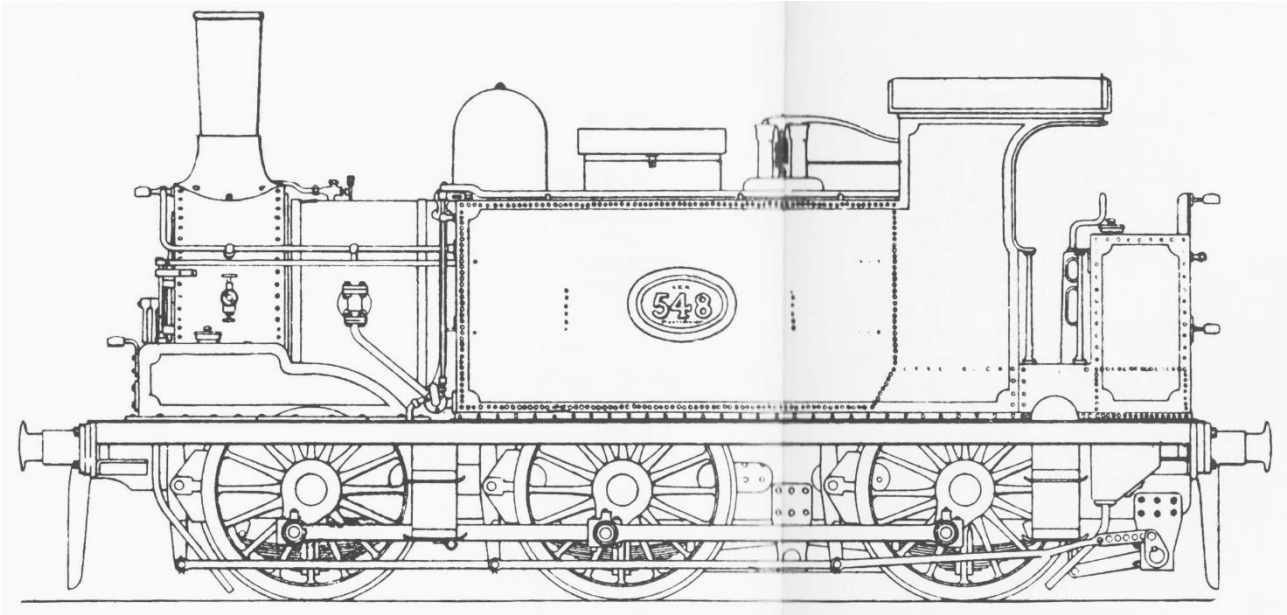
# A Blender Tutorial

*Building a Steam Loco for Trainz with Blender 3.0 or later*





## Contents

# 1. Introduction

## *Why Blender?*

Well, it's free, fast, and stable. It can also be used for very complex models – it's much more suitable than GMax for example when working with large and highly detailed models – and it includes everything you might need to make mapping and texturing easier in one package. It can render and 'bake' the results into the texture which means that highlights and shadows can be automatically generated, saving hours of work assuming you have the artistic ability to do this manually.

There also are numerous tutorials on the Internet covering every aspect of using the program and it is being actively developed.

Since I started to write these tutorials, it has become apparent that N3V are heading in the direction that Blender will be the free program of choice for creating assets for Trainz. GMax is no longer supported and cannot do the things that are needed for good looking efficient assets, especially for TRS19 and up. SketchUp may be at first glance easier to use but the models it creates are terribly inefficient unless you know what you are doing.

Interestingly it is claimed that Blender is the most popular 3D modelling program in the world with an estimated 4 million users and is also being used by the big Hollywood animation studios including Pixar.

You can get Blender here: http://www.blender.org. This tutorial assumes you will be using version 2.93 or later, this can be used without an installer so you can simply copy the whole Blender application anywhere you want without installing it, for example onto a USB stick. It's about 140Mb in size and very fast so it works well from a stick. However, I'd recommend that you use the installer and install it like any other Windows program, but in c:/Blender to avoid any permissions problems that might occur if you install it in the usual locations. This will allow you to double click on a file to open it for example which is not the case if you are using the 'portable' (i.e., from an USB stick) version.

Versions are available for Windows, Linux, and Macs (and it doesn't have any problems running under Vista or Windows 7, 8, 8.1 and 10 (unlike GMax).

## *Disadvantages*

The user interface is quite a bit different to 'normal' Windows programs although this is true of most 3D editors. Stick at it though, as once you are used to it you can work very fast and effectively.

## *Hardware Needed*

Although I claimed above that Blender is fast, I should qualify that by stating that later versions seem to have slowed down a bit as the program has become more complicated. For creating the models, it still works fine on (say) an i3 Netbook without a dedicated graphic card but once you start baking texture maps for a model with over 100K polygons it is not exactly fun, and you'll start wishing for more power especially when working with PBR materials which are essential for TRS19 and up.

For creating PBR materials I'm going to recommend another free program, Quixel Mixer, and this needs a lot of graphics processing power. By which I mean I would say that even an i7 processor and a GTX970 graphics card in a Desktop computer is starting to get borderline. You might also like to look at PBR Painter which costs a few dollars, but which (should) allow everything to be done inside Blender. I don't have any experience with this though.

Blender is also available for Mac OS, and from personal experience I can say that an 8 core CPU/8 core GPU MacBook Air M1 is more than enough for creating Trainz assets in Blender and is at least as fast as my i7/GTX 970 Desktop - it is optimised for Apple Silicon. Annoyingly, the hot keys are not the same in the PC and Mac versions of Blender so for this tutorial I'm going to be using the PC version and the menus and icons – it's well worth getting used to using the hot keys though, as it's far faster to work with them.

## *About This Tutorial*

This tutorial will attempt to guide you through the steps necessary to create your own 3D models which can be used in the Trainz Railway Simulator or any other program for which you can find an exporter. I won't attempt to explain all the possibilities that Blender offers as a lot of the more advanced stuff just isn't needed – instead I'll concentrate on techniques that are useful for modelling for Trainz rather than the basics.

The latest version of this tutorial as a PDF file can always be downloaded here: https://1drv.ms/b/s!AgW_lPDScEO9h652Vf9X1Fczdg_ukw?e=aAFQKw

You'll need to bring a certain amount of patience and perseverance with you, learning a new 3D application is a bit like learning a new language – you must work at it. Once mastered you can create any model that you desire though, so it is worth it. I couldn't use Blender either until I tried.

If you get stuck there are many, many other tutorials as PDFs or videos available on the Internet, but most of them use earlier versions of Blender. As Blender develops some major changes have been made to the interface which render the earlier tutorials out of date, so check to make sure that they are still relevant. Checking out some of the many tutorials on YouTube for the Blender interface would be useful though.

I would particularly recommend the series by Justin Geis, as he has almost 200 videos covering everything from basis to paid addons. Some things are not so useful for Trainz content creation, but he does have a nice 'don't panic' sort of style – check some of them out. In this tutorial I'm not going to explain everything you can do with Blender as it's not necessary for getting started, if you want more detail, it's all available on YouTube.

A useful resource is the Trainz WikiBook at http://en.wikibooks.org/wiki/Trainz/Tutorial_for_Blender which covers texturing, mapping, and animation. It also has links to videos and other tutorials.

If you have access to an IOS or Android device there are also training courses available which can be useful for learning the basics.

## *Useful Strategies for Modelling (Advanced Users)*

Here is a selection of tips that I've found useful over the years, if you are a beginner these won't make any sense yet, but we'll go more into the reasons I think they are a good idea when we start to create the model. If you already know Blender hopefully, they will also prove useful. Most of them are intended to reduce the amount of work needed to create a mesh and to make it easier to make corrections later without having to redo a lot of work.

- Model only one side of the mesh and mirror it to the other side using the mirror modifier. This way any changes can be made to only one side (by convention the left side when looking forward) will automatically be mirrored to the right side.
- Make use of copied instances as much as possible when modelling. Changing one instance will change all the others at the same time which saves a huge amount of

time and ensures that parts of the mesh that are supposed to be identical really are.

- Add seams and sharp edges to each mesh as you create it. You can do this later, but that means a lot of boring repetitive work. Having the seams done also means that you can stretch and adjust the mesh and then remap it to remove any distortion.
- Export the model to Trainz often to check things are working as planned. Waiting until the model is complete often means hours of frustrating problem fixing.
- Break the model down into meaningful sub-assemblies such as Boiler, Footplate, Frames etc. It's easier to see what you are doing if you can turn off the parts you are not working on. Don't overdo this though, this makes it harder to select everything for exporting.

## *The Interface*

Once Blender starts you are presented with the following screen:

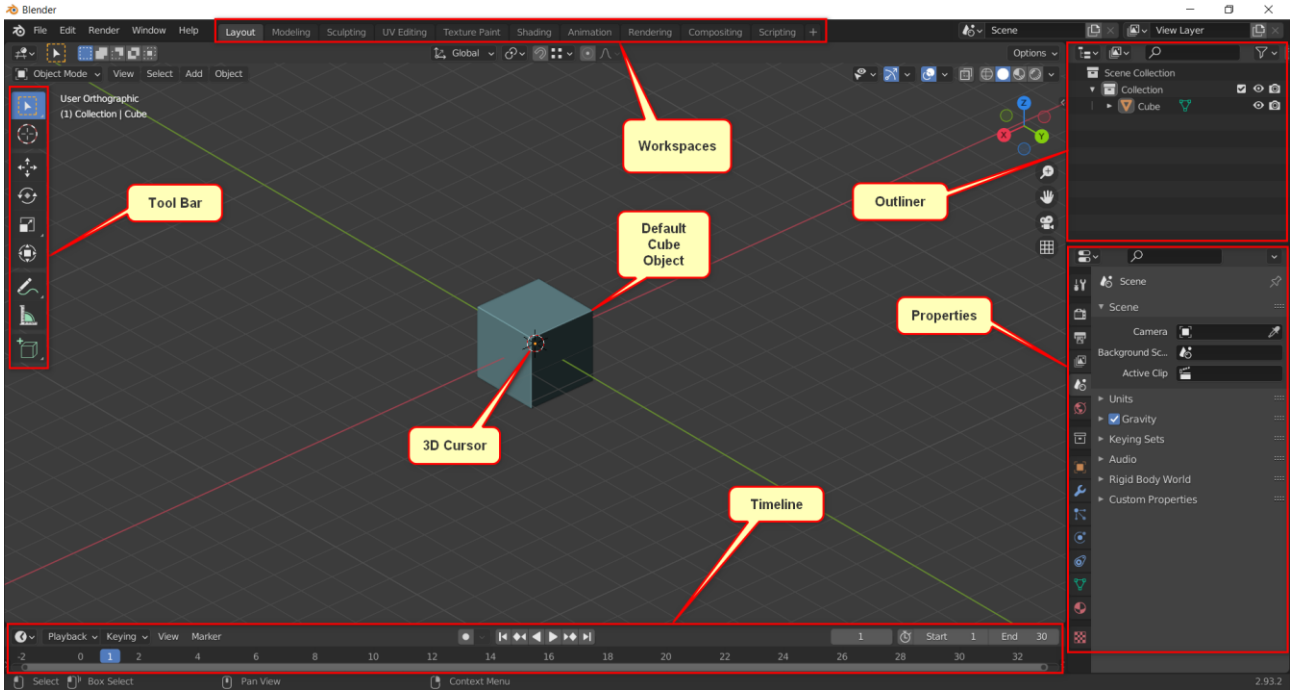



*Figure 1-1: The initial default screen when opening Blender for the first time*

Things may look a bit different when you open Blender for the first time as things change between releases and it's possible to customise almost every aspect of how Blender looks. I'm using the Blender dark theme here, this can be selected or changed via the ***Edit > Preferences > Themes*** Menu selection and then choosing a Theme.

The new interface by default has a more useful selection of windows open right from the start and it is possible to configure it for operations such as mesh modelling or UV mapping (via the 'Workspaces' tabs). In fact, it's possible to get in such a mess by changing things around one of the first things to do is to find out how to reset it back to the standard layout.

You do this by choosing ***File > Defaults > Load Factory Settings***. This will put everything back to the 'out of the box' state including the window layout. It's possible to change almost every aspect of how Blender looks, such as the size of the interface text, the fonts, and the highlight colours, but we won't be going into that here as the tutorial is long

enough as it is.

To keep the length of this tutorial down and so that we can concentrate on modelling Trainz assets I'm going to suggest that you watch at least the first few of this series of 47(!) short videos from the Blender Foundation:


instead of me writing another 30 pages about the basics.

If you don't want to view the videos, I will try and mention the essentials as we go.

## *What to Model*

I believe the best way to learn new 3D software is to use it, so we'll build a loco now and pick up the other techniques as we go along. You could equally well build a house or a wagon, but it's a good idea not to start with anything too complicated – my first loco was a simple diesel shunter which had the wheels hidden by skirts, not a BR standard loco with outside valve gear.

I'm not going to detail the making of a specific loco this time around as it's hard to pick one that will illustrate all the problems you might run into, plus I'd like people to build their own favourite loco instead of copying my example.

Whatever you want to make it's a good idea to have plans available and to use them to ensure that the proportions are correct, so the next thing to know is how to do this.

# 2. Preparing The Reference Drawings

## *Considerations When Using Available Drawings*

Firstly, don't assume that the drawing is correct – for example some drawings published in the last few years in the UK magazine 'Railway Modeller' were full of howlers or even incomplete. Especially as nowadays the drawings are usually prepared on a computer there is a tendency to use an existing drawing as the basis for another so you may find parts of the drawing may not apply to the locomotive (or whatever) it is supposed to represent.

Dimensions can also be downright wrong. I've seen drawings where the wheel sizes and wheelbases were clearly a long way out, so check any text supplied with the drawing for clues for what the dimensions should be.

It's a good idea to have photographs of the prototype as well to check the drawing, and although it's normally not recommended to use photos of models these can be clearer than prototype photos - just remember that not all models are correct either.

Secondly don't assume that if you've scanned a drawing in that the scale in both axes is correct. The scale in the scan direction along the axis of the scanning head will be constant, but probably not in the other direction as this is dependent on the speed that the scanning head moves. A cheap scanner may well not move at a constant speed, and this may mean that the scale in that direction could be variable. In the next section we'll look at a suggestion for minimising the impact this could have.

Lastly don't assume that the drawing is symmetrical about the centreline or that it has even been drawn correctly. For example, several of the drawings by Roche have different diameters of smokebox door in the front and side elevations which have mislead even commercial manufacturers over the years.

Although correcting mistakes in a Blender model is a lot easier than taking a real model apart it's still annoying to build a model that is out of proportion because the drawing was wrong in the first place.

## *Finding Reference Images*

Blender from v2.8x on has a way of displaying reference images which can be easily made semi-transparent, can always be displayed in front of the 3D model and be only visible when looking in an orthographic view, but I'm not going to recommend doing this. The problem with this is that it's not possible to correct distortions which always seems to creep in, at least when I scan in a drawing. You might have more luck.

I've had the best results by applying the scanned image to a Plane object and that's what I'll describe later.

A few words now on finding plans, I've created some Google spreadsheets which list drawings and other articles in the UK magazines 'Railway Modeller', 'Model Railway Constructor' and 'Model Railways' and here are links to them:

Index for Railway Modeller

Index for Model Railways

Index for Model Railway Constructor

In case you are wondering where to get ancient copies of these magazines I recommend Magazine Exchange, back numbers cost less than new magazines.

I announced the availability of the Indices on the Auran Forum, here is a link to the original thread:

http://forums.auran.com/trainz/showthread.php?95269-Railway-Modeller-Magazine-Index-%28UK%29

They can also occasionally be found on Ebay.

## *Using Imperial or Metric Units*

Blender normally uses 'Blender units' and the exporter assumes that one unit is one metre. I normally convert all measurements to metres – this isn't as bad as it sounds as you only need measurements for scaling the drawing and positioning the wheelsets, everything else being built directly from the scanned drawings.

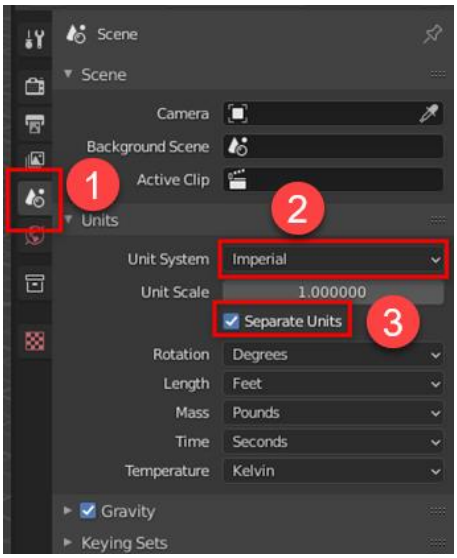



*Figure 2-1*

It is possible though to change the units to imperial here in in ***Scene properties*** tab, just select ***Imperial*** and ***Separate units*** this allows you to enter the measurements in feet and inches – otherwise you will be limited to feet and decimal parts (i.e., 6.5 for 6ft 6ins).

When entering the imperial values in numeric fields you just type the measurement in the form 6ft6in or 6'6" for example. Note that selecting ***Imperial*** changes all values displayed in the input fields instantly to the imperial values.

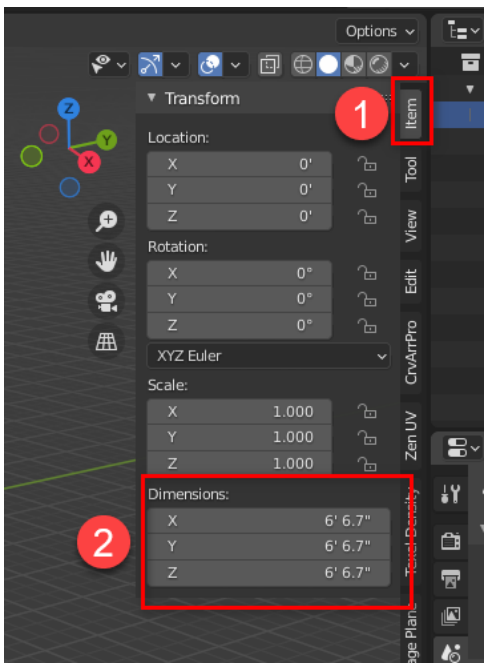



*Figure 2-3: The standard default cube dimensions in imperial units*

Selecting imperial units when the default cube is on the screen results in the dimensions being shown as seen here in the image to the left. Creating a new cube after changing the units to imperial will result in one that is 2ft x 2ft x 2ft.

Likewise selecting '***Metric***' and '***Separate units***' allows you to type the measurements in metres and centimetres. It's unlikely you'd want to do this of course as it's just as easy to use metres and decimal parts of metres.

You can also translate objects using imperial measurements, but one thing to watch out for is that you need to type a '**=**' symbol first before entering the imperial measurement. To translate an object by 6ft 6in in the Y direction you'd need to select it and then type ***GY=6ft6in***. Note that it is not possible to specify fractions of an inch, you'll need to use decimals, 3.25in for 3 1/4in.

Another problem with using imperial units is that you must be careful with negative values.

If you want to move an object to -20'3” you might expect that typing ***-20'3”*** in the input field would do the trick, but in fact this moves the object to -19'9”. This is because Blender treats the input like a formula and sees the value as -20' +3” - to get it to behave as you'd expect you need to input either ***-20'-3”*** or ***-(20'3”)***. Despite the somewhat shaky implementation (at least up to v2.93) it's still a useful way of avoiding the need to constantly convert imperial dimensions on drawings to metric.

## *Loading The Reference Drawings*

Prepare the plans by scanning them in, they can be any scale and (almost) any format (GIF, PNG or JPEG are good). The bitmaps don't have to be any particular size and do not have to be square – I usually scan the drawings at 300 dpi which gives good detail and a reasonable file size.

Mapping the image to a 2D plane is how we used to do it in the good old days of GMax, and it does have quite a few advantages over a background image in a viewport (the method usually recommended). These are:

- it's quite fiddly getting the background image in the right position and to the correct scale, if the image is on a plane the whole plane can simply be moved and scaled just like any other object in the 3D window which is a lot faster. Useful if you have a lot of reference images.
- Blender still (at least in versions up to 2.73) has the annoying habit of losing the background images in the views for some reason. If you haven't noted the position and scale of each of them, you need to set them up all over again if this happens.
- Using a 2D plane means you can subdivide it and distort specific areas of the image if necessary to match known dimensions (**this is important!**)
- When using the background image in a view method the image(s) need to be cropped to exactly the right size (i.e., overall length and height) as otherwise it's even harder to get the image positioned and scaled. Using a 2D plane means that the cropping need not be so exact, and you can easily make the image big enough to include any dimensions that might be present which is handy for reference.
- Blender includes an add-on as standard that creates a plane of the correct aspect ratio and maps an image to it automatically. The only thing you need to do is to point the add-on at an image and then position, rotate, and scale the plane.

I'm using in this case a drawing of a LMS Stanier 2-6-0 loco, but the principle will be the same for any loco, or indeed a wagon or a passenger coach. The scan doesn’t need to be prepared so that the horizontal lines really are horizontal, any errors can be corrected in Blender.

The first step is to activate the 'Import Image to Planes' addon by choosing ***Edit > Preferences***. This addon is built in by default, you don't need to download it.

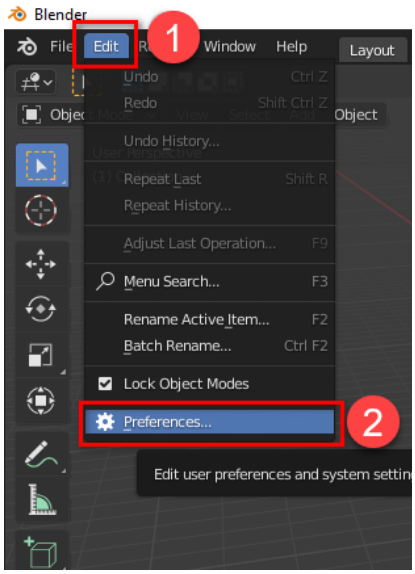
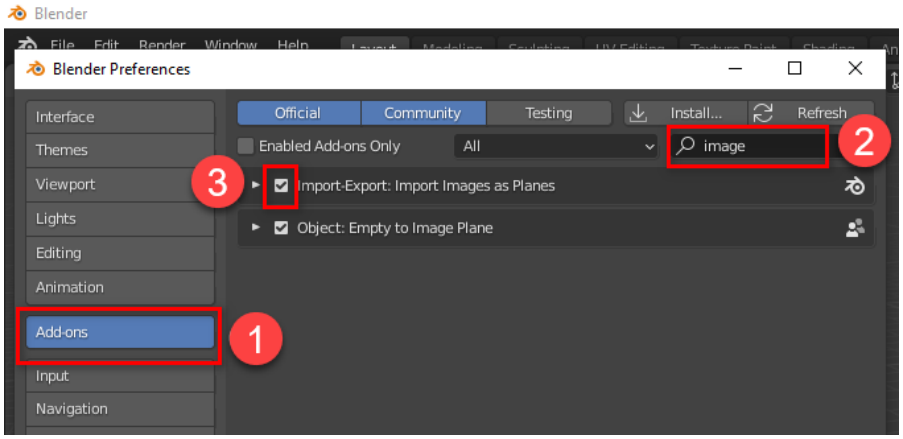



*Figure 2-4: Opening the Preferences dialog box*




Next choose ***Add-ons***, enter 'image' in the search field and finally activate the addon by ticking the box.

*Figure 2-5: Figure 2-6: Activating the Add-on*

If you now choose ***File > Import > Images as Planes*** you can navigate to where you saved the screenshots and open them. All the screenshots you select will then be loaded and appropriate Plane Objects created.

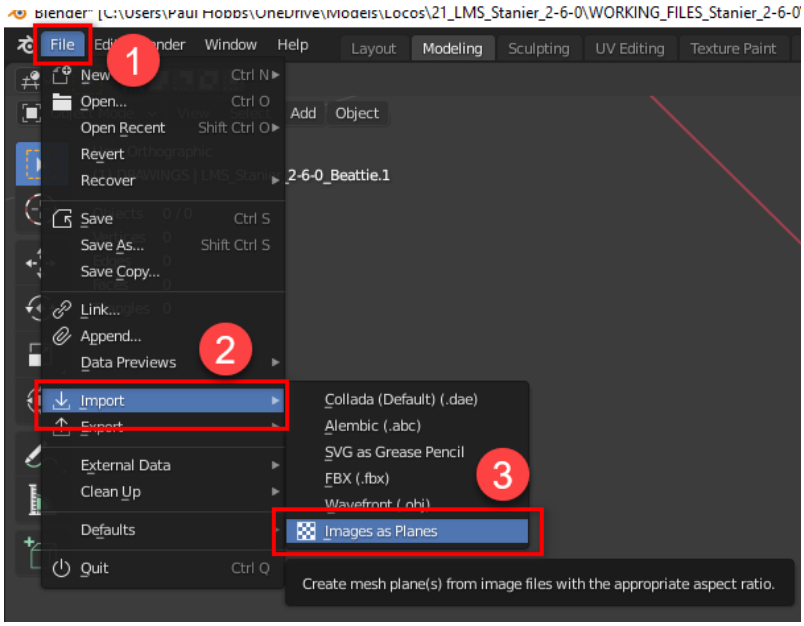



*Figure 2-7: Importing the image(s)*

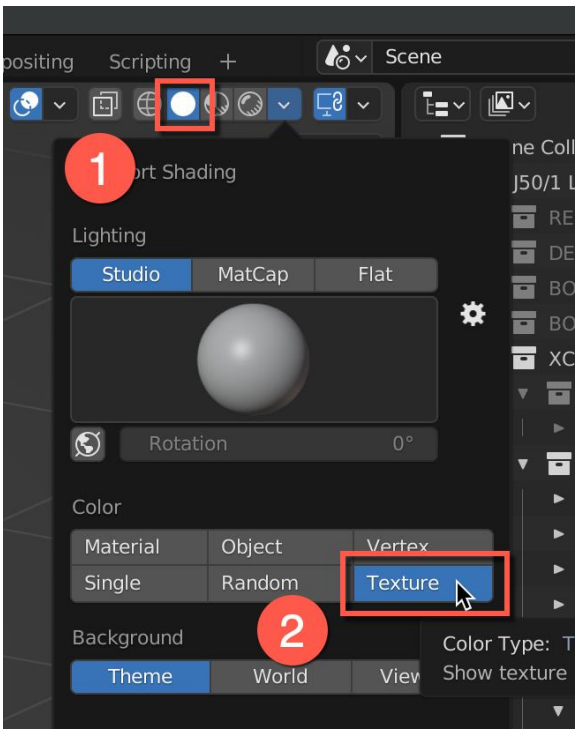



*Figure 2-8: Setting the viewport render mode to solid and the Colour to 'Texture'*

**Make sure that the viewport render mode is set to solid (1) and that the Colour is set to Texture (2), as otherwise the image(s) will not be visible!**

Also, it's useful to turn on the Viewport Overlay 'Wireframe' so that we can see where the vertices are that we want to snap to. I also prefer to work with this setting on so that I can always see the face edges, even in Object Mode.

This setting makes it easier to see where the image planes have been divided up.

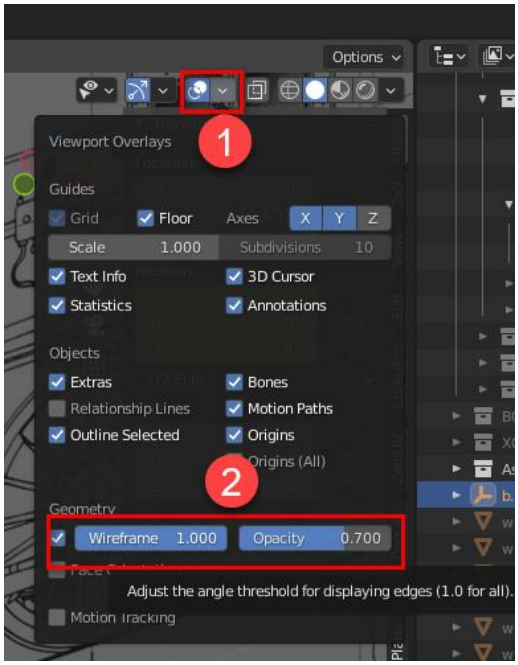



*Figure 2-9: Turning on the 'Wireframe' Viewport overlay*

Here I have selected and loaded one image for the loco and tender. Note that the Plane Object has the correct aspect ratio and has been positioned at the 3D Cursor. The position of the Object Plane in 3D space is immaterial as we'll be moving and rotating it later.

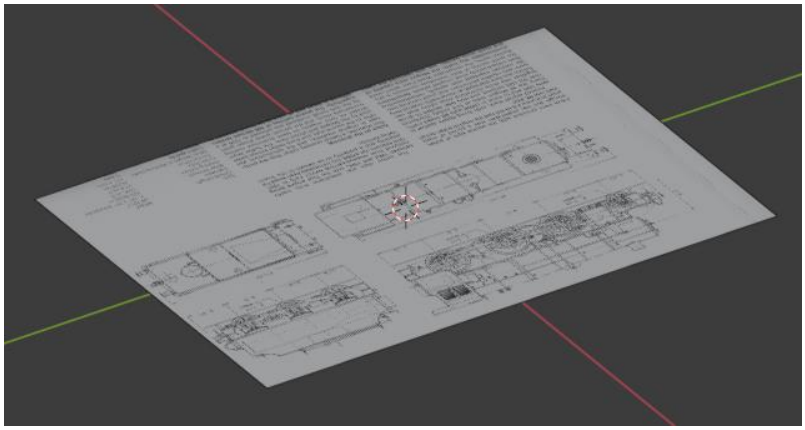


*Figure 2-10: Plane Object inserted with the image applied*

## *Rotating and Scaling the Reference Drawing*

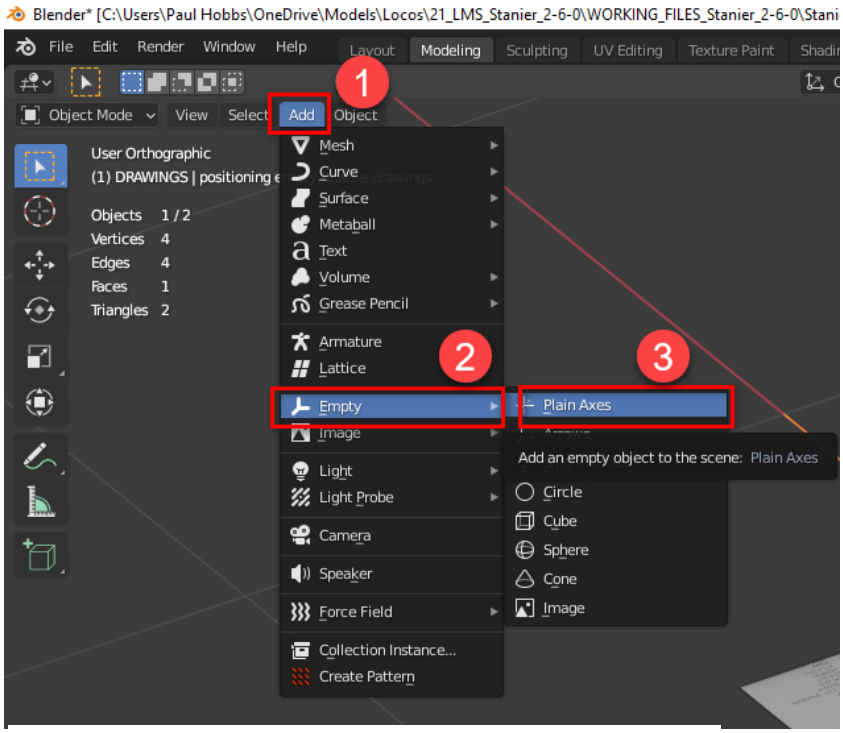



*Figure 2-11: Adding an Empty Object*

Next add an Empty Object by choosing ***Add > Empty > Plain Axes***

The Plain Axes are very useful for aligning Objects as you can easily change the size of them to create a crosshair effect.

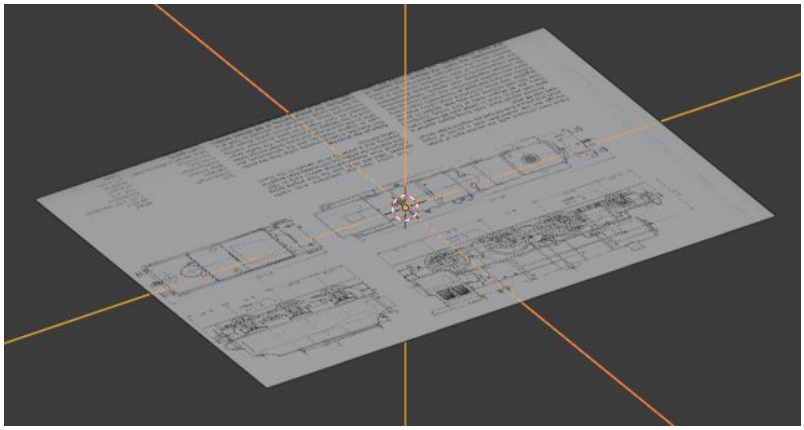
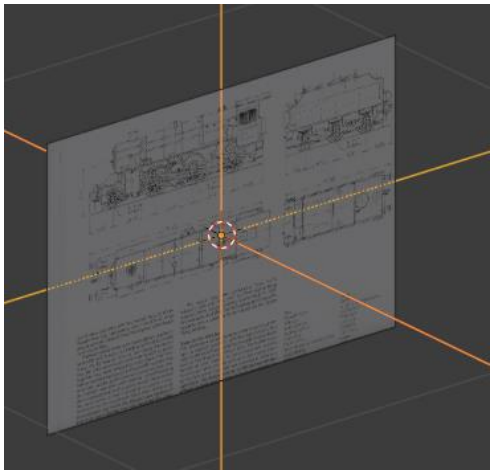


*Figure 2-12: Empty Object added and selected (to make it more visible)*



Next rotate the Plane Object, in this case by 180° in Z and 90° in Y, by selecting the Plane and then typing ***RZ180*** and then ***RY90*** in this case. **Important!** The front of the loco must point to the left when looking in the side view (click on the red X at the top right of the 3D view to switch the view to check this)

*Figure 2-13: Plane Object rotated*

With the Plane Object selected, move it by pressing ***G*** so that the end of the line denoting the top of the rail is directly under the cursor. As you can probably see, the scanned line is not exactly horizontal, and we need to correct this.

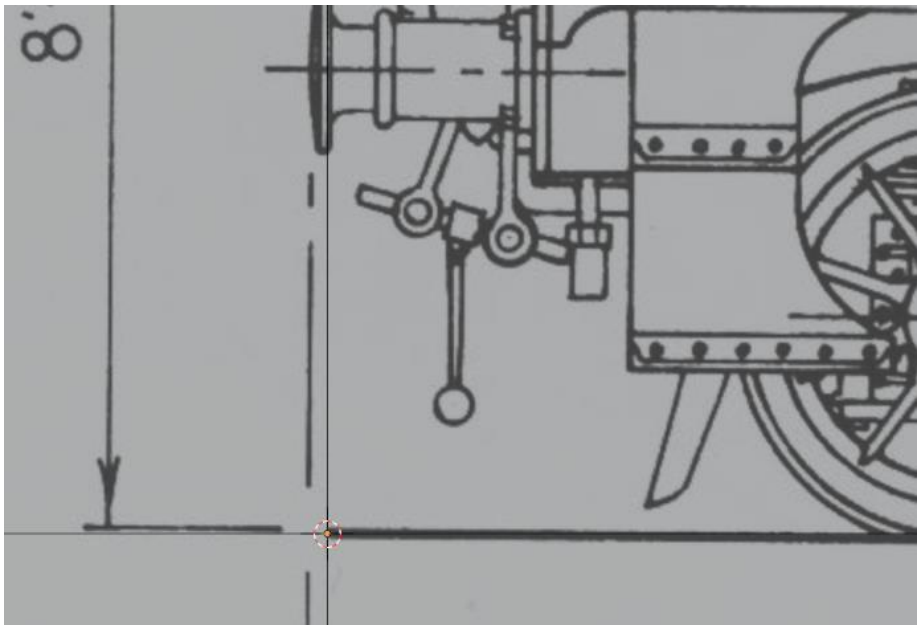


*Figure 2-14: Positioning the Plane Object*

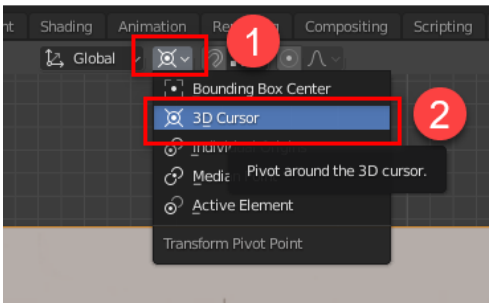



*Figure 2-15: Pivoting about the 3D cursor*

First, make sure that we are pivoting about the 3D cursor.

Now Rotate the Plane Object by pressing ***R*** so that the right-hand end of the line representing the top of the rail head is horizontal, using the Empty as a guide.

You can hold down ***SHIFT*** when dragging to give finer control over the rotation.

Note that as all of the views are on the same Plane Object, we have corrected the rotation of all of them in this step.

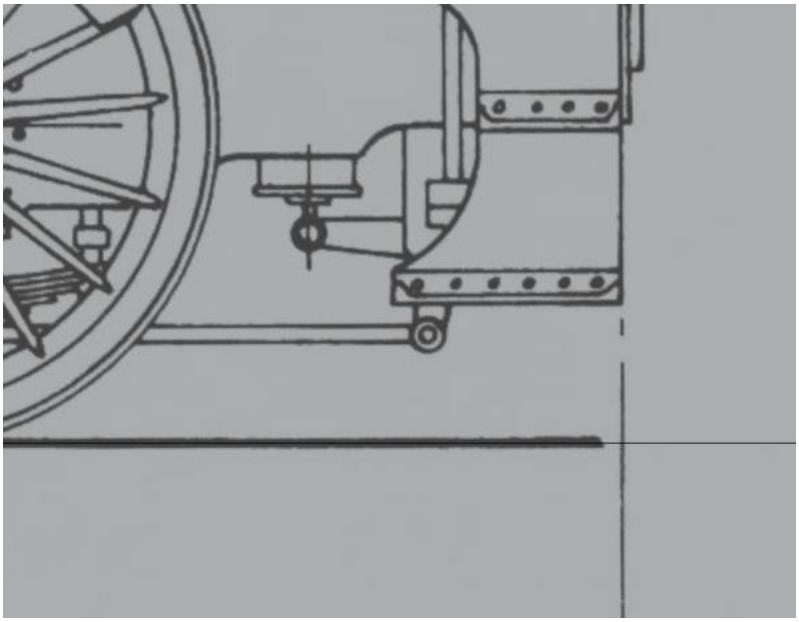
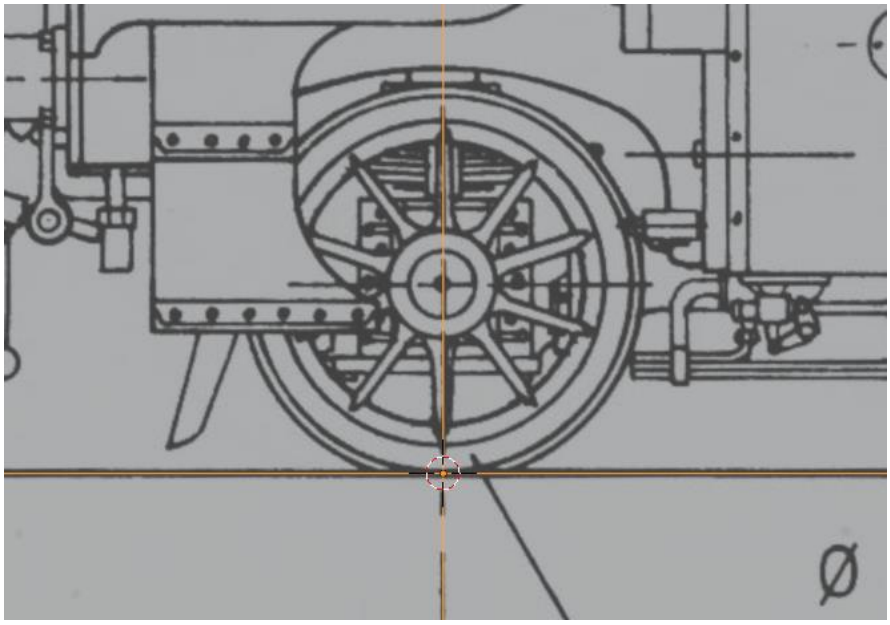


*Figure 2-16: The rotation of the Plane Object corrected*



Now move the Plane Object by pressing ***GY*** and dragging so that the Empty lines up with a suitable datum point, in this case the centre of the front wheel.

