segment is to map on a gable without any windows. The horizontal axis once again shows the length of the gable, between 0 and the maximum segment length. The vertical axis shows the slope of the gable in degrees. The quality indicator shown in Figure 7.23 is for the texture segment shown in 7.22. You can see that for gables longer than 16 meter it is very hard to map the texture, this is because these require a big area of the texture sheet without windows and at the higher slope angles the gable height might be higher than the height of the wall texture segment. But for slopes in the range of 30 to 45 degrees you can see that gables up to 8 meter in length can be mapped correctly without any windows on them.
- The **Gable (window)** indicator shows you how suitable the texture segment is to map on a gable with a window. The horizontal axis once again shows the length of the gable, between 0 and the maximum segment length. The vertical axis shows the slope of the gable in degrees. The quality indicator shown in Figure 7.23 is for the texture segment shown in 7.22. You can see that there are only a few gables where a window can be mapped correctly. This is partly due to the design of the texture segment, it only has a few windows in it that have enough empty space around them to fit on a gable. For gables with a small slope angle it is also hard to map a window, since these will not be very tall.

Chapter 8

Special tools

In this chapter all functionalities that are available via the special tools menu in the ModelConverterX toolbar are discussed.

8.1 GUID converter

With the GUID converter you can convert a GUID between the FS2004 and FSX notation. You enter the GUID in one of these two formats and then using the two arrow buttons you can convert it to the other notation. See Figure 8.1.

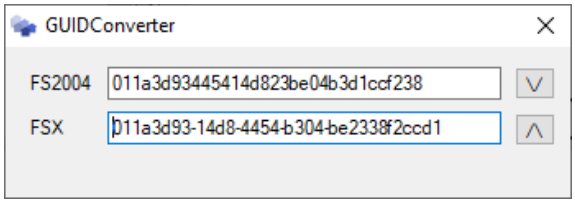


Figure 8.1: GUID converter

8.2 Coordinate converter

The coordinate converter can be used to convert coordinates between three different representations:

- Geodetic coordinates (latitude, longitude, altitude)
- Flat earth coordinates (XYZ from a reference position)
- Geocentric coordinates (XYZ from a reference position)

Figure 8.2 shows the converter window. When you change a coordinate in any of the representations, the other representations will be updated. This way you can for example calculate how big a polygon should be to accurately position in image for which you know the geodetic coordinates. But it can also be used to calculate how much offset the curved earth will have at a given location.

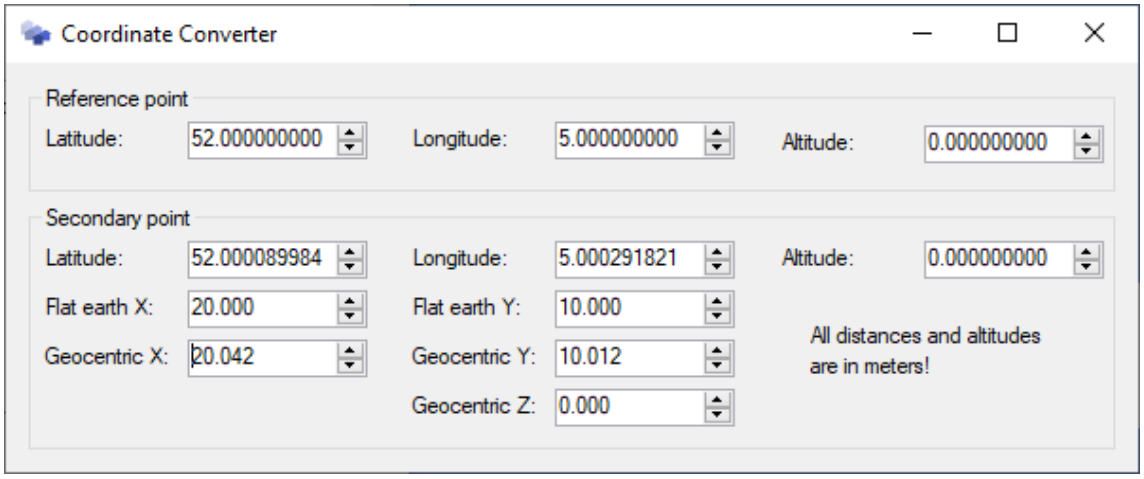


Figure 8.2: Coordinate converter

8.3 Texture converter

With the texture convert tool you can view and edit texture files, see Figure 8.3.

In the toolbar the following options are available:

- With the **Load file** button you can load a texture file from disk. See section 9.2 for more information on the supported formats.
- With the **Save file** button you can save the texture file to disk again. With the dropdown list next to the button you can select the format that it should be saved in. See section 10.3 for more details about the supported formats.
- With the **Set transparent color** button you can choose which color in the texture should become transparent. The alpha channel will then be generated based on this.
- With the **Convert normalmap to FS** button you can convert a normalmap to the FSX specifications. You load a texture made by a normalmap plugin and then save it to DDS after pressing this button.
- With the **Resize** button you can resize the texture to the specified size.
- Using the channel buttons you can select if you want to see all channels, or only the red (R), green (G), blue (B) or alpha (A) channel.
- With the **mip level** dropdown menu you can select which mip map of the image is rendered in the window.



Figure 8.3: Texture converter

8.4 Missing texture finder

With the missing texture finder, see Figure 8.4, you can inspect your scenery and check if all textures referenced by the objects are present in the texture library as well.

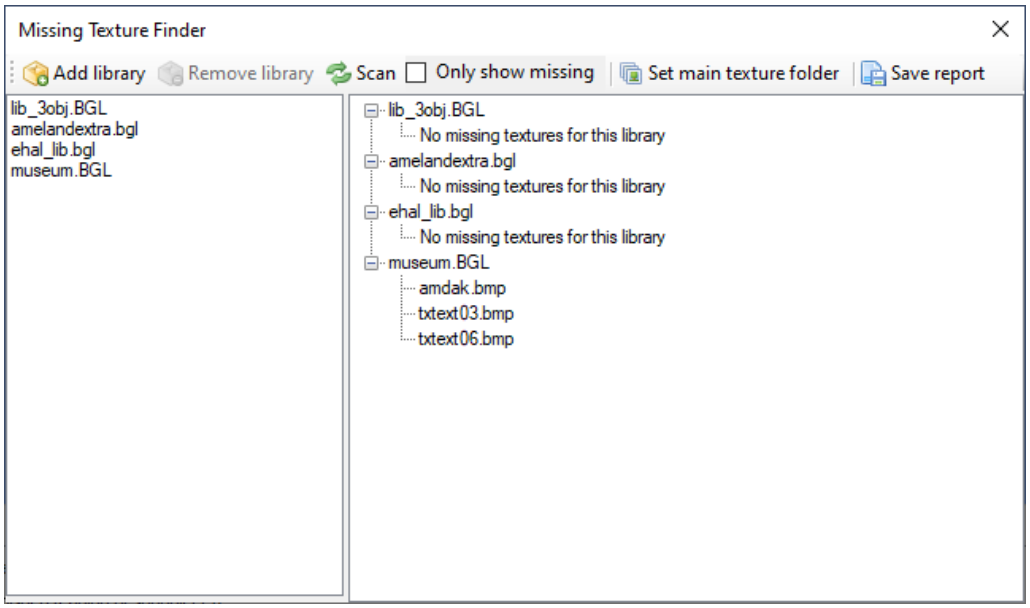


Figure 8.4: Missing texture finder

With the **Add library** button you can add the library BGL files for which you want the textures to be checked. The library BGL files are shown in the list on the left of the window. With the **Remove library** button you can remove the selected library BGL from that list again. Using the **Set main texture folder** button you can specify where the main texture folder is located, that way the missing texture finder will be able to check the texture folder of your scenery and the main texture folder as well.

When you press the **Scan** button the textures will be checked and in the treeview on the right the results are shown. For each library BGL file it will be reported which textures are missing. With the **Only show missing** checkbox you can configure that only library BGL files that have missing textures are shown in the treeview, library files without textures are then not be shown. With the **Save report** button you can save the results to a text file.

8.5 MDL tweaker

With the MDL Tweaker, see Figure 8.5, you can make changes directly to the MDL file. This is fundamentally different from importing the object into ModelConverterX and making the changes there. The difference is that when you import the model in ModelConverterX it is read into the internal representation and on export it is saved to the MDL format again. This can result in minor differences in the MDL file. With the MDL tweaker you can modify the binary code of the MDL file directly, so that you are sure no other changes are made. The MDL tweaker is only for FSX and Prepar3D MDL files.

Using the **...** button you select the MDL file that you want to tweak. Once it has been loaded the information fields are filled. You can now modify the following information in the MDL file:

- Name
- GUID
- Object radius

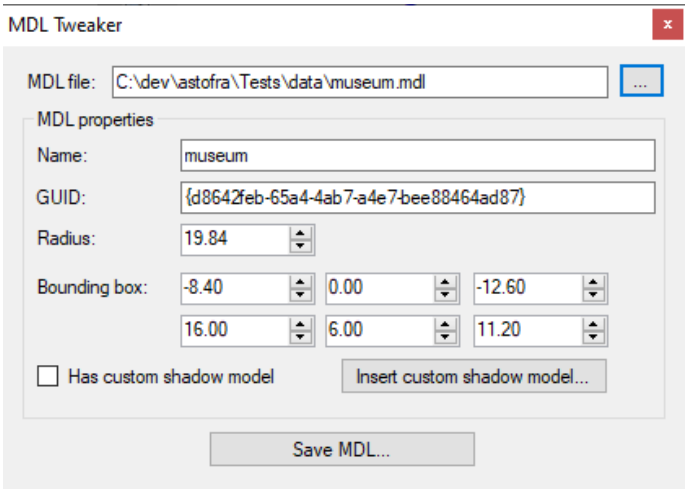


Figure 8.5: MDL tweaker

- Bounding box

With the **Insert custom shadow model** you can select another model that should be used as the shadow model for your object. Sometimes using a simplied geometry for the shadow model can result in performance improvements. When the **Has custom shadow model** checkbox is checked the MDL file you loaded already has a custom shadow model.

With the **Save MDL** button you can save the changes you made back to the MDL file again.

8.6 Animation tweaker

With the animation tweaker, see Figure 8.6 you can create an animated object for FS2004 that has more than 1024 frames. MakeMDL has a limitation of 1024 built in, which at 18 Hz gives you almost one minute of animation time. Sometimes this is not sufficient and you want a longer animation.

You need to make the animation from an FSX MDL, as XtoMDL does support longer animations. You select this FSX MDL in the **Object to import** field. Next you specify the name of the FS2004 MDL file in the **FS2004 model to export** field. Once you press the **Export** button the tool will create an FS2004 MDL with the long animation from your FSX model.

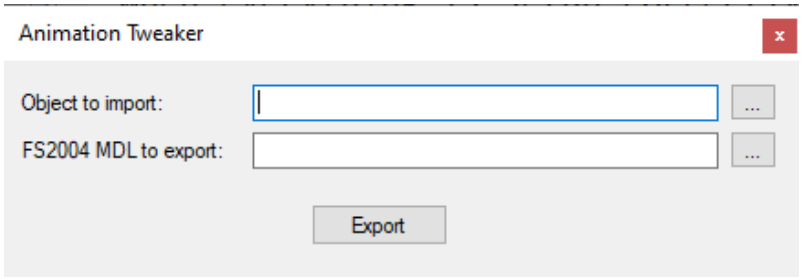


Figure 8.6: Animation tweaker

8.7 XML Apron to SurfaceType converter

With the XML Apron to SurfaceType converter, see Figure 8.7, you can make SCASM SurfaceType commands that ensure that your aprons are smooth and don't show dirt behind the wheels, based

on BGLComp aprons defined in a XML file. This technique can be used if you want to see the photoreal scenery below the apron.

After selecting the **XML file** to load and the **SCA file** that should be written, you can press the **Convert** button and the SurfaceType commands will be written to the SCA file you have selected.

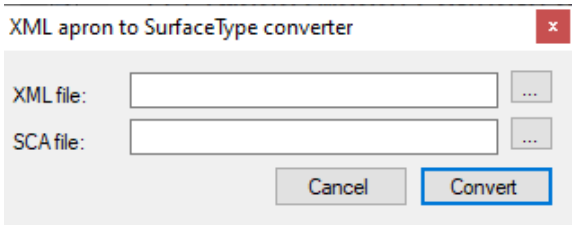


Figure 8.7: XML Apron to SurfaceType converter

8.8 Placement to SHP converter

With the placement to SHP converter, see Figure 8.8, you can create a SHP file for the position of each object in a scenery. This can be useful if you want to process the object placement information in a GIS tool or import it into another system that uses SHP files.

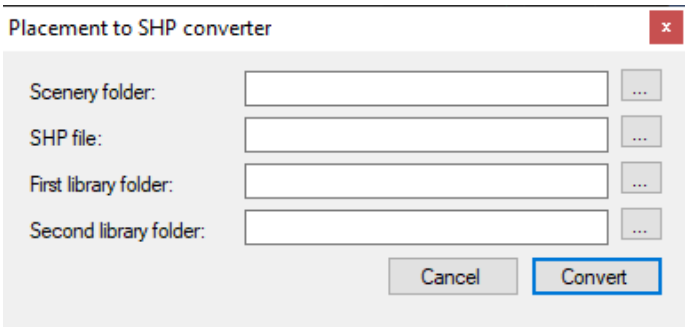


Figure 8.8: Placement to SHP converter

You need to specify the scenery folder of the scenery that you want to convert, the location where the SHP file will be written and optionally you can provide two additional folders where the tool will look for needed library BGL files. When you press **Convert** the SHP file will be written.

What the tool does is that it checks all BGL files in your scenery folder for object placement information. For each placement a point feature in the SHP file will be written, with attributes for the heading, scale, etc. The tool will also try to find the name of the placed objects, for this it will scan all BGL files in the scenery and the two library folders you selected. When the object is found in any of these files, an attribute with the name will be written.

8.9 SCASM Macro to XML placement converter

With the SCASM Macro to XML placement converter, see Figure 8.9, you can create a BGLComp XML file with object placement based on a SCASM file that contains Macro commands.

You need to specify the **SCA file** you want to read, the **XML file** that should be created. You also need to specify the **MDL folder** so that the object GUID information can be read from those files, the assumption is that the SCASM macro and MDL file have the same name. Lastly you need to specify the **API folder**, this is the folder where the SCASM macros are stored. This folder is used to find the long filename, in case the SCA file is using the short 8.3 character filename.

When you press the **Convert** button the XML placement file will be generated.

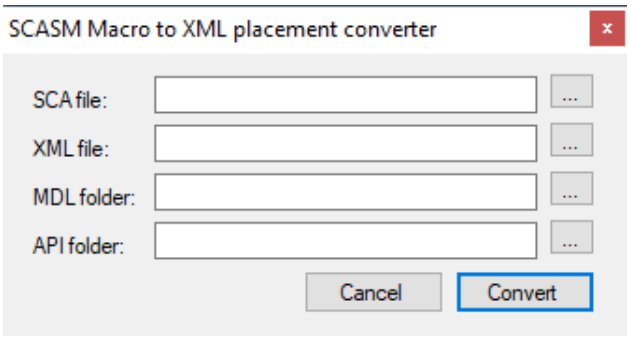


Figure 8.9: SCASM Macro to XML Placement converter

8.10 Replace double sided material by triangles

This function duplicates the triangles for all materials that have the double sided option enabled. It can be useful to have the double sided as geometry, if the system you are exporting to does not support this material attribute. Another advantage of creating them as geometry is that the lighting of the triangles improves.

8.11 Import names from XML

With this function you can load objects names from a XML file. When a FS2004 library BGL is loaded the objects have no names. With this function you can load the BGLComp XML file that was used to generate the library BGL. The filenames of the MDL files stored in the XML will then be used as names for the objects.

8.12 Export ground polygons to SHP

This function exports all ground polygons of the currently selected object to an ESRI Shapefile. This can be useful if you want to use the ground polygons in a GIS program, for example to overlay them on the photo scenery of the airport. This functions works best with ground polygons that are flat and haven't been corrected yet for the curve of the earth in FSX or Prepar3D.

8.13 Burn material colors into textures

When importing models from certain modeling tools it might be that a material has both a texture and a colour assigned. Some systems mix those two on rendering, while others don't do that. If your target system does not, you can use this function to mix the colour and the texture and store the result as a new texture.

8.14 Filter out ground polygons

This function removes all ground polygons from your object. Ground polygons are polygons that are level to the ground and within a certain distance. Sometimes old objects include ground polygons, like parking lots. In FSX scenery it is often more realistic to use photo scenery instead, so these have to be removed from the object.

8.15 Generate footprint object

When this option is selected a footprint of the currently selected object will be generated. This footprint projects all triangles of the object on the ground plane. The footprint object is then made the currently selected object.

8.16 Normals & Shading

8.16.1 Flip all triangles

When this option is selected all triangles in the object are flipped. This means they will be facing the other direction.

8.16.2 Flip triangles with inverse normals

When this option is selected all triangles where the normal of the triangle is inversed compared to the normals of the vertices are flipped. Having the triangle and normal vertices not the same can result in the triangle showing up dark in the object.

8.16.3 Flat shade object

When this option is selected all triangles in the object will be flat shaded. This means that the edges between different triangles will look hard, as the normal vector for the vertices on the edge is calculated to be equal to the normal of the triangle they are part of. See Figure 8.10 for an example of a flat shaded object.

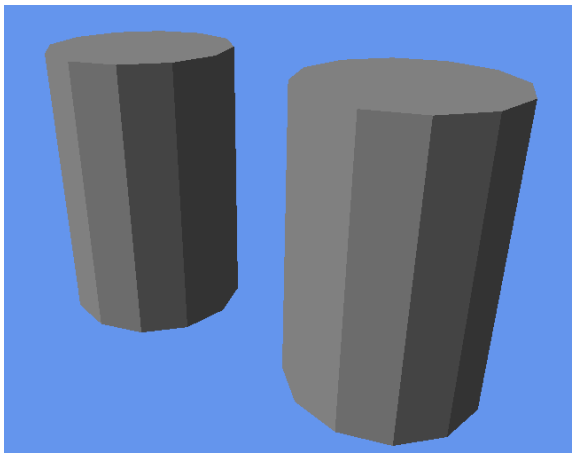


Figure 8.10: Flat shaded object

8.16.4 Smooth shade object

When this option is selected all triangles in the object for which the angle between them is less than the smooth shading threshold angle specified in the options will be smooth shaded. This means that the edges between different triangles will look smooth, as the normal vector for the vertices on the edge is calculated to be the average of the triangles that connect there. See Figure 8.11 for an example of a smooth shaded object.

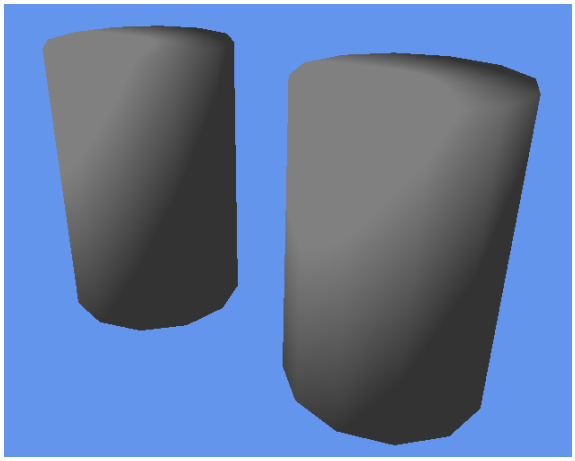


Figure 8.11: Smooth shaded object

8.17 Optimize scenegraph

This menu option will run the optimization of the scenegraph on your object. This optimization is normally run when you import an object. But with this option you can manually trigger it as well after you made changes to the object.

Chapter 9

Readers

ModelConverterX contains various readers to be able to load objects, sceneries and textures from different formats. In this chapter these various readers are discussed.

9.1 Scenery and object readers

This section describes the details and limitations of the various object readers. See Table 9.1 for an overview of the supported features of the different readers.

9.1.1 AC3D

The AC3D reader is used to read objects from the format of the AC3D modelling tool. This format is for example used by the FlightGear simulator. ModelConverterX can read the basic geometry and materials from this format.

9.1.2 Assimp

ModelConverterX makes use of the Assimp library to read certain 3D formats. Geometry, material settings and animations can be read using the Assimp library. It should be noted that not all formats in the Assimp library support all these functionalities, e.g. some formats don't support animations for example. The following formats are read by ModelConverterX using Assimp:

- AutoDesk FBX
- COLLADA (DAE)
- Wavefront OBJ
- Stereolithography (STL)
- AutoCad DXF
- Blender project files

9.1.3 BGL

The BGL Format is the compiled scenery format used by Flight Simulator and Prepar3D. ModelConverterX can read scenery objects and their placement, excludes and effects from the BGL format. The various versions of the BGL format are supported (e.g. FS2002, FS2004, FSX, P3D v4, etc). When 3D objects are read from FSX and P3D BGL files, ModelConverterX uses the MDL reader for that. It should be noted that ModelConverterX can't read airport or terrain BGL files.

Format	Object mesh	Basic materials	Advanced materials	PBR materials	LOD	Mouse rectangle	Visibility conditions	Animation	Skin and bone animation	Vertex color	Attached objects	Friendly name and GUID	Object placement	Special effects	Excludes
AC3D	✓	✓			✓										
Assimp	✓	✓			✓			✓		✓					
- FBX	✓	✓			✓			✓		✓					
- COLLADA	✓	✓			✓			✓		✓					
- STL	✓	✓			✓										
- DXF	✓	✓			✓										
- Blender	✓	✓			✓			✓							
BGL	✓	✓	✓	✓	✓		✓	✓	✓		✓	✓	✓	✓	✓
- Pre-FS2004 BGL	✓	✓			✓			✓			✓	✓	✓	✓	✓
- BGLComp BGL	✓	✓	✓	✓	✓		✓	✓	✓		✓	✓	✓	✓	✓
CFG	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓		✓	
DSF	✓	✓			✓								✓		✓
FLT	✓	✓			✓										
FSC	✓	✓			✓			✓							
glTF	✓			✓			✓	✓		✓	✓	✓			
KMZ	✓	✓											✓		
MDL	✓	✓		✓	✓	✓	✓	✓	✓		✓	✓		✓	
- Pre-FS2004 MDL	✓	✓			✓			✓							
- FS2004 MDL	✓	✓			✓	✓	✓	✓			✓			✓	
- FSX MDL	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓		✓	
- P3D v2 MDL	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓		✓	
- P3D v4 MDL	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓		✓	
- P3D v4.4 MDL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	
SCASM	✓	✓			✓			✓				✓	✓	✓	
Wavefront OBJ	✓	✓			✓										
X	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓		✓	
- FS2004 X	✓	✓			✓			✓							
- FSX X	✓	✓	✓		✓	✓	✓	✓	✓		✓			✓	
- P3D v2 X	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓		✓	
- P3D v4 X	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓		✓	
- P3D v4.4 X	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓		✓	
XML (BGLComp)	✓	✓		✓	✓		✓	✓	✓		✓	✓	✓	✓	✓
- FS2004 XML	✓	✓			✓			✓			✓	✓	✓	✓	✓
- FSX/P3D XML	✓	✓	✓	✓	✓		✓	✓	✓		✓	✓	✓	✓	✓
XML (MSFS model)	✓			✓	✓		✓	✓			✓	✓			
X-Plane OBJ	✓	✓			✓										
3DS	✓	✓			✓			✓							
3MF	✓	✓								✓					

Table 9.1: Object and scenery reader capabilities

9.1.4 CFG

The CFG reader can read information from aircraft.cfg files, for example points for lights, center of gravity and such. These points of the aircraft can be shown in the preview and edited with the Aircraft.cfg editor (see section 6.9).

9.1.5 DSF

The DSF format is used by the X-Plane simulator for sceneries. ModelConverterX can read objects and their placement from the DSF format. To read the actual object information the X-Plane OBJ reader is used.

9.1.6 FLT

The OpenFlight format is commonly used by professional flight simulators. ModelConverterX can read 3D objects from the OpenFlight format, including geometry, materials and light points. ModelConverterX does not support the concept of switches or degree of freedom nodes.

9.1.7 FSC

The FSC format is used by the Abacus FSDS modelling tool. ModelConverterX can read geometry, materials and animations from this format. Since support for this format is not complete, it might sometimes be easier to export the model to MDL format from FSDS and then import that file into ModelConverterX.

9.1.8 glTF

The glTF format is used by MSFS. ModelConverterX can import geometry, material settings and animation from the glTF format.

9.1.9 KMZ

ModelConverterX can read objects and their placement from KMZ files that are typically used by Google Earth. The 3D objects inside the KMZ file are stored in the COLLADA format, so the Assimp reader is used to read those.

9.1.10 MDL

The MDL format is the compiled format for aircraft and scenery models of Flight Simulator and Prepar3D. ModelConverterX can read MDL files from the following FS versions:

- Pre-FS2004 aircraft MDL files
- FS2004 (aircraft and scenery)
- FSX (aircraft and scenery)
- P3D v2 (aircraft and scenery)
- P3D v4 (aircraft and scenery)
- P3D v4.4 (aircraft and scenery)

Of these the FSX and P3D formats are supported almost completely. For the older versions of the MDL formats certain features might be less complete or more buggy.

Reading of vertex colors is only supported from P3D v6 MDL files.

Internally ModelConverterX uses the FSX/P3D animation names, in the animation mapping configuration file it is configured how FS2004 animation names are translated to teh FSX/P3D animation names. See section 13.3 for more details about this mapping.

9.1.11 SCASM

The SCASM format is the input of the SCASM compiler that can make BGL files. With this reader you can read SCA source files, but also API and SCM macros. These formats were mainly used before FS2004. Older versions of the BGL format are also decompiled into SCASM code first, before they are read by ModelConverterX.

Internally ModelConverterX uses the FSX/P3D animation names, in the animation mapping configuration file it is configured how FS2004 animation names are translated to teh FSX/P3D animation names. See section 13.3 for more details about this mapping.

9.1.12 Wavefront OBJ

Wavefront OBJ is a relatively simple format that is supported by many modeling tools. ModelConverterX can read geometry and materials from this format.

9.1.13 X

DirectX X files are used as input into the XtoMDL and MakeMDL compilers used to make MDL files. ModelConverterX can also read those X files directly.

9.1.14 XML (BGLComp)

The BGLComp compiler that makes the BGL files for FS2004, FSX and P3D uses XML source files that specify the placement of objects and which MDL files to include. ModelConverterX can also directly read those XML source files.

9.1.15 XML (MSFS model)

MSFS uses a model XML file to specify the GUID, LODs and the behaviour of the model. ModelConverterX can read this file and will then import all levels of details of the model at once.

9.1.16 X-Plane OBJ

X-Plane uses OBJ files for its scenery and aircraft models. ModelConverterX does not support all features of X-Plane, but it can read the geometry and materials from the OBJ files.

Internally ModelConverterX uses the FSX/P3D animation names, in the animation mapping configuration file it is configured how X-Plane OBJ animation names are translated to teh FSX/P3D animation names. See section 13.3 for more details about this mapping.