*Figure 2-17: Plane aligned to the first empty*

Now to scale the Plane Object to the correct size. In Object Mode, select the Empty and press ***SHIFT-D*** and then ***ENTER***. This creates a copy of the Empty in the same position and selects it. In this case we know that total wheelbase of the loco is 9ft + 8ft + 8ft 6ins = 25ft 6ins - we need to choose a large dimension to reduce the chance of scaling errors. With the copied Empty selected, type ***GY = 25'6"*** this will position the Empty correctly assuming that Imperial units are selected.

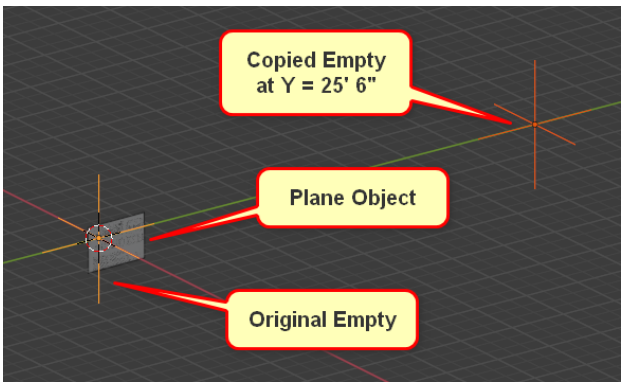



*Figure 2-18: Copying the Empty and translating it*

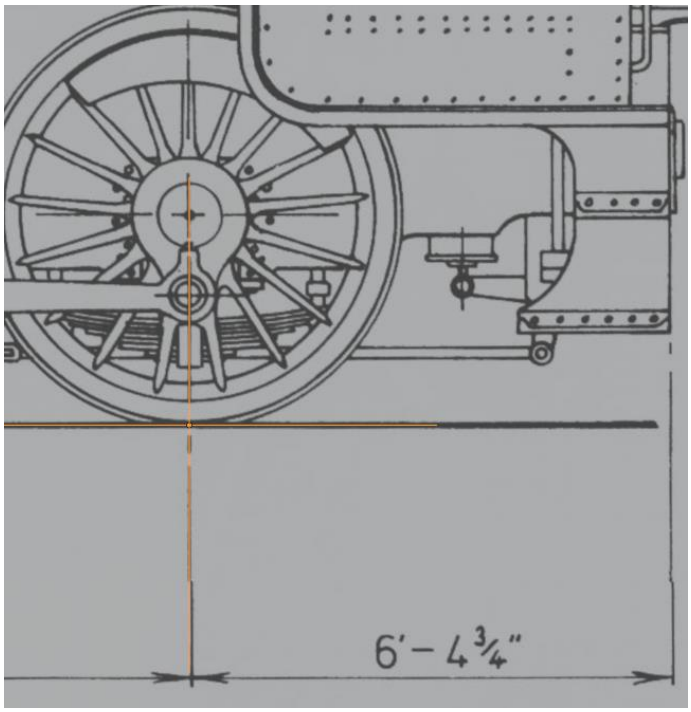



*Figure 2-19: Drawing scaled to match the position of the copied empty*

Select the Plane Object and then press ***S*** and drag the mouse to resize the Plane Object so that the rear loco wheel on the drawing lines up with the copied Empty.

The next step is to duplicate the first Empty with ***SHIFT-D*** and move it in Y by pressing ***GY*** and dragging to align it with the front of the buffer beam on the drawing.

Do the same to the second Empty that you created and align it with the rear buffer beam. We are going to use these two new Empties to move the Plane Object to the final position.

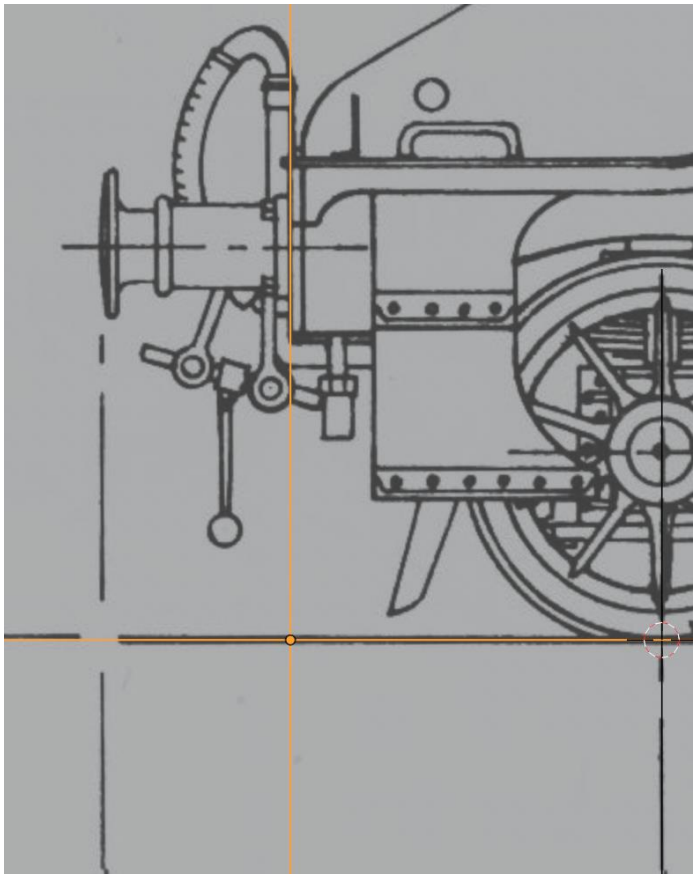


*Figure 2-20: Third Empty added and aligned to face of front buffer beam*

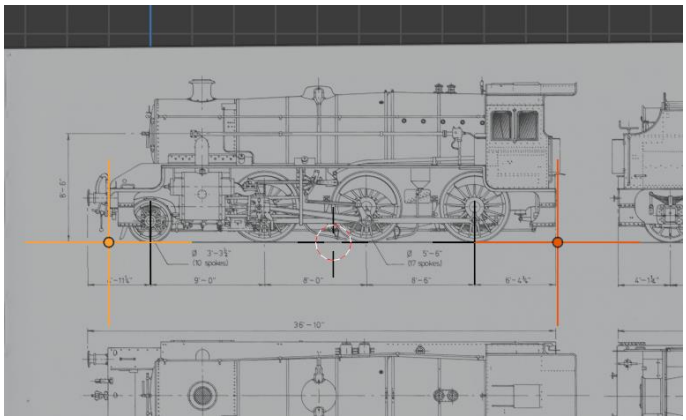


*Figure 2-21: The 3D Cursor moved to the mid-point between the third and fourth Empties*

Select the third and fourth Empties and then press ***SHIFT-S*** and choose ***Cursor to Selected*** from the pie menu to move the 3D Cursor to the mid-point between the two selected Empties.

Now Select the Plane Object and then choose ***Object > Set Origin > Origin to 3D Cursor*** to move the origin point of the Plane to the 3D cursor.

The point of doing this is that we can now set the position of the Plane Object to Y=0.

The third and fourth Empties will become the attachment points a.limfront and a.limback which should be equidistant from Y0.

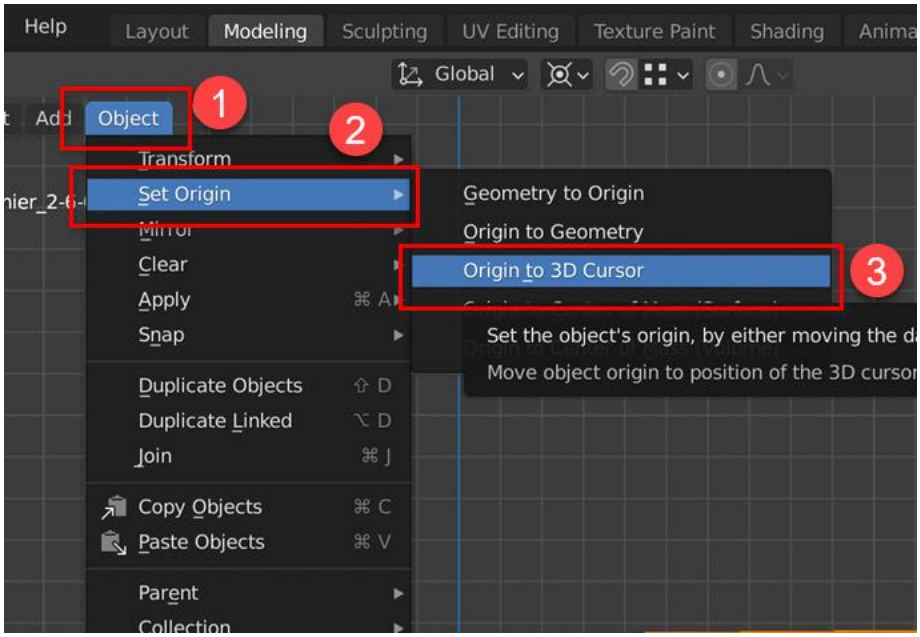



*Figure 2-22: Moving the origin of the Plane Object*

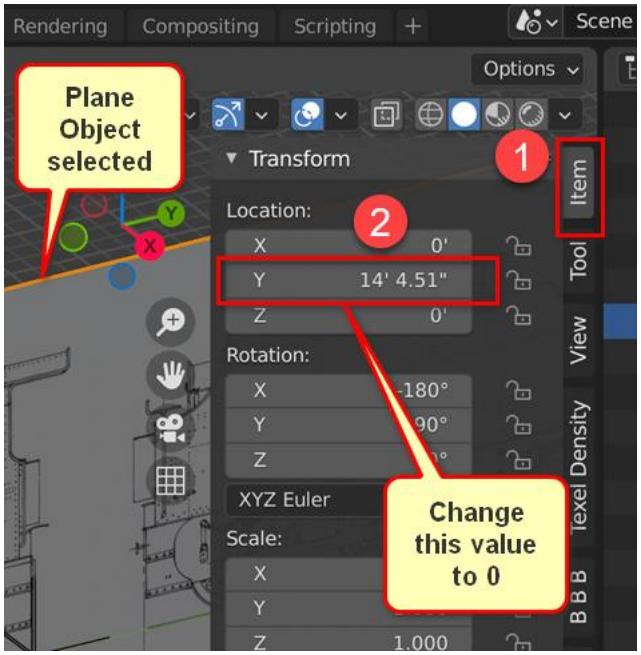



*Figure 2-23: Finding and changing the coordinates of the Plane Object*

Select the four Empties (probably easiest to do this in the Outliner Window) and then the Plane Object. It's important to select the Plane Object last, as we are going to parent the Empties to the Plane to move in one operation. Now press ***N*** to open the right-hand tool bar and change the Y location to 0. All the objects will be repositioned to the left.

Now select the four Empties and choose ***Object > Parent > Clear and keep Transformation*** to break the link between the Empties and the Plane Object.

## *Correcting Drawing Distortions*

Next, I like to correct any distortions in the drawing. To do this we need to create some lines that correspond to the known dimensions, such as the wheel centres, height of the boiler centreline and the chimney, and the rear end of the mainframes.

Making sure we are in Object Mode, create a cube by choosing ***Add > Mesh > Cube***. Enter Edit Mode by pressing ***TAB*** and make sure that the selection mode is Vertex as shown here.

Now choose ***Mesh > Merge > Collapse*** to make a single vertex out of the cube, and then press ***EZ*** and drag the mouse to extrude another vertex, creating a line.

For completeness, I should point out that there is a built-in addon called ***Extra Objects*** which can be activated in the Preferences which will allow you to add a new single vertex object directly, which saves a few mouse clicks.

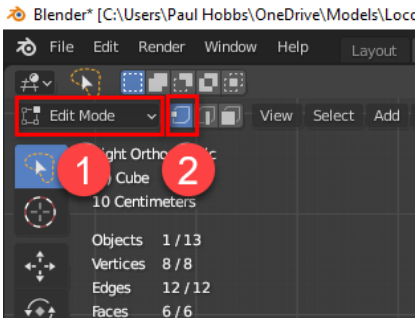



*Figure 2-24: Selecting vertex selection mode*

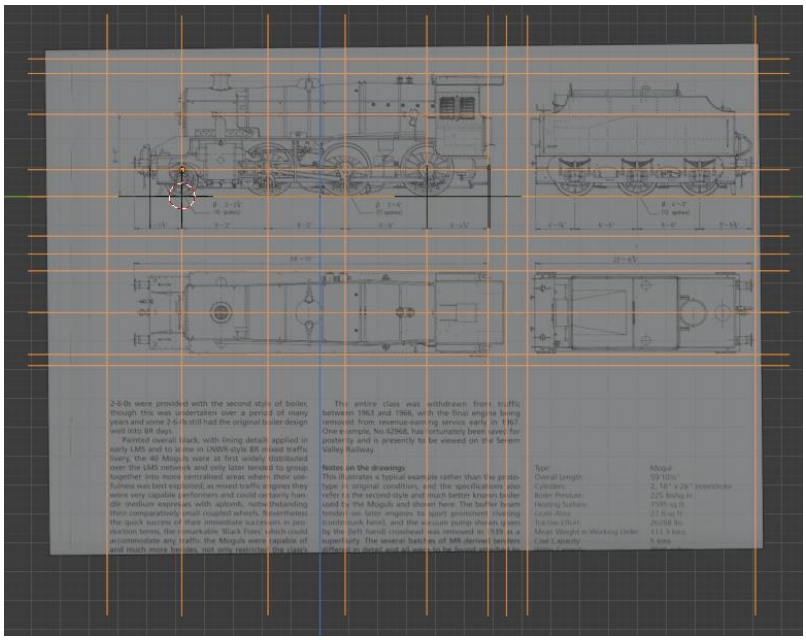


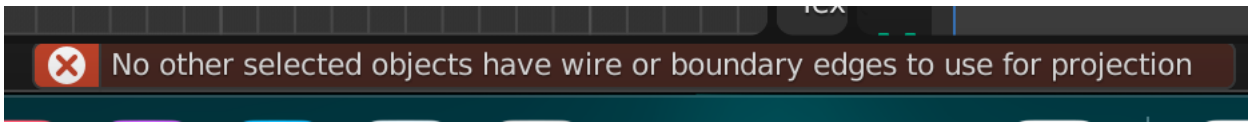
*Figure 2-25: The lines that will be used to divide up the Plane Object*

Repeat the procedure to create as many lines as you have dimensions for. The lines should be long enough to completely bisect the Plane Object.

I've added some extra lines for the centreline of the front pony wheel, the rear driving wheel and the top of the rail as I know these are correct and I don't need to change them. Also a few lines to remove areas of the Plane Object that are not needed, like the gap between the loco and tender for example, and the rest of the wheel centres.

To split the Plane Object up, join all the lines to a single object by selecting them and choosing ***Object > Join***.

Now select the Plane Object and press ***TAB*** to enter Edit Mode. Then, in the Outliner, select the Line Object with a left mouse button click and choose ***Mesh > Knife Project*** to divide up the Plane Object. Note that this behaviour was changed from version 2.93.5 onwards and therefore older tutorials will show different steps for using this function. If you follow an old tutorial, you will get the following error message:




So don't do that…

**As an aside, I've recently noticed that you can get the above error message even if you are using Blender 3.0 when working in a file that was created in an earlier version.**

**To fix this you must open a new, empty file in Blender 3.0 and then merge in everything from the older file – then Knife Project will work correctly. It is probably a good idea to do this anyway when working with older Blender files. I always include the Blender version in the filename so that I know which version of Blender was used to edit the file.**

Now we can check for distortions in the drawing and correct them. First turn on the display of edge lengths like so (we'll need to be in Edit Mode to see this setting).

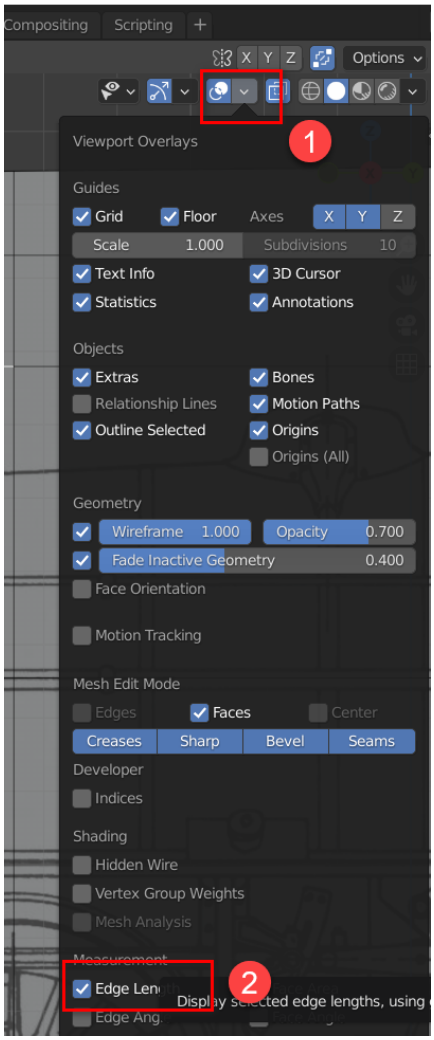
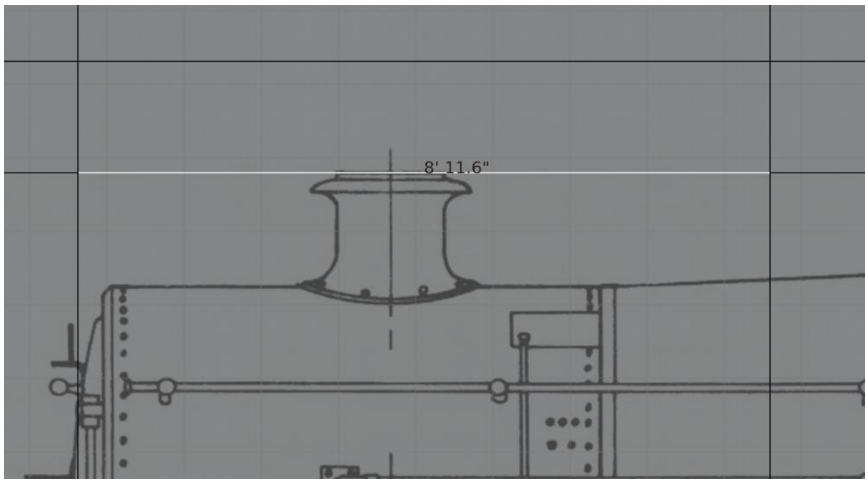



*Figure 2-26: Displaying edge lengths in edit mode*




By selecting this edge, we can see that the distance between the centreline of the front pony wheel and the front driving wheel is 8’ 11.6” when it should be 9’ (in this case, only being 0.4” out is pretty good!).

*Figure 2-27: Current measurement between the centreline of the front pony wheel and the front driving wheel*

To correct this all we need to do is to select the edges at the centreline of the front driving wheel by holding down ***ALT*** and left clicking one of the edges to select the entire loop then move them 0.4” to the right. This will bring the centre of the front driving wheel on the drawing into exactly the correct position. Repeat for all the other edge loops to correct all the distortions in the scanned image.

As an aside, I have two drawings of this loco, one from Ken Morgan and one from Iain Beattie – overlaying the two reveals that the only things the two drawings have in common is the wheelbase and the wheel diameters. Everything else is different. The Morgan drawing even has the crank throw of the drivers wrong which will mess up the animation of the valve gear, so I’m going to be using the Beattie version.

In fact, the only advice I can give is to pick one drawing and stick with it, checking it against known dimensions. In the light of this it seems a bit pointless to correct errors of 0.4” in the scanned drawing, but I feel it’s worth it to get the final model as accurate as possible.

## *Finalising the Reference Drawings and Tidying Up*

We can now delete the parts of the Plane Object that we don’t need and move the individual views into the final position.

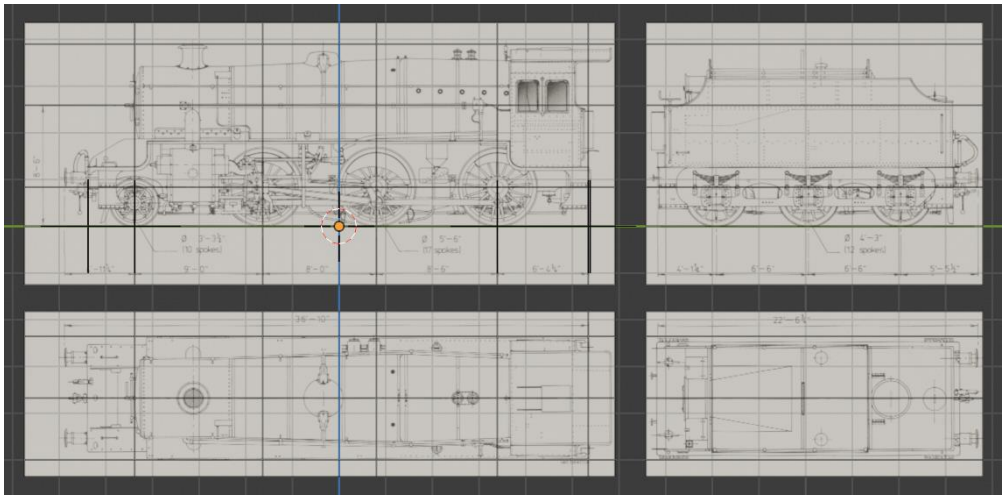


*Figure 2-28: Unnecessary faces deleted*

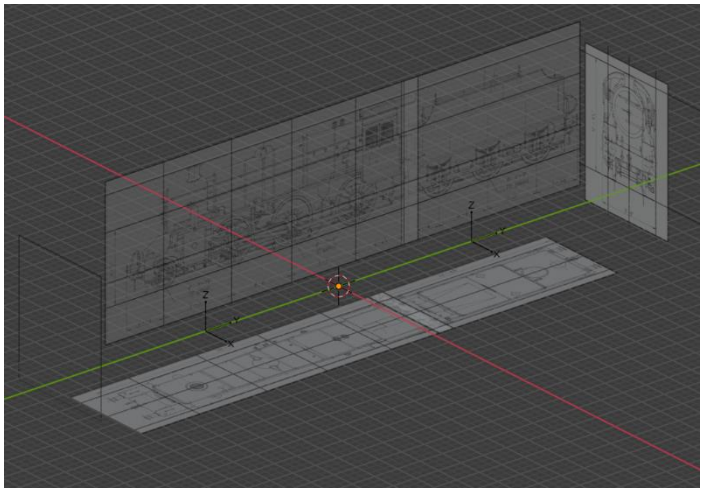


*Figure 2-29: Final position of reference images*

Select faces belonging to each view on the Plane Object and separate them by choosing ***Mesh > Separate > Selection*** to create separate objects from the Plane Object. These can then be rotated and moved into the final position like this.

XXX EDIT NEEDED!

Here I've also added the front view of the loco and the rear view of the tender which were in a separate scan.

The origin point of each scan has also been moved to 0,0,0 by setting the cursor position to 0,0,0 with '***SHIFT-C***' and then selecting each plane in turn and choosing '***Object > Set Origin > Set Origin to 3D Cursor***'. I find this to be a very good idea as it's possible to accidentally move one or more of the Image Planes and if the origin is set to 0,0,0 it's easy to check and move it back to where it should be if necessary.

I've also renamed the Planes to something more sensible and have created Collections for them which makes it easy to turn them all off if they are in the way. Clicking on the eye icons will turn the objects on or off, clicking on the arrow icons will make the objects selectable or non-selectable.

Clicking on the eye/arrow icons for the collection(s) will affect the whole collection.

It's a good idea to make the image planes non-selectable to avoid moving them accidentally!

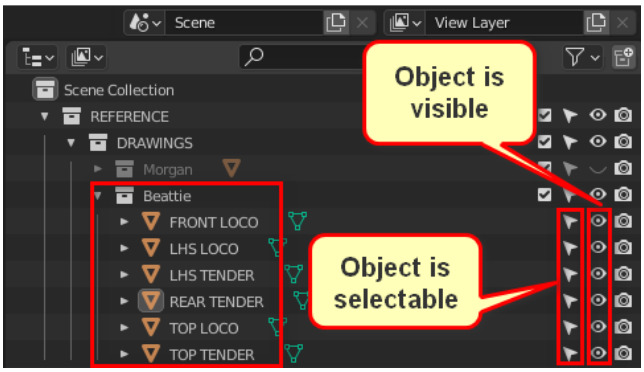



*Figure 2-30: The collections to organise the reference drawing planes and making the planes selectable/deselectable or visible/invisible*

I would also suggest making the Image Planes non-renderable as this will prevent them casting shadows on the model which you don't want when baking AO maps. This can be done by deselecting the camera icon for the Image Plane(s) in the Outliner.

One last thing to note before we leave the subject of reference drawings is that it could be a good idea to pack the images in to the *.blend file. This means that the one working file contains all the information need to display it properly and the images are always available no matter what computer you are using to edit the file.

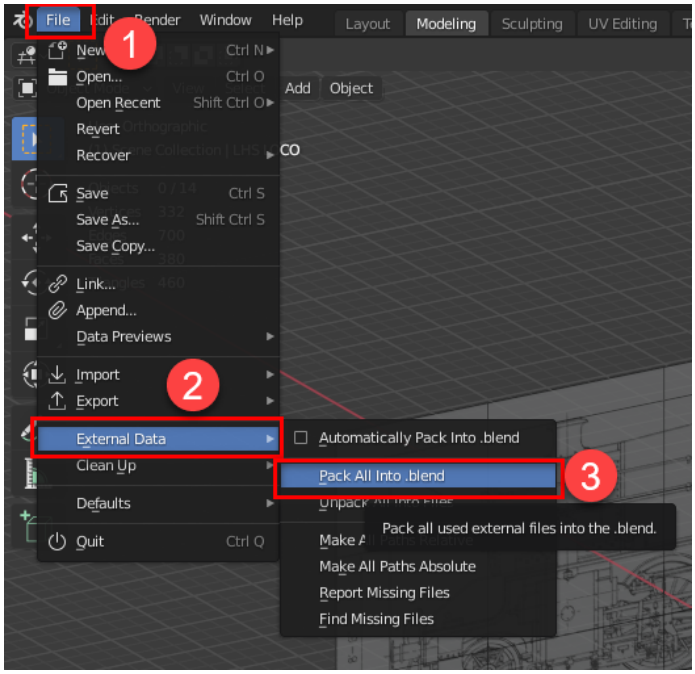



*Figure 2-31: Packing the external data into the *.blend file*

It's a good idea to avoid huge bitmaps though, these will increase the loading and saving times. The packing is carried out by choosing ***File > External Data > Pack All Into .blend***.

I must admit though that I don't usually do this, as I have all my working files stored on Microsoft's OneDrive so that all of the files are always available on all my machines as long as I have an Internet connection. If I was going to be working on a machine without a constant Internet connection though, I would pack the scans into the *.blend file.

Having set up the interface and prepared the plans we can now start to model. There are only a few basic techniques needed for this such as creating a profile and adding a thickness and rotating a profile to make a round object.

## 3. Creating The Wheelsets

I like to start constructing a loco with the wheelsets, and for this I'm going to recommend downloading a *.blend file to simplify this. The objects are built so that it's easy to change the diameter and the number of spokes in the wheelset.

This will also show you useful techniques such as creating round objects and radial arrays of objects. In fact, creating the wheelsets will introduce just about everything you need to build the rest of the model, so it's worth working your way through this section. If you wish to skip this step, there are loads of pre-made wheelsets available from my website from the BitsBox page.

Download the driving_wheel_kit.blend file from here and save it locally somewhere where you can easily find it again.

Before we really get started on creating a model, I would like to point out that at this stage we are creating a relatively high poly model – this will not be used in-game, for that we need to create lower poly models (known as LODs or Level of Detail meshes). Don't worry about the poly count.

### *Merging In Objects from Another Blender File*

Start by choosing ***File > Append*** to open the append dialogue box and navigate to where you saved the WheelKit file.

Double left click on the driving_wheel_kit.blend file and then open the 'Object' folder and

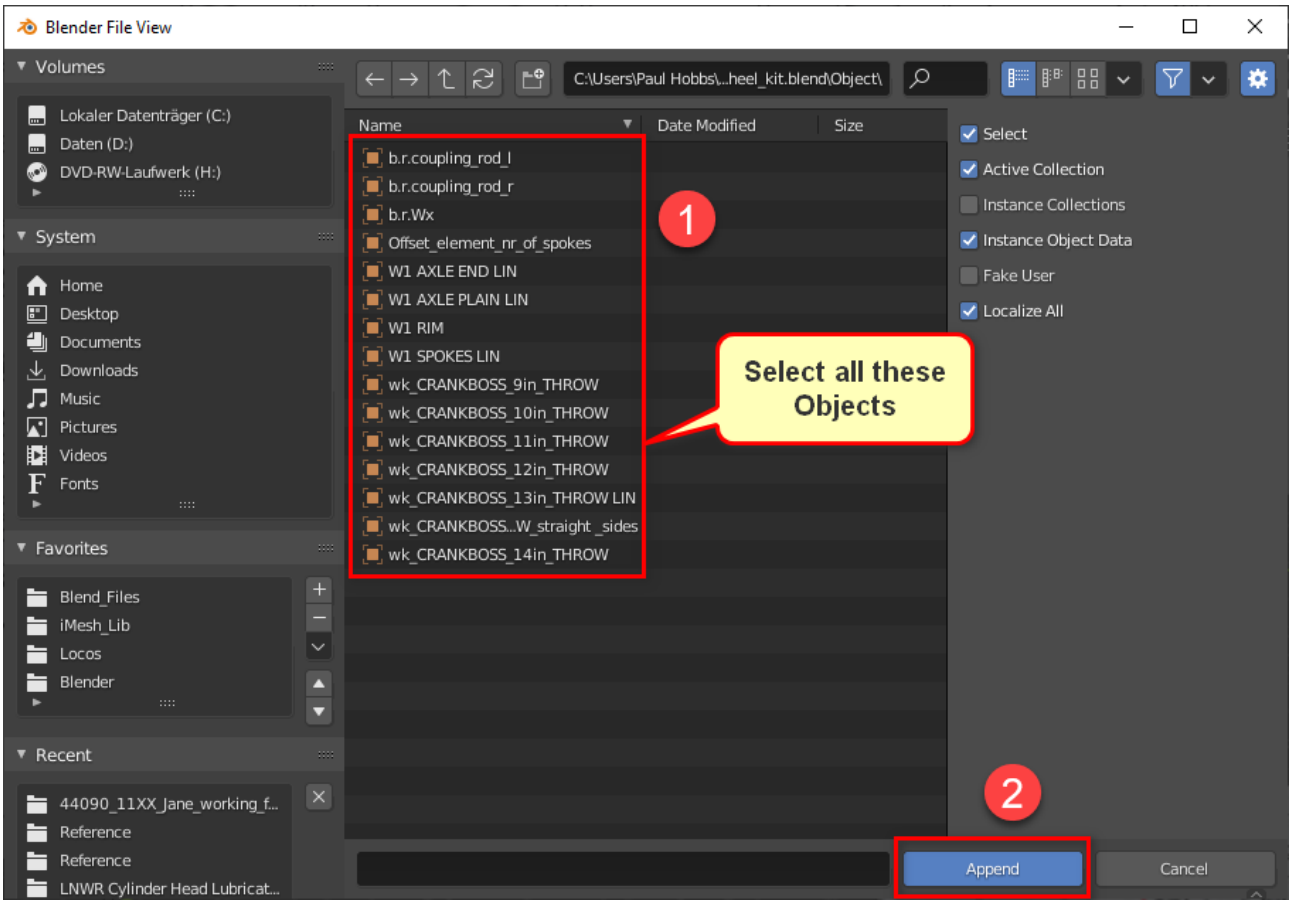



*Figure 3-1: Merging in the WheelKit Objects*

select all of the elements that are visible in the file by pressing ***A*** when the mouse cursor is over the window showing the object names they will then be highlighted. Finally click on the ***Append*** button.

All the objects from the WheelKit file have now been appended to the current file, retaining their positions relative to each other. Any Blender file can be used as a library in this fashion incidentally.

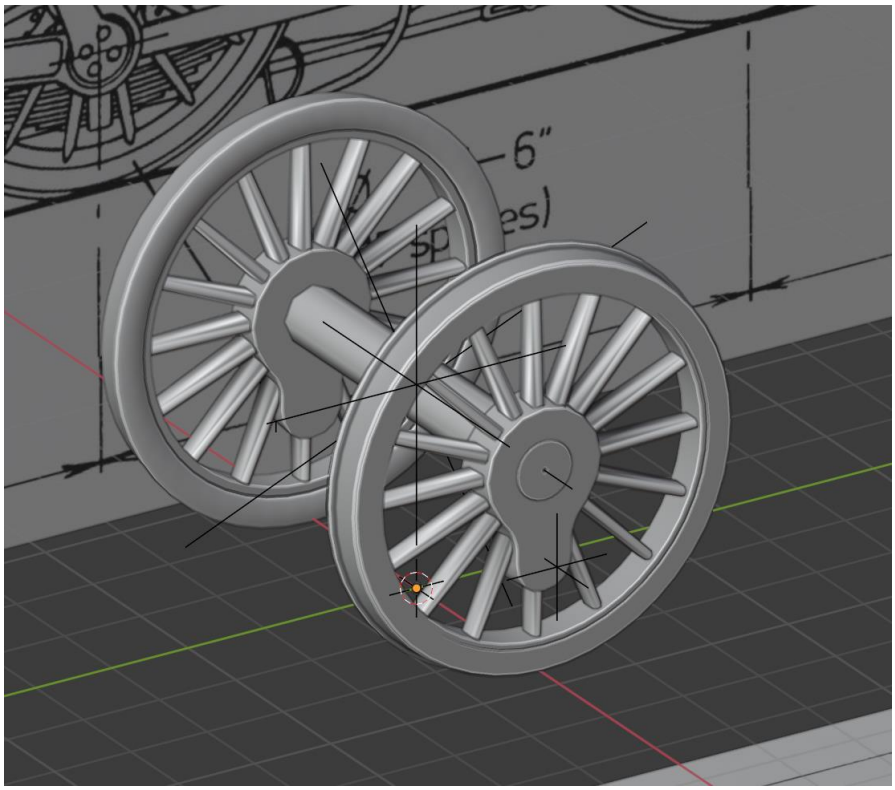



*Figure 3-2: WheelKit Objects loaded*

The WheelKit asset is for a 5’ diameter 16 spoke 26” stroke wheel, but this can all be edited to match the loco you are building. The asset also includes some alternative crank bosses which are hidden but can be made visible in the Outliner and used if they are more suitable.

Even if the Wheelset is exactly what you need, I’d strongly suggest that you go through the following steps to acquaint yourself with translating vertices and adjusting the Array Modifier as these skills will be useful later.

## *Moving the Wheel Objects to the Correct Position*

For this loco we need a 5’ 6” wheel with 17 spokes, the crank pin throw is correct at 13”. The next step is to move the Empty ‘b.r.Wx’ to the correct position in Y and Z, so select it and translate it up to 2’ 9” by changing the value in the Transform Properties (opened by pressing ***N***).

All of the Objects will move up by 3”, as they are all parented to ‘b.r.Wx’.

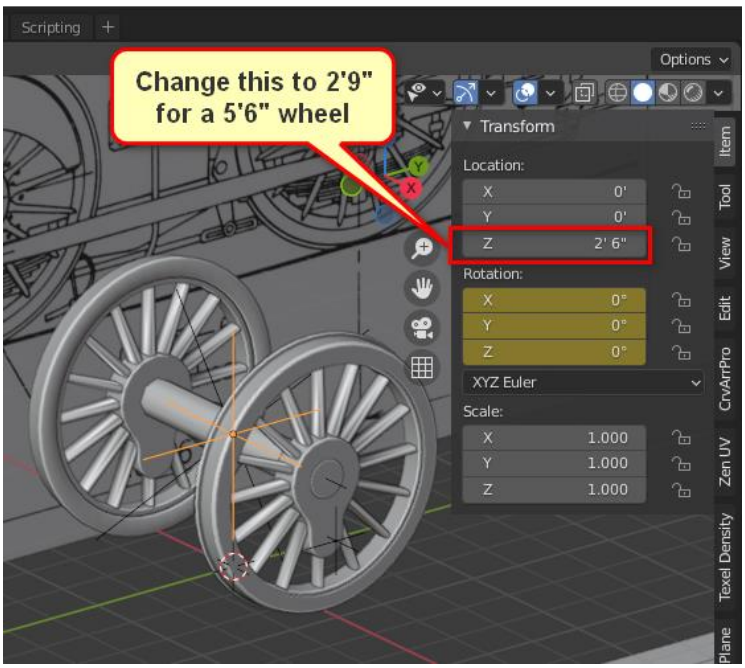



*Figure 3-3: Moving the WheelKit in Z*

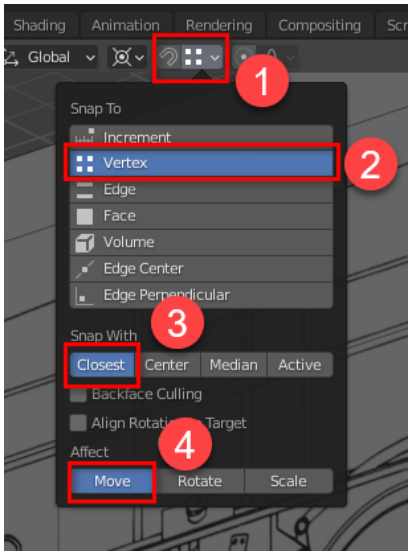



*Figure 3-4: Setting snap mode to vertex*

Now we need to move ‘b.r.Wx’ to the correct position in Y, and to do this we need to check a couple of settings first.

We need to ensure that the Snap Mode is set to Vertex when moving objects so that we can snap the position of ‘b.r.Wx’ to a vertex on the Image Plane. This avoids the need to calculate exactly how far we need to move it.

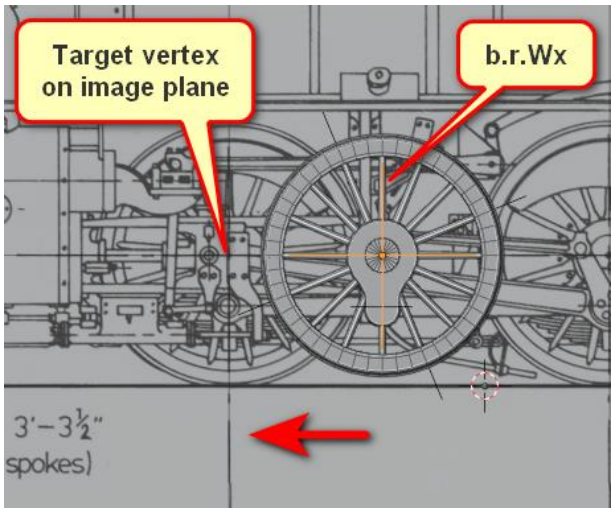



*Figure 3-5: Snapping 'b.r.Wx' to the wheel centre on the image plane*

With 'b.r.Wx' selected, type ***GY***, hold down ***CTRL***, and then drag it to the correct Y position using the vertex in the image plane at the front driving wheel centre, pressing ***ENTER*** to set the position. ***GY*** constrains the movement to the Y axis only and holding down ***CTRL*** activates the snapping mode.

And here are the WheelKit Objects positioned at the centre of the front driving wheel on the image plane.

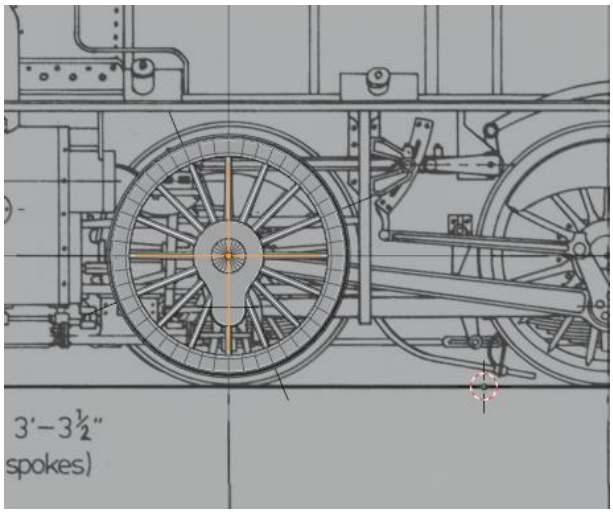



*Figure 3-6: 'b.r.Wx' snapped to the correct Y position*

## *Correcting the Wheel Rim Diameter*

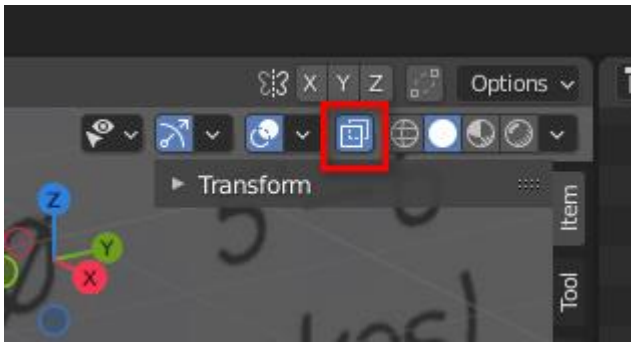



*Figure 3-7: Toggling X-Ray mode on/off*

To correct the wheel rim diameter, select the Object 'Wx RIM' and enter Edit Mode by pressing ***TAB***. Click on this icon, or press ***ALT-Z***, to switch to X-Ray mode so that all the vertices in the rim object can be selected. If you don't do this, you won't be able to select the vertices on the rear of the object.