9.1.17 3DS

The 3DS file format is the original format of the 3DS Max modeling tool. Many formats support this format, therefore it is common to use. However the format is also quite dated by now, e.g. having restrictions on the length of texture names from the DOS ages. ModelConverterX can read geometry, materials and animations from this format.

9.1.18 3MF

The 3D Manufacturing Format (3MF) is a format used by CAD applications to exchange models. ModelConverterX can read geometry, materials and vertex colors from this format.

9.2 Texture readers

To display the textures of your object in the preview window, ModelConverterX contains readers for different texture formats. The following texture formats can be read by ModelConverterX:

- BMP (8 bit, 16 bit, 24 bit, 32 bit)
- Extended BMP (DXT1, DXT3, DXT5)
- DDS (DXT1, DXT3, DXT5 and MSFS compressions)
- RAW/R8 format using the FS5 colour palette
- JPG
- PNG
- PSD
- SGI RGB/RGBA
- TGA
- TIF

Chapter 10

Writers

To export objects and sceneries from ModelConverterX you can use the different writers. There are two types of writers:

- **Object writers** that can save one single object and attached parts.
- **Scenery writers** that can save multiple objects, placement information, special effects and excludes.

Below the available writers are described in more detail.

10.1 Object writers

This section describes the details and limitations of the various object writers. See Table 10.1 for an overview of the supported features of the different writers.

10.1.1 AC3D

The AC3D format is used by the AC3D modelling tool and by FlightGear. ModelConverterX can export geometry and basic material settings to the AC3D format. Level of detail information is stored in the name of the parts.

10.1.2 AF2 TGI

The TGI format is used by AeroFly FS2 for 3D models. ModelConverterX can export geometry and basic material settings to the TGI format. Only the highest level of detail of the object is exported to the TGI format.

10.1.3 Assimp

The Assimp writer can export to various formats using the Assimp library. ModelConverterX can export geometry and basic material settings using the Assimp writer. Level of detail information is stored in the node names. The following formats are supported by this writer:

- COLLADA
- FBX (binary)
- Wavefront OBJ

Format	Object mesh	Basic materials	Advanced materials	PBR materials	LOD	Mouse rectangle	Visibility conditions	Animation	Skin and bone animation	Vertex color	Attachpoints
AC3D	✓	✓			✓						
AF2 TGI	✓	✓									
Assimp	✓	✓			✓					✓	
- COLLADA	✓	✓			✓					✓	
- FBX	✓	✓			✓					✓	
- Wavefront OBJ	✓	✓			✓					✓	
FBX (ASCII)	✓	✓			✓						
FLT	✓	✓			✓						
FSDS	✓	✓			✓						
glTF	✓			✓	✓		✓*	✓*		✓	✓
MDL	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
- FS2004 MDL (aircraft)	✓	✓	✓		✓	✓	✓	✓			✓
- FS2004 MDL (scenery)	✓	✓	✓		✓			✓			✓
- FSX MDL	✓	✓	✓		✓	✓	✓	✓	✓		✓
- P3D v2 MDL	✓	✓	✓		✓	✓	✓	✓	✓		✓
- P3D v4 MDL	✓	✓	✓		✓	✓	✓	✓	✓		✓
- P3D v4.4 MDL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Wavefront OBJ (old)	✓	✓			✓					✓	
X	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
- FS2004 X	✓	✓	✓		✓			✓			✓
- FSX X	✓	✓	✓		✓	✓	✓	✓	✓		✓
- P3D v2 X	✓	✓	✓		✓	✓	✓	✓	✓		✓
- P3D v4 X	✓	✓	✓		✓	✓	✓	✓	✓		✓
- P3D v4.4 X	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
X-Plane OBJ	✓	✓			✓						
3DS	✓	✓			✓						

Table 10.1: Object writer feature overview (to features marked with a * restrictions apply)

10.1.4 FBX (ASCII)

The FBX (ASCII) writer is an alternative to writing FBX files with the Assimp writer. ModelConverterX can export geometry and basic material settings to the FBX (ASCII) format. Level of detail information is stored as part of the object name.

10.1.5 FLT

The OpenFlight (FLT) format is used in many professional flight simulator and is used by the Presagis Creator tool. ModelConverterX can export geometry, basic materials and levels of detail to the OpenFlight format.

10.1.6 FSC

The FSC format is used by the Abacus FSDS modeling tool. ModelConverterX can export geometry, basic material settings and level of detail information to the FSC format.

10.1.7 glTF

The glTF format is an open standard and is used by MSFS. ModelConverterX can export geometry, material settings and levels of detail to the glTF format. Exporting animations and visibility conditions is only supported when the model has been imported from an open format and not when imported from compiled FS files (like MDL or BGL). An exception to this is that simple (scenery) objects that have maximum 5 animations and only use ambient animations can be exported to glTF with animations.

10.1.8 MDL

The MDL format is used by Microsoft Flight Simulator and Prepar3D for the 3D objects. ModelConverterX uses the MakeMDL and XtoMDL tools from the SDK to export to the MDL format. ModelConverterX can export to the following MDL versions:

- FS2004 scenery MDL file
- FS2004 aircraft MDL file
- FSX MDL file
- P3D v2 MDL file
- P3D v4 MDL file
- P3D v4.4 MDL file

For FS2004 aircraft MDL files some restriction apply. The virtual cockpit and mouse rectangles are not yet supported and will not be exported. Attached effects and attached lights are mapped onto the light types that MakeMDL supports based on their effect and/or visibility conditions, see section 13.4 for more details. Other attached objects that can not be mapped, are not exported.

Writing of vertex colors is only supported from P3D v6 MDL files.

10.1.9 Wavefront OBJ (old)

The Wavefront OBJ (old) writer is an alternative to writing Wavefront OBJ files with the Assimp writer. ModelConverterX can export geometry and basic material settings to the Wavefront OBJ (old) format. Level of detail information is stored in the name of the parts.

10.1.10 X

The X file format is the input file for the MakeMDL and XtoMDL tools from the SDK. ModelConverterX can save the X file for the following FS versions:

- FS2004 X file
- FSX X file
- P3D v2 X file
- P3D v4 X file
- P3D v4.4 X file

10.1.11 X-Plane OBJ

X-Plane uses OBJ files for the 3D objects. ModelConverterX can export geometry, basic material settings and levels of details to the X-Plane OBJ format. Spot lights are also written to the OBJ file. X-Plane only allows one texture per OBJ file, this means that if your object uses multiple textures it will be exported as multiple OBJ files.

10.1.12 3DS

ModelConverterX can export geometry and basic material settings to the 3DS format. Level of detail information is stored in the name of the parts.

10.2 Scenery writers

This section describes the details and limitations of the various object writers. See Table 10.2 for an overview of the supported features.

10.2.1 BGL

The BGL format is used by Microsoft Flight Simulator and Prepar3D to store scenery. The BGL files that ModelConverterX writes can contain 3D models, object placement, effects, attached objects and excludes. ModelConverterX uses the BGLComp tool from the SDK to compile the BGL files. The following versions of the BGL format are supported:

- FS2004, including FS2004 MDL files as objects.
- FSX, including FSX MDL files as objects.
- P3D v2, including P3D v2 MDL files as objects.
- P3D v4, including P3D v4 MDL files as objects.
- P3D v4.4, including P3D v4.4 MDL files as objects.

10.2.2 BGL flatten

ModelConverterX can create a terrain flatten BGL file using shp2vec based on 3D objects. This can be used to make an accurate flatten based on a 3D mesh in a modeling tool. See Figure 10.1 for an example of a flatten made from a building shape.

10.2.3 BGL ground polygon

ModelConverterX can write FS2002 style ground polygon BGL files using the SCASM tool. The BGL files are used for custom ground polygons in FS2004, FSX and Prepar3D v1. The ground polygon writer supports night and seasonal textures.

Format	Object models	Object placements	Special effects	Exlcudes	Attached objects	Ground polygons	Flatten
BGL	✓	✓	✓	✓	✓		
- FS2004 BGL	✓	✓	✓	✓			
- FSX BGL	✓	✓	✓	✓	✓		
- P3D v2 BGL	✓	✓	✓	✓	✓	✓	
- P3D v4 BDL	✓	✓	✓	✓	✓	✓	
- P3D v4.4 BGL	✓	✓	✓	✓	✓	✓	
BGL flatten							✓
BGL ground polygon						✓	
DSF	✓	✓		✓			
MSFS scenery package	✓	✓					
SCA ground polygon						✓	
TSC	✓	✓					
TXT	✓						
XML	✓	✓	✓	✓	✓		
- FS2004 XML	✓	✓	✓	✓			
- FSX XML	✓	✓	✓	✓	✓		
- P3D v2 XML	✓	✓	✓	✓	✓		
- P3D v4 XML	✓	✓	✓	✓	✓		
- P3D v4.4 XML	✓	✓	✓	✓	✓		
- MSFS XML	✓	✓	✓	✓	✓		

Table 10.2: Scenery writer feature overview

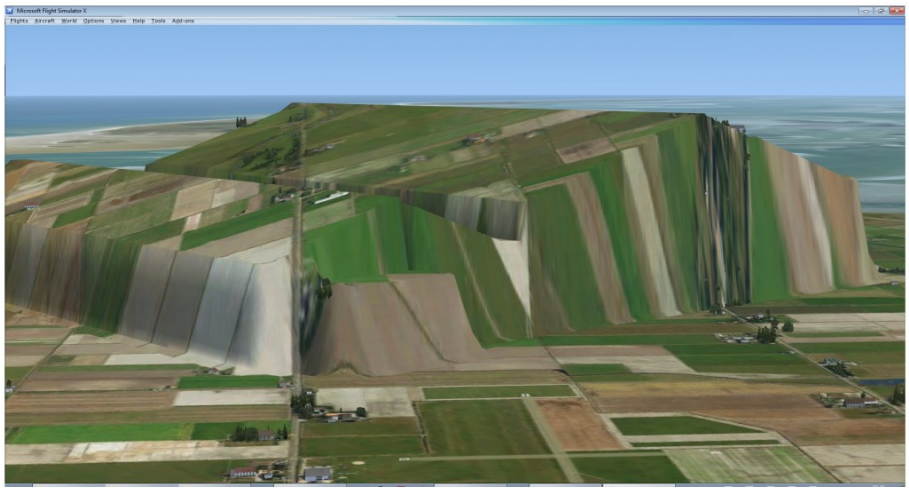


Figure 10.1: Flatten made from object mesh

10.2.4 DSF

DSF is the scenery format of X-Plane. ModelConverterX can export objects and their placement to the DSF format. The objects themselves are exported in the X-Plane OBJ format together with the DSF file.

10.2.5 MSFS scenery package

This writer creates a simple scenery package folder structure for MSFS, with a modellib for any objects and a scene folder for the placement of them. This writer should not be used to write into an existing package, as it will overwrite existing files and also not remove files that are not needed anymore. So use this writer only as a quick start to get a package for MSFS.

10.2.6 SCA ground polygon

This writer creates the SCASM source file used for FS2002 style ground polygons.

10.2.7 TSC

TSC is the scenery format of AeroFly FS2. ModelConverterX can export objects and their placement to the TSC format. The content converter tool from AF2 is used to create the TSC file.

10.2.8 TXT

The TXT format is used to list the GUID and names of objects in a scenery. Some tools use these TXT files to know which objects are contained in a library BGL file.

10.2.9 XML

The XML format creates the XML files that are used by BGLComp to create scenery BGL files. ModelConverterX can export the following versions of the BGLComp XML format:

- FS2004.
- FSX (also used for Prepar3D).
- MSFS (used by the MSFS package tool instead of BGLComp).

10.3 Texture writers

ModelConverterX can write textures to various formats, this allows you to convert objects and their textures to different applications. Below the supported formats are described.

10.3.1 Generic

ModelConverterX uses the .NET functionality to write textures to the following formats:

- BMP
- JPG
- PNG
- TIF

10.3.2 DXT compressed

ModelConverterX can save textures as DDS and DXT BMP files. These texture formats are DXT compressed, which means that they take less space and can also be efficiently processed by the graphics cards. These formats support mip maps. ModelConverterX automatically selects the best

compression format (DXT1, DXT3 or DXT5) based on the preferred FS version that has been set in the options and whether the texture has an alpha channel.

10.3.3 RGB

The RGB format is typically used in combination with the OpenFlight format. ModelConverterX can write textures to this format. This format uses no mip maps.

Chapter 11

Options

In the options window you can set all the options of ModelConverterX. There are different categories of options for the preview renderer, importers, exporters, etc. In the list on the left you can select which category of options you want to see and edit. The related options are then shown in the property list on the right side of the window. In the sections below all the available options are discussed.

The **Reset to default** button in the toolbar at the top of the window resets the currently selected category of options to their default values. The **Reset all to default** button resets all options categories at once to their default values

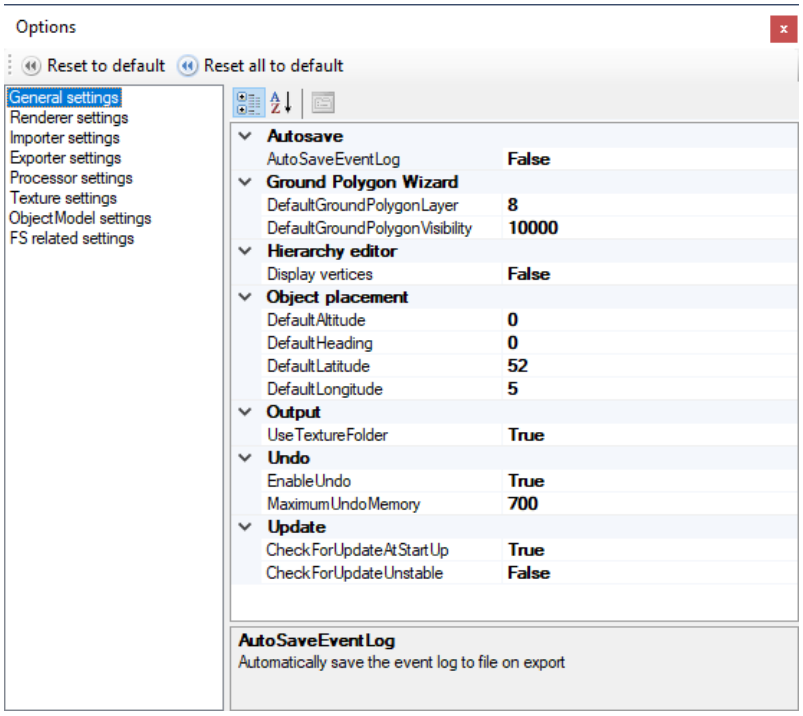


Figure 11.1: Options window

11.1 General settings

AutoSave

- **AutoSaveEventLog** Flag to determine if the event log should automatically be saved to a TXT file when you export an object.

Ground Polygon Wizard

- **DefaultGroundPolygonLayer** specifies the default layer value used for materials.
- **DefaultGroundPolygonVisibility** specifies the default visibility value used for materials.

Hierarchy Editor

- **Display vertices** specifies that information about all vertices of a modelpart is shown in the hierarchy editor.

Map

- **DefaultMapType** specifies which map provider is used by default when opening a map.

Object placement

- **DefaultAltitude** specifies the default altitude value used when creating a new object placement.
- **DefaultHeading** specifies the default heading value used when creating a new object placement.
- **DefaultLatitude** specifies the default latitude value used when creating a new object placement.
- **DefaultLongitude** specifies the default longitude value used when creating a new object placement.

Output

- **UseTextureFolder** specifies what the initial location of the texture folder in the material editor will be. When set to true the initial value is the last used texture folder, when set to false the object folder is used as initial value.

Undo

- **EnableUndo** specifies that the undo functionality is enabled. ModelConverterX will use more memory when it is enabled.
- **MaximumUndoMemory** specifies the maximum amount of memory in MB that can be used to store previous versions of your object. If this amount of memory is passed, teh oldest changes will be deleted and cannot be undone anymore.

Update

- **CheckForUpdateAtStartUp** specifies if ModelConverterX should check if a newer version is available at startup. When a newer version is found you will be notified with a message, see Figure 11.2. When you click on that message you are taken to the download page of the development release. An internet connection is required to check for updated versions.
- **CheckForUpdateUnstable** specifies if ModelConverterX should notify you of updates for the unstable development releases as well. When set to false only notifications for updates of the stable releases are given.

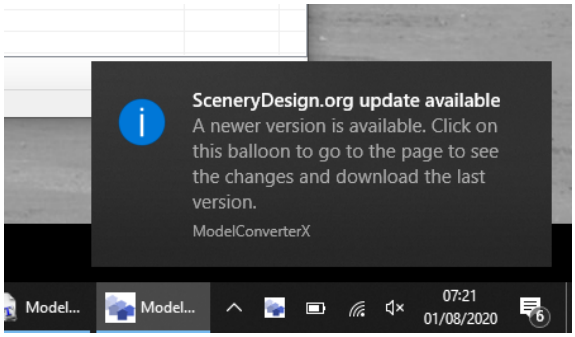


Figure 11.2: Update notification

11.2 Renderer settings

Colors

- **AttachPointColor** specifies the color used to render the attachpoint location.
- **BackgroundColor** specifies the color of the preview background at day.
- **BackgroundColorNight** specifies the color of the preview background at night.
- **BoneColor** specifies the color used to render the bones.
- **CFGCameraDefinitionColor** specifies the color used to render CFG camera definition locations.
- **CFGContactPointColor** specifies the color used to render CFG contact point locations.
- **CFGEngineColor** specifies the color used to render CFG engine locations.
- **CFGExitColor** specifies the color used to render CFG exit locations.
- **CFGFuelTankColor** specifies the color used to render CFG fuel tank locations.
- **CFGLightColor** specifies the color used to render CFG light locations.
- **CFGRotorColor** specifies the color used to render CFG rotor locations.
- **CFGSmokeSystemColor** specifies the color used to render CFG smoke system locations.
- **CFGStationLoadColor** specifies the color used to render CFG station load locations.
- **CrashBoxColor** specifies the color used to render the crashbox.
- **GridColor** specifies the color used to render the grid lines.
- **GridOriginColor** specifies the color used to render the grid lines that pass through the origin of the grid.
- **GroundColor** specifies the color used to render the ground plane.
- **NormalColor** specifies the color used to render the normal vectors.
- **SelectedAttachpointColor** specifies the color used to render the currently selected attachpoint location.
- **ShadowColor** specifies the color used to render the shadow.

General

- **PreviewEnabled** specifies if the preview image is enabled. Disabling the preview can reduce memory usage of ModelConverterX.

Grid

- **GridSize** specifies the total size of the grid in meters. For example if you specify 200 here and you have a step size of 10, you will see a grid with 20 squares.
- **GridStep** specifies the distance between the lines in the grid in meters.

Navigation

- **PanScale** scales the panning movement of the mouse. This allows you to tune how quickly you pan.
- **ZoomScale** scales the zooming of the mouse wheel. This allows you to tune how quickly you zoom in or out.

Particles

- **FadeParticles** specifies if particles of special effects are faded or not. Sometimes their behaviour can be studied better when fading is turned off.
- **PlaySounds** specifies if the sounds of special effects are played or not.

Rotation

- **DefaultRotationX** specifies the default rotation angle around the X axis. By changing this value you can modify the default orientation of your object in the preview.
- **DefaultRotationY** specifies the default rotation angle around the Y axis.

Shaders

- **ForceSimpleShader** specifies if the simple shader should be used, if set to false you can toggle between the complex and simple shader. On some older graphics cards the complex shader can't be compiled, so this can be used to force the simple shader.
- **ShowShaderWarnings** specifies if compilation warnings of the shader are shown in the event log.

Spot light

- **OnlyShowSelectedSpotLight** specifies if only the selected spotlight should be shown in the preview. When you have many spotlights this can make it easier to only see the one you are working on.
- **SpotLightStep** specifies the step in degrees between the lines that are drawn for the spotlight cone.

Wireframe

- **WireframeLineWidth** specifies the line width of the lines in wireframe rendering mode.
- **WireframeLineWidthHighlight** specifies the line with of the selected part in wireframe rendering mode. By making the lines of the selected part thicker, they are easier to spot.

11.3 Importer settings

BGLXReader

- **Import while saving MDL files** determines if the objects are still imported in ModelConverterX when the option Save MDL files is enabled. If set to false the objects are only saved to disk and not loaded in ModelConverterX.

- **Save MDL files** determines if the objects in a library BGL are saved to disk as MDL files while reading the library BGL. This can be used if you want to examine specific objects in more detail.

DSFReader

- **Library list** specifies a list of library.txt files that are used to find library objects.

FltReader

- **Read DOF as animation** specifies if animations should be created from DOF nodes. When set to false the current position of the DOF is used.

General

- **Animation scaling** specifies the scaling that will be applied to the animation frame values on import. This can be used if the source format uses a different scaling than FS. If you set the scaling to 10 an keyframe value of 10 will be imported as frame 100. Not all readers support this settting.
- **Debug mode** specifies if the debug mode is enabled. When debug mode is enabled additional information messages will be put in the event log. Not all importers support this function.
- **Load untextured** specifies if the object should be read without textures or not. If you don't need the textures this can speed up the loading time.
- **Replace effects by lights** specifies if ModelConverterX should replace special effects by lights on import. If set to true ModelConverterX will make light and spotlight attach points when effects for them are found. When they are imported as lights or spot lights it is easier to modify their settings. This function only works if you use the default ModelConverterX special effect names in your object.
- **Up axis** specifies which axis is used as the upwards axis of the model. Not all importers support this setting.

glTFReader

- **Empty Node as Attachpoint** specifies that empty nodes that have no child nodes or mesh should be read as an attachpoint.
- **Rotate 180 degrees** specifies that the object should automatically be rotated 180 degrees along the Z axis on import.

KMZReader

- **Use filename** specifies if the filename of the KMZ file is used as the object name. When set to false the name of the KML file that is stored inside the KMZ is used.

ObjReader

- **Read double sided** specifies if all faces of the OBJ file should be read as double sided.

SCASMReader

- **Apply default scale** specifies if the default scale as specified in a comment in API macro objects should be applied when importing the object.
- **Create empty LOD** specifies if an empty level of detail should be added to the object on import.

- **Empty LOD value** specifies the value used when automatically adding an empty level of detail.
- **Line extrude radius** specifies the radius of the circle that is used when creating a tube from lines. In FSX and Prepar3D it is not possible to draw lines anymore, so these are turned into a 3D tube object of triangles.
- **Line extrude vertex count** specifies the number of vertices that are used for the circle when creating a tube from lines. Keep this number low to minimize the number of triangles that are added to your object.
- **Replace user variables** specifies if user variables used in API macros are replaced by user specified values or not. When set to false a default value of zero is used. When set to true a window is displayed as shown in Figure 11.3 where you can enter the desired values.

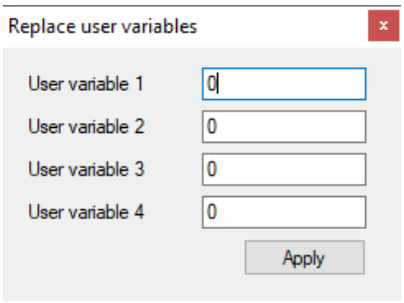


Figure 11.3: window to specify user variables of API macros

- **Use conditions** specifies how conditions in the SCASM and BGL code should be processed during importing. The following values are allowed:
  - **AlwaysTrue** means that any conditional check is considered to be true.
  - **AlwaysFalse** means that any conditional check is considered to be false.
  - **DefaultValues** uses the default values of all variables when checking the condition.
  - **UserSpecified** uses user specified values for the variables when checking the conditions. A window as shown in Figure 11.4 will be shown during import so that you can enter the values.
  - **VisibilityCondition** creates visibility conditions from the conditions.

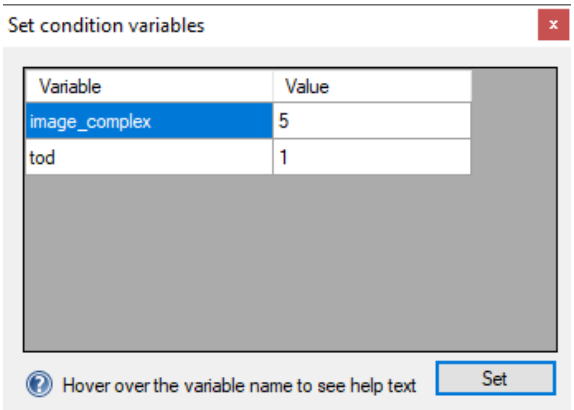


Figure 11.4: window to specify variables for conditions

XReader

- **Collapse gamepack matrices** specifies whether two specific scaling matrices that the gamepack adds to the X file should be collapsed into one matrix. The gamepack adds one matrix with scale 1024 and another with scale 0.000977, but due to rounding issues these do not result in an exactly identical matrix and leaving them can result in small offsets for big objects.
- **Parse frame name** specifies if the frame names should be parsed. If set to true frames with MR_ will be turned into mouse rectangles and frames with VT_ will be turned into visibility conditions.

11.4 Exporter settings

AC3DWriter

- **AC3D axes** specifies which directions are used for the XYZ axes when exporting to the AC3D format.

AeroFly

- **AeroFlyContentConverterPath** specifies the path to the content converter tool for AeroFly FS2, this converter is used to generate the AF2 specific formats.

BGLWriter

- **Create TXT list** specifies if a TXT file with the names of the library objects should automatically be created on export of a BGL file.
- **FS2004 BGLComp path** specifies the path to the FS2004 version of BGLComp.
- **FSX BGLComp path** specifies the path to the FSX version of BGLComp.
- **KeepMDL** specifies if the MDL files should be kept after exporting a BGL file.
- **KeepXML** specifies if the BGLComp XML file should be kept after exporting a BGL file.
- **P3D v2 BGLComp path** specifies the path to the Prepar3D v2 or v3 version of BGLComp.
- **P3D v4 BGLComp path** specifies the path to the Prepar3D v4 version of BGLComp.
- **P3D v4.4 BGLComp path** specifies the path to the Prepar3D v4.4 or v5 version of BGLComp.

BglGpWriter

- **AbsoluteReferencePoint** specifies if an absolute reference point is used in the ground polygon SCASM file. When set to false a relative reference point is used.
- **KeepSCA** specifies if the SCA file should be kept after compiling a FS2002 style ground polygon BGL file.

BGLXFlatWriter

- **shp2vecPath** specifies the path to the shp2vec tool from the FSX or Prepar3D SDK.

FLTWriter

- **IgnoreSpecularColor** specifies if the specular color of the material should be ignored. When set to true a black specular color is used instead. When set to false the specular color from the material is used.

- **LODMultiplier** specifies the multiplier that is used to calculate the LOD distance based on the size of the bounding box.
- **TexturedPolygonsWhite** specifies if textured polygons should use a white diffuse color. When set to false the diffuse color from the material is used for textured polygons as well.
- **WriteATTR** specifies if ATTR files are written for the textures of the object.

FSCWriter

- **SetSmoothing** specifies if smooth shaded polygons are exported as a separate part with smooth shading applied.