With the vertices selected (you must be in vertex selection mode of course), translate them down by typing ***GZ=-(3")***.

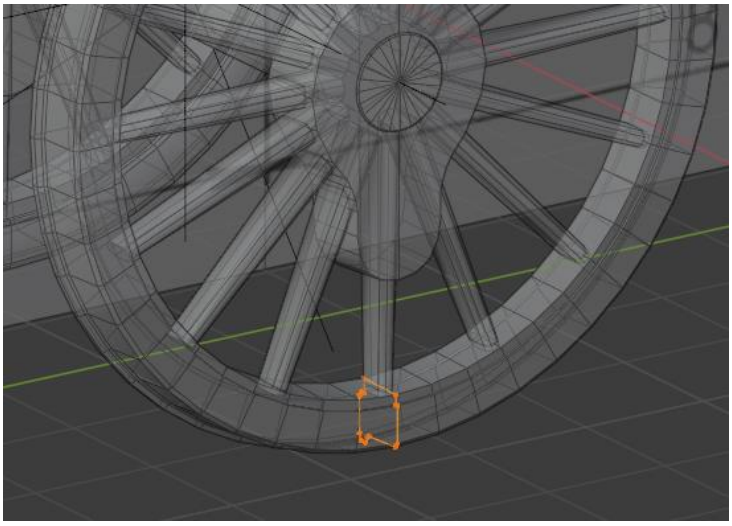


*Figure 3-8: Diameter of wheel rim corrected*

The vertices will move down and make the wheel rim the correct diameter. The reason that this works is that the wheel rim profile has a Screw Modifier on it which sweeps the profile object 360° around the origin point of the profile. The profile can easily be changed to thicken up the rim for example to match the drawing if needed. We'll see this in action later when we increase the taper of the rim section.

## *Adjusting the Number of Spokes via the Array Modifier*

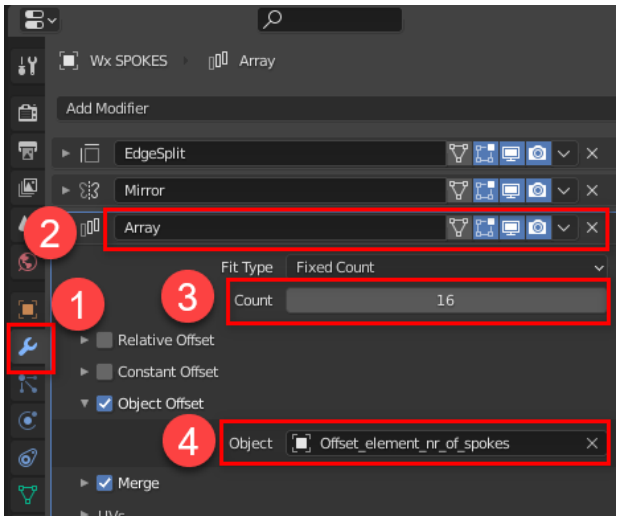



*Figure 3-9: Adjusting the number of spokes in the Array Modifier*

To correct the number of spokes from 16 to 17, we need to edit the Array Modifier on the Object 'Wx SPOKES'. Select the Object and in the modifier tab change the count to 17.

The angle between the spokes is controlled by the Empty ‘Offset element nr of spokes’, so nothing seems to happen as we haven’t changed the rotation of ‘Offset element nr of spokes’ yet. Do this now by selecting the Empty and setting the X rotation to 0°.

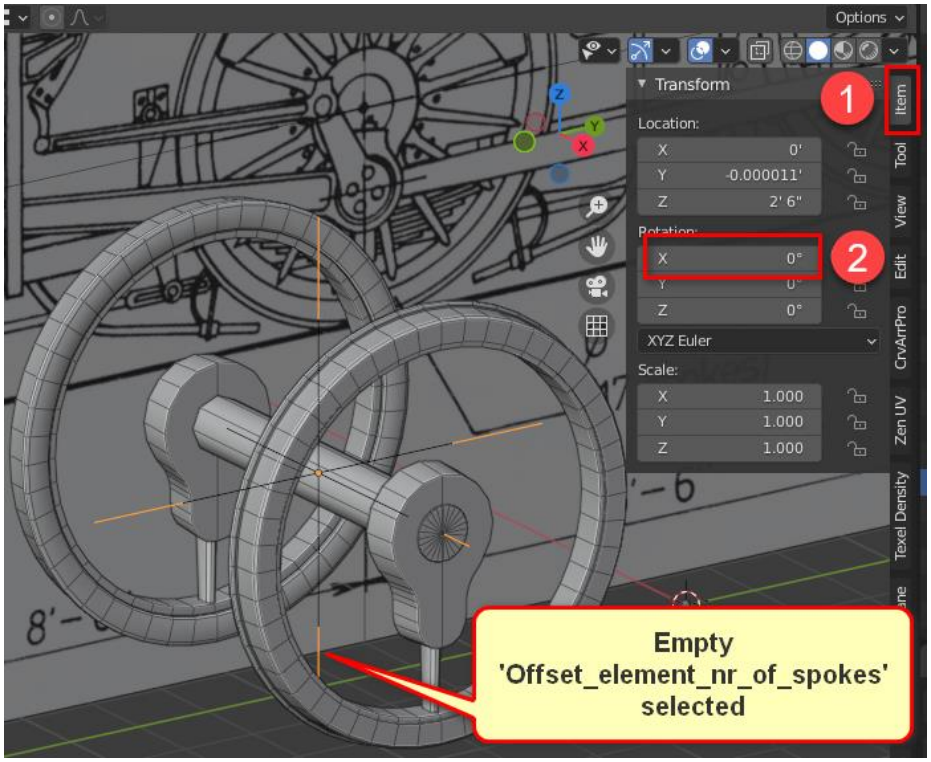



*Figure 3-10: Rotation of Empty controlling the angle between the spokes set to 0°*

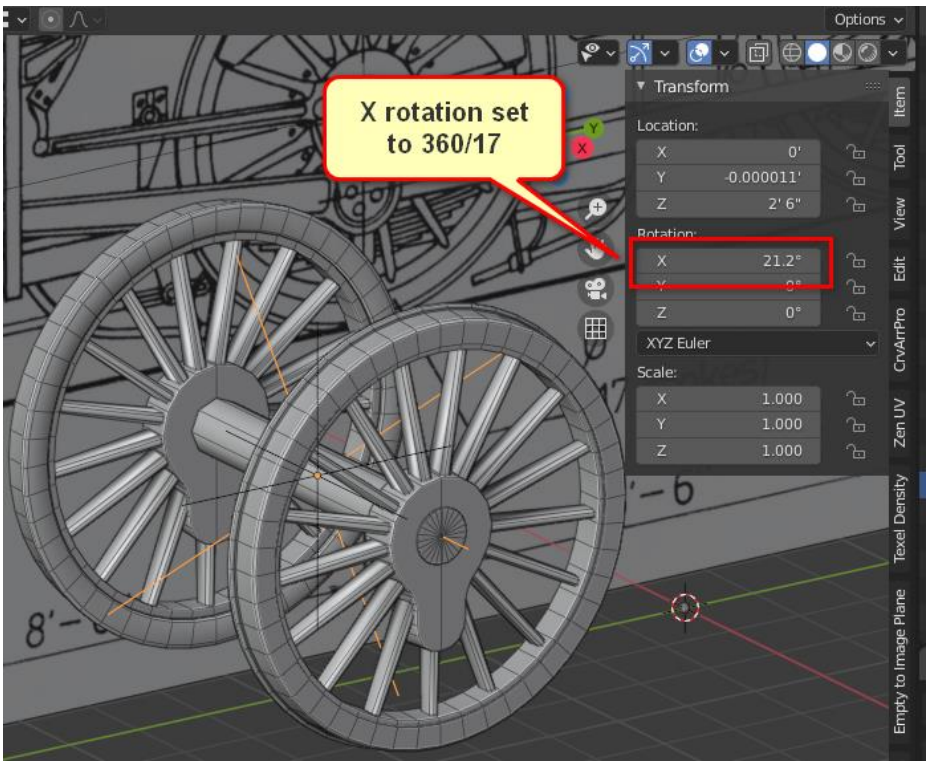



*Figure 3-11: Angle of the Empty in the Array Modifier corrected for 17 spokes*

The reason for setting the rotation to 0° first is to avoid having to calculate the difference between the angles for 16 and 17 spokes.

With the rotation set to 360/17 (the displayed value is rounded by default, but if you click on the input field it will show the correct value).

The Wheelset now has the correct number of spokes for this loco.

One more thing to do in this case is to correct the position of the spoke Object as the crankpin should be between the spokes and not in line with one of them. To do this, simply select the spoke Object, enter Edit Mode and select all of the vertices (turn on X-Ray Mode to ensure that you select all of the vertices). Then rotate them by typing ***RX10.5882*** (which is 360/17) to rotate all the spokes by half the angle between each one.

To avoid making this calculation we can also type ***RX=(360/(17*2))*** which would give us the same result.

We have now adjusted the parameters to get the Wheelset roughly as we want it.

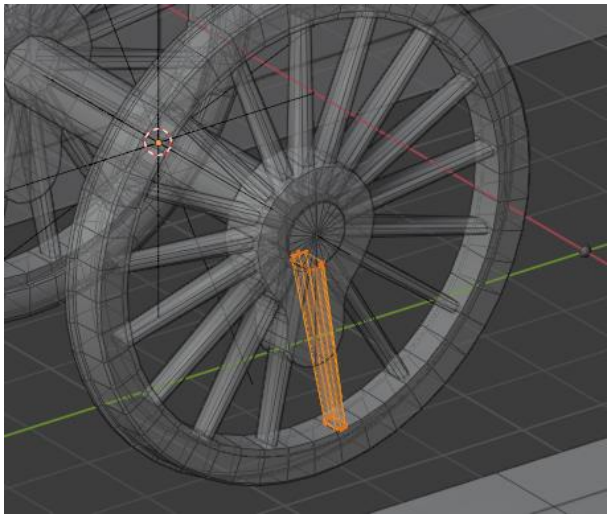


*Figure 3-12: Spokes rotated so that the crankpin is between the spokes*

The spokes contained in the WheelKit are basic – they should flare into the crank boss, have fillets where they join the rim and the crank boss, and they should also be oval in section and not rectangular. I must admit up until now I've never bothered with these refinements as it would make modelling the wheel a lot more complicated, but I see that some others have started to create better looking wheels than this...

Also, you might be interested to know that the wheels for an inside cylinder locomotive are different to those of an outside cylinder locomotive. The wheel bosses are usually much thicker, and the spokes are flared more.

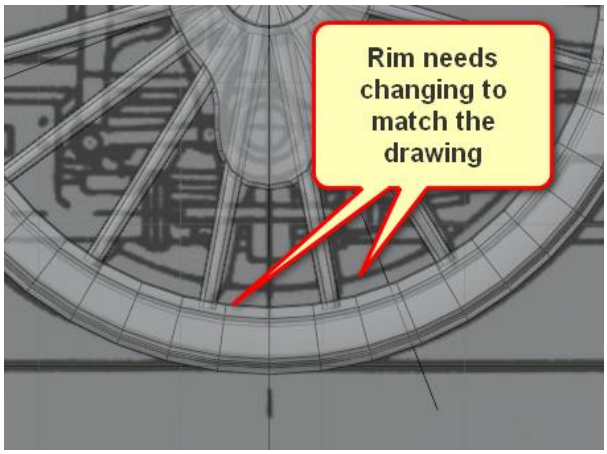



*Figure 3-13: Comparison of wheel asset and reference drawing*

Comparing the wheel asset to the drawing we can see that the rim profile needs to be changed – locos designed during the Stanier era on the LMS had an obviously heavily tapered rim which we can model just by changing one vertex.

Switch to the side view after selecting the vertex and drag it in the +Z direction until the rim matches the reference drawing.

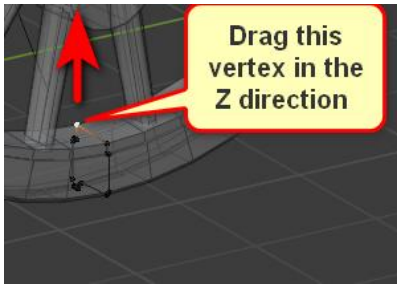
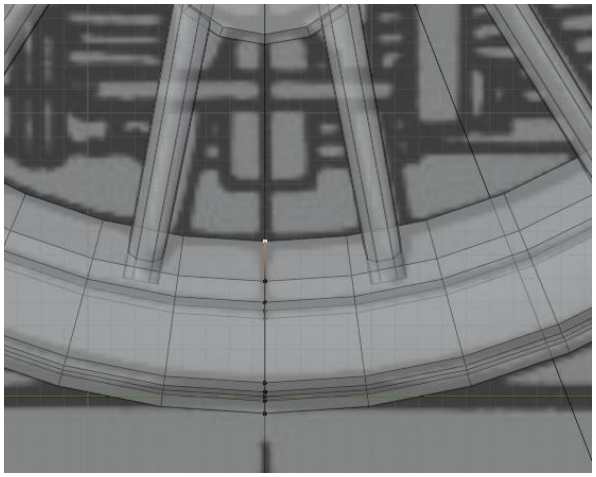



*Figure 3-14: Selecting the vertex to correct the rim profile*



Like so. A similar technique can be used to make the wheel tyre thicker or thinner if necessary.

*Figure 3-15: Rim profile corrected to match reference drawing*

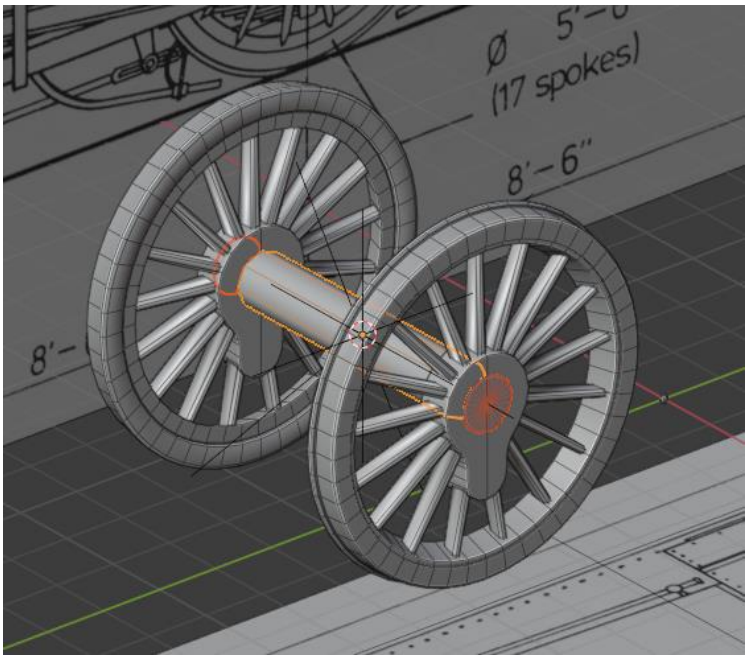



*Figure 3-16: Axle and Axle End scaled up to match the reference image*

## *Adjusting the Scale of the Axle and Axle Ends*

The diameter of the axle is also a little too small, here we can easily rescale these before we make the copies for the other driving wheelsets. Start by selecting the axle and the axle end objects and then press ***S*** and then ***SHIFT-X***. ***S*** starts the scaling operation and ***SHIFT-X*** locks the scaling in the X axis, this because we don't want the axle to get longer, just thicker.

Check that the 3D cursor is positioned at the centre of the axle and that the ***Transform Pivot Point*** is set to 3D cursor.

## *Applying the Modifiers and Creating the Copied Instances*

Before we apply the modifiers, we should consider whether the number of segments in the wheel rim is ok at 48, or if they should be increased or decreased. Decreasing them will reduce the poly count, which is good, but too low will result in a wheel with visible flat spots. Try out different values in the Screw Modifier to see what suits you.

You need the change the '***Steps Viewport***' value here.

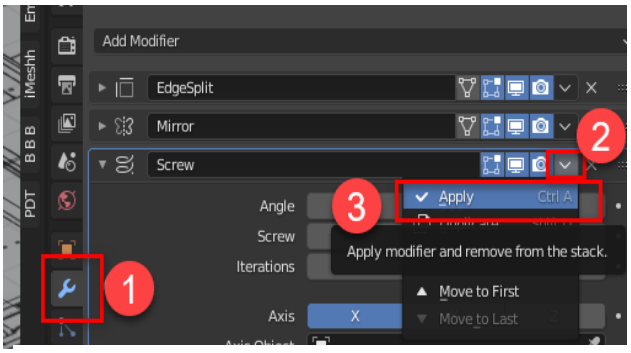
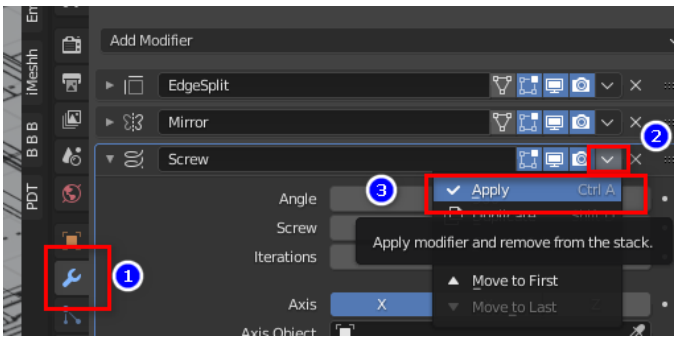



*Figure 3-17: Changing the number of steps in the Array Modifier*




Select the rim object and apply the modifier as shown here. The ***Edge Split*** and ***Mirror*** modifiers can stay for now. Repeat this step for the Array modifier for the spokes object.

The 'Offset_element_nr_of_spokes' object can now be deleted.

*Figure 3-18: Applying the modifiers*

The reason for applying the modifiers is that we are going to create copied instances of the wheel objects, and these will be linked so that changing the geometry of one wheelset will affect the others as well – unfortunately the modifiers can't be linked so it's better to apply them and work directly with the geometry. Until the modifier has been applied it's not possible to edit the vertices and faces of the objects as they don't exist except in the viewport.

In Object Mode, select all the objects that make up the wheelset. Then press ***ALT-D*** and then type ***GY*** and drag the copied objects, snapping them to the centre of the middle set of drivers on the reference image (the same technique we used to position the first wheelset). Repeat to create a third wheelset in the correct position.

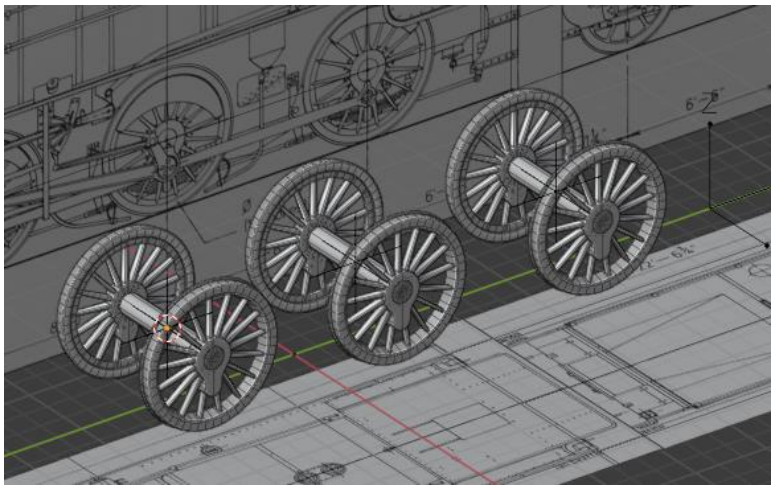


*Figure 3-19: All three driving wheels created using linked instances*

As all the objects in the wheelsets are linked instances, changing one of them will change the others as well. This is a very powerful technique to edit lots of objects all at once and

can be used for items like brake gear as well. When we come to the UV mapping, we'll break some of these links, but for now they should be kept linked until the modelling is complete in case changes are needed.

Try this out by selecting a vertex or a face in one of the objects and move it. All the instances will change as well.

This chapter has been quite long, but we have covered a lot of techniques that will be needed for the rest of the modelling.

## 4. Mesh Creation – Flat Parts

The easiest way to create flat parts such as mainframes, footplates, tanks or cab sides is to simply trace lines over the reference drawings and fill them with faces. To do this we need to create a new object consisting of a single vertex. This can be done by inserting a cube object and then entering Edit Mode by pressing ***TAB.***

Then choose ***'Mesh > Merge > Collapse***' from the menu.

The cube will collapse down to a single vertex. This is a very handy way of creating a new object with just a single vertex. There are addons available add a single vertex as a new object, but this functionality has now been added to Blender – just activate the built-in addon '***Add Mesh: Extra Objects***' and the option will appear in the '***Add > Mesh***' menu opened by pressing ***SHIFT-A***.

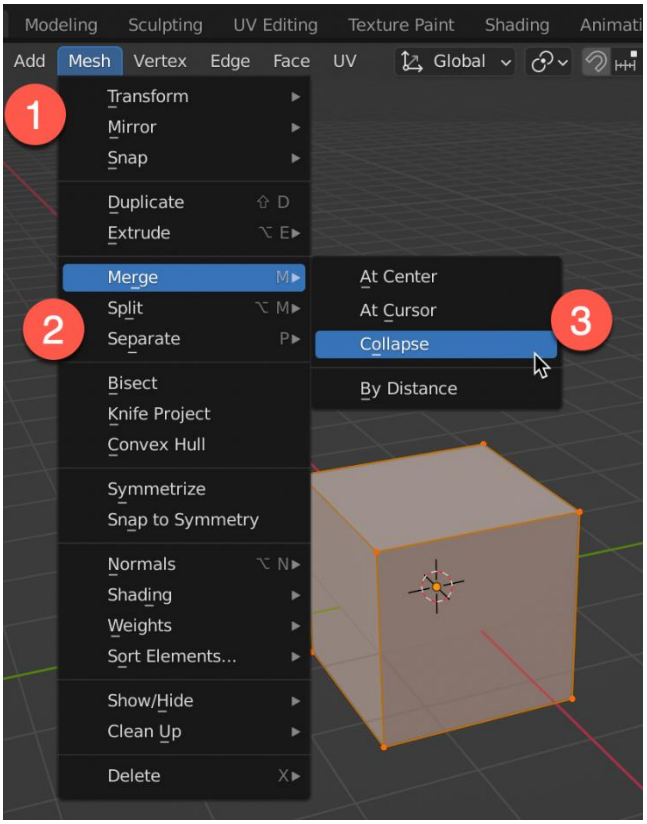
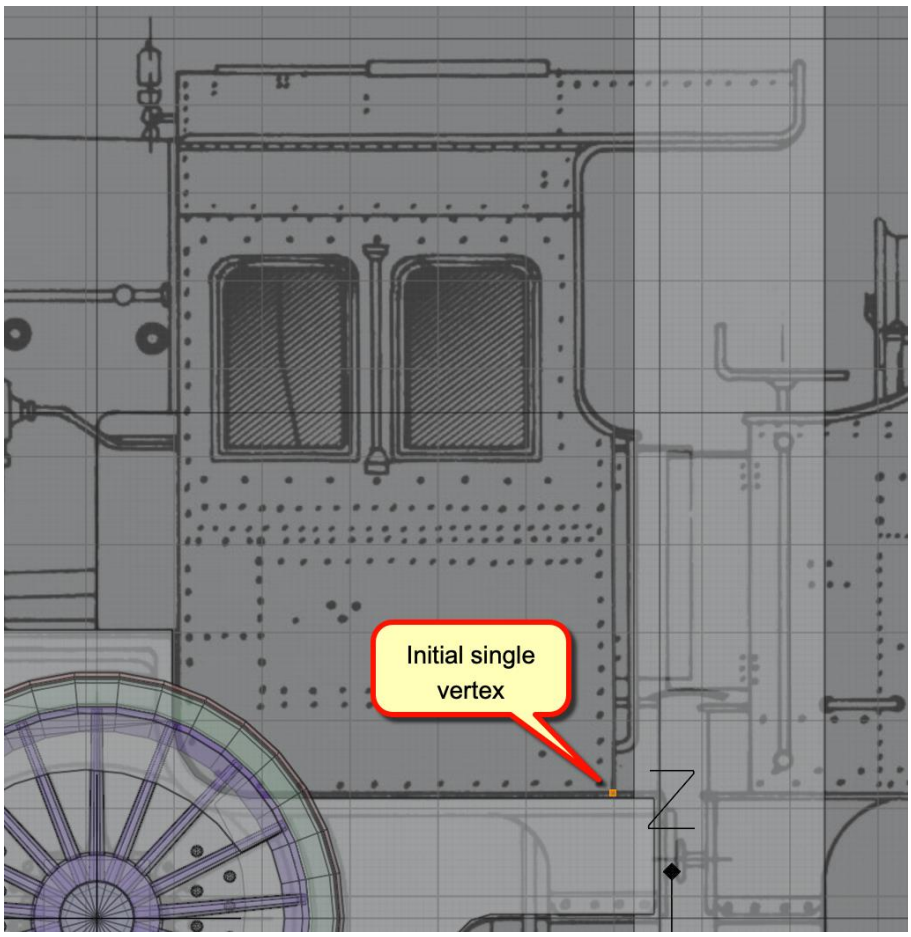



*Figure 4-1: Cube in Edit mode, all vertices selected*




Having created our single vertex, we can extrude it to add more points. Here I'm tracing the outline of a cab side, but this technique can be used for any flat parts and as we'll see, for profiles to create round objects or pipes.

Note that it is necessary to have the textures displayed and X-Ray mode on for this.

Move the vertex to a suitable position by pressing ***G*** and dragging.

*Figure 4-2: Adding the initial vertex*

Now extrude the vertex by choosing '***Vertex > Extrude Vertices***' in Edit Mode.

You can extrude the vertex in a particular direction by pressing ***X***, ***Y*** or ***Z***.

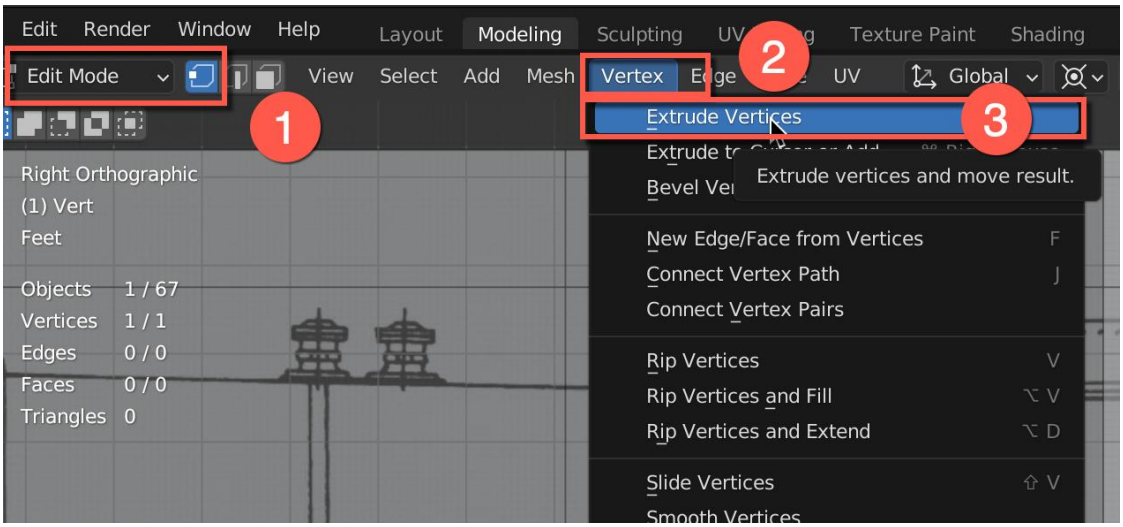
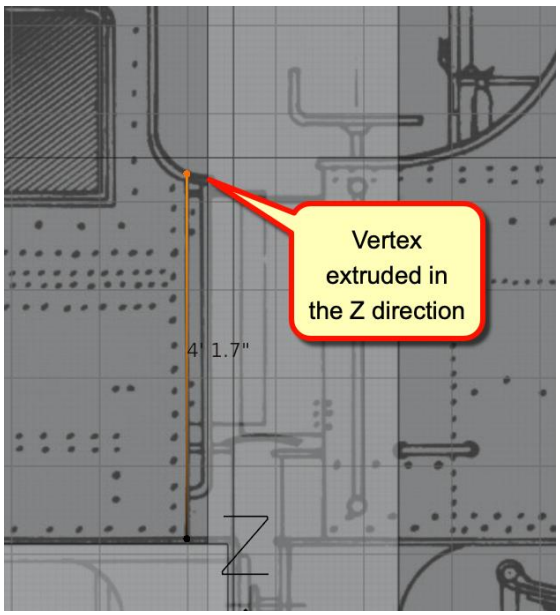



*Figure 4-3: Extruding Vertices*




Here I have extruded the vertex in the Z direction. Continue around the outline of the part to be created, ignoring any small radii to start with. Also, in this case, the cab side outline should include the beading as this will probably be normal mapped later and the cab side needs to be sized to include it.

*Figure 4-4: Second vertex created*

Here is the initial outline for the cab side, there are a few things to note. The outline is higher than it will finally be as we will trim it down once the cab front has been created (1).

Next, there is a small face at the rear of the cab side so that the beading can be added in the normal map later (2).

Rounded corner (fillets) will be added later (3).

Any openings such as windows will be left out at this stage and cut out later using the '***Knife Project***' tool (4). This is because when we create the faces for the object Blender does not handle holes in the mesh very well.

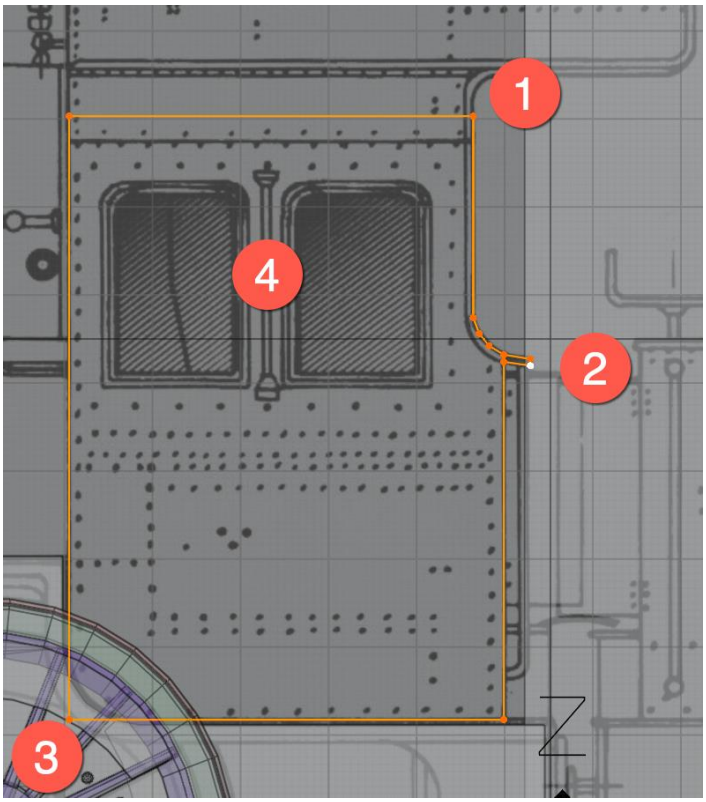



*Figure 4-5: Cab side initial outline*

Now select all the vertices in Edit Mode by pressing ***A*** while in Vertex Select Mode and then choose '***Mesh > Merge > Merge by Distance***' to make sure that there are no double vertices in the shape.

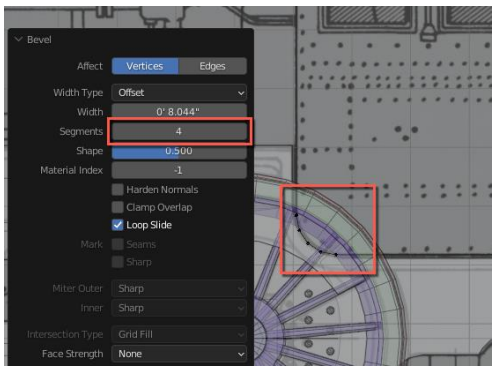



*Figure 4-6: Vertex bevel created*

Next, we'll create the fillet at the bottom left of the cab side using a bevel. Select the vertex and then choose '***Vertex > Bevel Vertices***'.

It's a good idea to use an even number of segments for a bevel, this is because it is then possible to delete every other vertex afterwards to reduce the poly count without affecting the start and end points of the bevel. You don't have to do this, but it does make the creation of reduced poly meshes later easier, so it is a good habit to get into.

To cut out the windows, create a new single vertex object, again ensuring that all the vertices are inside any beading present so that the beading can be baked into the normal map later. Any radii needed should be added after the basic shape has been created by bevelling the vertices as described above.

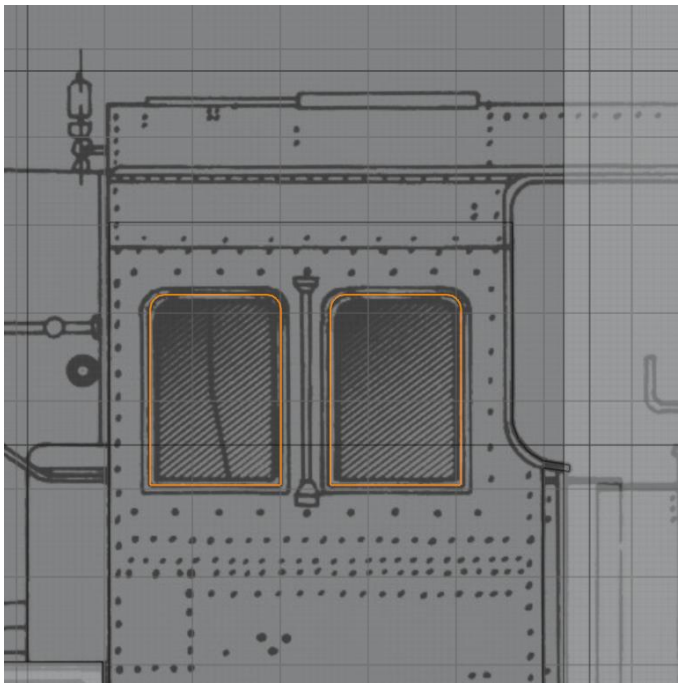


*Figure 4-7: New cutter object added for the window cut outs (here shown selected)*

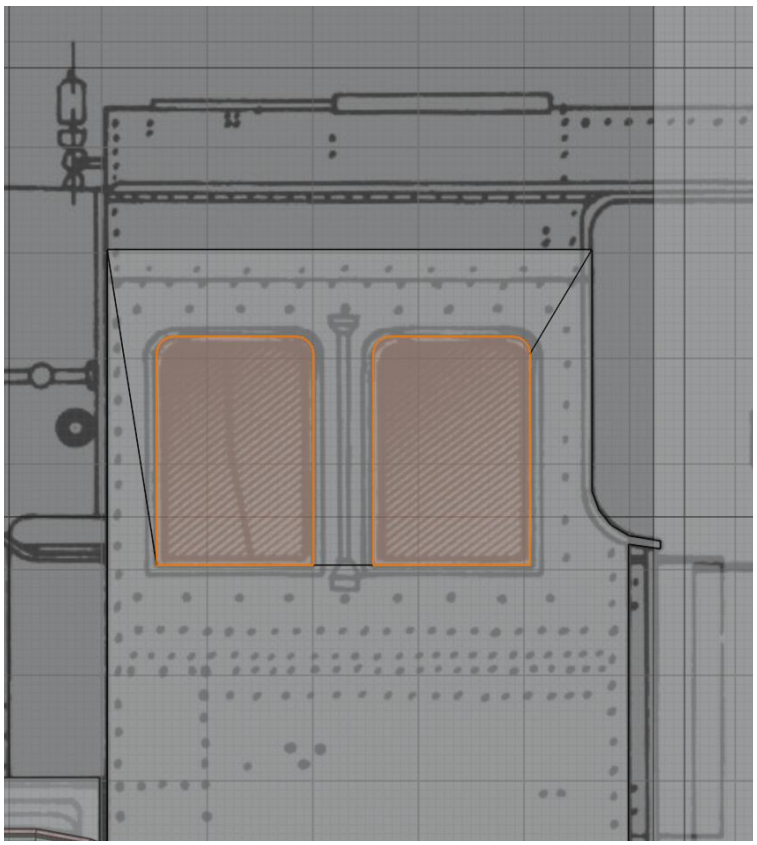


*Figure 4-8: Window cut outs created (shown here selected)*

We can now create the faces for the cab side by selecting the cab side object, entering Edit Mode, selecting all the vertices by pressing ***A***, and then creating the faces by pressing ***F***. Now select the cab side faces and, in the Outliner, select the cab window cutter object with ***CTRL LEFT CLICK.*** Finally, choose '***Mesh > Knife Projec***t' to cut out the windows. The faces in the window cut outs can then be deleted (or perhaps separated by choosing '***Mesh > Separate > Selection***' for use as the glazing if needed).

The cab side can now by moved into the correct X position by selecting it and moving it with ***GX*** and dragging, taking the position from the front or top views if you have them. If you know exactly how wide the cab should be you can just type the desired dimension in the Item Property field.

# 5. Mesh Creation – Round Parts

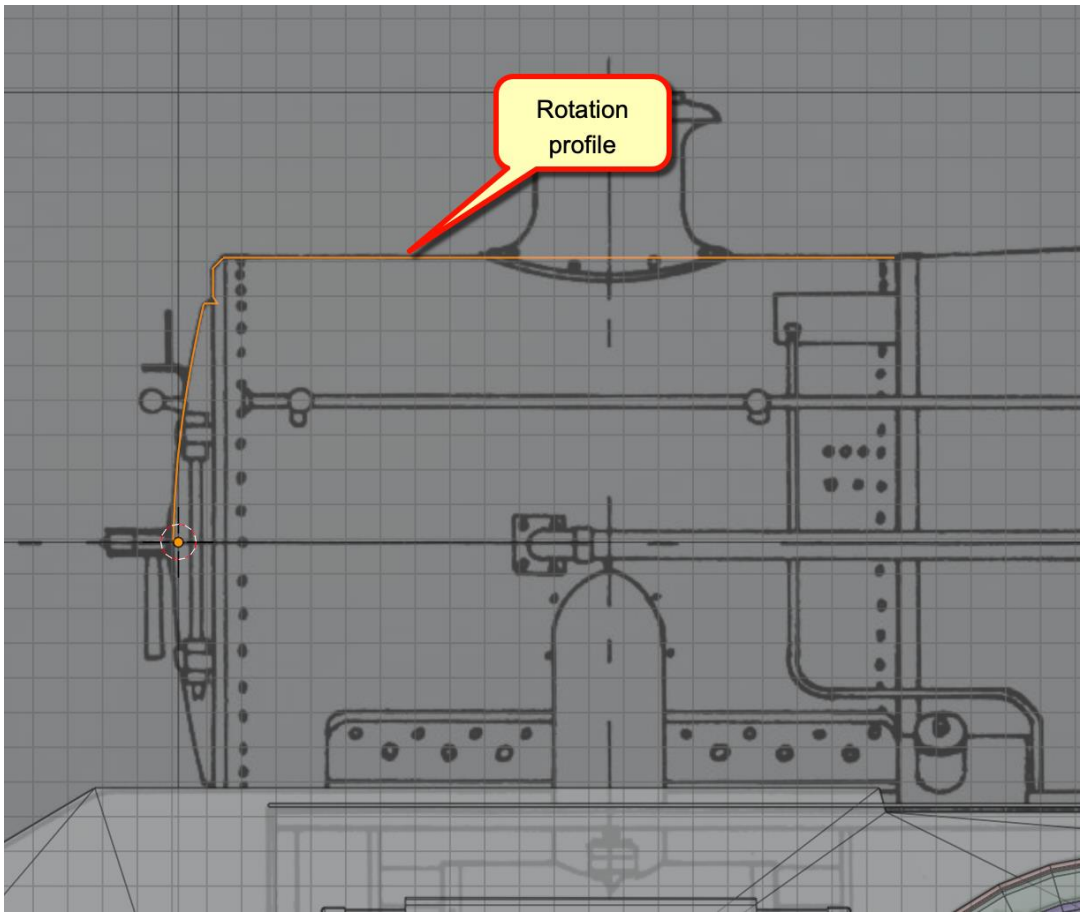



*Figure 4-1: Rotation profile for the smokebox*

These can be made using a 2D profile rotated around the 3D cursor. Here I've sketched the profile of the smokebox in a similar manner to creating the cab side and have set the 3D cursor to the centre of it (hopefully).

Now add a '***Screw***' modifier to the profile object in Object Mode using steps (1), (2) and (3) and the smokebox object will be created.

A few settings can be changed in the Properties Panel for the modifier after adding it with steps (4) to (7).

The '***Axis***' (4) is the direction of the centre of the object, '***Steps Viewport***' (5) is the number of segments in the rotation object, 'Merge' (6) will remove any double vertices and '***Smooth Shading***' (7) will shade the object smoothly which is usually what we want.

'***Calculate Order***' (7) will correct any faces that are facing in the wrong direction, i.e., if the object created looks inside out, then select this.

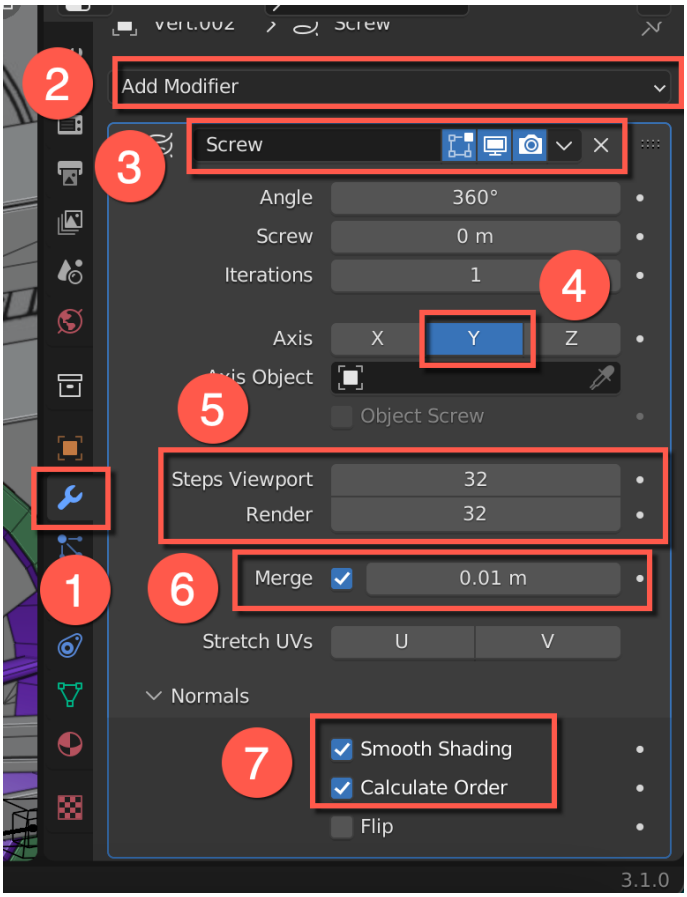



*Figure 5-2:Adding the Screw modifier and the options available*

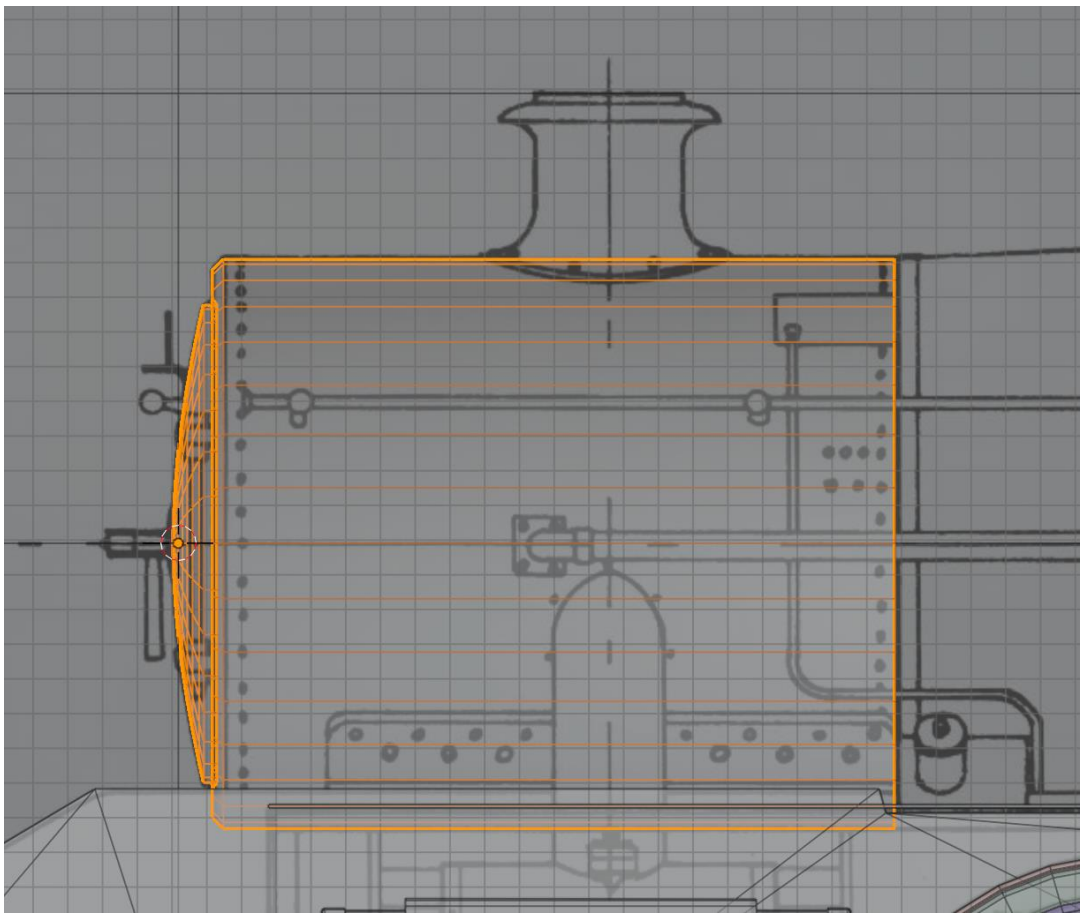


*Figure 5-3: Smokebox object created*

As an aside, although we are using the Screw modifier here to create a rotation object, its main purpose is to helixes such as springs. The number of turns is controlled by the '***Iterations***' field and the distance between the turns is controlled by the '***Screw***' field.

Have a play with this and see what you can do with it. I have used it for safety valve springs for example.

Try to keep this modifier active for as long as possible, it makes life easier if you need to modify the profile to add more detail or to adjust the number of segments later. As with bevels, try and keep the number of segments to an even number as this allows the possibility of removing every other edge loop to reduce the poly count.

Chimneys and Domes: It's worth only making a quarter of the object and then mirroring it though as the vertices at the base will need to be adjusted to fit neatly on the boiler or smokebox.

**[To be expanded…]**

# 6. Mesh Creation – Pipes, Handrails and Cables

I resisted changing to Blender 2.5 and up for a long time as a very important feature was missing until about v2.70. This is the ability to bevel a vertex which comes in very handy when making pipes and handrails.

If the pipe, handrail, or cable has bends in X, Y or Z then there are other possibilities, but this method works even if the pipe bends in some combination of these directions. The idea is to first sketch the centreline of the pipe with sharp corners and then to use the bevel tool to create the bends themselves. Finally, the pipe can be created by 'thickening' the line.

To illustrate this, I'll just create a pipe in a new blend file. Create a new object consisting of a single vertex as described in the Chapter ‘Mesh Creation – Flat Parts’.

Next, make sure you are in Vertex Edit mode.

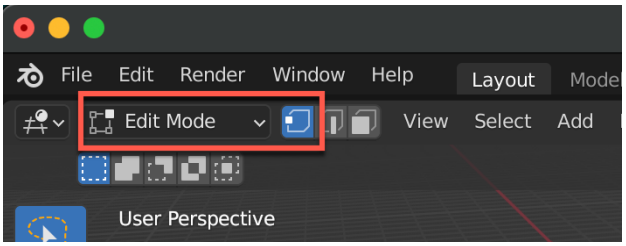
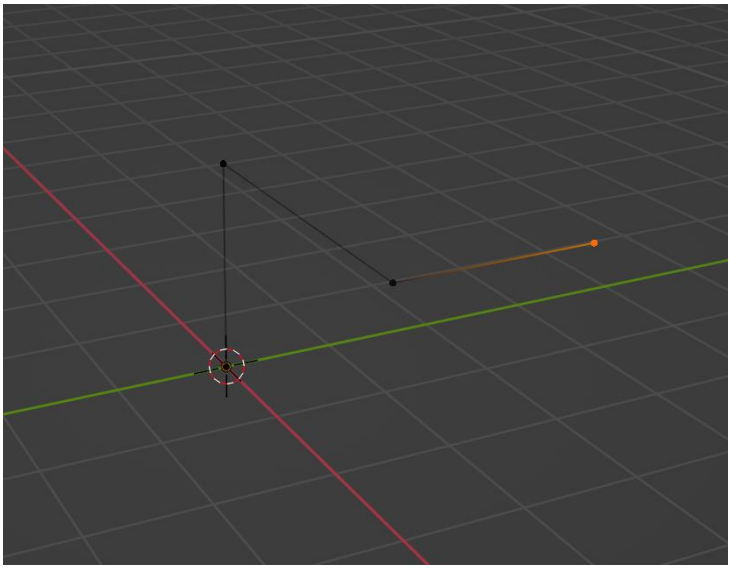



*Figure 6-1: Vertex edit mode*



Press ***E*** and then ***Z*** to extrude the vertex upwards by moving the mouse and then left clicking with the mouse to set the position. Repeat in the X and Y directions to produce the pipe centreline as shown here.

*Figure 6-2: Pipe centreline created*

Now select the vertices where the bends should appear.

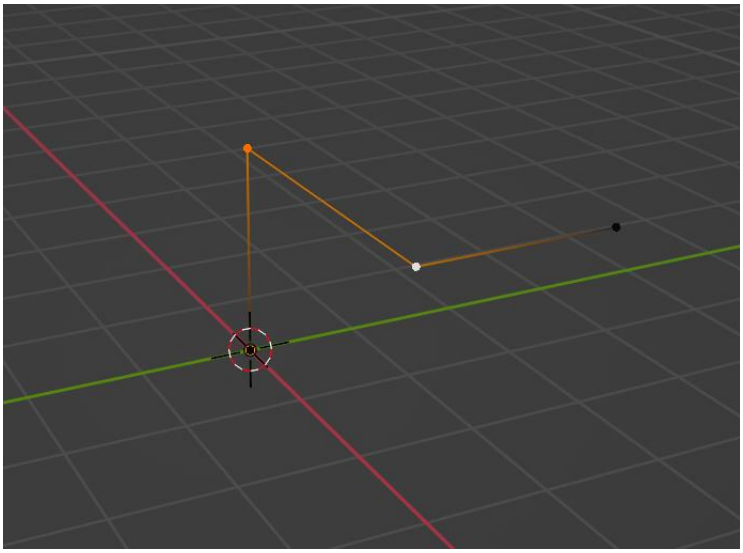


*Figure 6-3: Vertices selected*

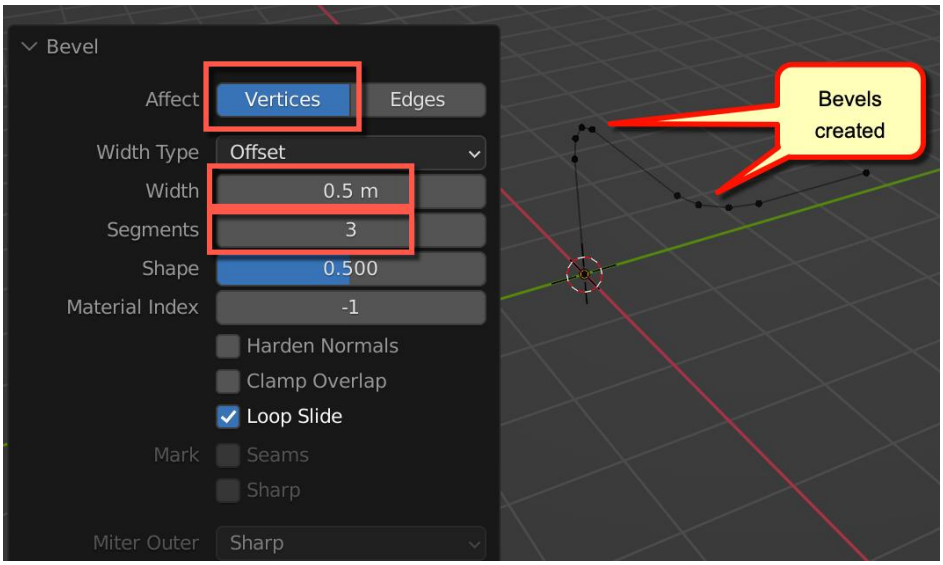



*Figure 6-4: The bevel options and the completed pipe centreline*

Then choose ***Vertex > Bevel vertices*** to start the bevel operation and click with the left mouse button near the selected vertices. In the bevel options dialog box (this appears collapsed in the bottom left of the 3D window, so you'll need to expand it to see the options) make sure that the '***Vertex***' button is selected and type in the bevel distance in the '***Width***' field. You can also drag with the left mouse button over this field to increase or decrease the value – this works with all the numeric input fields in Blender.

The '***Segments***' field allows you to increase the resolution of the bevels, a value of 1 would be a 45° chamfer. A value of 3 segments in the centreline is usually enough for most pipe bends.

Now we have the pipe centreline we need to convert this into a curve, so in Object mode select the centreline and choose ***Object > Convert > Curve***. A new tab will appear in the Properties window to allow the curve object to be modified. Note that this only appears when a curve object is selected.

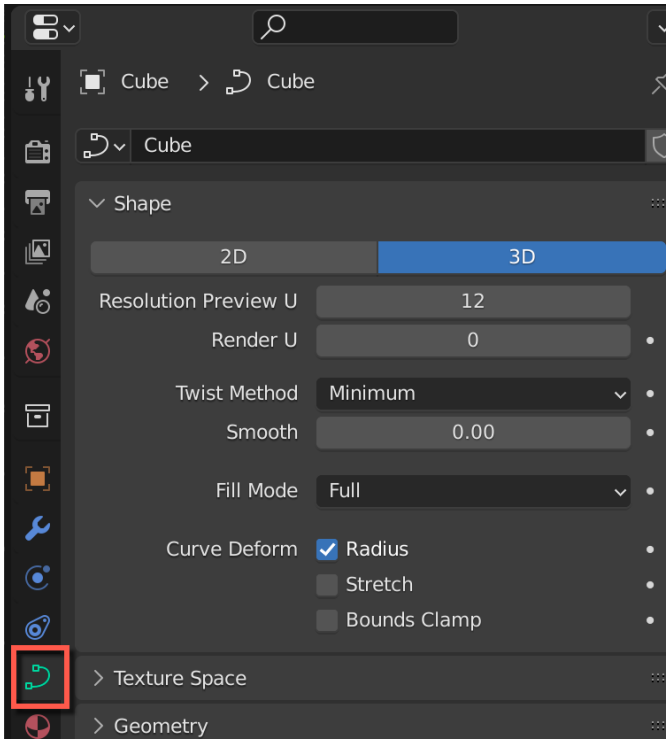



*Figure 6-5: Curve tab in the Properties window*

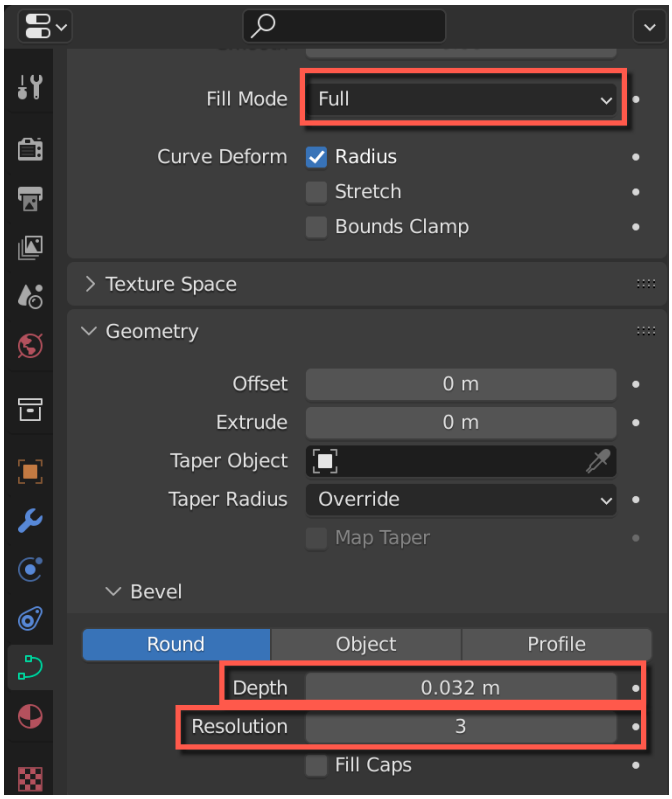



*Figure 6-6: Options for a Curve object in the Properties window*

Change the Options in the Geometry section of the properties window as shown here. For '***Fill***' you can choose '***Half***' or '***Full***', '***Full***' being a 360° pipe, '***Half***' is only 180°.

The '***Depth***' value is the radius of the resulting pipe and '***Resolution***' defines how many segments will be created per segment of the centreline. 1 is 6 sided, 2 is 8 sided and so on. Depending on the size of the pipe you might want to adjust this value. For reference a typical handrail would be about 32mm diameter.

The Pipe object has now been created – at this stage it is still possible to change the properties to adjust the diameter and the resolution of the pipe.

Blender v2.49b could also do this, but it didn't work as well. Often it would twist some segments of the pipe which needed sorting out manually.

To convert the Curve object back to a mesh, choose '***Object > Convert > Mesh***' (with the pipe object selected of course...).

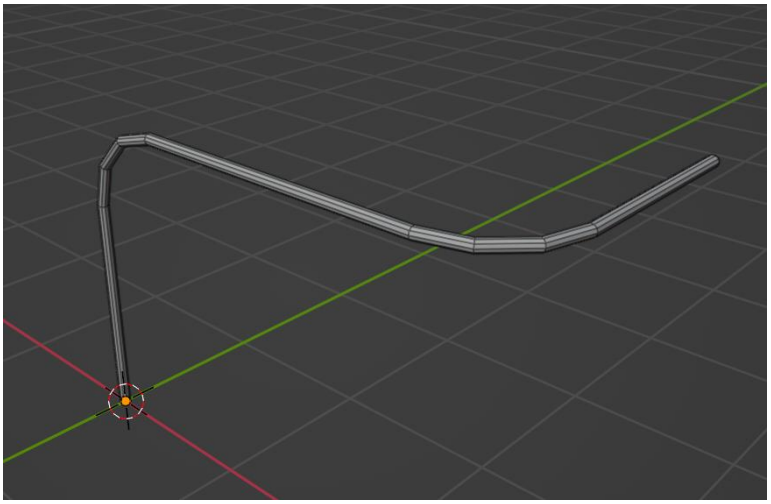
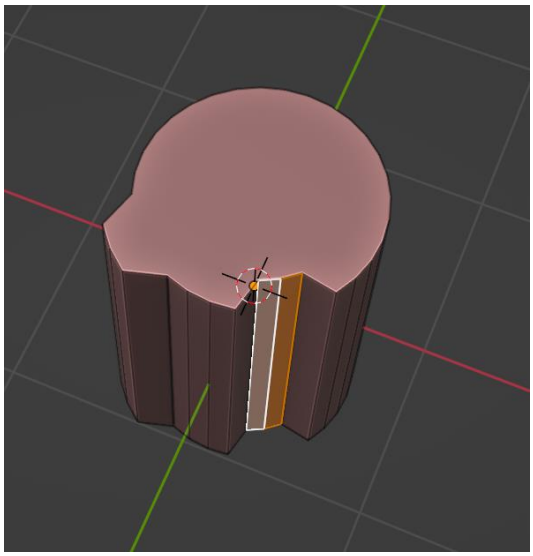


*Figure 6-7: Pipe object created*

Once you have converted the Curve to a Mesh you lose the ability to edit the diameter of the pipe, but there is a way to do this without creating the curve again. Select all the faces of the pipe and then press '***ALT S***' (on a Mac '***Option S***') to shrink or fatten the pipe – this will of course work with any mesh object which can be very handy.



And, veering completely off topic, this also allows you to create shapes like this, which would be quite tricky by other means.

This operation allows you to scale faces along the normal directions.

*Figure 6-8: Shrinking and fattening faces with ALT S/Option S*

## *Using the Tube Tool Add-on*

The method described above is simple and quick but creates pipes etc that are very regular with clearly defined bends. In real life, especially in steam loco cabs, the pipe routing was much more free flowing as the aim was to reduce the amount of copper tubing needed, as this was/is expensive. Luckily, there is another add-on that will create such a pipe, called ***TubeTool***.

Here is a link to a short video explaining it and where to download it:
https://www.youtube.com/watch?v=-Cr2e_gkee0

Basically, it creates a free form pipe curve between two faces of the same mesh which can then be edited afterwards to add more control points, or to increase/decrease the number

of segments. The guy who created the video was a bit confused as to why the add-on was creating tapered pipes, but this is because he forgot to check the box to make the pipe diameter constant. By default, the pipe diameter at each end is based upon the size of the faces used to generate it.

Let's have a look at how you can use this to add a pipe to a cab brake valve. Start by creating two planes at the desired ends of the pipe. Join these two planes together.

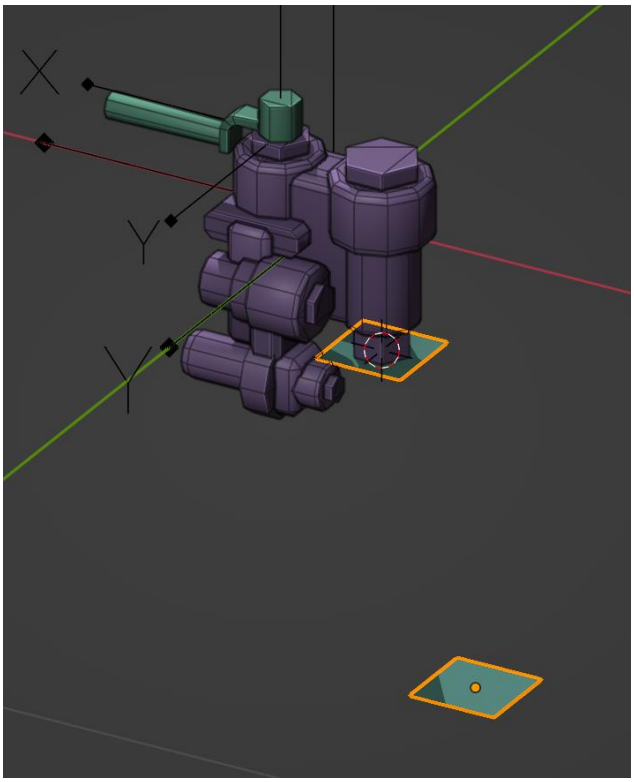
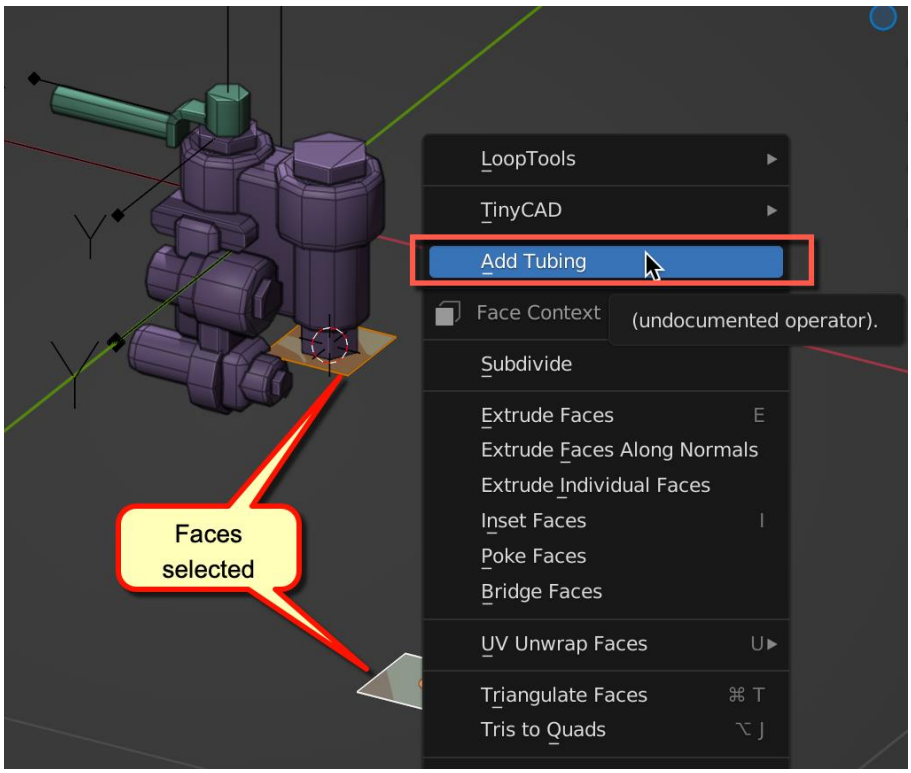


*Figure 6-9: Adding the planes*

Select the upper face of the joined planes and flip it by choosing '***Mesh > Normals > Flip***' in Edit Mode. This is to ensure that the tube or pipe will go down from the top face which is what we want. Next, in Object Mode select the joined plane object and choose '***Object > Apply > Scale***'. This is important as the scaling of the plane object will affect the scaling of the tube or pipe. We want the scale to be 1 so that we can more easily set the size of the tube or pipe.




Switch to Edit Mode and select both faces of the joined plane object and then right click and choose '***Add Tubing***' from the menu.

*Figure 6-10: Adding the tube using the selected faces*

The tube will be created but doesn't look too good yet.

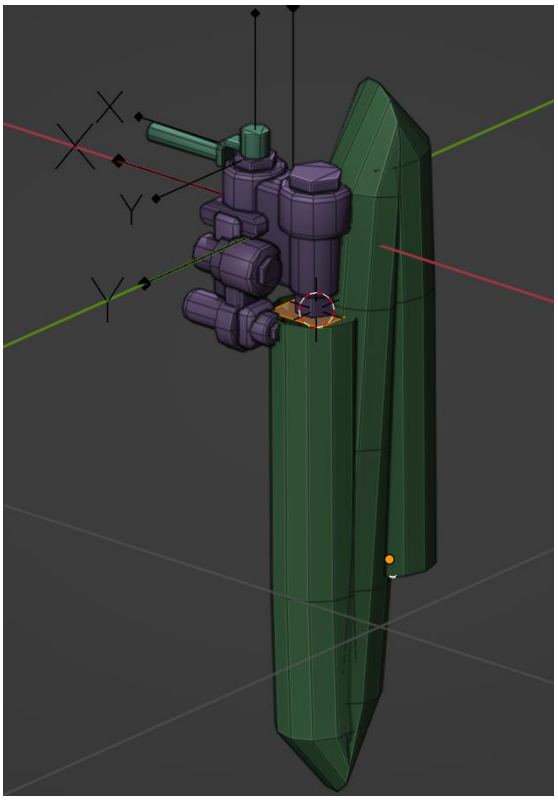
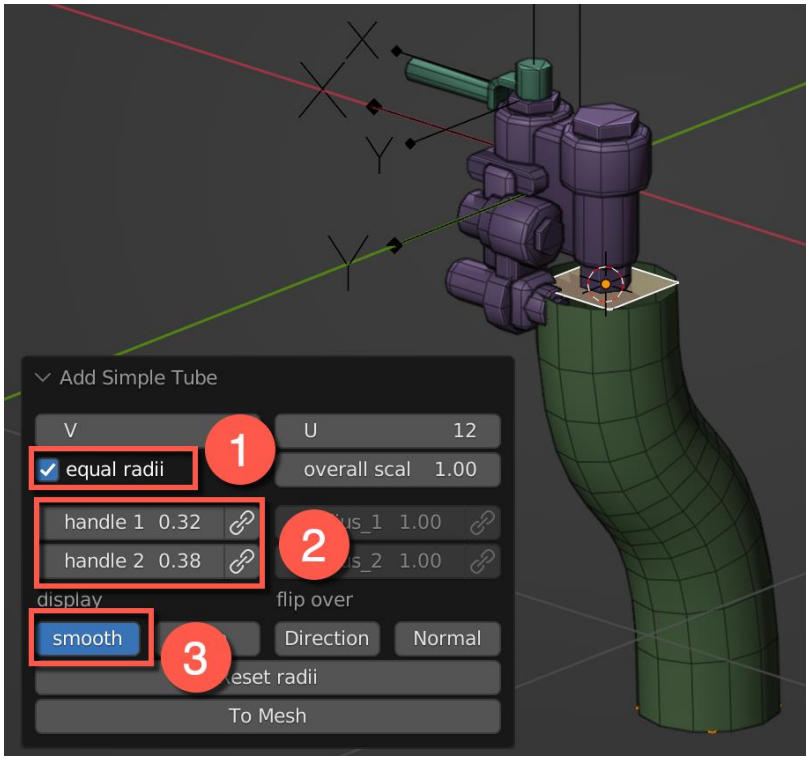


*Figure 6-11: Tube created*




To fix this, change the values for the handles in the dialog box as shown.

*Figure 6-12: Adjusting the geometry of. the tube*

Now in Object mode select the tube and in the Object Data Properties and change the diameter of the tube to the desired diameter in the '***Geometry > Bevel***' section.

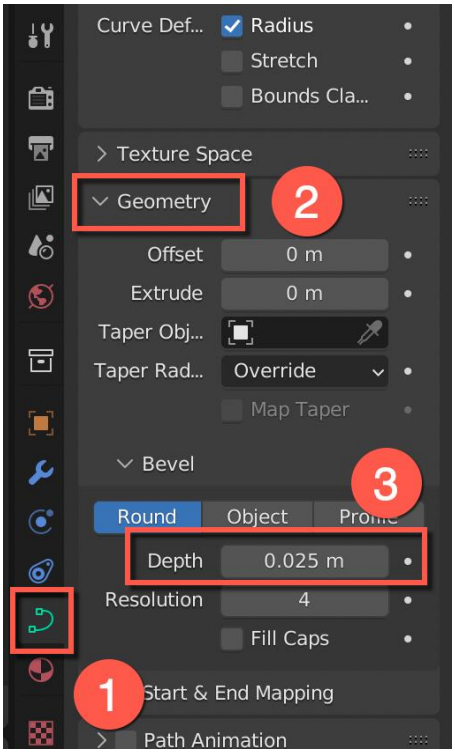
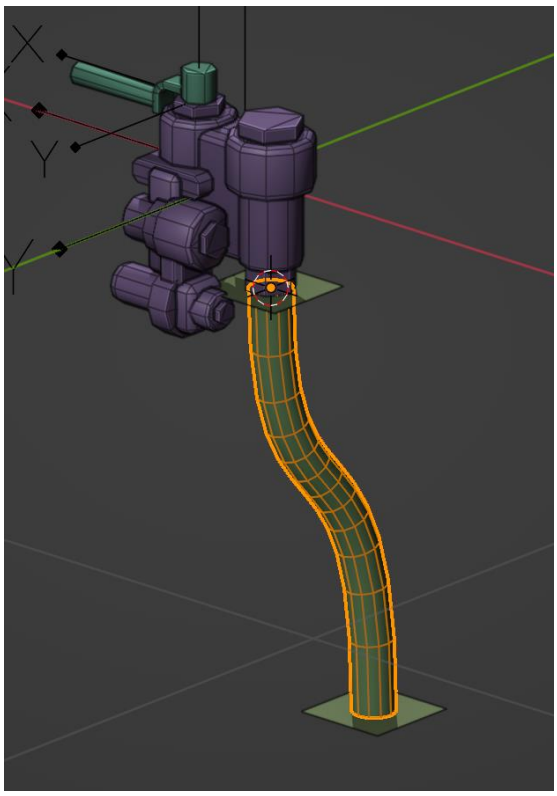



*Figure 6-13: Setting. the diameter of the tube*



Here is the result. The joined plane object can now be deleted.

*Figure 6-14: Tube resized*

As the tube is a curve object, you can further edit it to fine tune the route. For example, selecting the tube and entering Edit Mode allows you change the length of the control handles at the ends of the tube, or to rotate them. You can also subdivide the curve to add more control points or extrude the curve.

Of course, it is still possible to change the diameter of the tube if needed.

## *Creating or Modifying Pipes Using Bridge Edge Loops*

A useful way or creating or modifying pipe runs is to use the '***Bridge Edge Loops***' tool. To use this, you select edge loops that belong to the same mesh object…

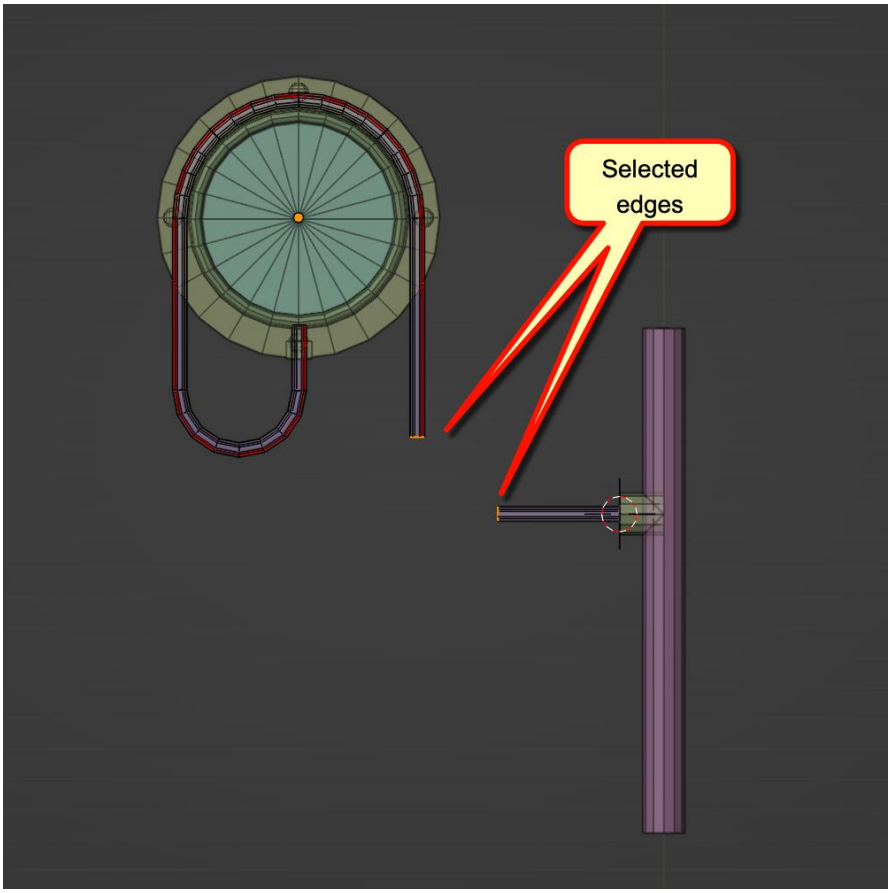
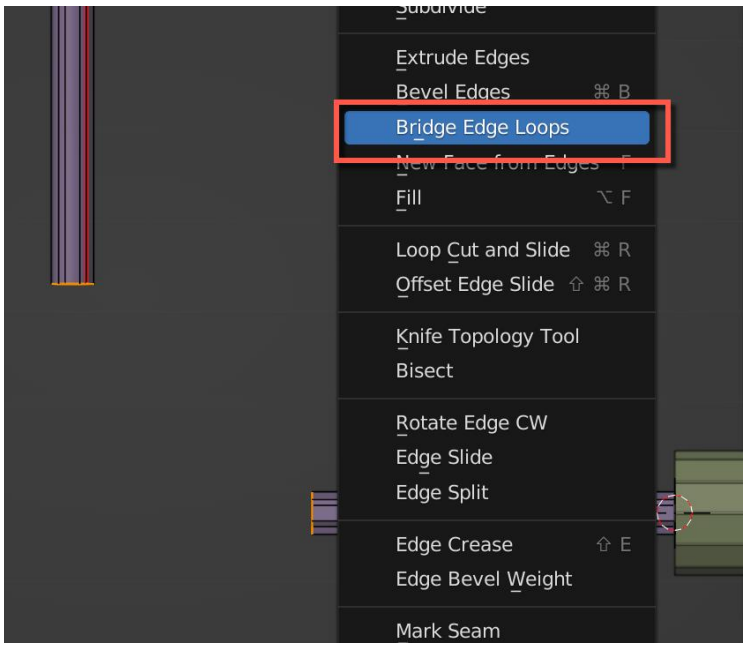



*Figure 6-15: Selecting the edge loops to bridge*




and then choose '***Bridge Edge Loops***' from the right click menu.

*Figure 6-6-165: The menu*

After selecting '***Bridge Edge Loops***' the pipe segment will be created. The dialog box allows you to specify how many segments the new pipe should have and how it should blend with the existing geometry.

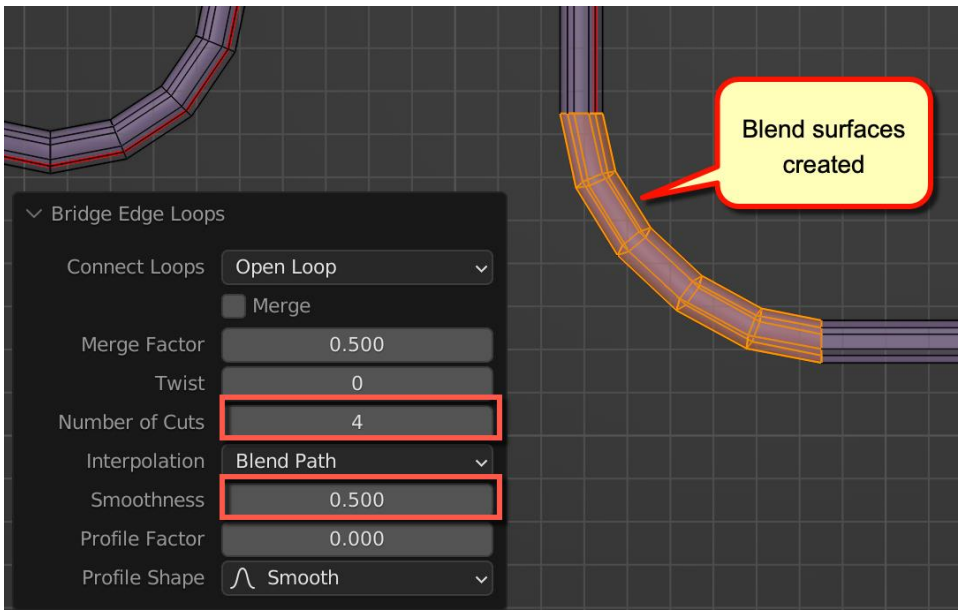
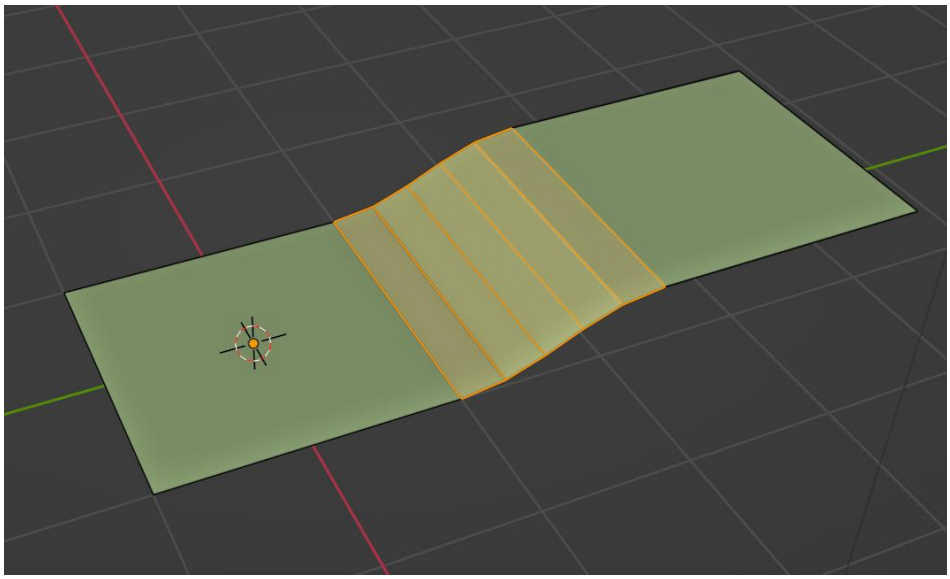



*Figure 16-16: Blend surfaces created, controlling the geometry*

This is especially handy for creating smooth pipe segments quickly. In fact, a good way of creating pipes is just to create short sections at the start and end of the pipe and any other sections that need to be in a particular location (for example where the pipe needs to pass behind another one) and then to use '***Bridge Edge Loops***' to fill in the missing bits.