General

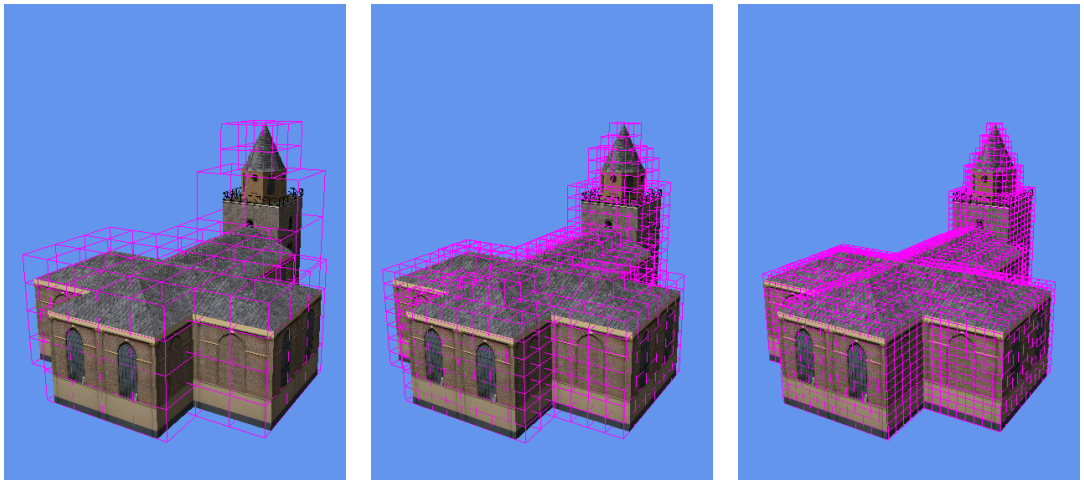
- **Animation scaling** specifies the scaling that will be applied to the animation frame values on export. This can be used if the target format or application uses a different scaling than FS. If you set the scaling to 0.1 a keyframe value of 100 will be exported as frame 10. Not all writers support this settting.
- **Author company** specifies the name of the company of the author. Some formats support storing the author company within the object.
- **Author name** specifies your name as author. Some formats support storing the author name within the object.
- **Debug mode** specifies if debug mode is enabled. In debug mode the writers provide additional debug messages in the event log. Not all writers support the debug mode.
- **Export highest LOD** specifies if only the highest level of detail should be exported. When set to false all levels of detail are exported.
- **Multiple representation preference** specifies how objects with multiple representations (e.g. external and internal model) are exported by writers that do not support writing multiple representations. The preference can either be for the active representation, the external representation or the internal representation, or each representation can be exported as a separate file. See section 13.6 for more details about the representations.
- **OnlyWriteTextureName** specifies if only the texture names should be written to the object. When set to false the (relative) path is also written. Not all writes support this setting.

glTFWriter

- **Rotate 180 degrees** specifies that the object should automatically be rotated 180 degrees along the Z axis on export.
- **Use MSFS extensions** determines if the glTF file is written using the MSFS extensions or not. To store all MSFS (material) settings you need to use the extensions. But for applications that can't handle them you can disable them.
- **Write model XML** specifies if a MSFS model XML should be written together with the glTF file. This model XML contains the GUID and LOD definitions.

MDLWriter

- **CrashboxGranularity** specifies the granularity used for creating the crashbox. The lower the value the more accurate the crashbox, but also the bigger your MDL will be. See Figure 11.5 for an example of the impact of different values. Only used when SetCrashboxGranularity is true.
- **FSX XtoMDL path** specifies the path to the FSX version of XtoMDL.



(a) (b) (c)

Figure 11.5: Example of different crashbox granularities for an object.(a) default value of XtoMDL (b) value of 2 (c) value of 1

- **Keep** specifies if XtoMDL should save an XML file with the object definitions.
- **KeepASMFile** specifies if the ASM files are kept when exporting a MDL file with MakeMDL.
- **KeepXFile** specifies if the X file should be kept after exporting a MDL file with MakeMDL or XtoMDL.
- **MakeMDLPath** specifies the path to the FS2004 MakeMDL tool.
- **NoCrash** specifies that no crashbox should be generated when exporting a MDL file.
- **Part vertex limit** specifies the maximum number of vertices allowed per model part. When there are more vertices the part is split into multiple parts. Lowering this limit can help to prevent XtoMDL vertex list has too many vertices error.
- **P3D v2 XtoMDL path** specifies the path to the Prepar3D v2 or v3 version of XtoMDL.
- **P3D v4 XtoMDL path** specifies the path to the Prepar3D v4 version of XtoMDL.
- **P3D v4.4 XtoMDL path** specifies the path to the Prepar3D v4.4 or v5 version of XtoMDL.
- **PreserveGUIDOnExport** specifies if the GUID should be preserved when overwriting an existing MDL file. When set to true the GUID will be taken from the MDL file that you are overwriting.
- **SampleXML** specifies if a sample placement XML file should be saved with the object.
- **SetCrashboxGranularity** specifies if the CrashboxGranularity setting should be used. When set to false XtoMDL will use its default granularity.

SceneryPackageWriter

- **CompilePackage** specifies if the package should be automatically compiled using the FSPackageTool tool after saving it.
- **FSPackageTool path** specifies the path to the MSFS FSPackageTool tool.
- **OverwriteExistingTextures** specifies if existing textures should be overwritten when writing the package textures.
- **PackageTextureFormat** specifies the format used for saving the textures of the package.

- **WritePackageTextures** specifies if the textures of the object should be written to the package as well.

X-Plane

- **Point light name** specifies the name of the named light that is written for point lights.
- **Write light color** specifies if for point lights the color should be written after the position.

XWriter

- **DrawcallBatching** specifies if drawcall batching should be enabled when exporting an XML file. This can help improve performance, see section 13.5 for more background information.
- **DrawcallBatchingWorkingLOD** specifies if levels of detail should be retained when drawcall batching is enabled.
- **OnlyWriteTextureName** specifies if only the texture names should be written to the X file. When set to false the (relative) path is also written.
- **Use controller effect** specifies that a controller effect file should be used when writing light effect files.
- **Use season material script** specifies if a material script is used for seasonal textures in Prepar3D MDL files. When set to false a visibility condition will be used instead.
- **ZBiasOffset** specifies the offset that will be given to polygons when exporting to FSX MDL files based on their ZBias value.

11.5 Processor settings

Color

- **EmissiveTextureWhiteColor** specifies if the emissive color of materials should be set to white for materials that have an emissive texture assigned. It only affects object readers that have this processing step enabled for them.

Texture

- **BumpTextureSuffix** specifies the suffix added to the diffuse texture name when adding bump textures.
- **DetailTextureSuffix** specifies the suffix added to the diffuse texture name when adding detail textures.
- **FresnelTextureSuffix** specifies the suffix added to the diffuse texture name when adding fresnel textures.
- **MaterialTemplateOverwriteTexture** specifies if material templates should overwrite existing textures in the material or not. When set to false a texture filename is only modified when it does not yet exist. When set to true it is always updated.
- **MetallicTextureSuffix** specifies the suffix added to the diffuse texture name when adding metallic textures.
- **NightTextureSuffix** specifies the suffix added to the diffuse texture name when adding night textures.
- **SpecularTextureSuffix** specifies the suffix added to the diffuse texture name when adding specular textures.

Texture Converter

- **TextureConverterRemoveDuplicateExtensions** defines if the texture converter should remove duplicate extensions when converting textures. When set to true MSFS textures like texture.PNG.DDS will be stripped of both extensions before saving to a different format.

11.6 Texture settings

TextureLoader

- **BMPDDSEquivalence** specifies if BMP and DDS files should be considered equivalent when looking for textures. This is the default behaviour of FSX and Prepar3D. So if the model references a DDS file, but the texture folder contains a BMP file with the same name it will be used instead.
- **TextureOrigin** specifies the location of the texture origin when reading textures (not all formats use this setting). For DDS files FSX/P3D use a top origin, and MSFS/X-Plane use a bottom origin.
- **TextureSearchDebugOutput** specifies if the texture search logic will write debug messages to the event log. These can be useful to debug issues when textures are not found.
- **TextureSearchPath** specifies the search path that ModelConverterX uses when it looks for a texture. ModelConverterX will first look for a texture in the folder where the object is located, next in a texture subfolder of the scenery or aircraft, and finally it will check all folders specified in the texture search path. The first folder where the texture is found is used for loading it.

  Two special values can be used in the texture search path:

  - `[MCXTEMP]` expands to the temp folder where ModelConverterX extracts zipped formats, like KMZ files.
  - `[FSPATH]` expands to the path of the preferred FS version. You can specify this preferred version in the FS related settings of the options.

  When editing this setting the texture search path editor, as shown in Figure 11.6, will be displayed. With the **New entry** button you can add a new entry to the search path where you can type your desired path. With the **Add path** button you can select a folder that you want to add to the path. The **Remove path** button removes the currently selected path from the search path. The **Edit path** button toggles the edit mode, so that you can make changes to the currently selected path. With the **Up** and **Down** arrow buttons you can move the priority of the currently selected path in the search path, the higher in the list the more priority it has. When you press the **Cancel** button any changes you made are discarded, while if you press the **OK** button they are stored in the settings.

TextureWriter

- **AddMipMap** specifies if mip maps are added when saving DDS or DXT BMP textures.
- **BlackWhiteAlphaAsDxt1** determines if a 1 bit (black or white) alpha channel is written as DXT1 or DXT 3 or 5. When set to true the alpha information is stored in DXT1, which can reduce the quality of the image but saves disk space.
- **CompressTexture** determines if textures are compressed or not when writing DDS and DXT BMP file. When set to true a DXT compression is applied, which reduces the size of the texture files.
- **DefaultType** specifies the default texture type selected for exporting in the material editor.
- **IncludeWhiteAlpha** determines if a fully white alpha channel is also saved to a DDS or DXT BMP file. When set to false a 24 bit or DXT1 image without alpha channel is made.

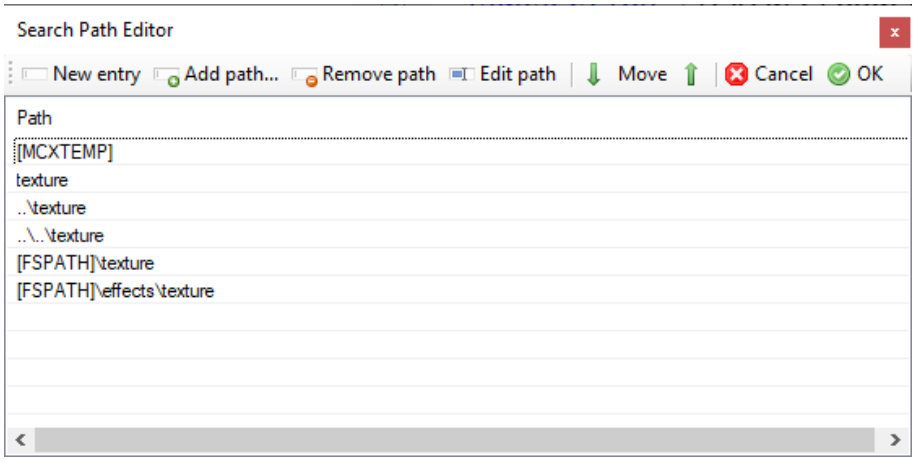


Figure 11.6: Texture search path editor

- **TextureOrigin** specifies the location of the texture origin when writing textures (not all formats use this setting). For DDS files FSX/P3D use a top origin, and MSFS/X-Plane use a bottom origin.

Transparency

- **AlphaDetectionThreshold** specifies the threshold for detecting an alpha channel. Values below this value are not considered transparent. This setting is used to prevent incorrect reading of compressed formats like JPG or PNG.

11.7 ObjectModel settings

Level of detail

- **DefaultLOD** specifies the default level of detail value.
- **LODFOV** specifies the horizontal field of view in degrees that is used for calculations between viewing distances and level of detail values.
- **LODResolution** specifies the horizontal screen resolution in pixels used for calculations between viewing distances and level of detail values.

Material

- **UseMaterialName** determines if the material name of a material is used for the display name. When the material name is not used, the display name is derived from the color and texture of the material.

Optimize

- **CollapseModelParts** specifies whether modelparts with the same material should be combined into one modelpart. They are also moved as far upwards in the scenegraph as possible.
- **DisableOptimize** specifies that the scenegraph optimization is disabled, this is not advised in general but can be useful if you want to see the original structure of an imported file.
- **RemoveMirrorTransform** specifies whether transformations that have a mirror effect on the object should be removed during optimization. When exporting animations to glTF this option should be true.

- **SmallTriangleLimit** specifies the area in square meters of a small triangle. Triangles with an area less than this value are removed during optimization of the object.

Placement

- **DefaultImageComplexity** specifies the default image complexity used when placing an object.

Season

- **DefaultSeason** specifies the default season settings that are used for the object. Clicking on the ... button opens the season editor to change the season definitions, see 6.11.

Shading

- **SmoothShadingThresholdAngle** specifies the threshold angle that is applied when smooth shading a model part. Only vertices for which the angle between the normals of the different triangles is below the specified threshold angle are are smooth shaded.
- **SplitSmoothPolygonsThreshold** specifies the threshold that is applied when determining if a polygon is smooth shaded or not. This is used when splitting smooth shaded polygons from a modelpart. When the dot product of the vertex normal and polygon normal is below this threshold it is considered smooth shaded and split.

Texture

- **AutumnTextureSuffix** specifies the suffix added to the diffuse texture name when adding autumn season textures.
- **HardWinterTextureSuffix** specifies the suffix added to the diffuse texture name when adding hard winter season textures.
- **SpringTextureSuffix** specifies the suffix added to the diffuse texture name when adding spring season textures.
- **SummerTextureSuffix** specifies the suffix added to the diffuse texture name when adding summer season textures.
- **WinterTextureSuffix** specifies the suffix added to the diffuse texture name when adding winter season textures.