Finally, note that this works for any edge loops, not just pipes

*Figure 6-17: Bridge Edge Loops between the edges of plane joined together as one object*

.

# 7. Mesh Creation – Bulk Loads

This has its own section as bulk loads (usually coal loads in bunkers or tenders) need a few techniques that we haven't covered yet.

I'm going to describe a method of creating a simple coal load object that has a fixed shape. It is also possible to deform the mesh as the load is animated, but this is usually not required for tenders and bunkers as it is almost always possible to hide the load when the tender or bunker is 'empty' even if the shape of the load mesh doesn't change.

The first step is to create a Plane Object with '***Add > Mesh > Plane***' in a suitable position.

Select the edges and snap them to the existing geometry to size the Plane Object to fit the available space.

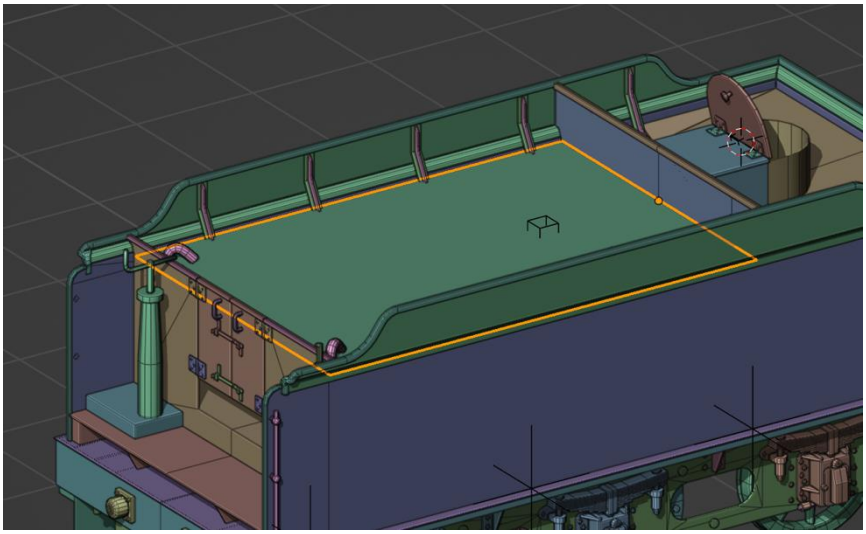
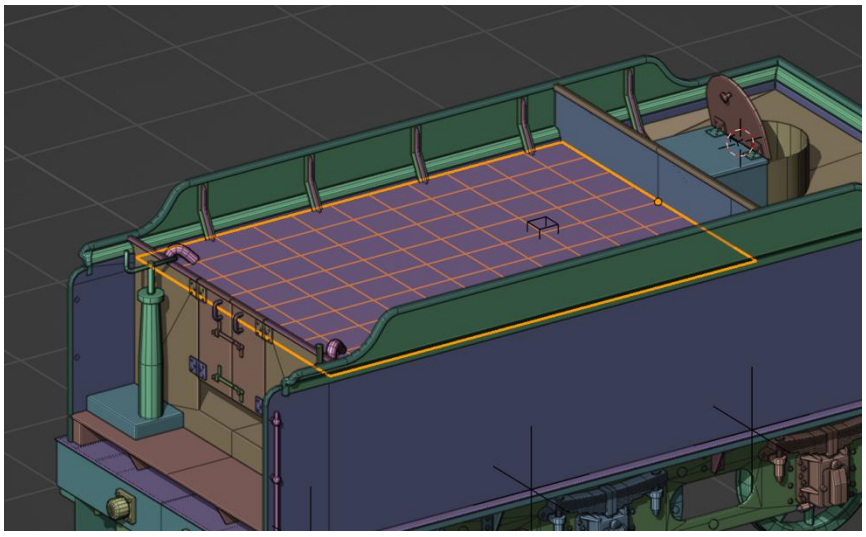


*Figure 7-1: Initial Plane Object added and resized*



Then subdivide the plane by selecting the faces and pressing '***CTRL-R***', rotating the mouse scroll wheel to increase or decrease the number of cuts. Do this in both directions to divide up the Plane Object so that there are more vertices that can be moved to create an uneven surface.

*Figure 7-2: Plane Object subdivided*

Next, switch to Edit Mode and in Vertex Select Mode select all the vertices in the Plane Object. Then choose '***Mesh > Transform > Randomize***'. In the dialog box choose about 0.1 for the '***Amount***' and set '***Normal***' to 1.000.

The '***Amount***' value is the maximum distance the vertices will be moved, and '***Normal***' 1.000 ensures that the vertices are only moved in the Z direction.

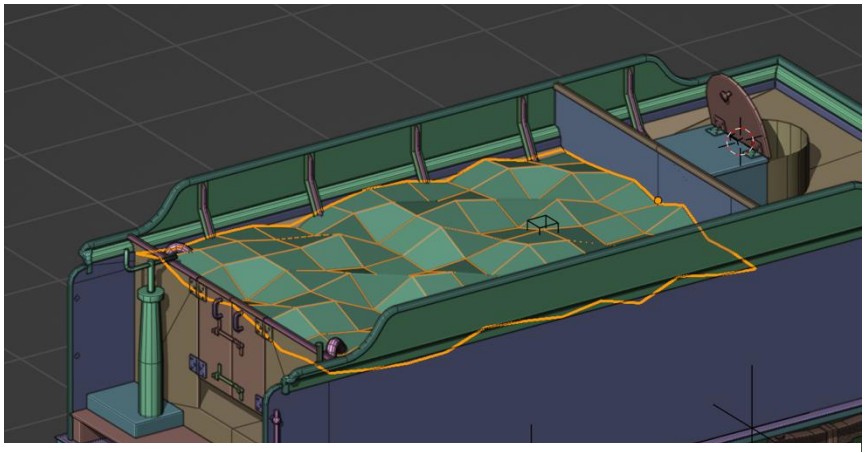


*Figure 7-3: Vertices in the Plane Object randomised in the Z direction*

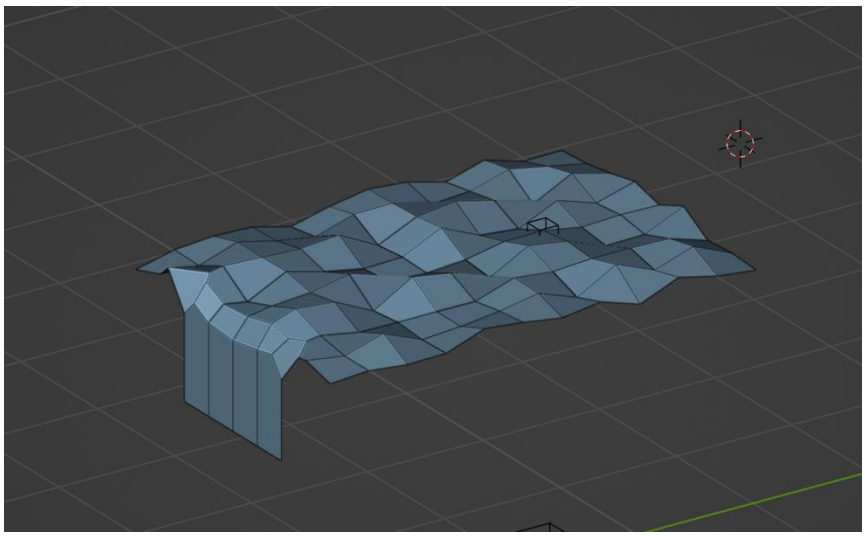


*Figure 7-4: Coal load mesh with blocker for the coal hole added*

Here is the (almost) final version of the load mesh for this tender after tweaking some of the vertices manually by using '***GZ***' and dragging to adjust their positions, plus adding a portion at the front to block off the coal hole.

This front portion will need to be split and animated to fold back when the tender is almost empty as otherwise part of it would be visible under the tender. Folding it back will allow this part of the load to be hidden in the well tank under the tender floor.

It might be worth considering adding individual pieces of coal as meshes to the coal hole and shovelling plate area, bearing in mind that this area can be inspected at very close range when you are driving the loco from inside the cab.

Another technique that you can use to enhance coal loads is to use proportional editing which allows you to select one vertex and to move it and affect the surrounding vertices without having them selected. This allows you to create a mound or a depression in the coal load.

To do this, press '***O***' or select this icon:

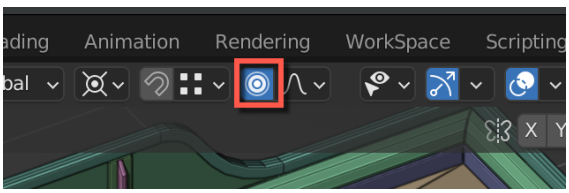



*Figure 7-5: The proportional editing icon*

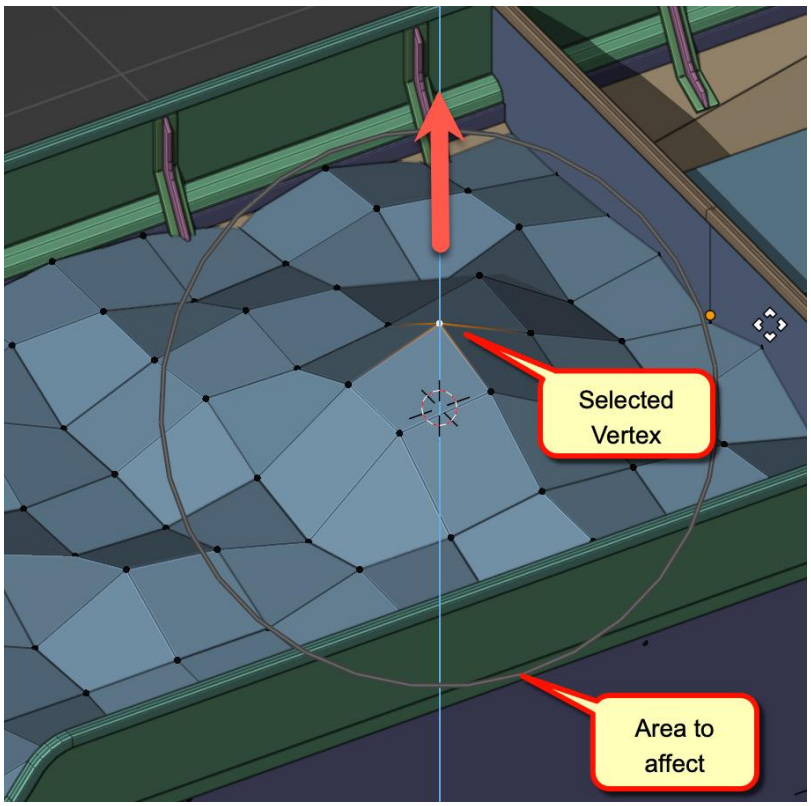



*Figure 7-6: Using proportional editing to create a mound in the coal load mesh*

Now in Edit Mode and Vertex Select Mode select a vertex and move it with '***GZ***' and dragging.

The 3D cursor should be positioned at the vertex you are moving as this will be the centre of the affected area.

'***CTRL – Mouse Wheel***' can be used to increase or decrease the affected area on Windows computers, on MAC OS laptops it is adjusted by using two fingers moving up or down on the trackpad (confusing, isn't it?).

**Make sure that you turn off proportional editing as soon as you are finished as it can cause a lot of confusion later as you wonder why vertices that are not selected are being moved or scaled…**

The animation of the coal load will be covered in a later chapter.

## 8. Checking the Mesh for Flipped Normals

A common problem when creating meshes is flipped normals, as they are sometimes difficult to detect until you bake an AO map, and you can't work out why parts of the map are coming out black when they should be light – the reason is usually that the normal are facing 'inside' the mesh instead of 'outside'.

Before we start the UV mapping of the mesh it is a good idea to check this - luckily, Blender now includes a tool for graphically verifying that the normal are pointing in the expected direction.

To do this we need to be in Object select mode, and the Viewport shading needs to be set to solid. Lastly, the check box '***Backface Culling***' needs to be deselected, as flipped faces need to be invisible.

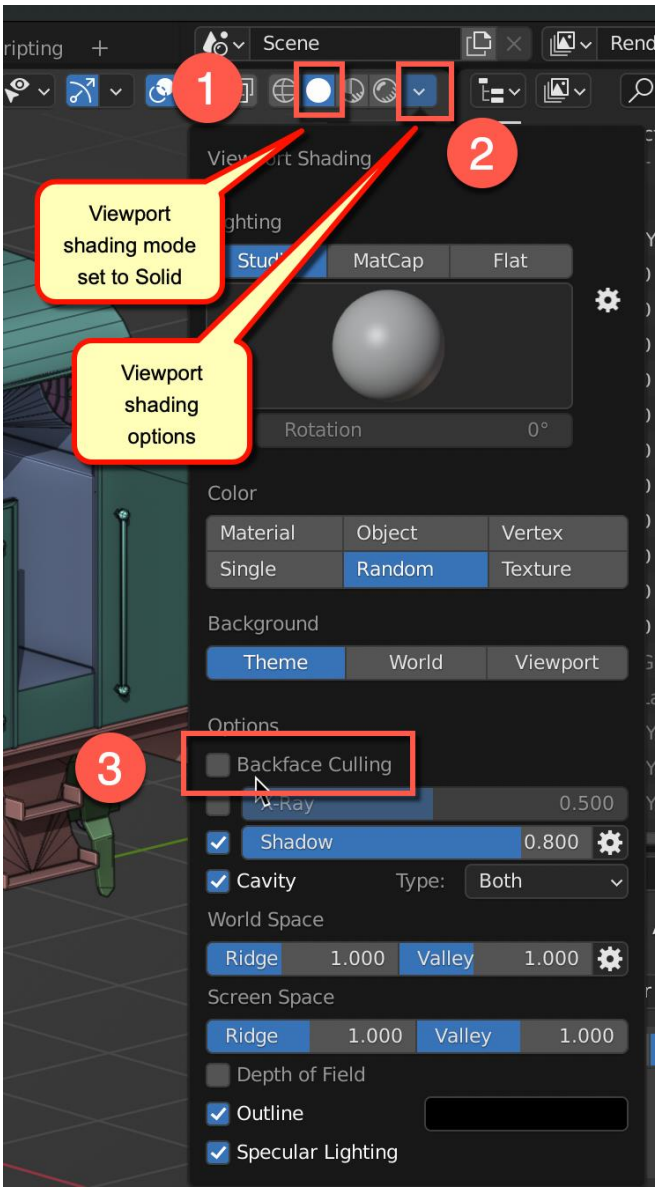
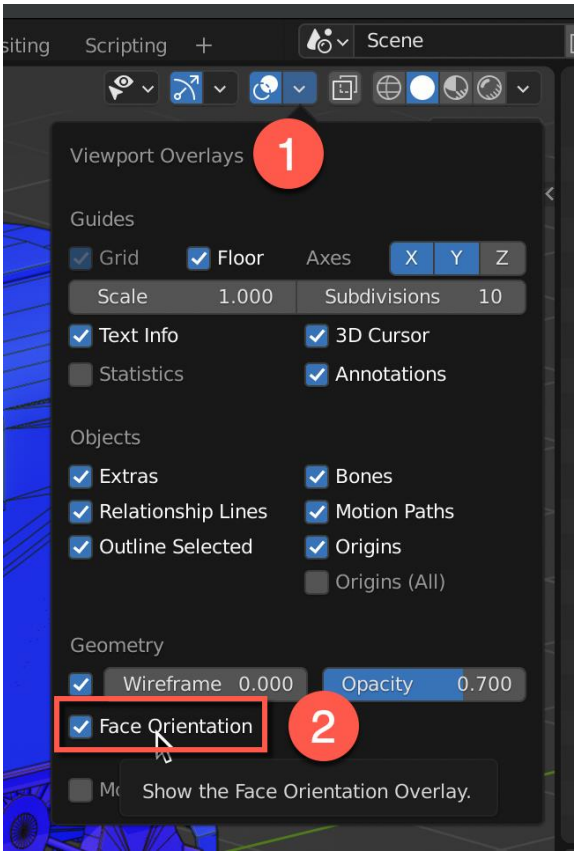



*Figure 8-1: Setting up the Viewport for the face orientation display*




Select the check box for '***Face Orientation***' as shown in the '***Viewport Overlays***' dialog, the faces will turn blue if they are facing towards the camera and red if they are facing away from it.

This allows you to easily spot any missing or flipped faces in the model by rotation it around and looking at it from the normal in-game viewing angles. If you see red faces they are either missing or they are facing in the wrong direction.

*Figure 8-2: Turning on the face orientation display*

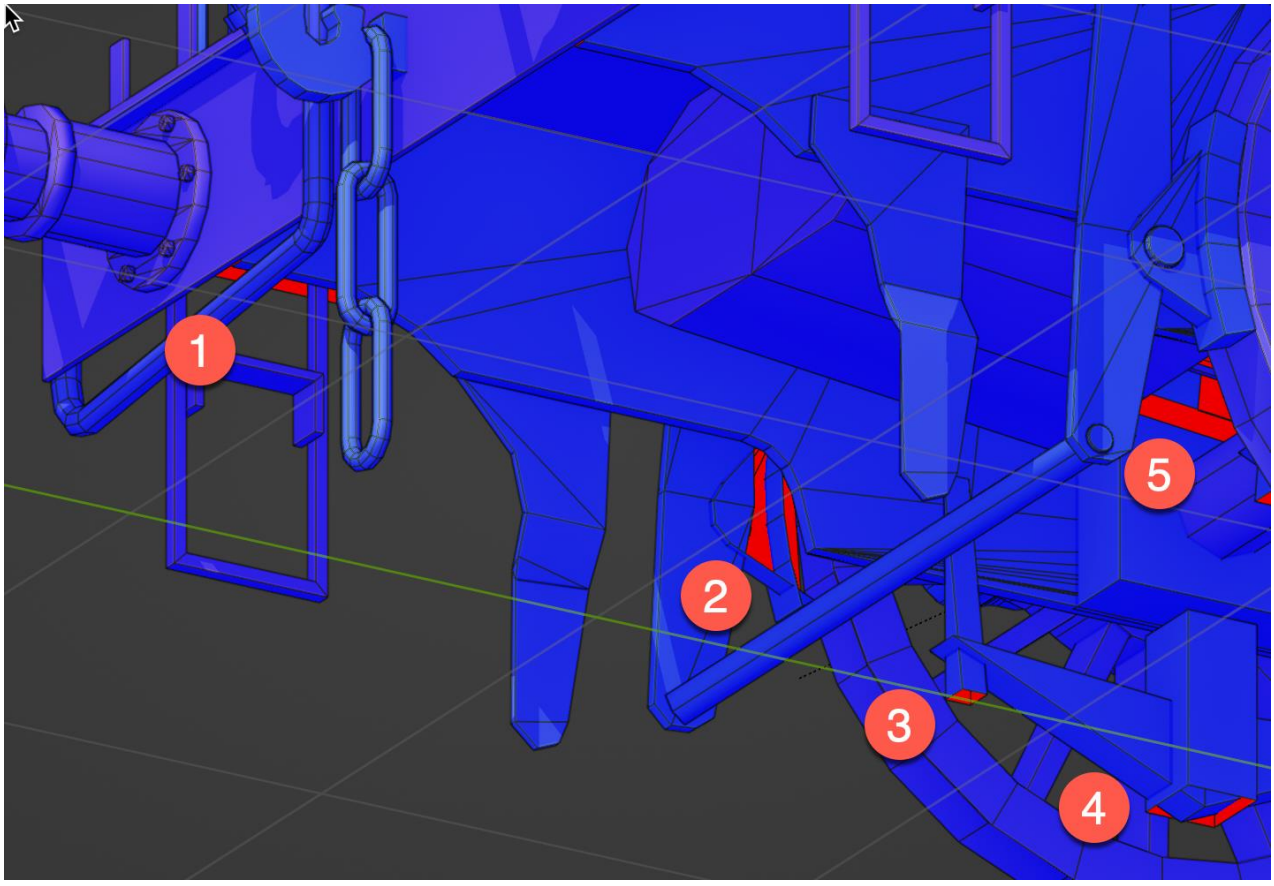



*Figure 8-3: Result of turning on Face Orientation*

In this screenshot we can see 5 areas that might need attention before we start the animation and UV mapping of the mesh.

1. The valences are modelled as single sided faces, in this case though they are very hard to see. It's hardly worth creating the missing faces and using up space on the UV map for these
2. The rear faces of the brake blocks are missing, these are too visible to leave out, at least on the highest LOD mesh and need to be added. They can take up very little space on the UV map and can all share the UV coordinates
3. The downward facing faces on the spring hangers have been left out, these will not be visible
4. Ditto for the spring buckle
5. Ditto for the inside motion components

It is obviously better to be sure that all the needed faces are present in the mesh before proceeding to the UV mapping step.

# 9. Animation

## *Animating Wheelsets*

If you have downloaded any of the wheelset library assets from my 'BitsBox' page, then they already include an Empty that has been animated over 30 frames. Also, there is a separate library file that includes pre-animated Empties and Bones for valve gear, so I'm going to recommend downloading this. Once you have an animated wheelset Empty available in your library you just need to copy it into your latest project and you are done.

**[To be expanded…]**

## *Animating Connecting Rods and Crossheads (Including Inclined Cylinders)*

To animate the connecting rods and crossheads I use a technique that I found here: https://blenderartists.org/t/piston-animation-with-bones/461537 which also allows for inclined cylinders and uses an armature system to produce perfect results quickly. Now, although I know I followed and slightly simplified this idea, I am not sure what exactly I did (it was many years ago now).

Not that this really matters as once you have the armature set up once you can just merge it into any new model you are making.

Please download the partially finished model of a Hawthorn Leslie 14in 0-4-0ST from the download page on my Web Site here to see how this is set up.

To animate the coupling rods, it is necessary to create an Empty at the centre of one of the crankpins, which one you pick is not important, but I prefer to use one that does not also carry the connecting rods, usually the front one.

To do this, set the 3D Cursor to the centre of the crankpin and then choose '***Add > Empty > Plain Axes***'.

You can choose from any of the available shapes for added Empties, including a Cube, but this looks like a Lattice Object and could lead to confusion, so I prefer to use the Plain Axes.

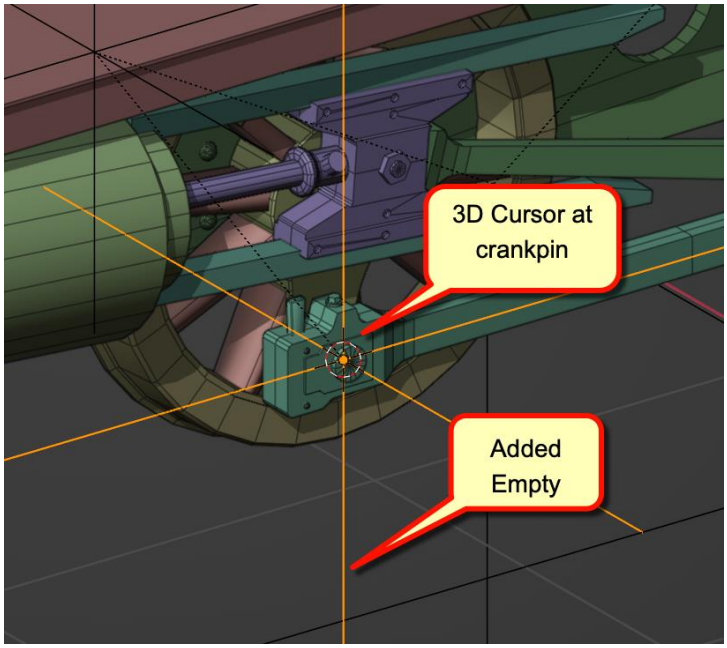



*Figure 9-1: Adding the Empty for the left hand coupling rod*

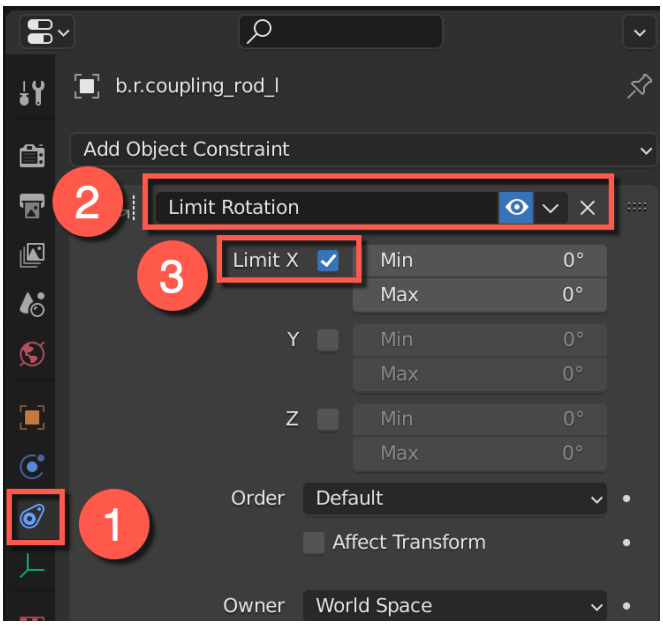



*Figure 9-2: Locking the X rotation of the crankpin Empty*

This Empty must then be parented to the Empty which is animating the wheelset which has the crankpin that we are using, in this case the front wheel. The coupling rod can then be parented to the crankpin Empty. The last step is to lock the rotation of the crankpin Empty in the X axis.

The coupling rod will now stay horizontal while rotating around the wheel centre which is exactly what we need.

To animate the connecting rods, we need to create two more Empties, one at the centre of the big end of the rod and one at the little end. I like to create these 2m away from the centreline of the loco so that the animation bones are not in the same X position as the rod. This just makes it easier to see them.

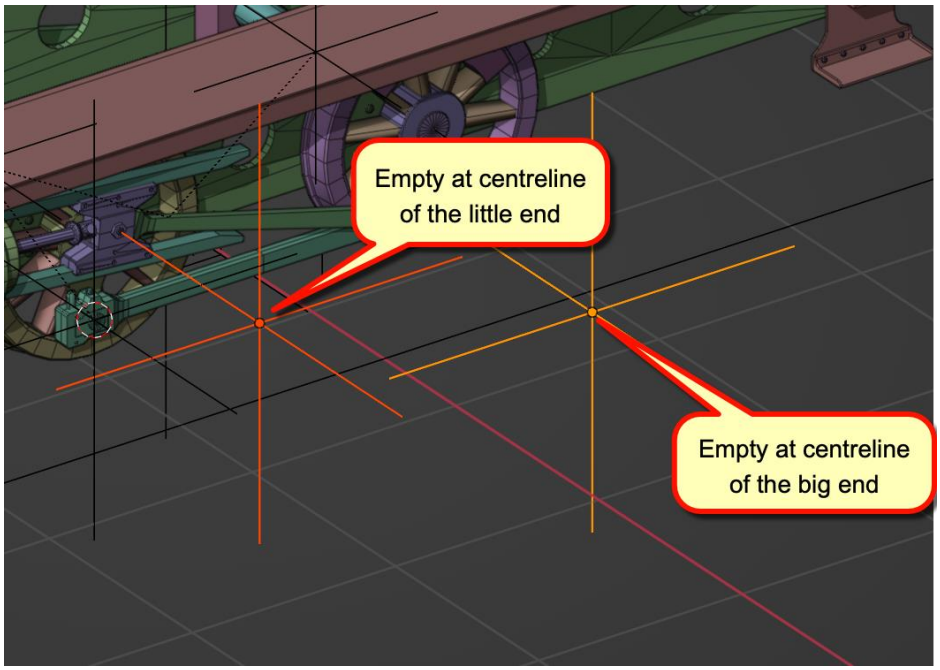



*Figure 9-3: Additional Empties created at the positions of the two ends of the connecting rod at X = 2m*

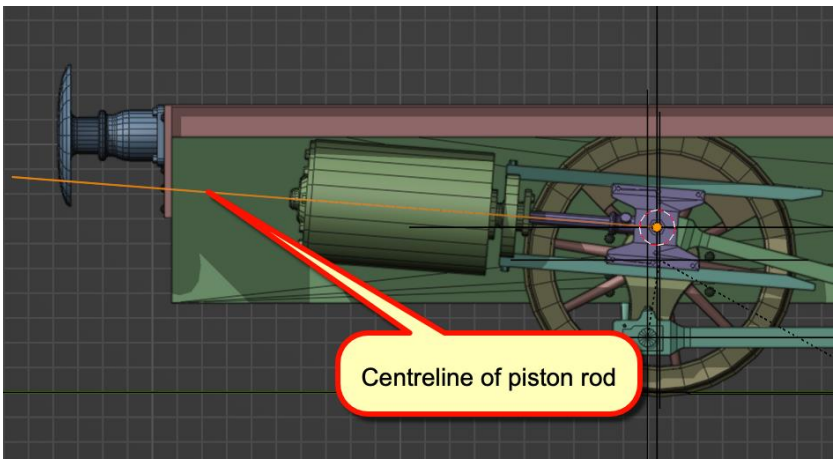



*Figure 9-4: Object at centreline of piston rod added*

Now we need a construction line that lies at the centre of the piston rod. The easiest way to create this is to insert an Object consisting of a single vertex positioned at the origin of the Empty at the little end of the connecting rod and then to extrude this in the -Y direction. Finally, this line should be rotated in the X axis by the amount of the inclination of the cylinders. Hopefully you made a note of what this should be when modelling the cylinders, slidebars and crossheads… If you didn't, it is easy enough to eyeball it based on the geometry that is already there.

Next, merge in the Bone System that can be downloaded in the '***Animations***' zip file from my Website from this link. You want the '***Bone System Connecting Rods***' Blend file.

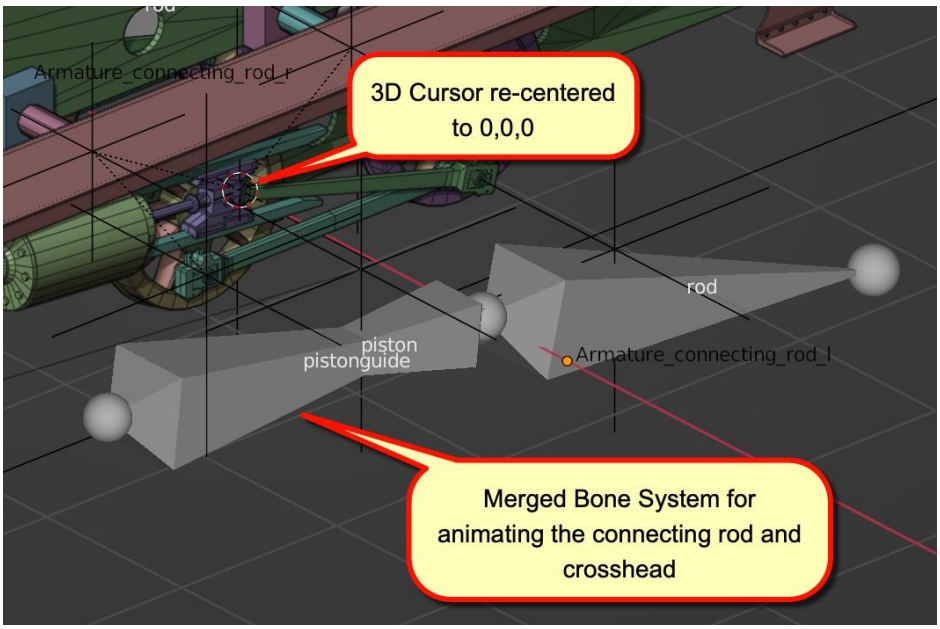
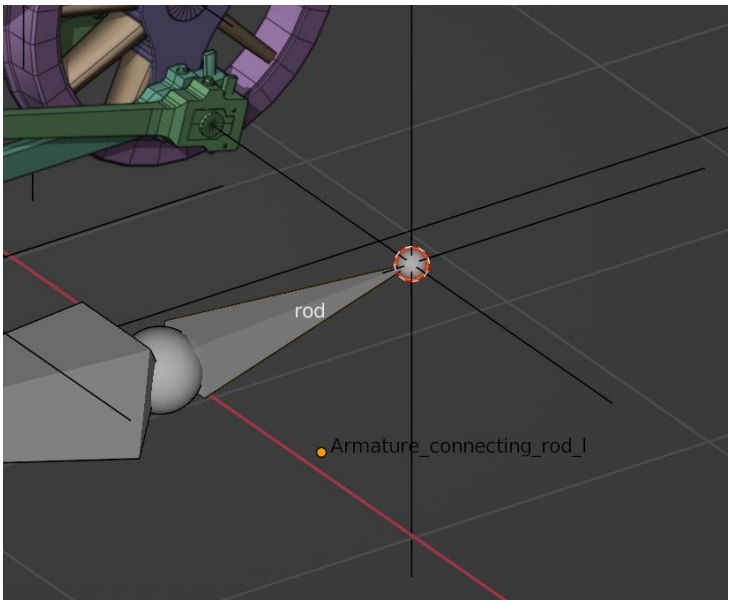



*Figure 9-5: Bone System merged in*




The next task is to move the end points of the bones to match the geometry of the loco we are building which is why we created Empties at the big and little ends of the connecting rod and the construction line for the piston rod centreline.

Start by moving the rear end of the ‘rod’ Bone to the big end of the connecting rod. Position the 3D Cursor at the origin of the Empty and in Edit Mode select the end of the Bone and move it by pressing ‘***SHIFT-S***’ and choosing ‘***Selection to cursor***’ from the Pie Menu.

*Figure 9-6: End of Bone 'rod' moved to the position of the big end of the connecting rod*

Now do the same for the little end of the connecting rod.

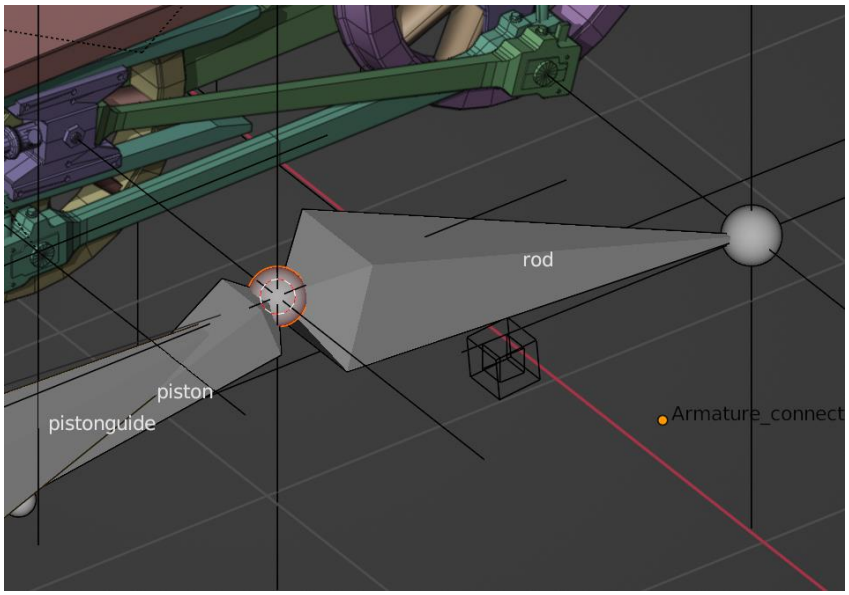



*Figure 9-7: End of Bone 'rod' move to the position of the little end of the connecting rod*

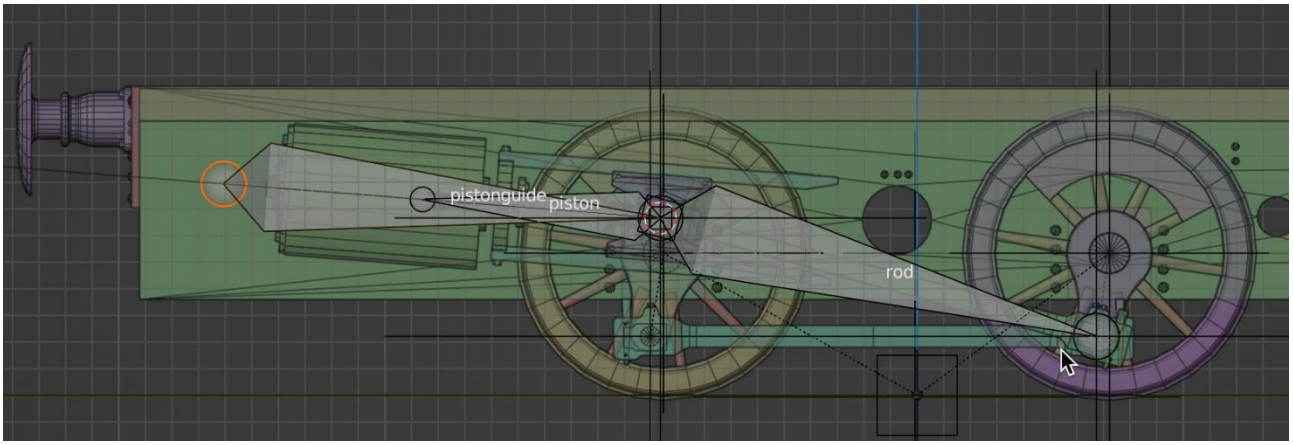
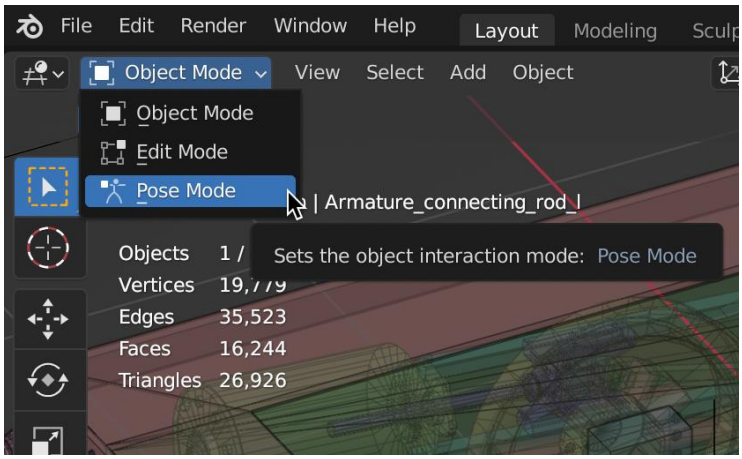



*Figure 9-8: Front ends of the 'piston' and 'piston guide' Bones moved to the centreline of the piston rod*

The end ends of the '***piston guide***' and '***piston***' Bones now need to be moved to coincide with the construction line at the centreline of the piston rod. Do this in Edit Mode and just move both end points until they are on the line. The Y position (I.e. the length of these Bones) is not too important, but if they are longer it is easier to position them accurately.

In the illustration above I have switched to X Ray mode and have set the Viewport properties of the piston rod centreline to '***In Front***' to make it easier to see what is going on.




The Bone System must now be animated. Select the Bone System and switch to '***Pose Mode***'.

*Figure 9-9: Switching to 'Pose' mode*

Select the ‘rod’ Bone, it will turn blue.

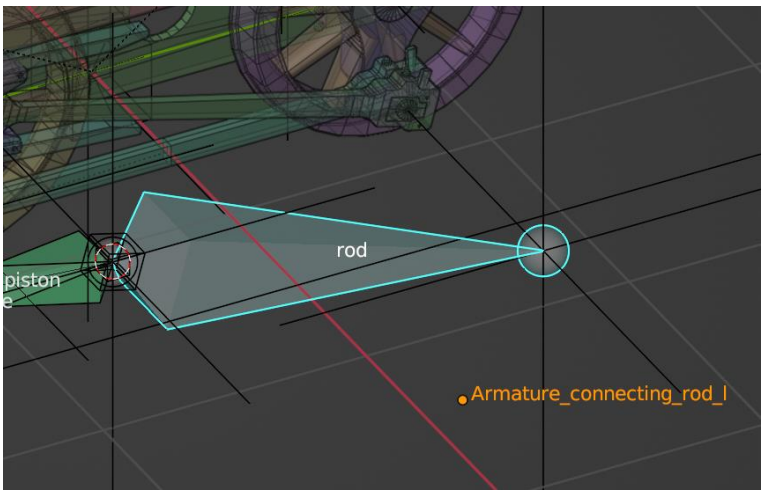
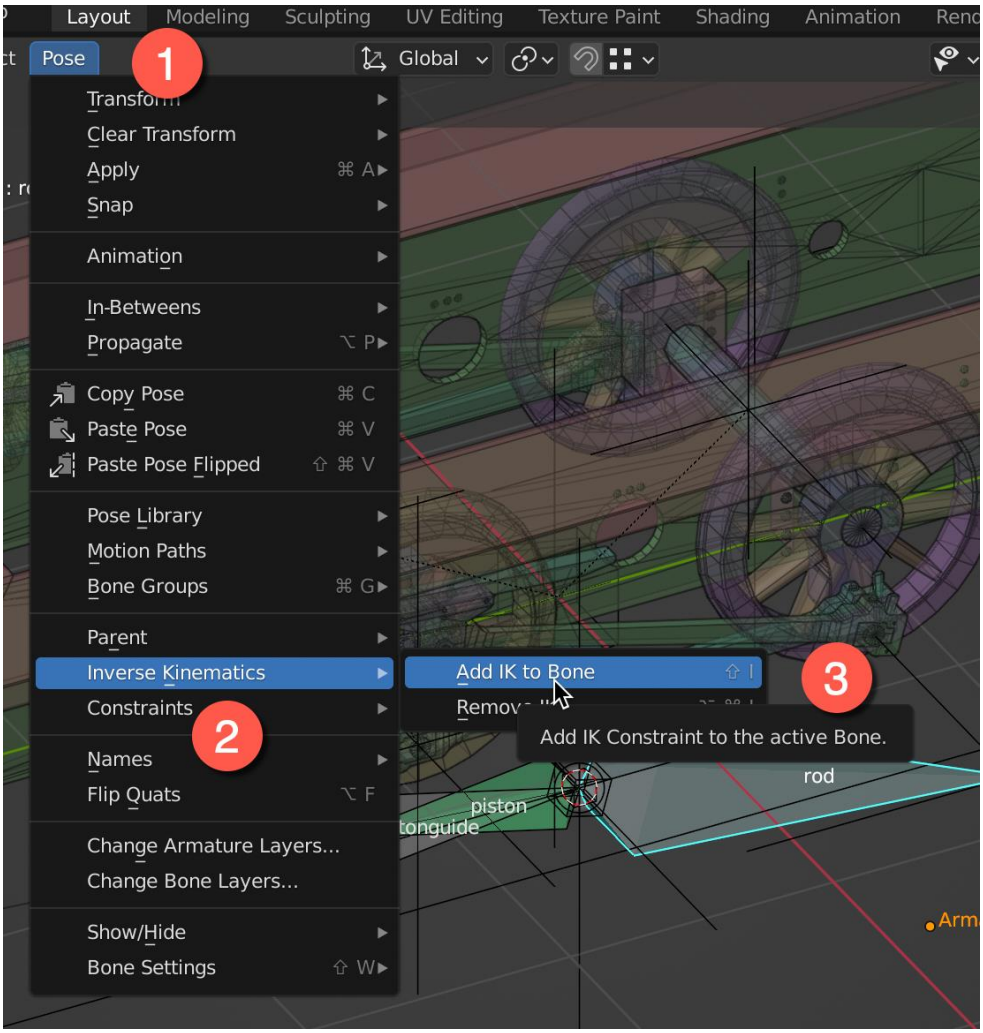



*Figure 9-10: 'rod' Bone selected in Pose Mode*




Now choose ‘***Pose > Inverse Kinematics > Add IK to Bone***’. This will calculate the position of each of the Bones in this system for each frame of the animation.

*Figure 9-11: Adding an IK Solver to the 'rod' Bone'*

Choose '***To New Empty Object***' from the popup menu.

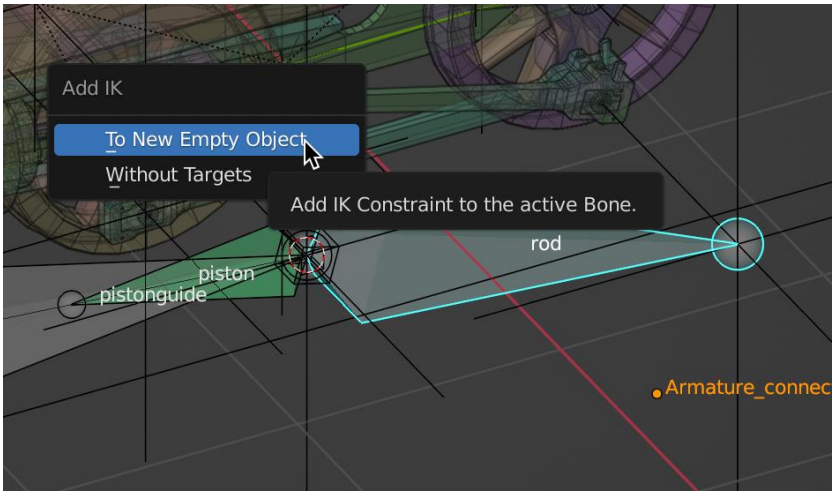



*Figure 9-12: Targeting the IK Solver to a new Empty object*

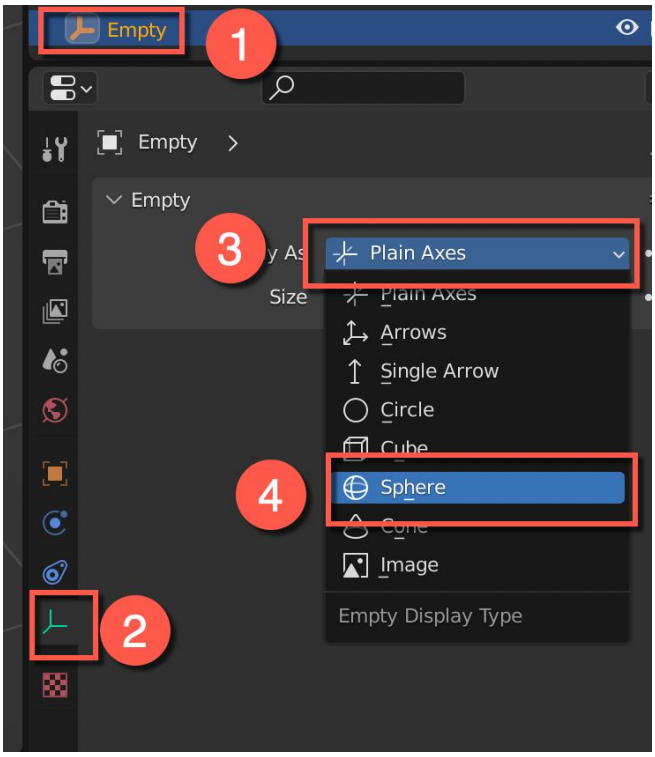



*Figure 9-13: Changing the appearance of the IK Driver Empty*

A new Empty will be created at the rear end of the 'rod' Bone. Switch back to Object Mode and select this Empty. It is a good idea to rename it and possibly change the appearance to that you know that is an IK driver Empty, I like to change it to a sphere.

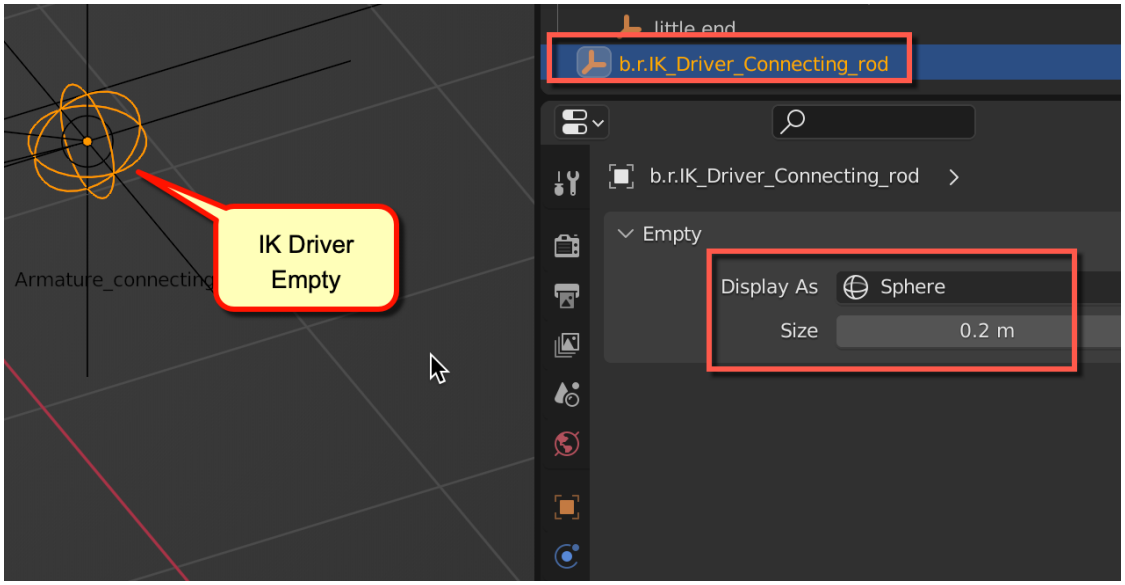
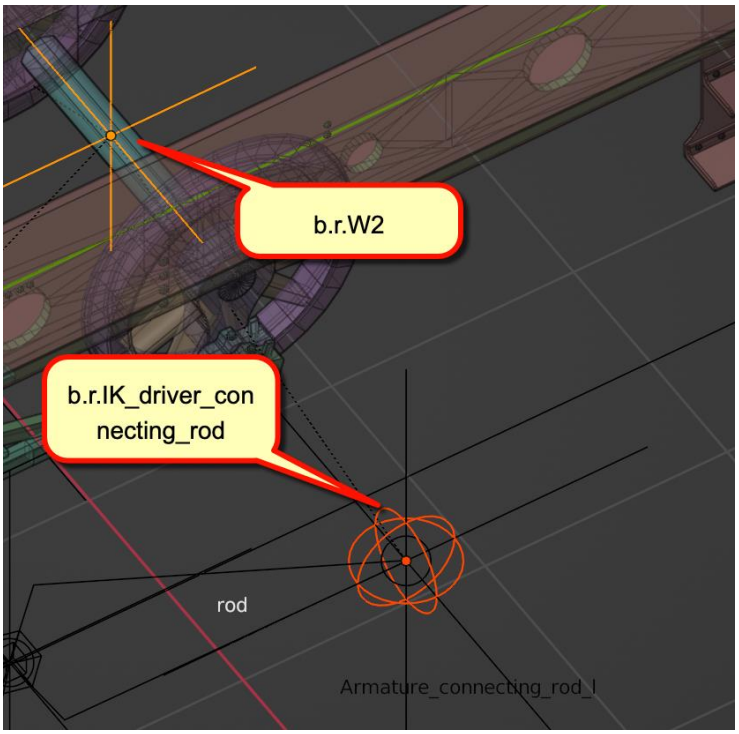



*Figure 9-14: IK Driver Empty change to a sphere, renamed and resized*

Another handy option in this dialog box is 'Size' which changes the graphical size of the Empty without scaling it.

**Note that the Empty must be name 'b.r.something' as otherwise the animation will not be exported to Trainz!**




This Empty must now be parented to the wheel Empty, in this case '***b.r.W2***'.

If you now start the animation, you'll see the Bone System moving correctly – the '***piston guide***' Bone will stretch and the '***piston***' Bone will follow it moving in a straight line along the centreline of the piston rod.

*Figure 9-15: Parenting the IK Driver Empry to 'b.r.W2'*

If you look closely at this area in X Ray mode, you'll see that there are already two Empties that are parented to the Bone System. All that remains is to parent the crosshead object to '***b.r.crosshead_l***' and the connecting rod object to '***b.r.connecting_rod_l***'

And that's it, the crosshead and connecting rod should now be animated correctly.

This Blend file at this stage can be downloaded from my website, you can try and animate the right hand side if you like…

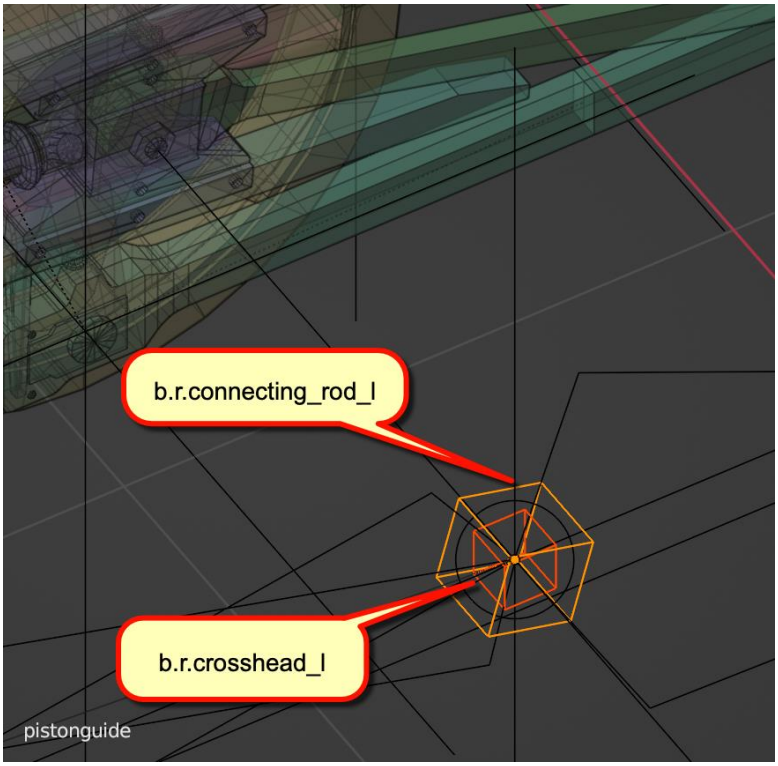
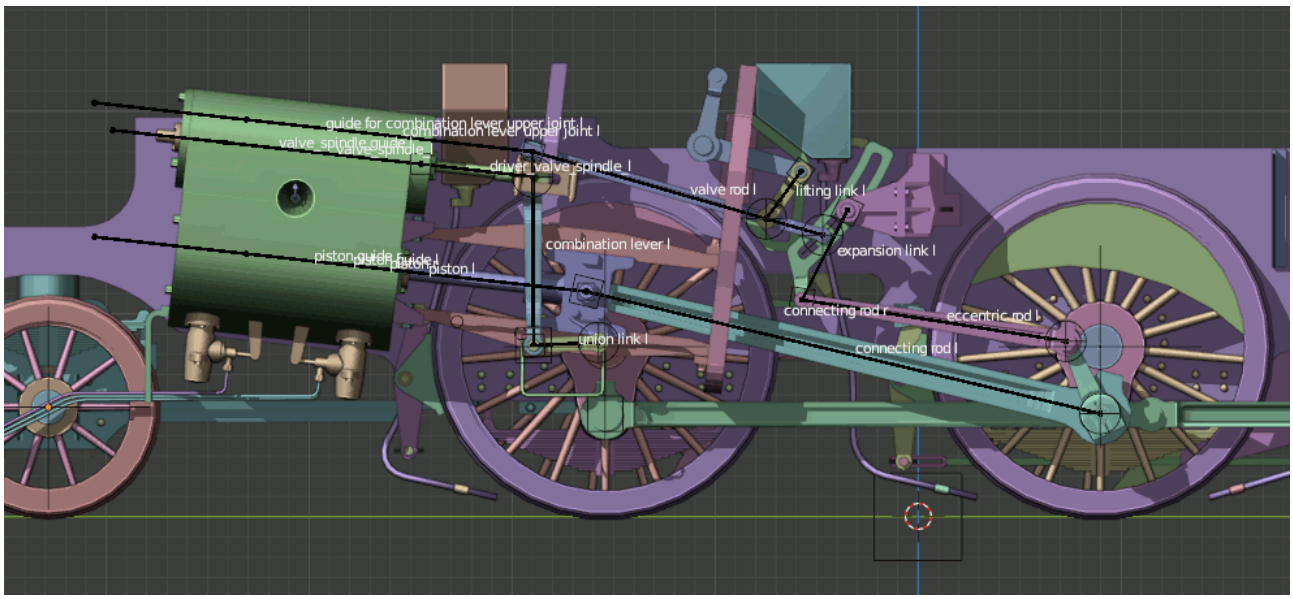



*Figure 9-16: The Empties for parenting the crosshead and the connecting rod*

## *Walschaerts Valve Gear*

**[To be expanded]**

This section will be expanded with a detailed procedure for animating all of the valve gear, based on this Forum post: https://forums.auran.com/trainz/showthread.php?162376-Some-tips-on-how-to-rig-Walschaerts-valve-gear-in-Blender&highlight=walschaerts




## *Copying Animation Frames from One Object to Another*

FBX exports require that you use Empties instead of Lattices for animations, so if you have a complex animation (Walschaerts Valve Gear for example) that uses Lattices you have a problem – the prospect of re-doing the whole animation to use Empties is probably not very appealing, especially if you spent ages manually positioning each Lattice for each of the parts of the valve gear.

Fear not, however, there is a way to copy Keyframes from one object to another and it's as easy as falling off a log.

Obviously the first thing to do is to create Empties in the same position as the existing Lattices. This is just a case of selecting the Lattice and pressing '***SHIFT-S***' and then choosing '***Cursor to Selected***' from the Pie Menu. Then add an Empty at the Cursor Position with '***Add > Empty > Cube***' (or any other style of Empty you want).

To copy the Keyframes from the Lattice to the Empty select the Lattice and then the Empty and choose '***Object > Link/Transfer Data > Link Animation Data***'. Job done, except of course you'll need to link the meshes to the Empties, but this sure beats doing the animation all over again…

Of course, if the Lattice is parented to an Armature it won't have Keyframes, in this case just create new Empties centred on the existing Lattices and parent them to the Armature, re-linking the meshes.

## 10. Creating The First LOD Mesh (LOD0)

It might seem a bit strange to start discussing LOD (Level of Detail) at this point, but up until now we have been working on a high poly mesh that is not what we want to export to Trainz. This mesh will only be used to bake a normal map to a lower poly model that will look almost identical to the high poly one, but which will be much more efficient for the Game Engine to handle.

This high poly mesh does not need to be UV mapped, or to have materials and as only the geometry is important, all the modifiers, curves that are used for pipes and linked duplicates of meshes can be left active. This is handy as it makes changing the mesh afterwards a lot easier.

Of course, if you want to skip this step, you can simply start UV mapping the high poly mesh and use it as the LOD0 one.

I'll assume that we'll rework the highest poly mesh to create the LOD0 one as this will introduce some techniques that will be useful for cleaning up the mesh and reducing the poly count.

The first step is to create duplicates of all the meshes that make up the highest poly mesh and to move them to a separate collection. The next step is to do any tidying up of the LOD0 meshes before we start the UV mapping process. LOD0 will be the highest poly mesh that we will export into Trainz.

Here we have a vacuum pipe mesh that has a standpipe with 1328 polys and a hose which is a curve object.

So, first convert the hose curve object to a mesh by choosing '***Object > Convert > Mesh***'. This adds another 224 polys to the mesh… The original curve object is still available in the highest poly version of course should we want to change it.

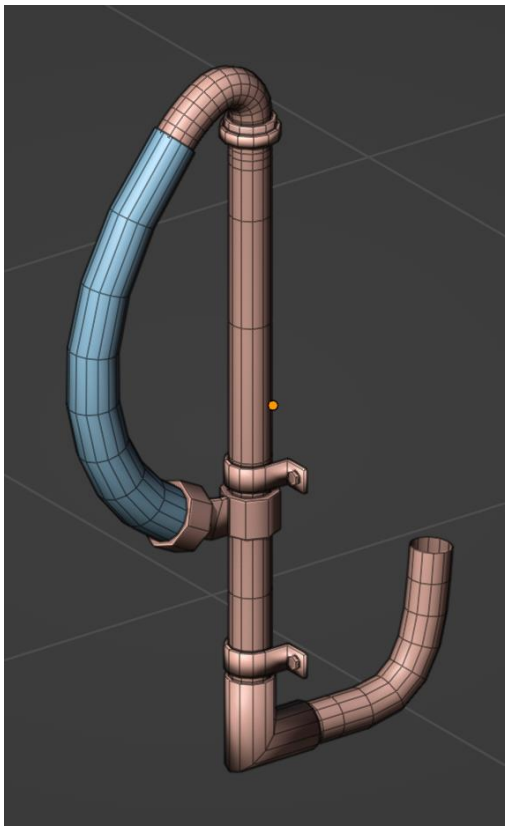
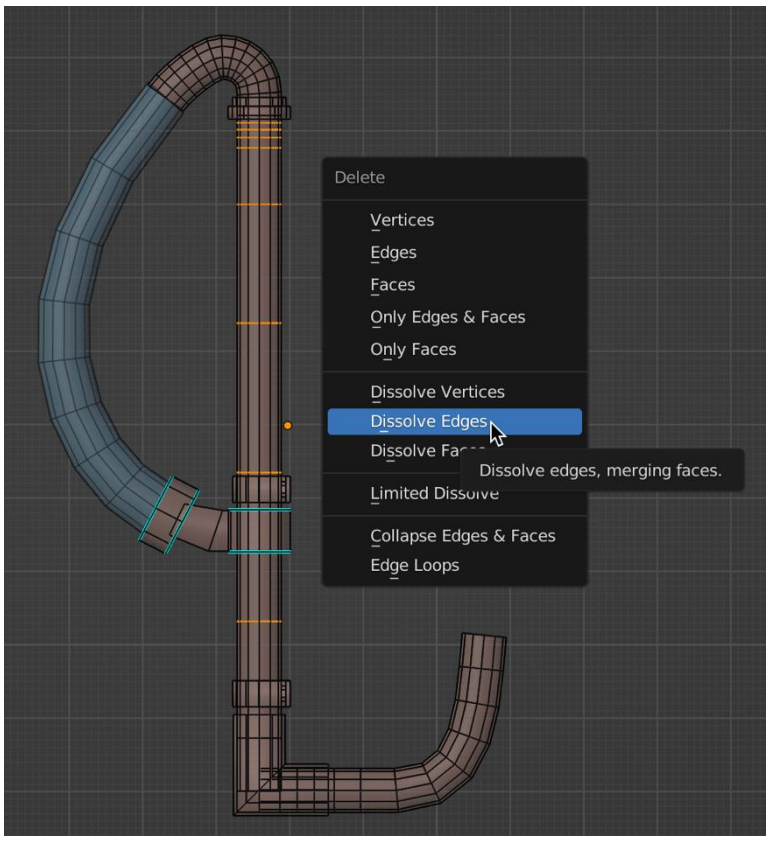


*Figure 10-1: High poly vacuum pipe mesh*




Looking at this mesh we can see that there are a lot of edge loops in the standpipe which are not needed. To remove these, we need to select them and then press ***X*** to open the delete menu.

*Figure 10-2: Removing unnecessary edge loops*

This one step has reduced the poly count by 192.

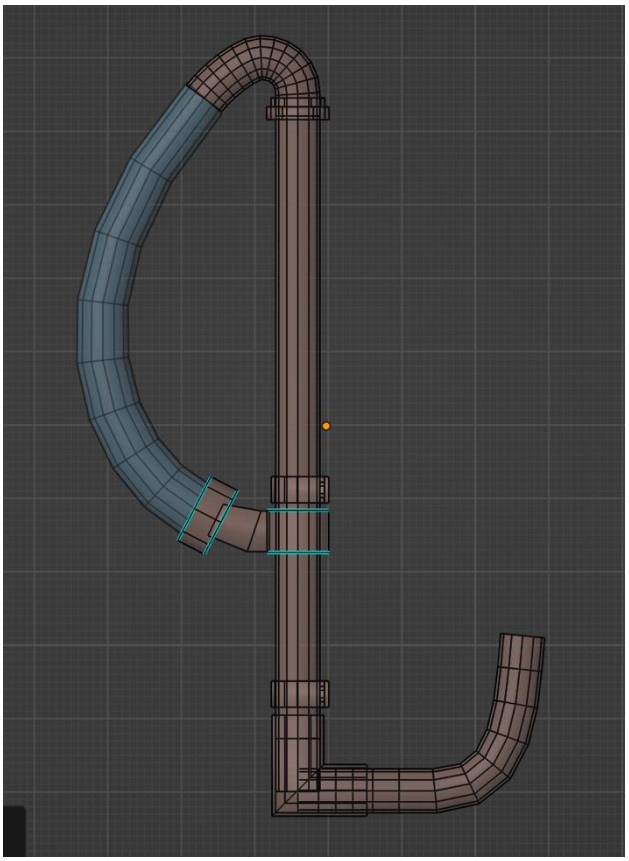


*Figure 10-3: Reduced poly LOD0 mesh*

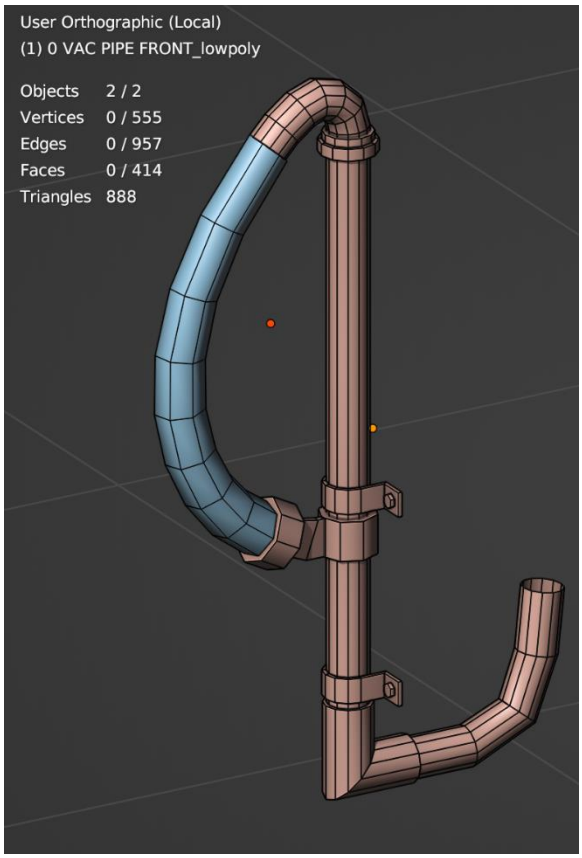



*Figure 10-4: The final reduced poly LOD0 vacuum pipe mesh*

Continuing along these lines the poly count can be easily reduced to 888, which is a reduction of 664. Baking a normal map from the highest poly mesh will improve the look of the LOD0 mesh so that it will be almost impossible to tell the difference.

Continue optimising all the LOD0 meshes in a similar manner.

The next thing to look at is if any of the surface detailing can be removed from the LOD0 meshes, these will be replaced by baking a normal map. Obvious candidates for this would be rivets (always massively poly heavy), beading and boiler bands for example. Looking at a boiler mesh before and after you can see what I have removed from this LOD0 mesh.

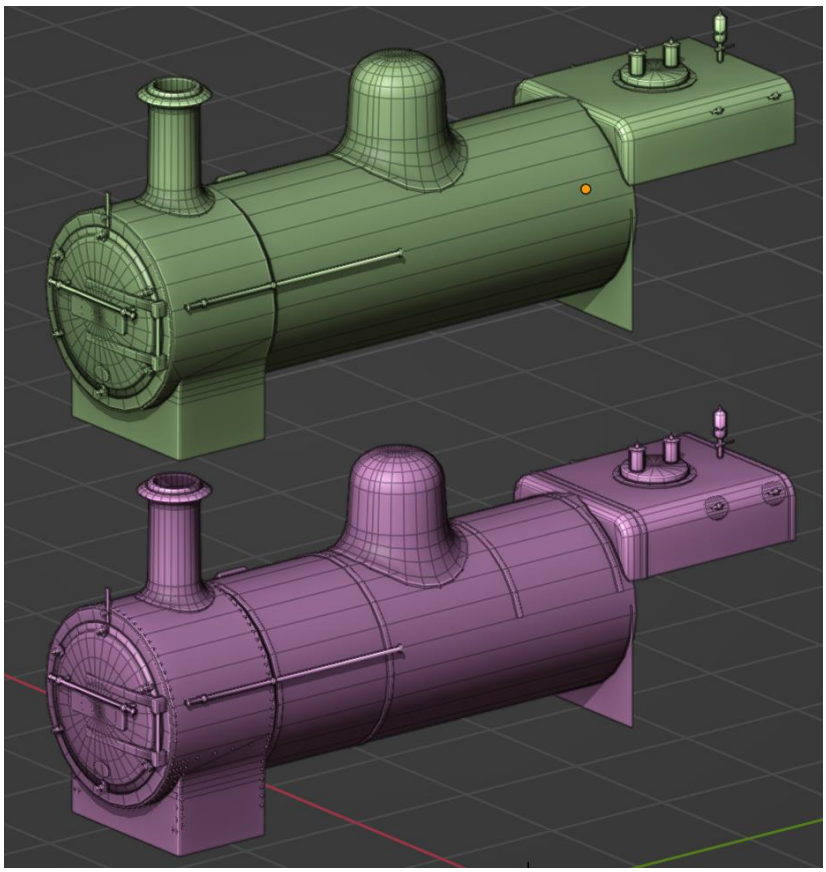


*Figure 10-5: Comparison between the high poly (bottom) and the LOD0 (top) boiler meshes*

I made things easier for myself when creating the high poly mesh by assigning a vertex group called ‘SD’ (doesn’t matter what the name is if you can remember it) to anything that I thought could be dropped from the LOD0 mesh. Then it was just a case of duplicating the boiler mesh, selecting the faces with the ‘SD’ vertex group and then deleting them.

Here I went from 15027 polys to 3723 polys. There are a lot of edge loops that can also be deleted to reduce the poly count further, and possibly the base of the safety valve cover could also be normal mapped.

Finally have a look around if there are any access hatches or cut outs that could also be normal mapped - basically, anything that could be described as surface detailing that does not define the form of the object, and remove them from the LOD0 meshes.

## 11. Setting up Vertex Groups

I'm going to detail now how to set up vertex groups, which will make the texture painting easier, especially if it to proves to be necessary to remap the complete model multiple times (which I always seem to need to do).

It is not essential, but vertex groups have lots of uses so they are a useful trick to master. Vertices can be assigned to one or more named groups, and they can then be selected and deselected as required – as Texture Painting can be used to paint selected faces it is very handy to be able to easily select the faces as required.

In the 'Start Model' that can be downloaded from my Web Site here are several vertex groups set up as shown which can be expanded as needed.

The 'Start Model' includes a Cube which has these Vertex Groups, if you attach this Cube to your meshes then the Vertex Groups will be transferred to your meshes (in the same order), which means that you don't need to define them for each mesh. The Cube from the 'Start Model' can then be deleted.

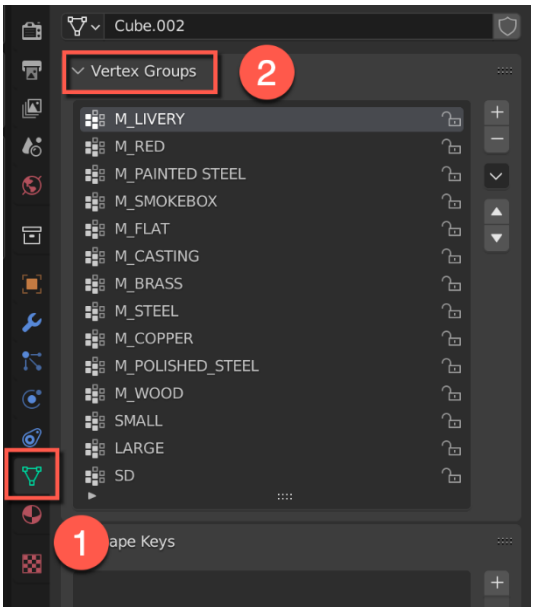


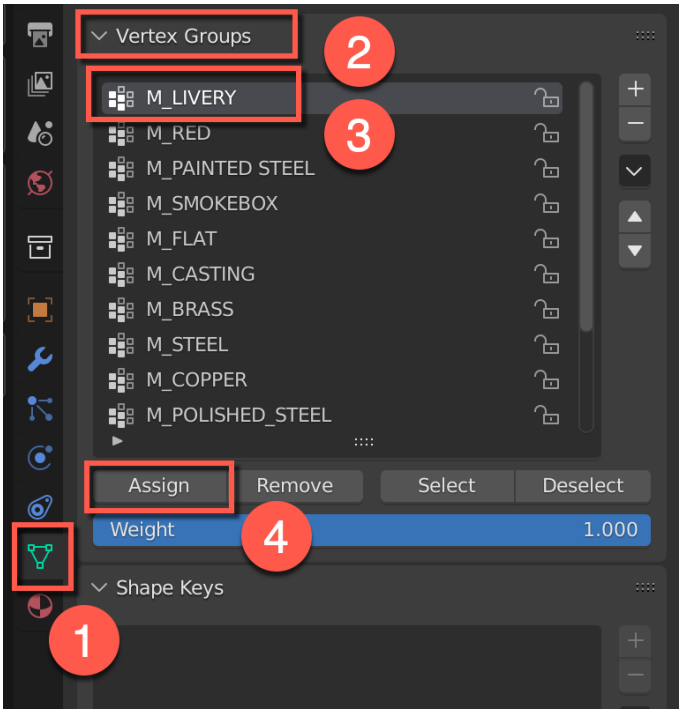

*Figure 11-1: The vertex groups included with the start model cube object*

**Note that you must attach the Cube to your mesh and not the other way around – otherwise you'll lose any modifiers that you have applied to your mesh!**

**It is also very important to note that if the Cube has a UV map on it that has a different name to any UV map on your meshes then the joined mesh will have two UV maps on it which will cause no end of grief when you export the asset into Trainz.**

To avoid this problem the Cube in the 'Start Model' has no UV mapping applied to it, but always, always, check before you join two meshes that they both have the same name for the UV map (or that one of the meshes has no map).

In fact, once the meshes are joined together always check that you only have one UV map on the resulting mesh. I can't emphasise enough how important this is.




To assign Vertex Groups to faces all you need to do is to select the faces (in Edit Mode) in the 3D Viewport and then the desired Vertex Group. Finally selecting 'Assign' will complete the operation.

Note that faces can be assigned to more than one Vertex Group, so you could have a Group for the Boiler assembly which then has groups for the various materials. This allows you to join parts together and then separate them later without having to select each face. However, I would suggest always keeping the Boiler as a separate mesh for simplicity.

*Figure 11-2: Assigning Vertex Groups to selected faces*

Obviously, you can remove selected faces or vertices from a Vertex Group with the '***Remove***' button or select/deselect vertices or faces that belong to a Vertex Group with the '***Select***' and '***Deselect***' buttons.

We now have a way of selecting groups of faces for texture painting which makes life a lot easier if, for example, it is necessary to remap the model. All we need to do in this case is to repaint the model which is relatively simple. The whole process of assigning the Vertex Groups is made much easier if you do this right from the start as you create each mesh, and I strongly recommend that you do this.

It is also a good idea to assign the Vertex Groups before applying any modifiers (such as mirror) to the meshes as the mirrored instances automatically get the same Vertex Group assignments as the original mesh – this can save a bit of time.

## 12. UV Mapping

UV Mapping is simply defining what areas of the skin (the bitmap that controls the colouring of the model) are 'mapped' to which faces. This must be done in order to export the model to Trainz, so it is the next logical step after creating the mesh.

Probably the most efficient way to tackle this is when building the mesh, as the mesh needs to be prepared with mapping in mind.

### *Auto UV Mapping*

The quick (and lazy) way to UV map the model is to use auto mapping which just involves selecting the mesh, entering Edit mode by pressing ***TAB***, selecting all the faces by pressing ***A*** and then pressing ***U*** with the mouse cursor over the 3D window. Then just choose ***'Smart UV project'*** from the menu.

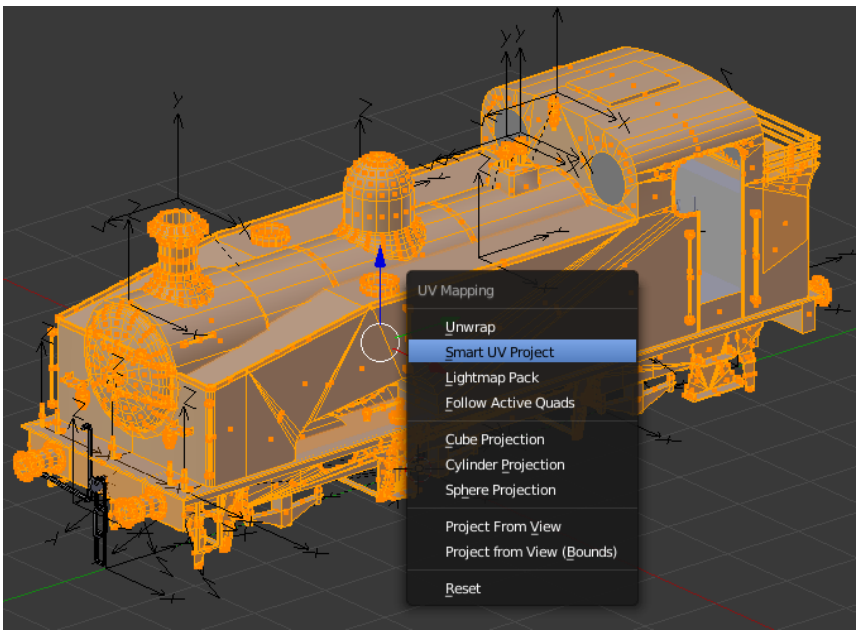



*Figure 12-1: Choosing the type of UV Mapping, all faces selected*

A few remarks on the settings – setting the ***Angle Limit*** as high as it will go (89) tends to create fewer islands. The '***Island Margin***' separates the islands slightly which helps when texturing, but this can also be adjusted afterwards if necessary. '***Area Weight***' I've never used (so I don't know what it does).

The result in the UV editor window looks like this:

At first glance it doesn't look too bad, but the disadvantage is that round objects such as the boiler have been split up into four separate islands and there are a lot of islands which wastes space on the texture map if we want a couple of pixels between each one – it's a good idea to do this as otherwise you can get textures bleeding onto other faces.

Mind you, having a lot of small islands can allow for better packing if you are using the UV PackMaster Blender add-on, as it has a better chance of arranging the islands in the most optimal layout.

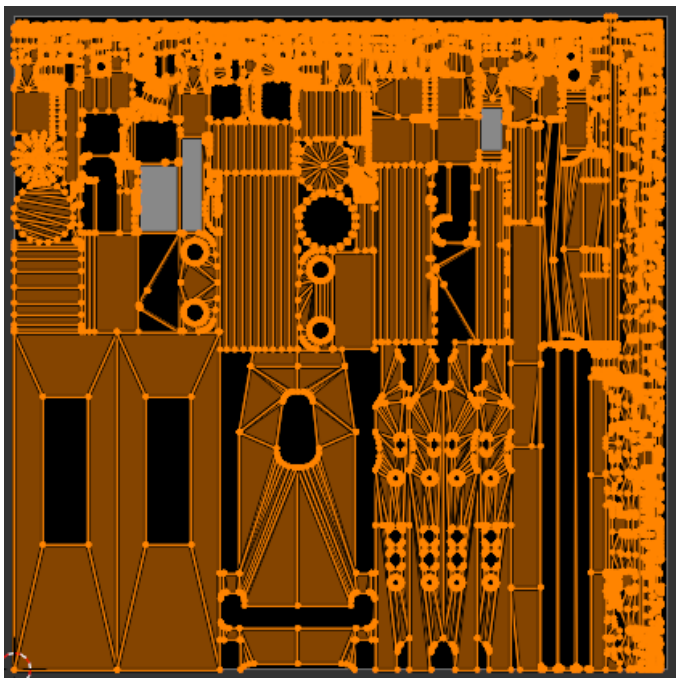


*Figure 12-2: Resultant UV Mapping (all faces selected)*

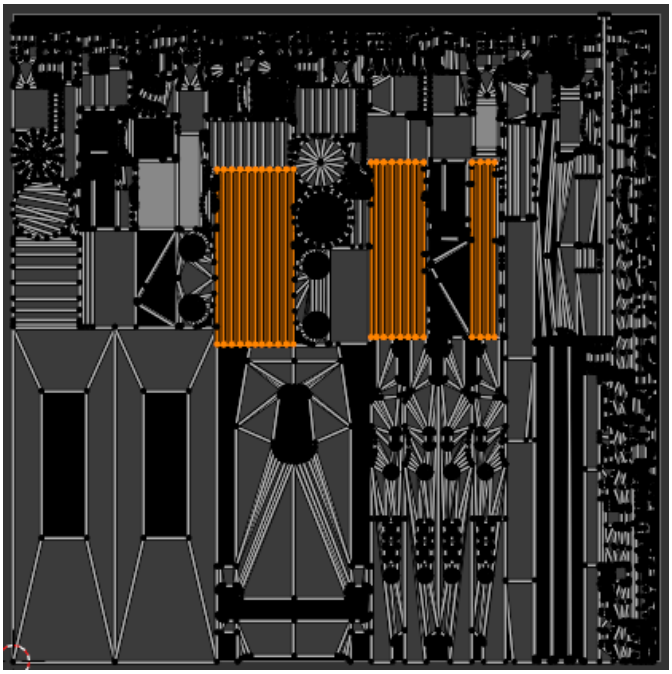


*Figure 12-3: Boiler faces selected*

Here I've selected the boiler to show the splitting of the UV islands – this makes re-skinning harder for third parties if the source *.blend file is not available although if you do have the *.blend file it doesn't really make a lot of difference.