11.8 FS related settings

FS

- **FS2004Path** specifies the path where FS2004 is installed.
- **FSXPath** specifies the path where FSX is installed.
- **MakeMdlPartsPath** specifies the path of the FS2004 MakeMDL.parts.xml file that ModelConverterX uses to read FS2004 specific animation, visibility condition and mouse rectangle definitions. When you are importing FS2004 aircraft models setting this paths will give better results.
- **ModelDefPath** specifies the path of the ModelDef.xml file that ModelConverterX uses to read animation, visibility condition and mouse rectangle definitions.
- **MSFSPath** specifies the path where the MSFS packages are installed. This should point to the folder containing the Official and Community sub-folders.
- **P3DPath** specifies the path where Prepar3D v1 is installed.

- **P3Dv2Path** specifies the path where Prepar3D v2 is installed.
- **P3Dv3Path** specifies the path where Prepar3D v3 is installed.
- **P3Dv4Path** specifies the path where Prepar3D v4 is installed.
- **P3Dv5Path** specifies the path where Prepar3D v5 is installed.
- **P3Dv6Path** specifies the path where Prepar3D v6 is installed.
- **PreferredFSVersion** specifies the preferred FS version as used by ModelConverterX. This preferred version influences from where textures and effect files are read, but it also influences the format in which GUIDs are displayed.

Chapter 12

Command line

ModelConverterX is primarily a graphical tool, which means that you control which object is loaded, how it is shown and how it is exported from the graphical user interface. But it is also possible to provide command line arguments at startup to ModelConverterX that influence this behaviour. You can influence which object is loaded into the graphical user interface and you can perform a conversion from the command line. But options are described in this chapter.

If you provide the path to a file as argument to ModelConverterX, this file will be loaded into ModelConverterX directly after starting the tool. The example below loads the file `myfile.mdl` into ModelConverterX.

```
ModelConverterX.exe c:\path\to\myfile.mdl
```

It is also possible to select which livery of the object should be shown from the command line, that way the preview shows that livery directly after loading the object. This is done with the `-livery` option as shown in the example below. In this case `livery3` is selected.

```
ModelConverterX.exe c:\path\to\myfile.mdl -livery livery3
```

The two examples before show how command line arguments can be used to control how ModelConverterX starts up. But sometimes you might want to integrate the conversion capabilities of ModelConverterX in your own workflow and not use the graphical user interface. That is also possible and the rest of this chapter shows the arguments that can be used for that. With the command line conversion the graphical users interface is not shown at all and all output goes to the command prompt window, see Figure 12.1. The usage of the command line mode of ModelConverterX is printed to the command prompt when you use the `-help` argument.

```
ModelConverterX.exe -help
```

The most simple command line conversion is done by loading an object as the first argument and specifying the output format you want with the `-format` option. Possible format values are shown in the usage. This will convert the object `myfile.api` to a FSX MDL file, it is exported as `myfile.mdl` in the same folder.

```
ModelConverterX.exe c:\path\to\myfile.api -format MDL_X
```

It is also possible to write the converted file to a different filename, in that case you need to provide the path you want with the `-out` argument.

```
ModelConverterX.exe c:\path\to\myfile.api -out c:\another\path\new.mdl -format MDL_X
```

If you want to convert multiple objects, you can also use a wildcard in the input to select all of them. The example below shows how all API files starting with an `a` are converted to FSX MDL files. When using a wildcard in the input, the filename of the `-out` argument is ignored and only the directory to export to is taken from the argument.

```
ModelConverterX.exe c:\path\to\a*.api-format MDL_X
```

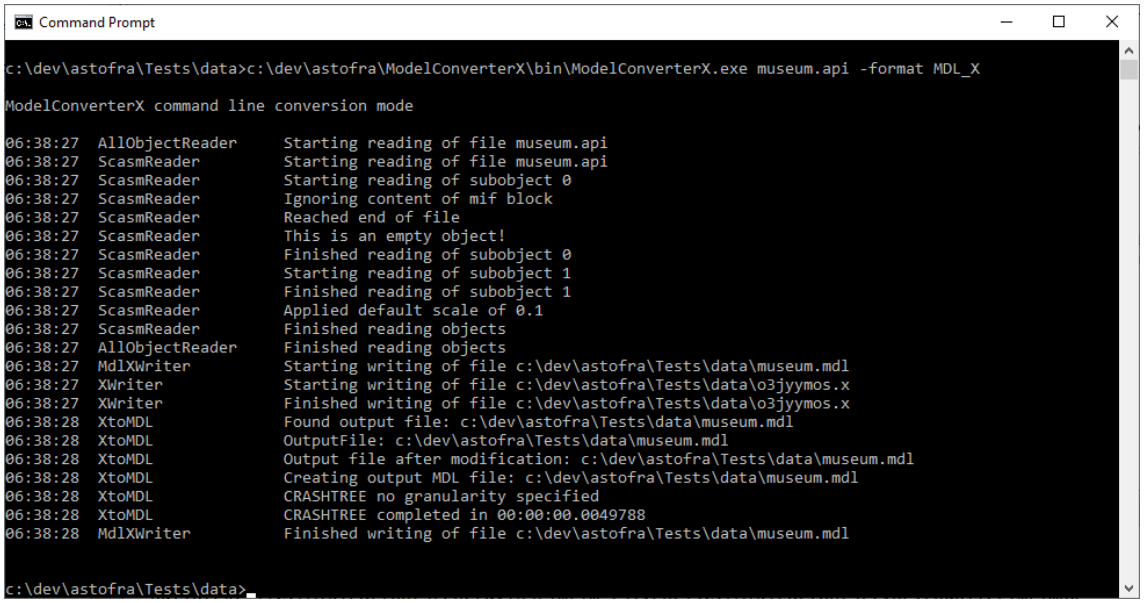


Figure 12.1: Command line conversion output

If you want to apply certain modifications to the object before exporting it, that can be done in the command line mode as well. You can apply batch operations on the object, similar to the Batch Convert Wizard, see section 7.3. The batch convert configuration you can have saved from the wizard can be applied in command line mode as well. Only the batch operators are used in that case, any input or output settings are ignored (these are driven by the command line arguments). The example below shows how batch settings are applied with the `-batch` argument.

```
ModelConverterX.exe c:\path\to\myfile.api -batch c:\path\to\batch.mbc -format MDL_X
```

It is also possible to only apply the batch operations, without exporting the object afterwards. This can be useful if you use the batch operator to convert textures or to generate a screenshot. In that case you don't provide a `-format` argument, see the example below:

```
ModelConverterX.exe c:\path\to\myfile.api -batch c:\path\to\batch.mbc
```

You can also merge two objects from the command prompt. The command below merges object2.mdl into object1.mdl and then saves the result as a FSX MDL file named object1.mdl.

```
ModelConverterX.exe c:\path\to\object1.mdl -merge c:\path\to\object2.mdl -format MDL_X
```

It is also possible to specify as which level of detail the object should be merged, in the example below object2.mdl is merged as LOD 40. When no LOD is specified the object is merged as the highest LOD of the primary object.

```
ModelConverterX.exe c:\path\to\object1.mdl -merge c:\path\to\object2.mdl -mergeLod 40 -format MDL_
```

Chapter 13

Background info

This section provides some interesting background information that might be useful to you while using ModelConverterX.

13.1 Position control

Multiple editors and wizards require you to specify a position, therefore these all use a common control that allows you to specify and save these positions. This section explains this control, which is shown in Figure 13.1.

In the latitude, longitude, altitude, heading and scale fields you can specify the details of the position that you want to use. To prevent you having to enter this information multiple times, you can save it using the button with the earth icon and the plus. This will add the position to a list of saved positions with the name you give it. The next time you want to use the same position you can just select it from the dropdown list and all details of the position will be filled in automatically.

If you want to remove a position from the list, you first select it in the list and then you press the button with the earth icon and the minus to remove it.

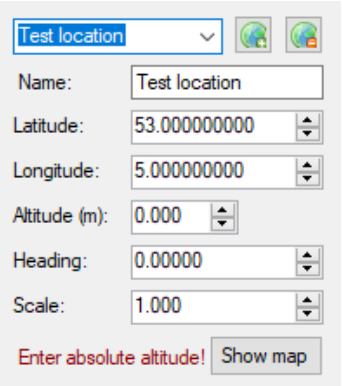


Figure 13.1: Position control

When you press the **Map** button a map window opens where you can see your position on a map, see Figure 13.2. You can select different map providers for the map image using the dropdown list at the top left. With the search box at the top right you can search for a named position, for example a city name.

Dragging with the left mouse button will move the currently selected position. The coordinates are automatically updated in the position control. Dragging with the right mouse button will pan

the map itself. With the mouse wheel you can zoom the map in and out.

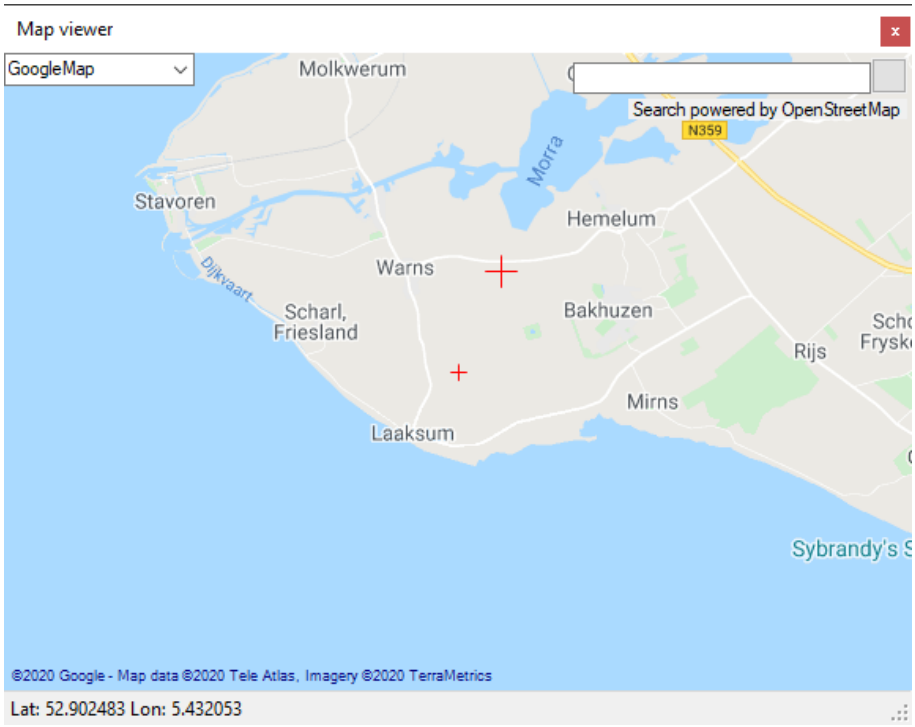


Figure 13.2: Map view of position

13.2 Rotation order

When you want to rotate an object around its axes there are different orders in which you can do this. In the **Rotate editor**, the **Transformation editor** or the **Animation editor** you can specify the order you want. But what do all those choices mean?

The first thing you can choose is in which order the different axes are rotated. For example you can first rotate along the Z axis, then along the Y axis and then along the X axis. That will obviously give a different result then when you would first rotate along the X axis, then along the Y axis and finally along the Z axis. So in the selection list you will be able to choose from the different orders in which the axes are rotated.

But you will also see that you have two choices for each order the axes are rotated, one being intrinsic and the other being extrinsic. What is the difference between them? Simply said an intrinsic rotation means that the axis are modified when you rotate the object. So for example when you first rotate along the Z axis, this means that the orientation of the X and Y axes also change. See Figure 13.3 for a graphical representation of this.

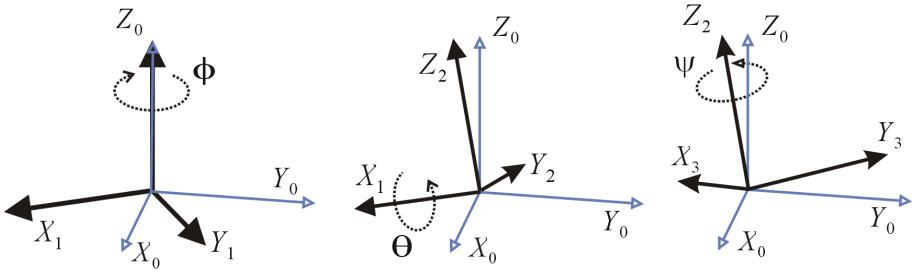


Figure 13.3: Example of intrinsic rotation

On the other hand extrinsic rotations means that the axes do not change. So if you rotate along the Z axis first, this does not affect the orientation of the other axis. See Figure 13.4 for a graphical representation of this.

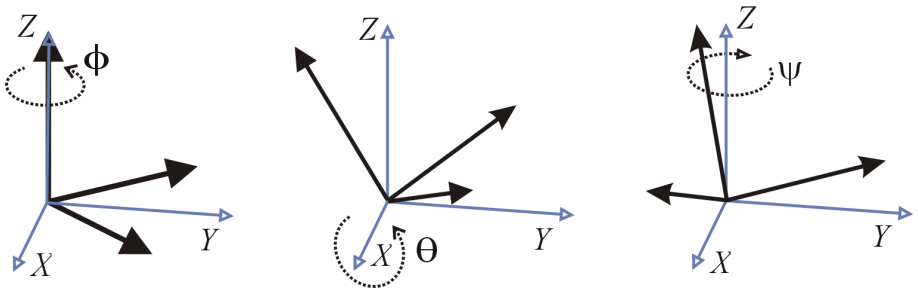


Figure 13.4: Example of extrinsic rotation

13.3 Animation mapping

Internally ModelConverterX uses the animation names as defined in the modeldef.xml file of FSX or Prepar3D. But some file formats use their own animation names, for example FS2004 and X-Plane both use different animation names. In the animation_mapping.ini file, that you can find in the ModelConverterX folder, it is defined how these animation names should be translated to the names as defined in the modeldef.xml file. This mapping has been made configurable, so that as an user you can modify or extend it yourself, while converting your models. The configuration is just a plain text INI file that you can edit with any text editor.