All the round parts of the mesh have been split in to three separate islands which isn't ideal.

The main problem with auto mapping is that it makes a messy map although it is very quick to do and is handy if you have pressed on and built the mesh without manually mapping as you did so. I'd recommend mapping as you go though as mapping a whole loco at the end can be a bit tedious.

## *Manual UV Mapping*

In order to get a much more optimised UV map it is better to do it manually and this is not really that hard to do if you follow the workflow that I am going to suggest. The first thing is to mark all the sharp edges of the mesh, and Blender can do this automatically.

In Edit Mode and in Edge Select Mode, choose '***Select > Select Sharp Edges***' and then '***Edge > Mark Sharp***'. The default angle for edges to be regarded as sharp is 30° and this usually works fine. It is also a very good idea to mark seams in the LOD0 mesh. Tthis can

be done by selecting the sharp edges again and choosing '***Edge > Mark Seam***'. This will create rather too many seams, but it is a good start and the extra seams can be deleted afterwards.

**[To be expanded…]**

## *The UV PackMaster Add-on*

I'm now going to suggest something. That is a big no-no for some content creators, and that is investing some money in an add-on for Blender. Although you can get away without this, this add-on makes creating an optimised UV map so much easier to produce, and if I could only afford one add-on this would be it. I really can't imagine creating content without it now.

### *Why Use this Add-on?*

It's important to make the best use of the available space on a texture map as this potentially improves detail on the textures and the performance in-game. It is especially important when using PBR materials which always include a normal map which allows things like rivets to be baked on to the low-poly in-game mesh, saving many, many extra polygons and making the asset look better.

Blender has a built-in UV packer, but quite frankly, it's not that good - this is the same mesh repacked using UV PackMaster. The automatic UV unwrap used 71.9% of the available space, UV PackMaster 87.3%, which is a worthwhile improvement.

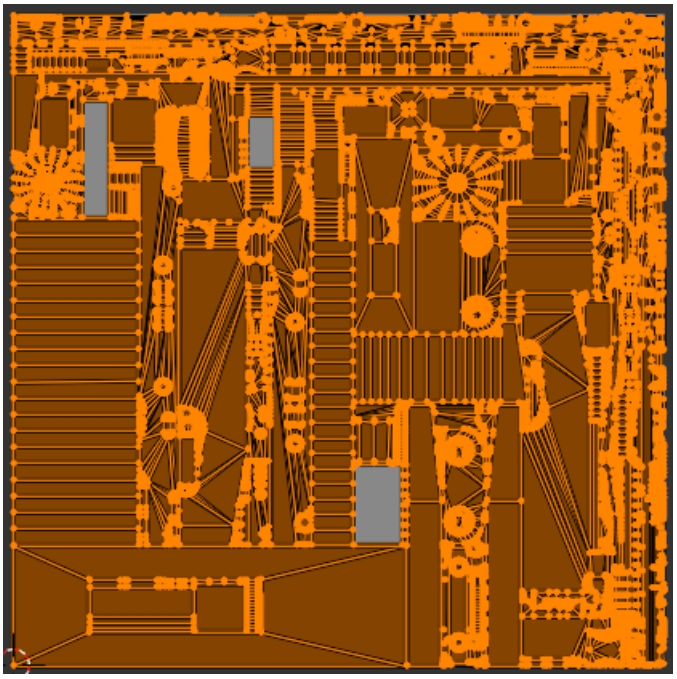


*Figure 12-4: UV pack using UV Packmaster*

### *Where to Get It*

The UV PackMaster WebSite can be found here: https://uvpackmaster.com/, to buy it you'll need to go to GumRoad here: https://glukoz.gumroad.com/l/uvpackmaster3, or Blender Market here: https://blendermarket.com/products/uvpackmaster. All these sites have videos and descriptions of what it is and what you can do with it.

The bad news is that it does cost 39USD at the time of writing, but it is worth noting that it is much more than a UV packing tool as it covers just about everything you need for working with UV maps, replacing several other add-ons.

### *Using the Add-on*

**[To be added…]**

## *Strategies for Improving the UV Map*

So, what can we do to further improve the UV map? As good as the result from UV PackMaster is, you may find that the Texel Density of the map is not high enough to get a good result when baking normal mapped rivets for example. I usually try to get a Texel Density of 1.8-2.0 px/cm and the UV map shown above is only 1.395 px/cm.

One possibility is to consider whether parts of the map can be reduced in size to say, half. Candidates for this would be the underside of the footplate, the insides of the frames and

indeed anything which is not particularly visible on the final model. Unfortunately, I'd already implemented this on the example model…

Downward facing parts of the mesh don't need to have a high texel density, certainly not as high as more visible parts and just need to be there so that there are no visible holes in the mesh.

Another is to see if any parts of the mesh are completely hidden and can therefore be removed. In this model the front of the tanks is partially obscured by the smokebox, so we can cut this area out of the tank mesh to free up some space on the UV map.

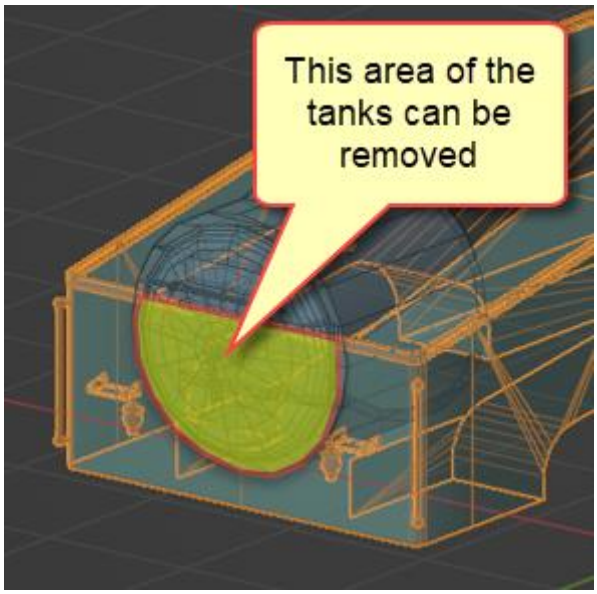



*Figure 12-5: Front of tanks are completely hidden by the smokebox and can be removed*

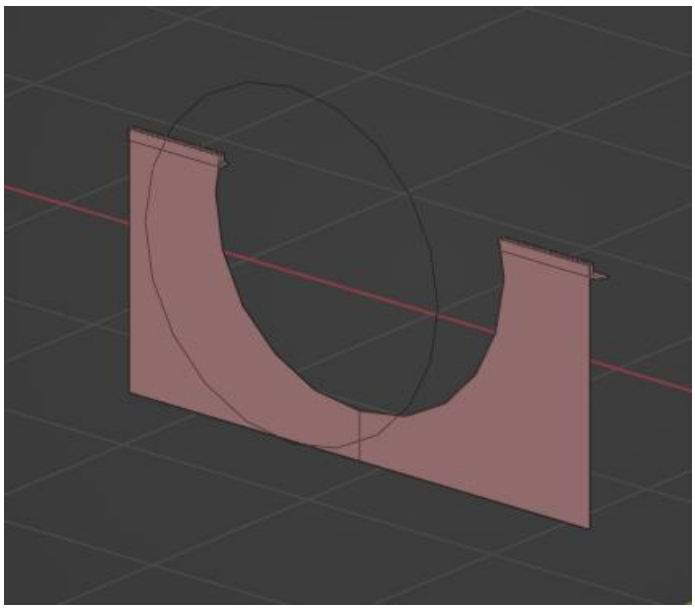


*Figure 12-6: Front of tanks cut away to save space on the UV map*

Probably the best way to do this is to select the edges of the smokebox front and separate them into a new object which can be used in a Knife Project operation, or indeed just use the knife tool to cut the area away as it doesn't need to be so exact. Here I've done just that – obviously we have increased the poly count a bit, but this is not that important.

In the case of this loco a large amount of the footplate, smokebox and boiler are hidden by the tanks, so these areas can be cut away to save space on the UV map like so:

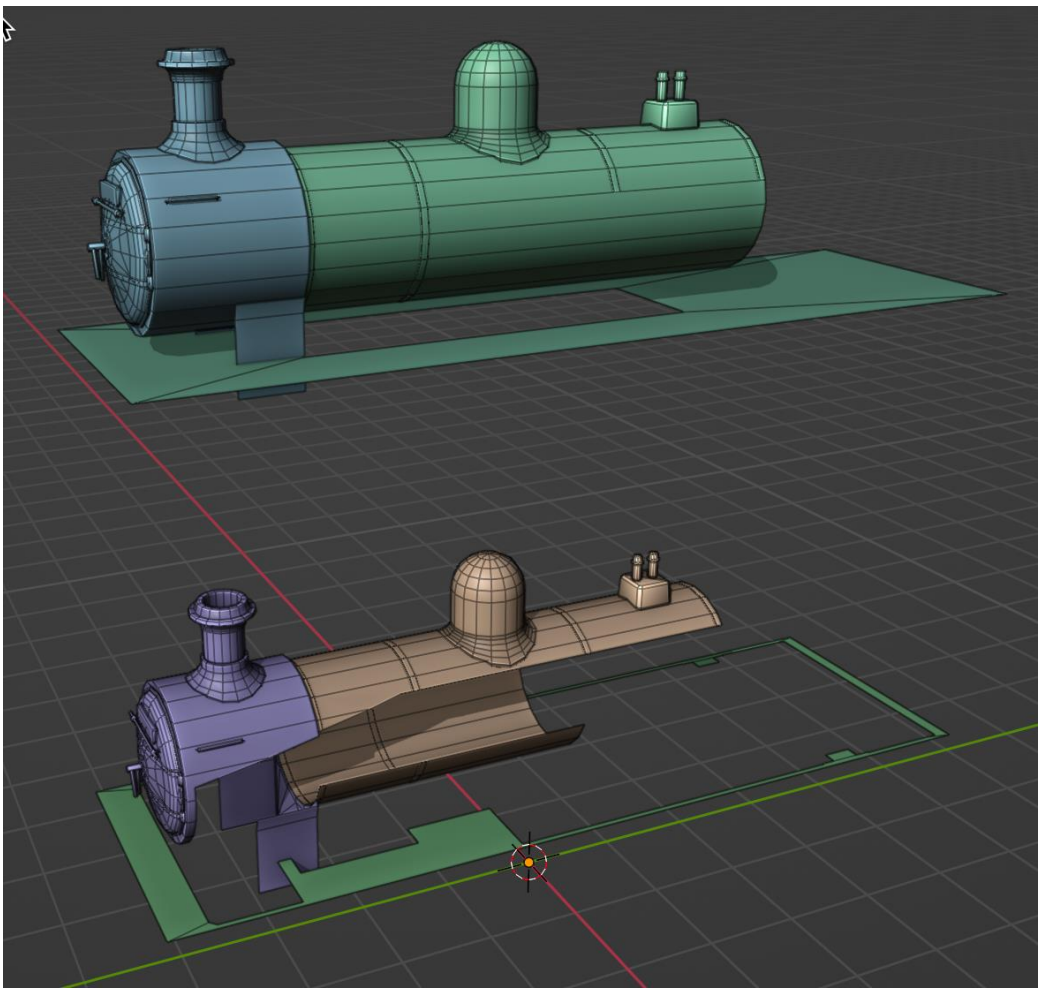


*Figure 12-7: Before (top) and after (below)*

This is an extreme case but is a good way of freeing up UV space to use for additional detail meshes or to increase the Texel Density of the whole map.

The poly count is also reduced from 2549 (before) to 2477 (after).

A further possibility to save space on the UV map is to overlay certain UV islands which have the same geometry and can share space on the UV map which, UV PackMaster can do automatically. An example would be the brake blocks perhaps.

For this loco I had quite a job to achieve a Texel Density of 1.8 px/cm for some reason, and in the end, I moved the mainframes and brake gear from the body UV map to the bogie UV map, both being 2048 x 2048. This sort of thing is why I recommend assigning Vertex Groups to the meshes as it makes redoing the textures a lot easier if you find it necessary to completely reorganise the UV maps.

The absolute last resort would be to admit defeat and move some parts of the mesh onto a separate UV map, which could mean a new material and an extra draw call, so should be avoided if possible.

## 13. Baking The AO and Normal Maps

An ambient occlusion map is effectively a way of generating 'soft' shadows on the mesh which helps to bring out the detail. Without it the mesh will look flat and frankly not very good. The AO map is a separate bitmap file that can be overlaid on the skin to add fake shadows. Look here for a fuller explanation of what it is and why you'd want to use it.

In addition, when using Quixel Mixer or Substance Painter to generate the final PBR export textures the AO map gives the painting software important information that helps it to create realistic looking textures. Quixel Mixer is not able to bake textures itself, so these maps must be created in Blender or some other software. Substance Painter can do baking, but I have found that you have better control over the baking in Blender.

Baking the AO map is also useful for checking the mesh and the UV map for any problems before we get too far with creating the textures. If you have forgotten, for example, to properly unwrap something you will see artifacts in the AO map.

I am going to suggest baking the AO and normal maps in a separate Blender file as it is useful to move some parts away from their final export positions to make the baking work better. In the case of the AO map, you don't want the wheels casting shadows on the mainframes for example, so it is useful to move them away from the main meshes. For the normal map if two meshes are too close together you may get artifacts in the normal bake, and it is a good idea to animate them to explode them away from the other meshes. Animation is handy as you don't usually want the exploded positions for doing the AO bake.

Before we create a new Blend File for baking the textures, we need to create a new collection that will be the first LOD mesh, i.e. the mesh that you see when you are close to the model in Trainz. The high poly meshes will only be used for baking and will include such details as rivets and boiler bands. These can easily be included in the normal maps and save a lot of polys in the finished model.

So, create new collections for the LOD0 meshes for the Body, Bogie and XCab meshes. Select all the high_poly meshes and copy them into the '***0 BODY***' collection. Here you can see that I have also created '***1 BODY***', '***2 BODY***' and '***3 BODY***' collections for the LOD1, LOD2 and LOD3 meshes.

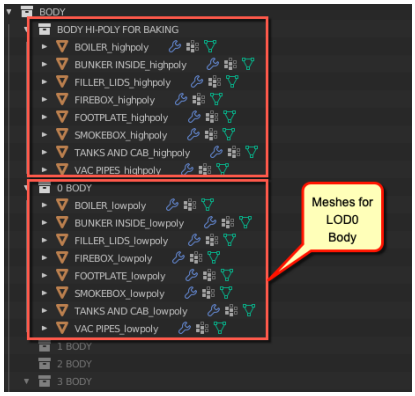



*Figure 13-1: The collections for high and low poly meshes for the Body*

You'll notice that the meshes have been renamed with '***_highpoly***' and '***_lowpoly***' at the end of the mesh name. This is important for using the '***TexTools***' addon which I use to bake the AO and Normal maps. This addon is built-in to Blender and only needs to be activated. The advantage of using this is that it bakes maps that have anti-aliasing to smooth out any jagged edges in the bake. The standard Blender baking in v3.0 can't do this.


Switch to the 'UV Editing' tab and click on the '***TexTools***' tab (press '***N***' in the UV Editor window if the tab is not visible). Set '***Size***' as required, the '***Padding***' to 1 and open the '***Baking***' section of the dialog box.

Then set 'AA' (anti-aliasing) to 2x and in the '***Preview Texture***' section set '***Tangent Normal***'.

With all the high poly and the low poly meshes selected, tick the '***Single Texture'*** option (1) and click on the '***Explode'*** button (2).

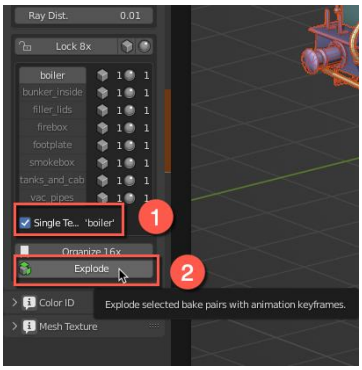



*Figure 13-2: Selecting the 'single Texture' and 'Explode' options*

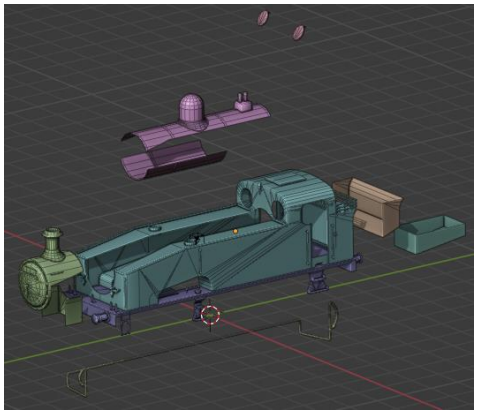


*Figure 13-3: The high and low poly meshes for the body exploded by the 'TexTools' addon*

The mesh objects will be animated from 0 to 50 frames to explode them. We may need to help things along a bit at this stage, as I want the filler lids to remain attached to the tank tops and the footplate has not been animated for some reason (the '***TexTools***' addon seems to be a bit unreliable, unfortunately).

The filler lids can be fixed by deleting the animation data on the meshes created by TexTools so that they are only animated by the '***b.r.main_filler_lid***' Empty, and the footplate by manually adding a Key Frame at frame 50 to move it.

Don't forget to move both the high poly and the low poly meshes, as they must be in the same position as each other when baking.

Here I've adjusted the position of some of the meshes to ensure a better normal bake.

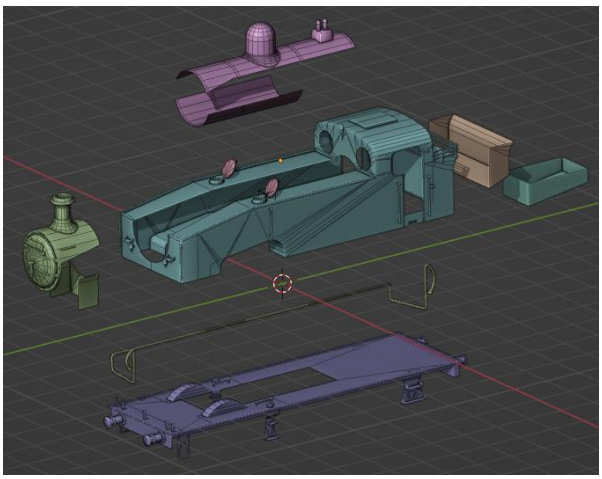


*Figure 13-4: Optimised exploded meshes*


*Figure 13-5: Setting the 'Ray Dist' value*

Before baking the normal map, we need set the '***Ray Dist***' to a suitable value such as 0.03. Then just hit '***Bake 1x***' – this will take quite a long time depending on your hardware, on an Apple MacBook Air 8/8 around 6.5 minutes for example. You'll need a computer with a fair bit of graphics power for this step. Also don't believe the YouTubers who say that you'll never see a beachball (the Mac OS 'busy' cursor icon) if you have a M1 Apple, you will be doing this sort of thing as it hits the hardware hard…

Turning off the anti-aliasing will reduce the time by about half, but then the quality of the bake won't be as good.

Once the bake has finished, save the texture that has been created. Saving as *.png with say 15% compression will produce much smaller files compared to *tga, so I would recommend that.

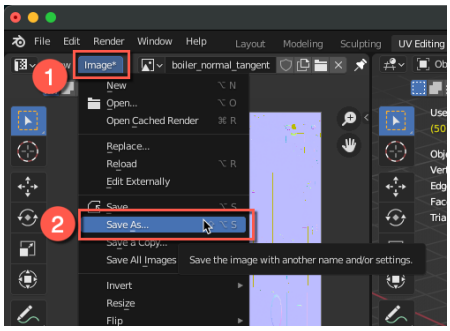



*Figure 13-6: Saving the texture*

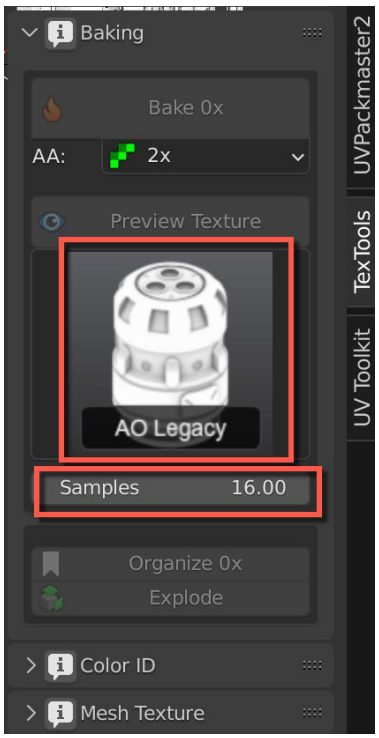



*Figure 13-7: Settings for the AO bake*

Before doing the AO bake, set the animation frame back to 0 to set everything back to the starting position, then setup the TextTool addon to bake AO, setting the '***Samples***' to around 16. Higher numbers mean a less 'grainy' AO bake but will take longer to render.

There is a problem that you can have with the AO bake, caused by overlapping UV islands which is something that normally you would try and avoid.

The problem here is that the faces have a particular order in which the faces are rendered out to the bake. For the solution to this, read on…

I don't like using so much space on the UV map for things like smokebox doors, when overlapping the islands seems to be an easy solution. In addition, you can simply remove all the 3D detail at the lower LODs and replace it with a flat face to save polys.

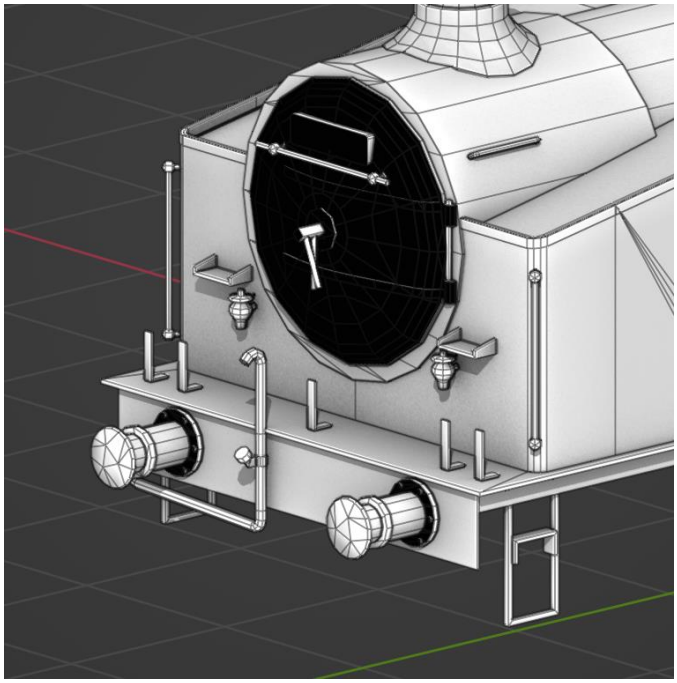
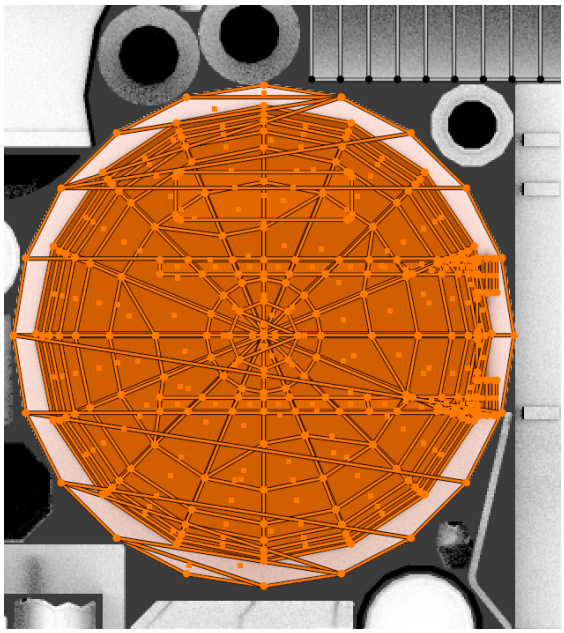


*Figure 13-8: The AO bake problem with overlapping islands*



Here is a view of the area of the smokebox door on the UV map with the faces highlighted.

*Figure 13-9: The UV map of the smokebox door area*

There are a couple of things you can do here to correct the black faces in the AO bake. The simplest solution is to temporarily separate the black faces and to just bake them again, as once they are separated you can decide yourself which faces to bake first. This is quick and easy to do, as usually there will not be many problem areas. In the case of this loco it is only the smokebox door and the buffer beams.

Another possibility is to avoid the overlapping faces by using the '***Knife Project***' tool to split the mesh up so that the faces that are overlapped can just be deleted. This would mean more polygons in the higher LOD meshes but might be a better solution.

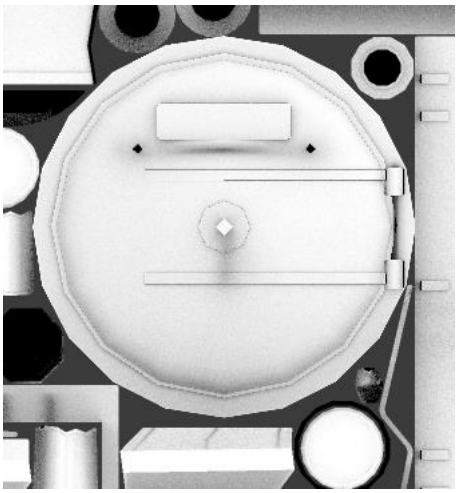


*Figure 13-10: AO bake for the smokebox door manually corrected*

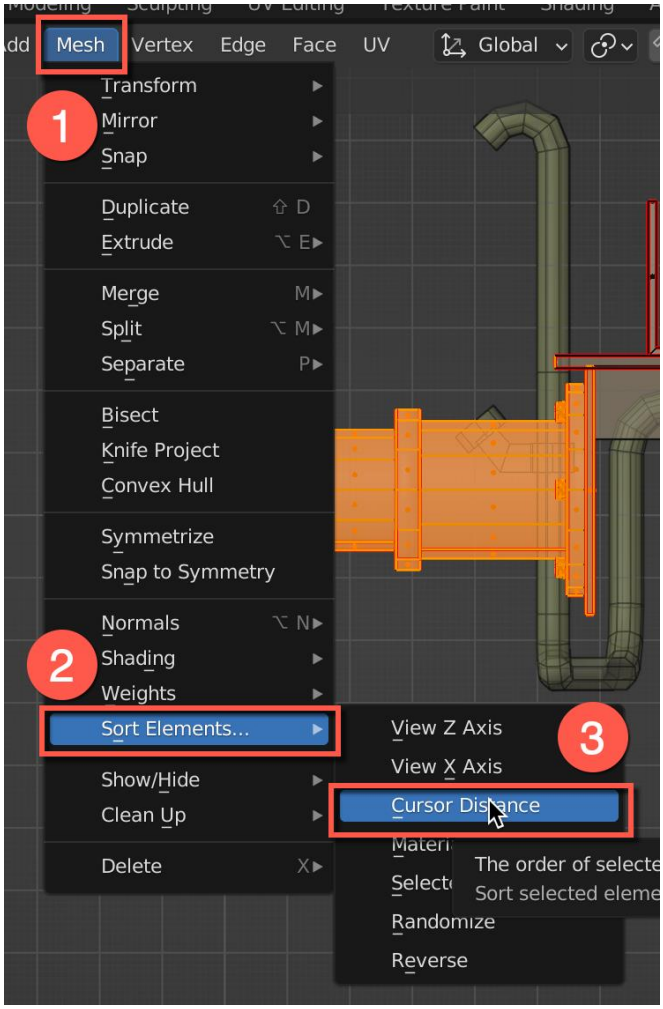



*Figure 13-11: Changing the order of selected faces*

However, the proper solution to this problem is to simply reorder the faces by placing the 3D cursor in a suitable position and then selecting the affected faces in Edit Mode and then choosing '***Mesh > Sort Elements > Cursor Distance***'.

Unfortunately, '***View Y Axis***' is not an option, if it was, we could avoid having to place the 3D cursor.

By a suitable position I mean that the cursor should be positioned in this case in front of the buffer beam, so that the render order of the faces depends on the distance to the 3D cursor.

It only took me 15 years of using Blender to find this out, things like this never seem to get mentioned in tutorials…

In practice I would only do this if problems really do occur with the AO bake, after all sometimes the faces do render in the correct order!

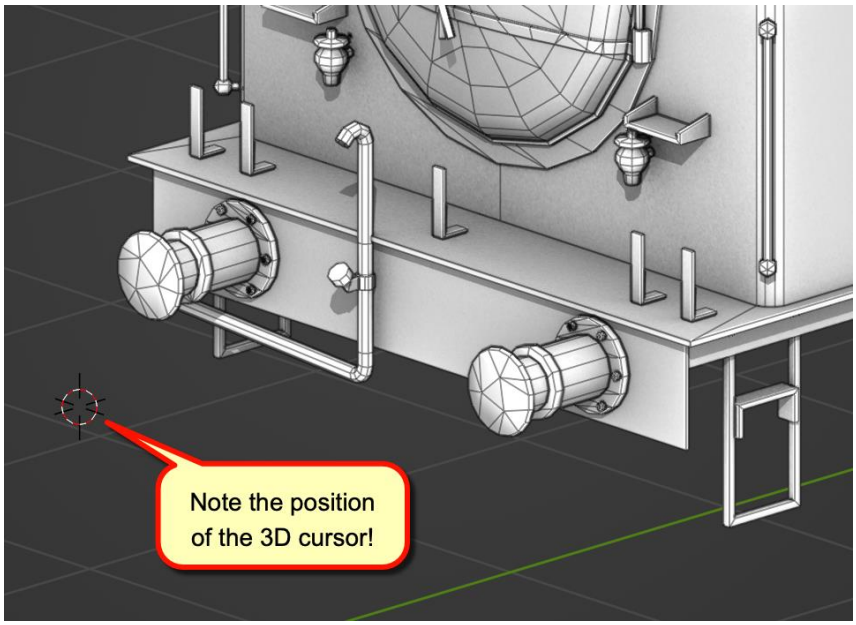



*Figure 13-12: AO bake after changing the ordering of the faces*

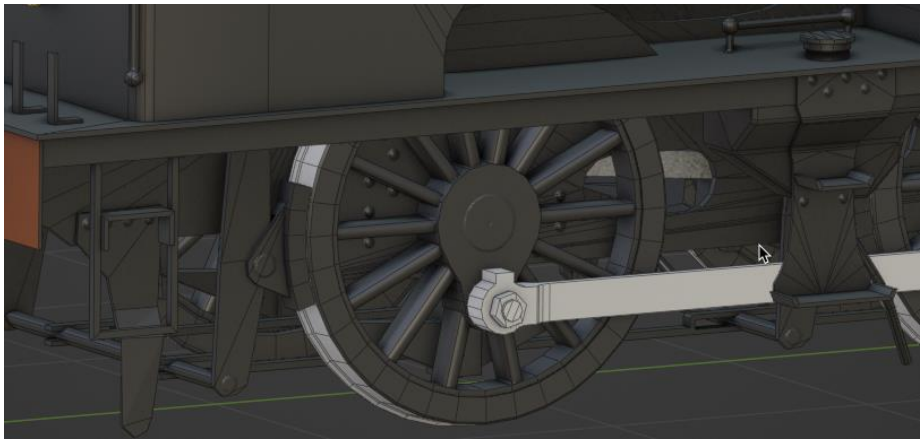


*Figure 13-13: Final normal mapped model with basic colours and AO bake*

Here is a view of this loco with the normal mapped textures, all the rivets on the mainframes and steps are normal mapped, as are the shaft ends on the brake shoes and the brake hangers and the axle ends. In this case there are not too many rivets, but sometimes there are hundreds of them which cost a lot of polys which are not needed. I think this looks pretty good, and usually you wouldn't get this close to a loco in Trainz in any case.

# 14. Texture Painting

This is a technique that makes the creation of textures for a 3D model much easier and quicker. Basically, it allows you to paint the 3D model in real time using a virtual airbrush and allows for effects that would be almost impossible in any other way using just Blender – for example a rust streak that starts on the base of a chimney and continues onto the smokebox.

Having said that though, I'm not going to recommend using texture painting for creating weathering effects as there are now much better tools available outside Blender for doing this (see the Chapter on Quixel Mixer for example).

The way I use texture painting now is just to apply the basic colours to the model and to create a material ID map, which is easy and quick to do, leaving the more complicated stuff to Quixel Mixer or Substance Painter.

## *Setting Up the Materials*

The prerequisite for using texturing painting is a UV mapped 3D model which has a material applied to it, so the next step is to define the materials. It is important to keep the number of materials to a minimum as this is better for in-game performance, so I try to have just three main materials for the loco body, the bogie (assuming there is only one, otherwise a material will be needed for reach bogie) and the cab that is visible in exterior mode. Others will probably be needed for the coal load, glass parts and perhaps numbering, but these can be relatively small.

I should just mention that the material setups are included in the 'Start Model' that can be downloaded here if you don't want to go through this step. Just merge the materials into your Blend File making sure that the '***Fake User***' flag is set in the '***Append***' dialog.

Start by clicking on the '***Shading***' tab at the top of the screen to enter the Shader Workspace. Then click on the '***+ New***' button to add a new material.

The model at this stage is white as it doesn't have any materials applied to it.

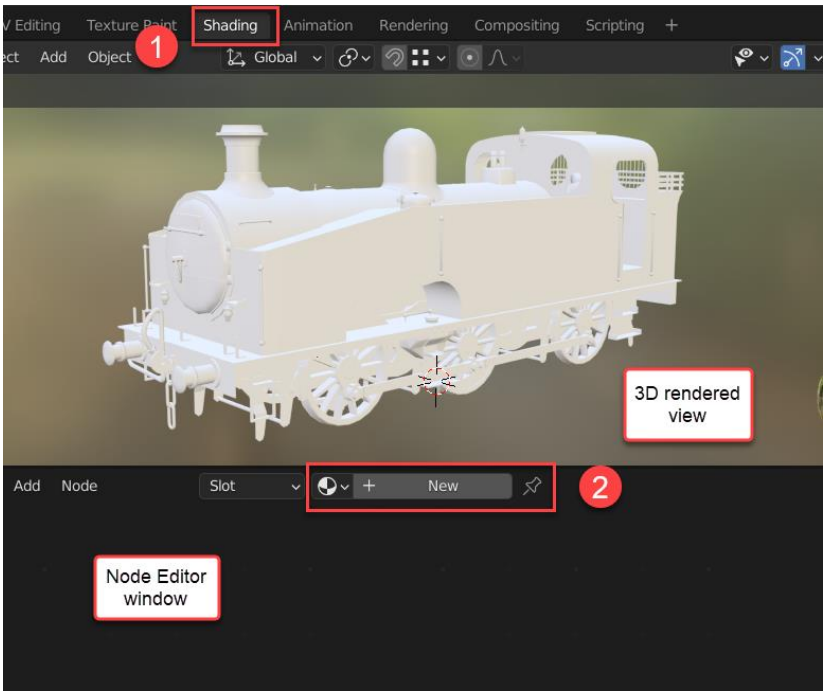
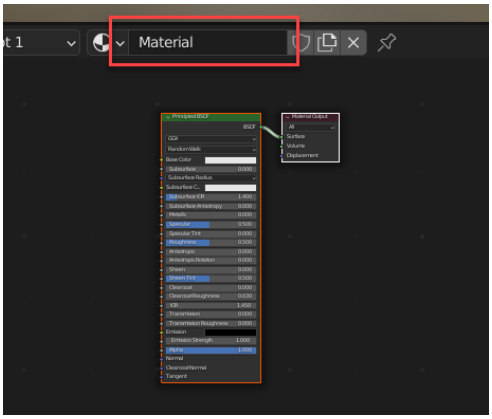



*Figure 14-1: Entering the Shading Workspace and the Node Editor window*




Double click on the Material name field and change the name of the material to '***BODYm.pbrmetal***'. The ***BODY*** part can be anything you like, but the name must end with '***m.pbrmetal***'.

*Figure 14-2: Editing the material name*

Now choose the menu option '***Add > Texture > Image Texture***' to create a new texture node.

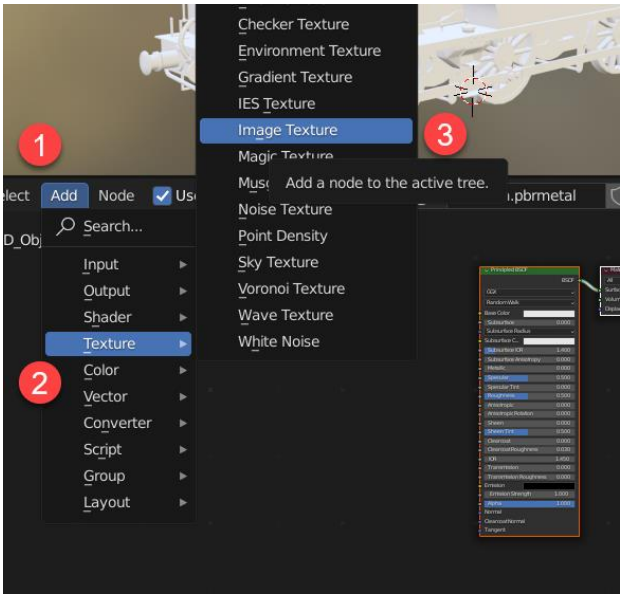
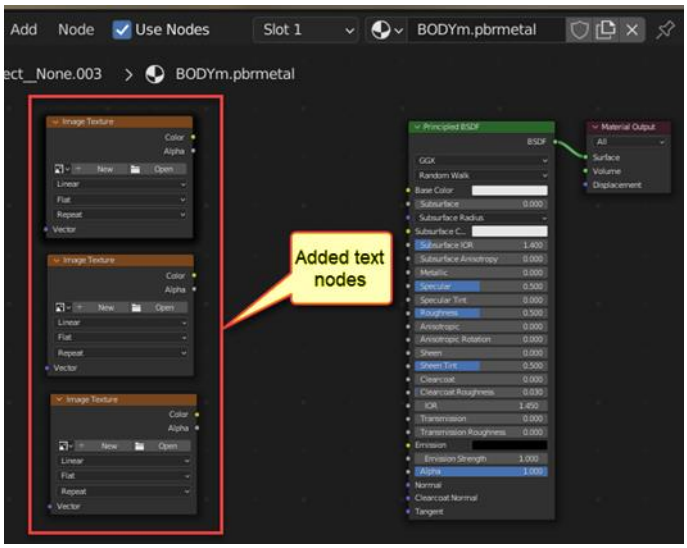



*Figure 14-3 : Creating a new image node*




With the newly created node selected, press '***SHIFT-D***' to duplicate it twice and arrange the nodes as shown here. These nodes will be used for the working textures, not for the final export.

*Figure 14-4: The first three texture nodes added*

Now choose '***Add > Color > Mix RGB***'.