The INI file contains different sections, each of which defines the mapping between the animation names of a specific format or simulator and the modeldef.xml animation names. The following sections are supported:

- [FS2004_FSX]
- [FSX_FS2004]
- [XPLANE_FSX]

Each section define a mapping between the names used by the formats mentioned in the name. The first name is the name in the source format and the second name (seperated by a space or tab) is the name in the destination format. Below you see the partial mapping from FS2004 animation names to FSX animation names.

```
[FS2004_FSX]
c_tire c_tire_blurred_key
elevator elevator_percent_key
engine0 prop0_still
l_flap_key l_flap_percent_key
```

This mapping defines that the FS2004 name `c_tire` should be replaced by the modeldef.xml name `c_tire_blurred_key`. Similarly the FS2004 name `engine0` is replaced by the modeldef.xml name `prop0_still`.

There are also three special sections that deal with FS2004 animations.

- The [FS2004_FSX_REVERSED] section is a list of animation names that should be reversed on import into ModelConverterX.
- The [FS2004_AXIS_CHANGE] section lists all animations for which the axis as defined in the MDL file should be changed to the specified axis. This is used to correct animations that are otherwise distorted.

- The [FS2004_AUTOMATIC] section lists all animations that are automatically generated by MakeMDL on export to a FS2004 object. These animations don't have to be included in the MakeMDL.parts.xml file and the X file also does not need keyframes for them. The section defines the name of the animation and the axis along which the automatic animation is generated. ModelConverterX will ensure that the object is exported with the correct axes for the automatic animation to work. The axis is defined by a normal vector, where the elements are separated by a colon. Below is partial list of the automatic animation definitions.

```
[FS2004_AUTOMATIC]
prop0_still 0,-1,0
prop0_slow 0,-1,0
prop0_blurred 0,-1,0
c_tire 1,0,0
```

If you have suggestions of animation mappings that should be added to the default mapping file that ships with ModelConverterX, please post them in the support forum so that I can include them in future releases.

13.4 Light mapping

The light_mapping.ini file specifies how lights are mapped between different versions of Flight Simulator, at the moment conversion between FS2004 and FSX/Prepar3D is support. FSX uses effects to represent lights, while FS2004 uses fixed BGL lights. The mapping determines how these are translated. The internal representation of ModelConverterX follows FSX, so the lights are mapped when you read a FS2004 MDL file and are mapped again in reverse direction when you export to a FS2004 MDL file again.

The following light types are supported in FS2004:

- Beacon
- Landing
- Logo
- Nav
- Reco
- Strobe
- Taxi
- Wing

The FS2004_FSX section of the INI file specifies the FS2004 light type in the first column and the effect that is used for FSX in the second column. When a light type is not listed in the INI file the light will be inserted as an attached light in the model and on export to FSX this will automatically become a new light FX file that ModelConverterX generates.

The FSX_FS2004 section of the INI file specifies the mapping from FSX effect to BGL lights. The first column is the FX file (without the FX extension). The second column specifies the FS2004 light type and color to be used. This is done using the format Type_R_G_B_A. For example the line below specifies that the fx_beacon effect should become a red BGL light of the type Beacon.

```
fx_beacon    Beacon_255_0_0_255
```

The VISCOND_CAT section of the INI file specifies how attached lights are mapped onto a FS2004 BGL light based on their visibility condition. The first column lists the visibility condition name from the modeldef.xml file and the second column specifies the FS2004 light type.

If you have suggestions of lights mappings that should be added to the default mapping file that ships with ModelConverterX, please post them in the support forum so that I can include them in future releases.

13.5 Drawcalls

One of the most important aspects for the performance of your object in the simulator is the amount of drawcalls that it has. So if you can minimize the number of drawcalls that will improve the performance a lot. In the **Material editor** there is a function for this. But what is a drawcall actually?

When the simulator needs to render your object on screen, it will first have to set the rendering state to the correct values. This for example means that a specific texture is activated or that settings affect the display of transparency are configured, etc. Once all these settings have been set up, the rendering engine will start to draw all triangles that use this material. This is called one drawcall, as all these triangles can be rendered in one call with the same settings. Switching to a different rendering state is a relatively expensive operation. So this means that every time you use a different texture, a different colour or some other material attribute that is slightly different, the rendering state has to be updated.

This means that to minimize the amount of drawcalls you need to use exactly the same material on your object. As soon as you change one attribute, you will get a different material and thus a different drawcall. With the dozens of attributes that an FSX material offers, it is quite easy to make a new drawcall, even though you might use the same texture.

So what can you do to minimize the amount of drawcalls? If you are using multiple texture sheets, it might help to put all of them on one bigger sheet instead. Or if you are using untextured materials with a colour, it might be easier to paint a corner of your texture that color and use that instead. Or if you have different material attributes set you could check if they really need to be different or not.

Figure 13.5 shows an object of a church as example. When I first designed this model it used about 10 different textures. The front, the side, the roof all had their own textures that I could easily map on the object. Later I combined all these textures into one sheet, see Figure 13.6 and this improved the performance of the object a lot. Originally I had part of the wall texture tiled, when I combined all the textures that was not possible anymore. But adding a few more polygons to get rid of the tiling was outweighed by the performance improvement of reducing the drawcalls.

Until now we have been talking about the performance improvements of reducing the number of drawcalls of a single model. But FSX and Prepar3D also include the concept of drawcall batching. This means that the rendering engine will combine the rendering of different instances of the same material into one single drawcall. So lets say that you use exactly the same material on two objects and that you placed 5 instances of each object. This means the rendering engine has to render 10 objects that use the same material. What drawcall batching does is render all triangles of these 10 objects in one go. So that means even more efficiency is gained.

There is one catch to this drawcall batching and that is that when enabled it will prevent the levels of detail, see section 13.7 from working. For relatively simple objects the drawcall batching will save more performance than adding levels of details, but for more complex objects it might be more useful to have the levels of detail working. So that is a choice you have to make as developer.

By default ModelConverterX will export all objects without levels of details in a such a way that their drawcalls can be batched. In the options you can specify if the level of detail or the drawcall batching has priority for objects that have levels of detail.

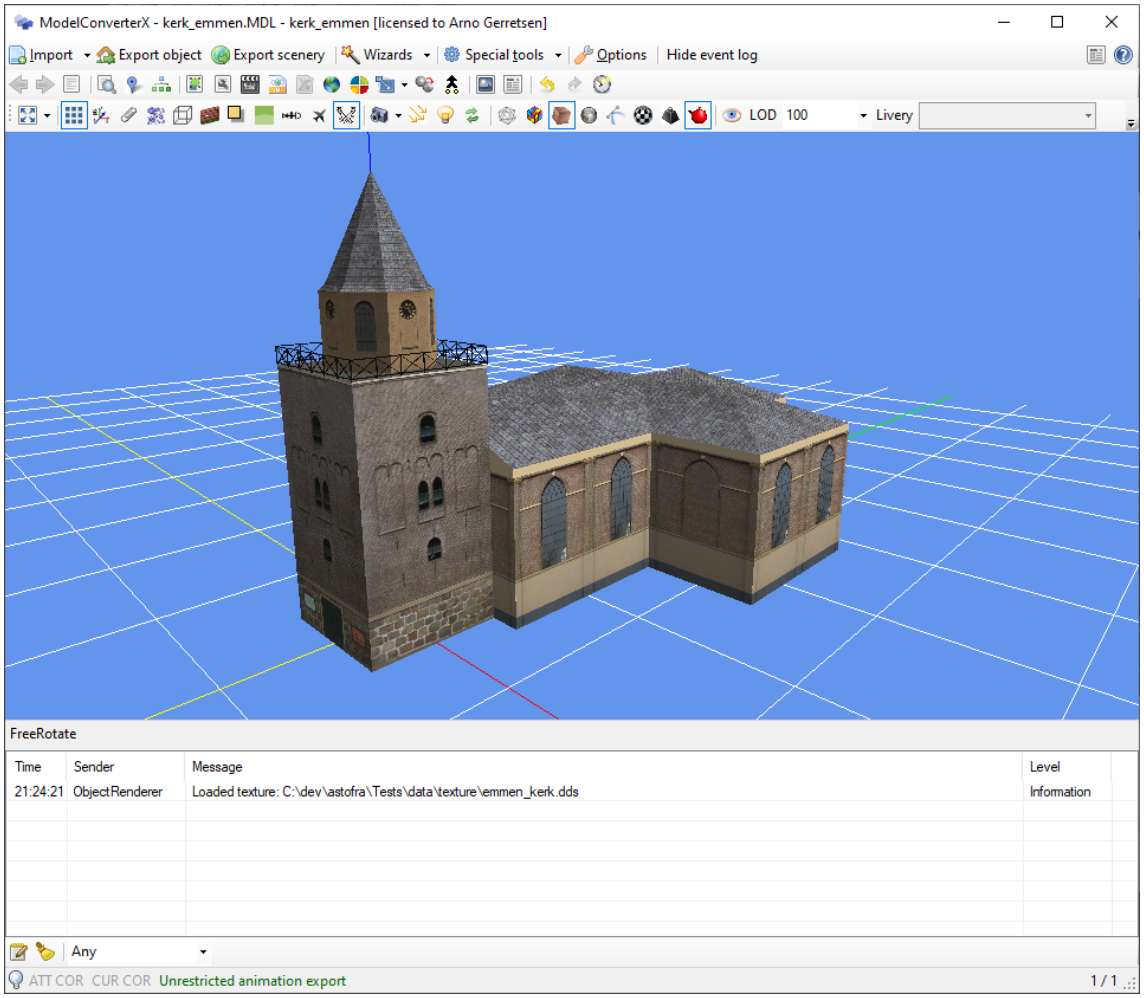


Figure 13.5: Model of a church

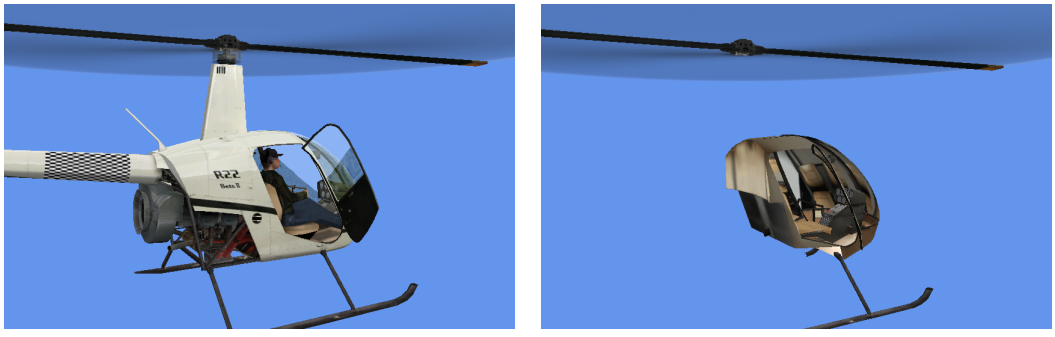


Figure 13.6: Example of a combined texture sheet

13.6 Representations

Within Flight Simulator different representations of an object can be used. For example aircraft models typically have an external representation and an internal representation that is used for the virtual cockpit. But it is also possible to have a different representation that is used to generate the shadow of the object. See Figure 13.7 and Figure 13.8 for two examples. Within ModelConverterX the following representations are supported:

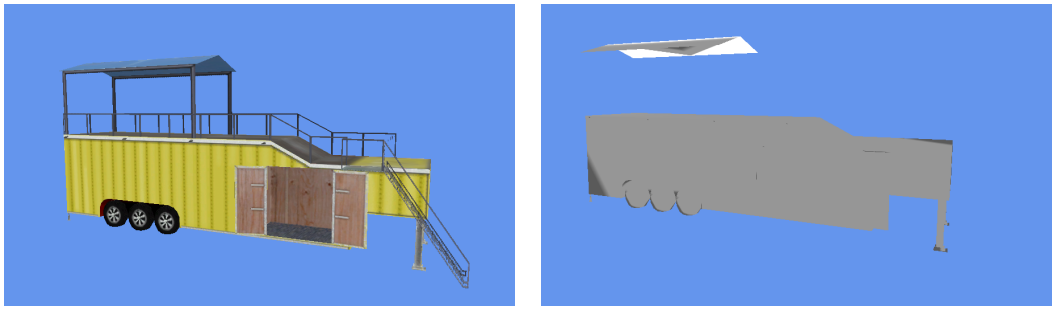
- External
- Internal
- Shadow



(a) (b)

Figure 13.7: Example of different representation. (a) External representation. (b) Internal representation.

It differs per version of Flight Simulator how these representations are stored. For example FS2004



(a) (b)

Figure 13.8: Example of different representation. (a) External representation. (b) Shadow representation.

has the external and internal representation in the same MDL file, while FSX uses two separate MDL files. ModelConverterX will read the representations from the different formats and combine them all in one object in the tool. If you load a FSX aircraft via the aircraft.cfg fil the external and internal representation are combined into one object.

On export it depends on the exporter how the representation are saved. Some exporters can write multiple representations to the same file, for example the FS2004 MDL export can do so. When an exporter only supports one representation the **Multiple representation preference** setting in the eporter settings section determines which representation is saved. You can either prefer the currently active representation, the external or the internal representation, or the different representations can be exported as separate files.

13.7 Level of detail

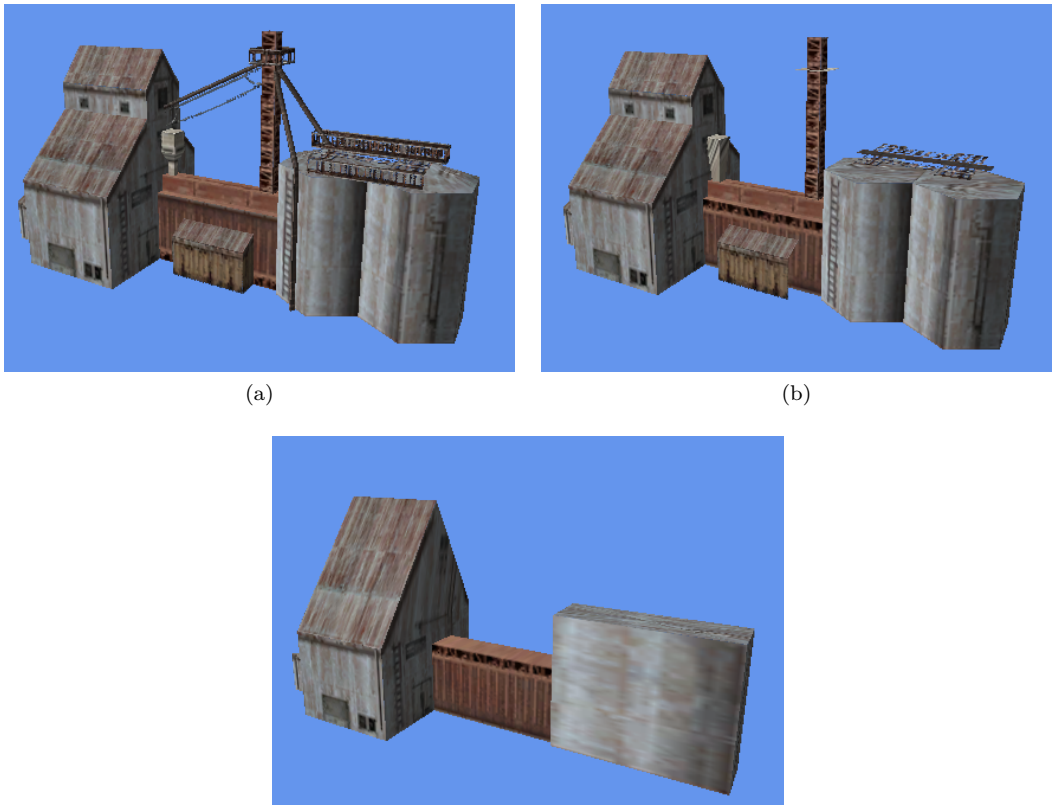
The concept of levels of detail is that you have different representations of your object, that vary in their complexity. And based on the how close you are to the object a different representation is shown. Figure 13.9 shows an example of the levels of detail in a default object in FSX. As you can see there are three representations that vary in complexity. When you are far away from the object the simplest representation is shown and as you get closer more detail is added.

But if the object switches to the more detailed representation too late, that will be a visual distraction. So therefore it is important that the LOD values are tuned correctly. But what do those values 100, 50 and 10 as used in the example above mean? In Flight Simulator you don't specify the distance at which the levels of detail should switch. And that is a good thing, as the contribution that the object makes visually is not only determined by the distance. Somebody running the simulator on a monitor with a higher resolution has more pixels available and will therefore be able to see the detail from further away.

The LOD values represent the size that the object has on screen. Level of detail 100 will be shown when the is object roughly 100 pixels wide(horizontally) on your screen, while the LOD 40 is shown when the object is around 40 pixels wide. So the higher the LOD value that you use, the later this version of the object will appear. Many objects use 100 as highest LOD, but you can also use higher values like 200 or 400 if your objects has small details that only need to show from very nearby.

Of course each level of detail that you add also makes your object bigger, as there are additional triangles and vertices that need to be stored. So you need to find a good balance there. A good rule of thumb is that each level of detail should contain around half of the triangle count of the next level of detail above it. So it probably doesn't make sense to add a level of detail when you only eliminate a handful of triangles.

If you want your object to disappear completely you can also add an empty level of detail. This



(a) (b)

(c)

Figure 13.9: Example of level of detail. (a) LOD 100 with 257 triangles. (b) LOD 50 with 121 triangles. (c) LOD 10 with 38 triangles.

will force the object to disappear before the normal distance that the simulator uses to display your object. If you have small objects that can't be seen from far away anyway, this can help to improve performance.

13.8 MSFS texture distortion

Models that have been processed by the MSFS packaging tool can show texture distortion in some cases. This section explains the technical background of why this happens.

Let's start with a little background information on how textures are mapped on a model. Each vertex has a set of texture coordinates (also called UV coordinates) that determine what position of the texture should be shown on that vertex. Figure 13.10 shows an example mapping where the bottom wall segment of the texture is mapped on a polygon. The resulting UV coordinates are shown as well. As you can see the whole width of the texture gives a U range from 0 to 1.

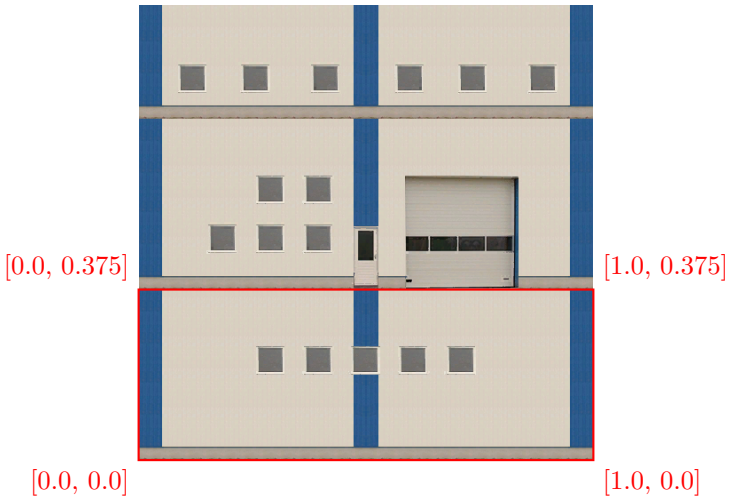


Figure 13.10: Example of texture mapping

If we would want to repeat the wall texture multiple times on a polygon that is also possible. We just extend the UV coordinates above 1 and then the texture gets tiled. Figure 13.11 shows an example where the wall segment is repeated three times on the polygon, the U range goes from 0.5 to 3.5. As you can see we also shifted the texture left a bit.

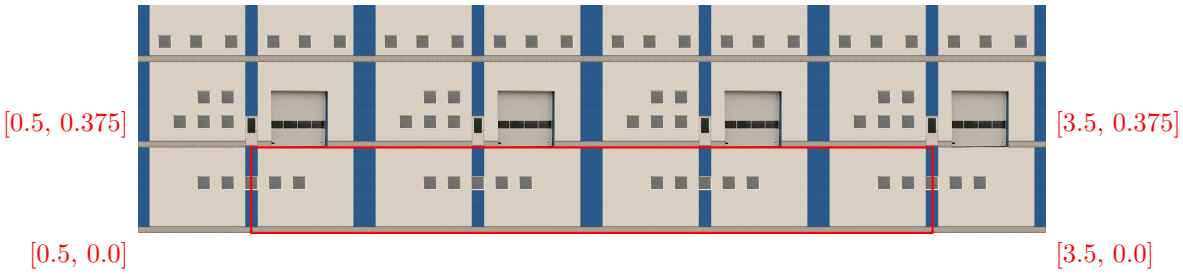


Figure 13.11: Example of tiled texture mapping

So far for the basics on texture mapping. These UV coordinates are stored in your model file and normally they are stored as single precision floating point numbers. But MSFS does optimize the models when you run the package tool and part of this optimization is that the texture coordinates are stored as half precision floating point numbers. These use less memory and storage space to store, but the downside is also that their accuracy depends on the number that is stored. And that causes the distortion you sometimes see. Table 13.1 shows the accuracy that the half precision floating point numbers have for different values (taken from Wikipedia).

Range	Accuracy
$\frac{1}{8}$ - $\frac{1}{4}$	$\frac{1}{8192}$
$\frac{1}{4}$ - $\frac{1}{2}$	$\frac{1}{4096}$
$\frac{1}{2}$ - 1	$\frac{1}{2048}$
1 - 2	$\frac{1}{1024}$
2 - 4	$\frac{1}{512}$
4 - 8	$\frac{1}{256}$
8 - 16	$\frac{1}{128}$
16 - 32	$\frac{1}{64}$
32 - 64	$\frac{1}{32}$
64 - 128	$\frac{1}{16}$
128 - 256	$\frac{1}{8}$
256 - 512	$\frac{1}{4}$

Table 13.1: Resolution of half precision floating points

These accuracy numbers might look very small to you, but let's remember that the UV range for the whole texture is from 0 to 1. If we have a texture of 2048 pixels width, that means that each pixel is $\frac{1}{2048}$ apart. So this already shows that if are UV values are higher than 1, the accuracy of the half precision floating point number is not enough anymore to map exactly onto a specific pixel! Instead those values will be modified due to the reduced accuracy and that is what causes the distortion you see.

Does that mean that you can't tile a texture at all? No, as long as you tile the texture an whole number of times, like 2 times, 4 times, 100 times, the accuracy will not give you distortion. Because even at 500 times tiles the accuracy of $\frac{1}{4}$ will still allow you to store the UV coordinates of 500 without distortion. But if you want to tile it 500.126 times, the UV coordinate will be rounded to 500.25 and you have some distortion.

You will notice the distortion most when you try to map a complex shape accurately and your UV coordinates are not in the range of 0 to 1. For example SketchUp has the habit to shift the entire mapping and that could cause distortion. See Figure 13.12 for an example texture mapping where the UV coordinates have been shifted. The red rectangles shows the intended mapping and the blue rectangle the resulting mapping after the MSFS optimizations have been applied. For this rectangular mapping it causes an offset, but you can imagine that if a more irregular shape is mapped the impact and distortion will be even more.

When the distortion is caused by such an offset, the solution is easy luckily, just shift the texture mapping back so that it is within the range of 0 to 1 again. In the hierarchy editor you can perform this action on a model part by using the normalize texture coordinates option in the context menu, see section 6.4.



Figure 13.12: Example of texture mapping with an offset

Chapter 14

Support

14.1 Support forum

If you have any problems while using the program or if you have suggestions and other feedback to improve the tool, please let me know. Any report about an object or a texture not loading correctly will help me to improve the tool, so please be sure to report those. You can either contact me directly or visit the ModelConverterX subforum at FSDeveloper.com.

14.2 Reporting crashes

If you experience a crash while running ModelConverterX you will get a dialog to report the crash as shown in Figure 14.1. Please send error reports using this dialog, as that will ensure that I get all the details of where the crash happened. That's much more efficient than telling me in the forum that something crashed.

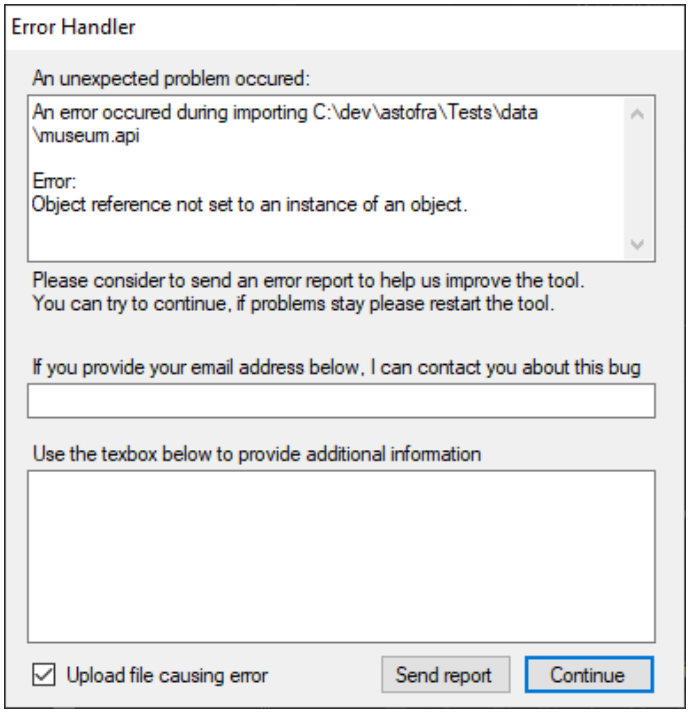


Figure 14.1: Error reporting dialog

Chapter 15

Credits

The following persons or organizations deserve credit for helping me while creating this tool:

- Everybody who reported feedback on the previous versions, this helped me to improve the tool. A special thanks goes to the beta testers who dared to try my development releases that were sometimes a little buggy.
- Michael Garland for his papers on object simplification
- The ASSIMP for the library to read and write various 3D file formats
- Manfred Moldenhauer for the SCASM compiler and the permission to redistribute SCASM with ModelConverterX.
- Tom Gibson for reviewing this manual.
- OpenTK for the C# OpenGL wrapper.
- Assimp for the library that supports many 3D model formats.
- Icons8.com for some of the icons used in the tool.

Chapter 16

User license

When using ModelConverterX you agree to the following user license:

The fact that ModelConverterX can read or convert a model, does not mean you are allowed to modify or distribute it. It is your responsibility to verify that you are allowed to use, modify and distribute the models as part of your addon.

In general you will need the permission of the author of a model before you are allowed to distribute a derived or converted version. This includes freeware models, which are still covered by the copyright of the author.

ModelConverterX is provided free of charge to flight simulator addon developers. You are not allowed to sell ModelConverterX or ask money for its distribution.

Content created with ModelConverterX can be used in commercial flight simulator addons.

The copyright and any intellectual property relating to this program remain the property of the author.

The software distributed in this way may represent work in progress, and bears no warranty, either expressed or implied.

©2007-2024 SceneryDesign.org / Arno Gerretsen