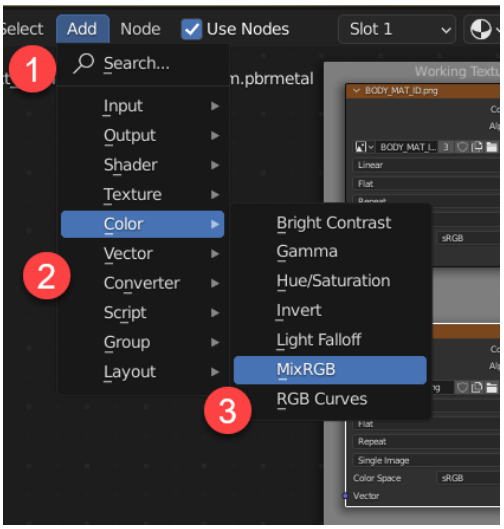



*Figure 14-5 : Adding a 'Mix RGB' shader*

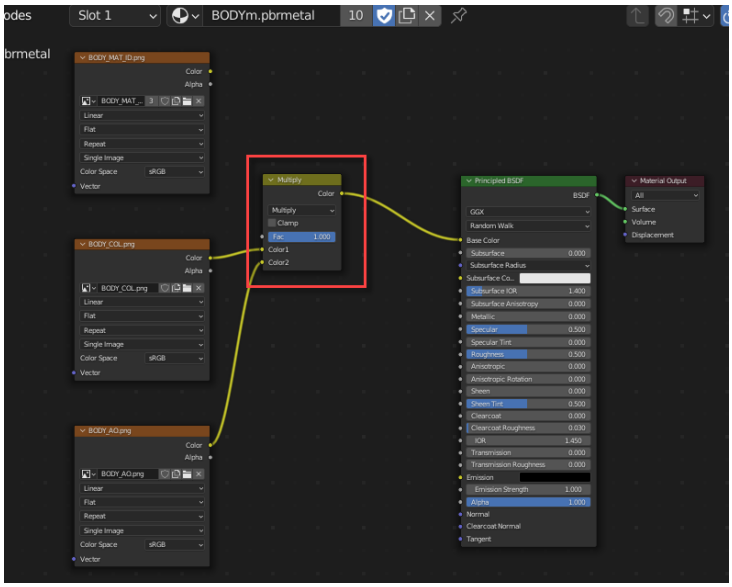



*Figure 14-6: 'Mix RGB' mode added and connected*

The nodes then need to be connected as shown here. All we need to do is to drag between the connection points on the nodes to achieve this.

Set the blend mode of the Mix RGB node to '***Multiply***' and the '***Factor***' to 1.0

By dragging a box around the four nodes and pressing '***SHIFT-P***' a frame can be added around them to the make things a bit clearer.

With the frame selected '***F2***' can be pressed to add a name to the frame.

Note that for these two steps to work the '***Node Wrangler***' add-on (built-in to Blender needs to be activated! This also allows the nodes to be aligned to make things tidier (use the Node Wrangler '***SHIFT-W***' menu).

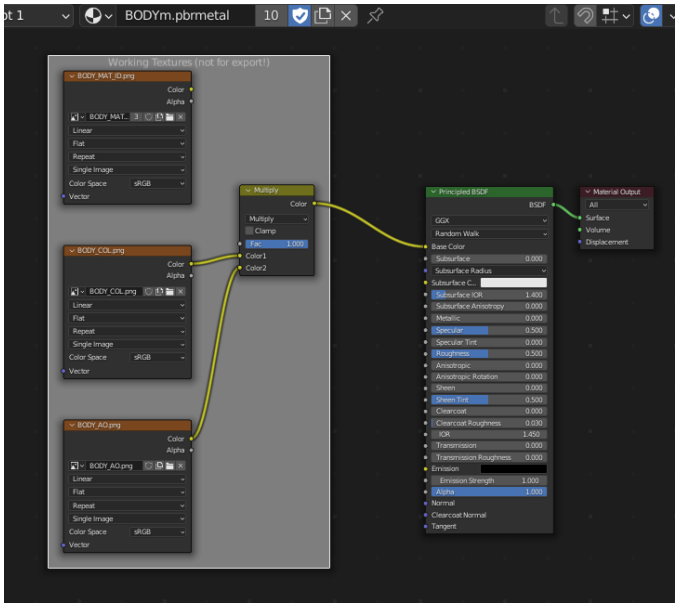



*Figure 14-7: Frame added and named*

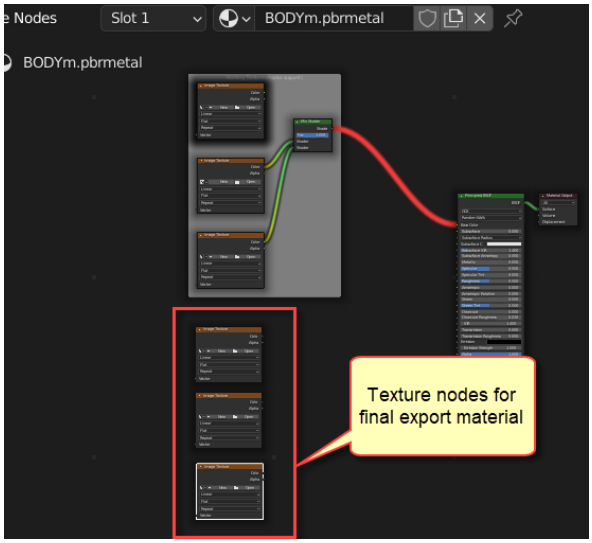



*Figure 14-8: Adding the texture nodes for the export material*

Now create another three texture nodes for the final export textures.

And now add a normal map node and connect it as shown.

This probably all seems a lot of work but having created one material we can copy it to produce the others we need and then create all the textures that the materials need.

This material setup only needs to be done once and can be merged into other Blender files as required.

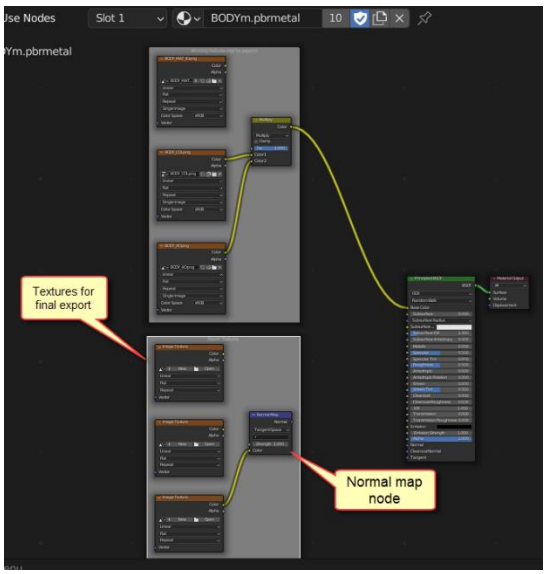



*Figure 14-9: The nodes for the export textures added*

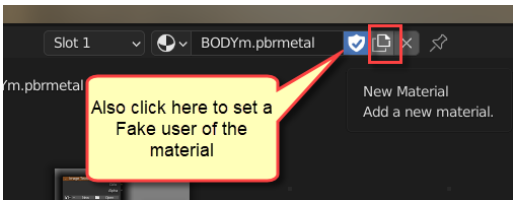



*Figure 14-10: Creating a duplicate material*

Click on this icon to create a new material based on the current one. The new material will be named BODYm.pbrmetal.001 in this instance, this must be renamed to change the 'BODY' part and to remove the '.001' as this will cause problems when importing the asset into Trainz.

Setting the Fake User flag is also important as if nothing is using the material it will be deleted when you save the Blender file and reload it(!).

Now we can create the textures – click on the '***New***' button in this texture node:

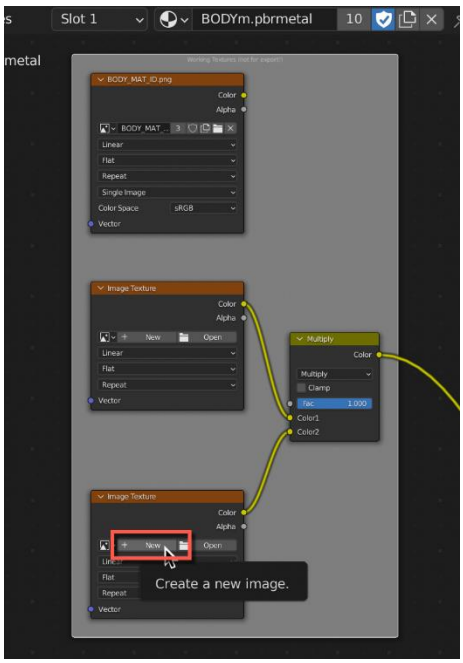



*Figure 14-11: Creating the first texture*

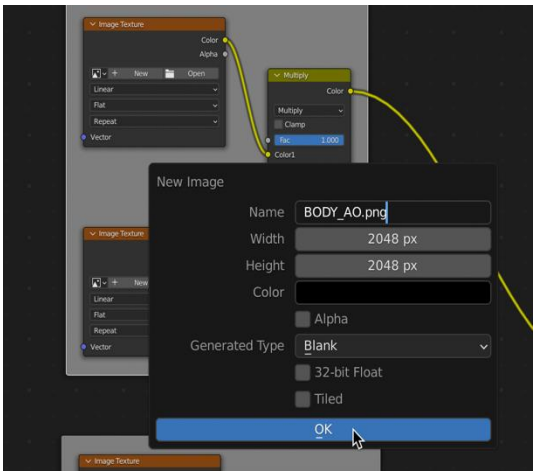



*Figure 14-12: Defining the texture properties*

Enter the texture name and the desired size in the fields. **Note that the dimensions of the textures MUST be a power of 2! If they are not, you are in for a lot of grief at the export stage…**

**Although I've described creating a new texture here, it is of course possible to load an existing one into the texture node (click on the '*Open*' button in the node). You can also just drag a texture file into the Node Editor window to create a new texture node automatically.**

Define the other textures and materials in a similar manner. I would suggest 2048 x 2048 for the main body textures, and 1024 x 1024 for the cab for the external view and the main bogie. These can be changed later, but these should be a good starting point.

## *Applying the Materials to the Mesh Objects*

Now we must apply the material to all the mesh objects so that we can see the results. It helps to enter the Texture Painting Workspace by clicking on the appropriate tab at the top of the screen first, as this automatically changes the 3D viewport to material preview mode so that the materials can be seen.

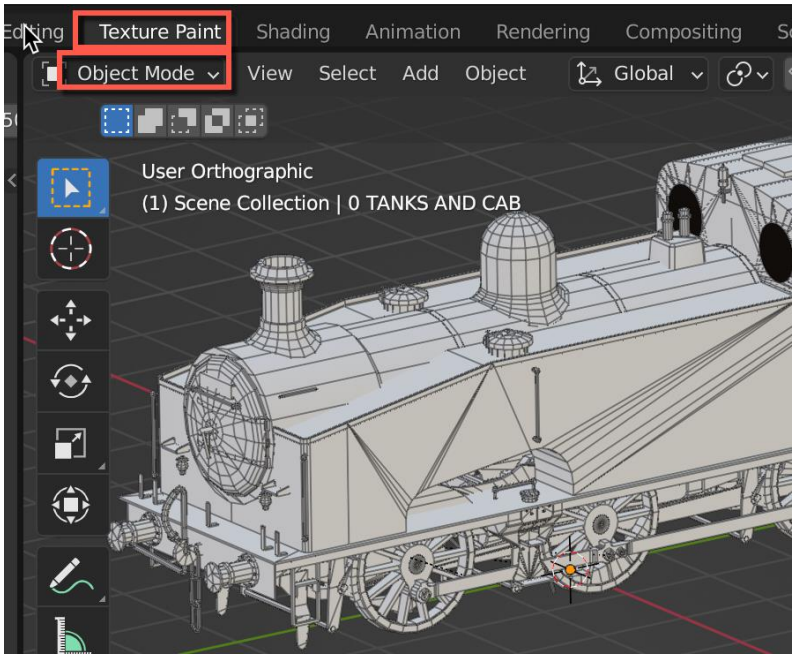



*Figure 14-13: The Texture Painting Workspace*

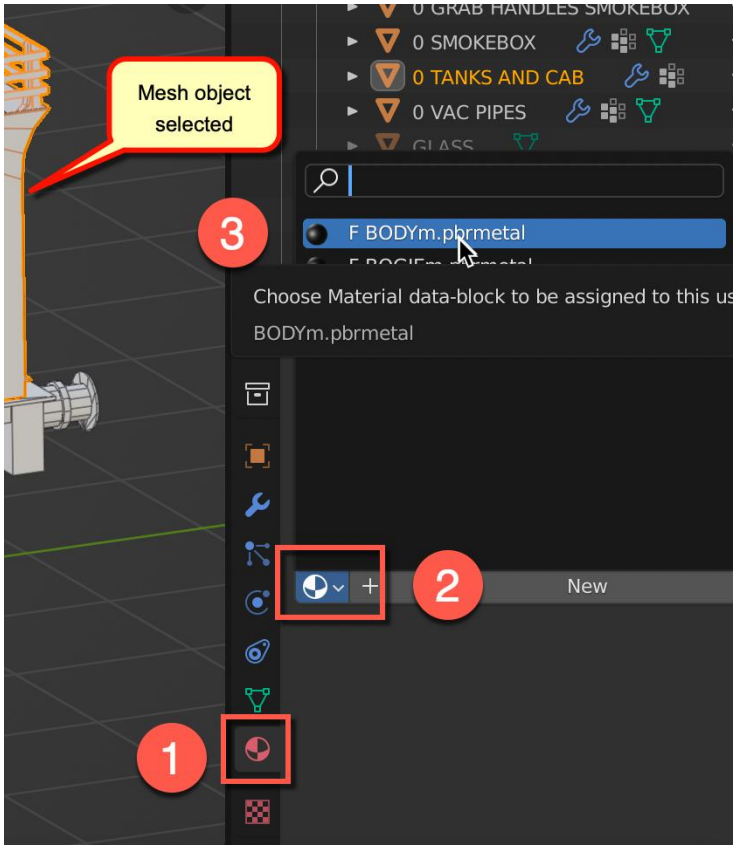



*Figure 14-14: Assigning a material to a mesh object*

Click on one of the mesh objects in the 3D view, select the '***Material Properties***' tab (1), open the Material Browser (2), and finally select the appropriate material from the drop-down selection (3).

The material will be applied. This should be repeated for each mesh object and material, so that each object has the correct material.

Once a material has been applied to one object you can copy this to other objects by selecting the objects that should have the material applied and then the object that has the material to use. Press ***CTRL-L*** and choose '***Link Material***'.

## *Painting the Colour Texture*

First, make sure that the texture that should be painted on is selected in the Shader Editor. Just left click on the node, it will then be highlighted as shown here.

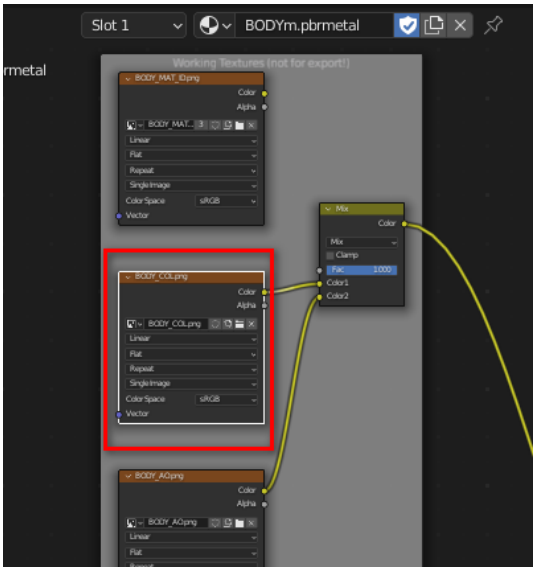



*Figure 14-15: Selecting the active texture to be painted on*

A couple of colour palettes for use when texture painting can be downloaded in this *.blend file here, just merge the palettes in to your model to make them available. One ('***basis***') is intended to be used to create the basic colouring of the model and includes common colours for brass, steel, copper and so on. The other ('***Material ID***') is intended for the creation of the material ID map.

The material ID map allows you to paint the mesh quickly in Quixel Mixer or Substance Painter without tediously selecting the individual faces. You can assign the colour yellow in the ID map to a brass PBR texture for example which makes painting the mesh far faster.

Assigning the Vertex Groups has already been covered, so let's see this in action. I suggest painting each mesh separately when doing this for the first time as then it is easier to spot if any faces have been missed or have the wrong Vertex Group assigned. The fastest way to paint the model though is to temporarily join all of the meshes that have the same material into one mesh, so if you do this save the *.blend file first so that you can get back to a version where the objects are not joined together.

## *Selecting the Texture to Paint*

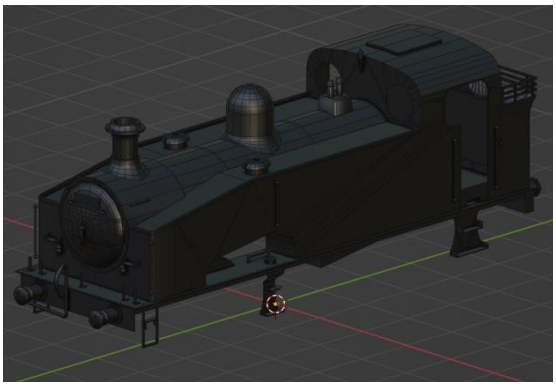


*Figure 14-16: Body mesh(es) with BODYm.pbrmetal material applied*

Starting with the material ID map, here I've applied the BODYm.pbrmetal material.

The mesh has turned black because the BODY_MAT_ID.png file is dead black as we haven't painted it yet.

Now click on the ‘Shading’ tab at the top of the screen, and in the Shader editor window click on the ‘***BODY_MAT_ID.png***’ node to select it. This ensures that we are painting on the correct texture.


*Figure 14-17: BODY_MAT_ID.png node selected*

Next switch to ‘***Texture Paint***’ mode by clicking on the tab at the top of the screen and select Object mode. Now select the body mesh and enter Edit mode.

## *Painting Only Selected Faces*

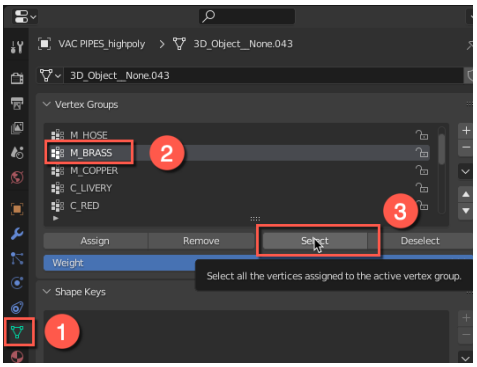



*Figure 14-18: Choosing a Vertex Group*

Choose the ‘***Object Data Properties***’ tab (1) and pick one of the Vertex Groups (2). Then click on the ‘***Select***’ button.

The vertices assigned to that Vertex Group will be selected.

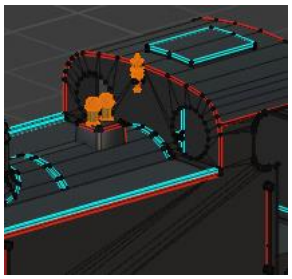
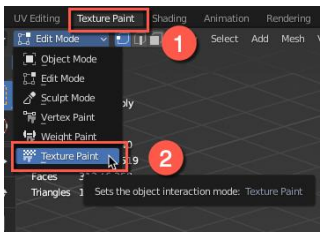
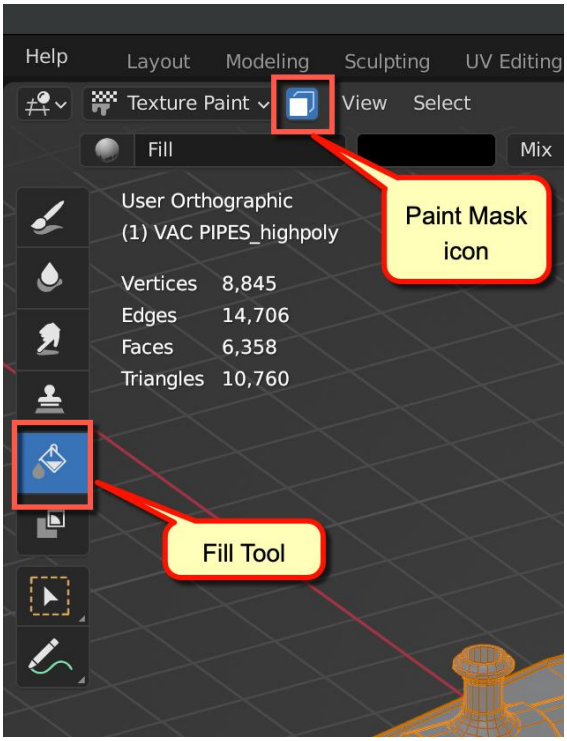


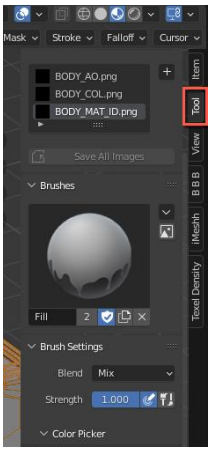
*Figure 14-19:Vertices assigned to the Vertex Group are now selected*




Switch back to Texture Paint mode by clicking on this drop-down menu.

And finally, click on the '***Paint Mask***' icon and the '***Fill***' icon. The '***Paint Mask***' icon allows us to paint only the selected faces when it is selected. If it is not selected then the painting operation will affect the whole mesh, which is useful sometimes.





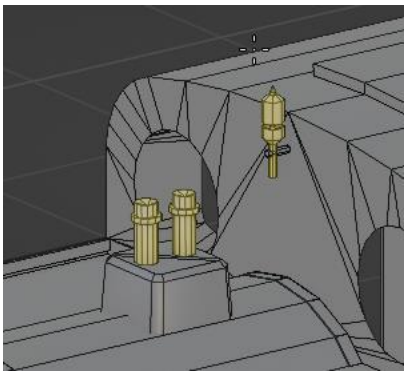

Now click on the '***Tool***' tab on the right side of the 3D Viewport (press '***N***' to open the tool bar first if the tabs are not visible) to open the dialog box for the active tool, in this case the texture painting tool.

*Figure 14-20: Opening the texture painting tool dialog*

## *Choosing the Paint Colour and Painting the Mesh(es)*

Here we can pick the colour that we want to paint with, for brass I use yellow, but you can use any colour in the palette. In the colour palette section choose the '***Material ID***' palette, the colour to paint with and set the '***Bleed***' value to 1px. 'Bleed' is the number of pixels that will be painted outside the actual area covered by the UV map. This reduces the chance of having visible seams in the final texture.


*Figure 14-21: Setting the colour to paint with*

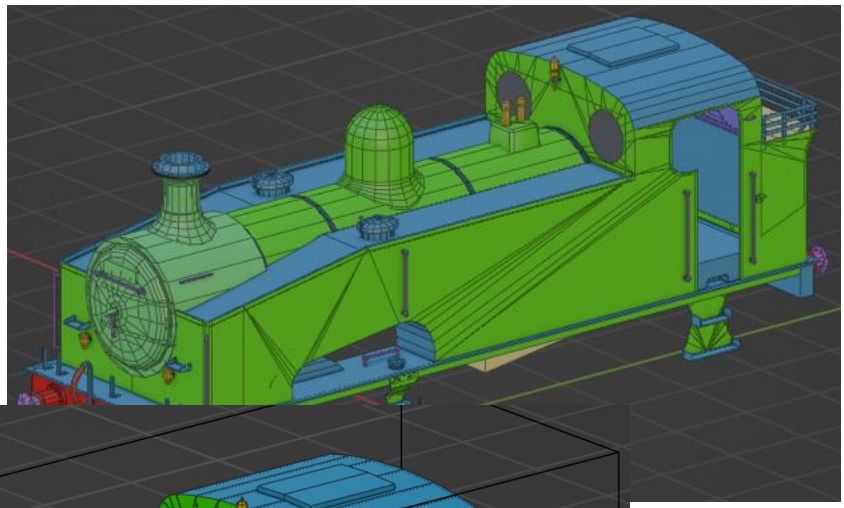


The selected faces will now have the colour that you painted with.

*Figure 14-22: 'Brass' parts of the mesh painted*

That all probably seemed very long winded, but it is far quicker to do than describe. Deal with the other Vertex Groups in the same manner and paint the whole mesh.

Here is the body mesh with the BODY_MAT_ID.png texture completely painted.



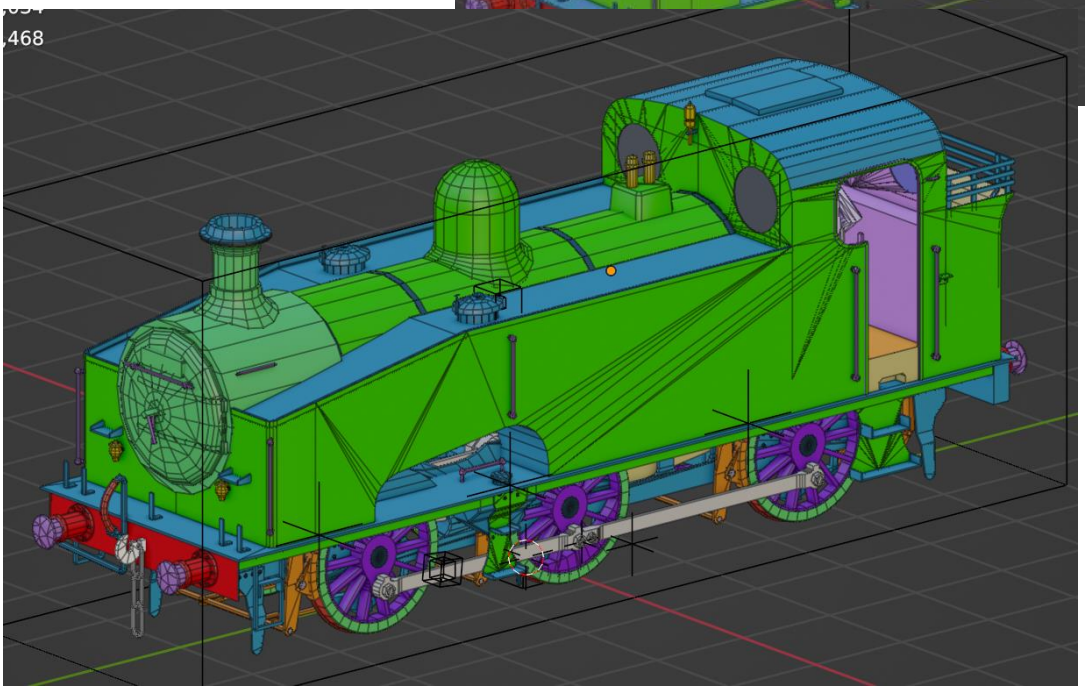
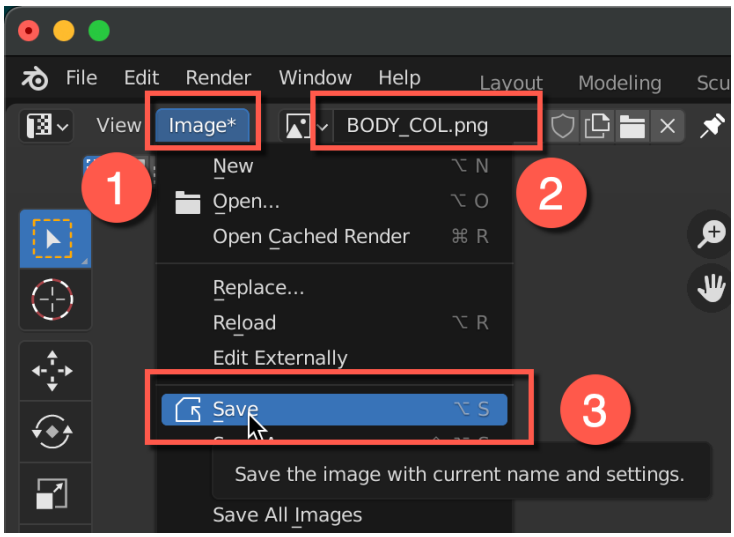


*Figure 14-24: All Material ID maps completed*

Working with Vertex Groups in this way allows you to completely remap the mesh if needed and to quickly repaint it. Remapping means of course than the UV islands will move around on the map. Continue in the same vein until all the needed Material ID maps have been painted.

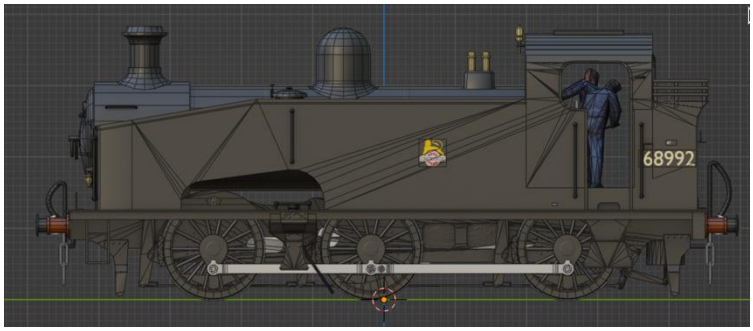
Don't forget to save the painted image when you are done, if the image has been modified there will be a star next to the '***Image***' menu button (1).

You will be prompted to save modified images if you try and close Blender without doing so, but if you don't then they are gone for good…




The next step is to paint the colour maps, so select the '***BODY_COL.png***' node in the Shader window, switch back to the '***Texture Paint***' tab at the top of the Blender window and paint this texture as well, using the 'basis' colour palette or one that you have created yourself.

This texture will provide the base colours for the texturing in Quixel Mixer or Substance Painter and provides some visual feedback at this stage as to how the model will look in Trainz. Also, any lining, insignia or numbering can be added to this so that the weathering (if any) will affect these as well.




Once the bogie and xcab colour textures have been created the result will look something like this.

*Figure 14-25: Colour and AO textures applied to the whole model*

## *Adding Modulation to the Colour Textures*

Something else that can be done to the colour textures to break up the solid colours is modulation, which is a technique used by modellers of physical models, especially of military subjects. This is simply adding a lighter tone in the middle of large surfaces to create a sort of paint fading effect.

Here I have painted areas of a separate texture with white to create this effect. You don't need to be very accurate with this!

Obviously, this is far too strong, so the transparency of the layer needs to be increased to almost 100% to create a subtle effect.

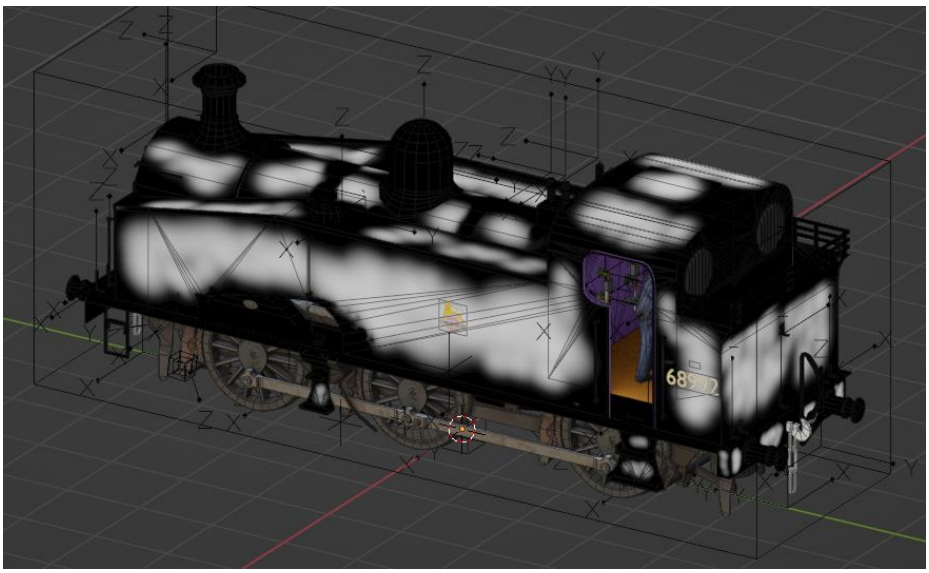
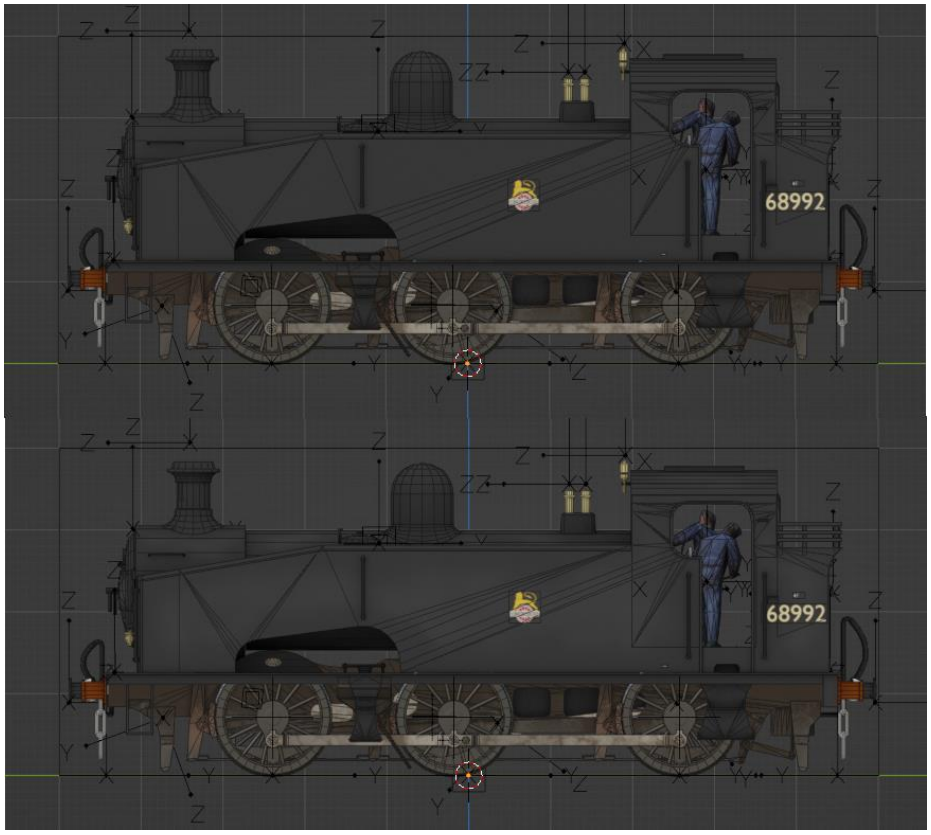


*Figure 14-26: Body with modulation added at full strength*




This is the result with the transparency increased. I think this is a worthwhile improvement and is especially effective with black locos which tend to look very flat without it.

The modulation layer can easily be added to the final textures in Quixel Mixer or Substance Painter.

*Figure 14-27: Body with no modulation (upper) and final modulation (lower) for comparison*

This is the node tree used to add the modulation layer and to control the transparency.

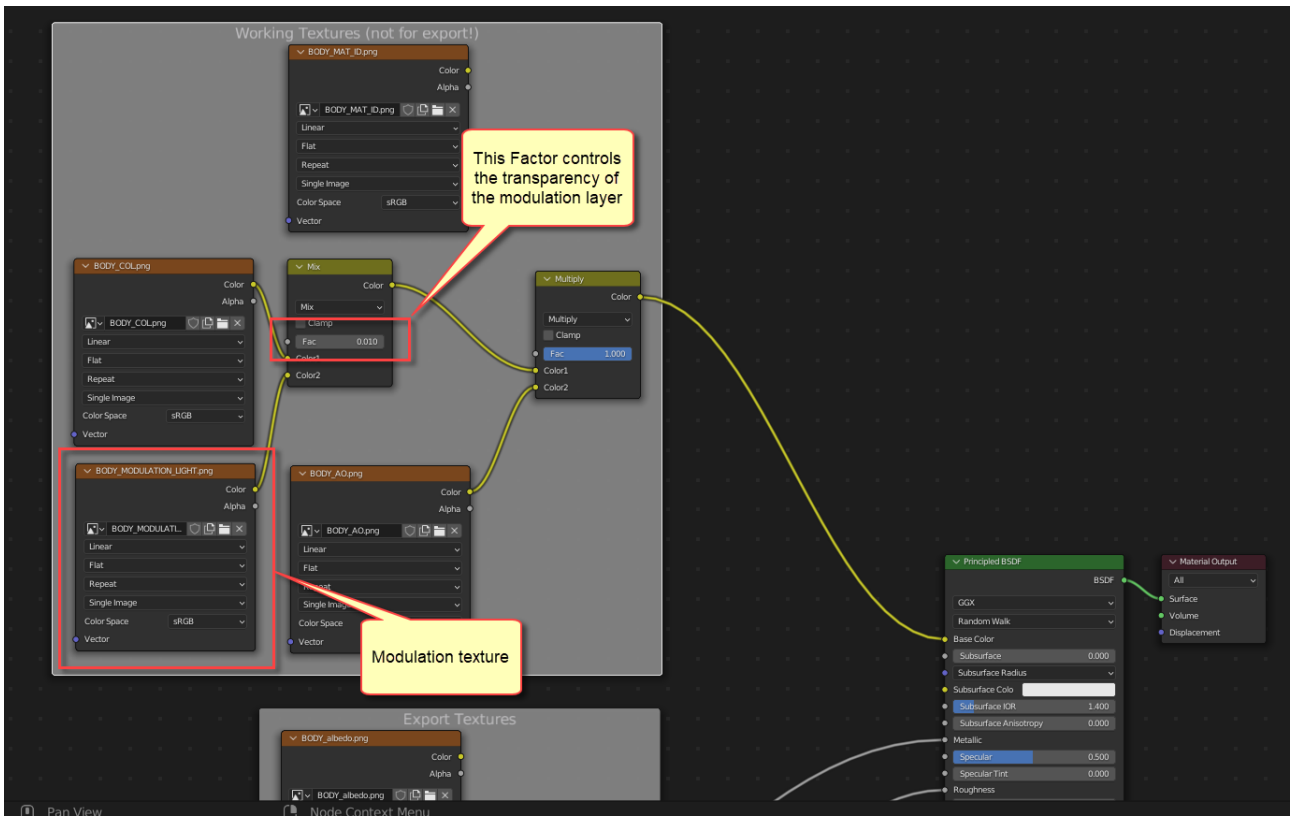



*Figure 14-28: Node tree for the modulation effect*

Although in this case I painted the modulation layer directly on the model using Texture Painting, I think it might be more controllable if the modulation layer is created in GIMP using the colour texture as a guide.

**[To be continued…]**

# 15. PBR Texturing Using Quixel Mixer

## *Why Quixel Mixer?*

Texturing Trainz models has become a lot more complicated with the introduction of PBR (Physically Based Rendering) materials which provide for a much more realistic appearance for the assets. While it is possible to do all the texturing inside Blender (with or without using paid add-ons), it tends to be a lot of hard work. While it is possible to cut down a tree using an axe, it is a lot faster to use a chain saw if you have one.

Quixel Mixer is free, and it is easy to get good results without knowing much about it. The main rival to Quixel now (2022!) is Adobe Substance Painter which is the industry standard, but this comes with a hefty price tag. I would strongly suggest using Quixel unless you have access to Substance Painter – both have very similar workflows, so it is relatively simple to transfer from one to the other.

**[Lots more to come here…]**

## 16. LOD (Level of Detail) Meshes

LOD stands for level of detail and basically means that as you get further away from a model the number of polygons can be reduced to make less work for the graphics card. Less polys means higher frame rates and a smoother performance in game, but unfortunately this means creating extra meshes with less polys which has led to this being rather unpopular with content creators.

Luckily Blender has the possibility of creating LOD meshes almost automatically although for the very best results you need to help it along a bit.

There have been various threads discussing LOD on the N3V Forum, one of which can be found here.

Note that some of the postings from content creators are aimed at outwitting the system so that the highest poly model will be visible for longer – I strongly urge you not to follow this advice and instead take note of what the N3V programmers have to say about this. I suspect they know more about what is good for frame rates than we do.

To save you going through the thread and picking out the relevant bits I'll re-quote what they had to say here:

*[ZecMurphy]*

*The stitched LOD system for scenery is designed to automatically adjust the transition distance depending on your detail settings, and the scene load. However, it does require you to be a little more careful with your LOD meshes, since you'll need to design them to work with this LOD system's distances (Rather than the other way around, as with the lm.txt files). This does mean you'll need to adjust/tweak LOD meshes so the transitions are smooth (such as by reducing detail, rather than removing it - or ensuring that it transitions to a good texture representation with normal maps).*

*Even if you personally experience poor performance, you should test LODs with the detail sliders at maximum in an 'average' scene. This way you can ensure that the transitions are least noticeable at the highest detail settings. Your transitions generally should become more noticeable at lower detail settings, as this is the way that LOD-vs-detail works (lower detail settings cause LOD transitions to occur earlier). You may also see earlier transitions when there is very high load as well. Trying to work around this isn't appropriate, as you are essentially preventing the detail sliders from working correctly.*

*I generally actually test the transitions on a blank baseboard to ensure they occur 'smoothly' at highest detail, then do tests in a 'normal' scene to see if I notice them (generally there's not too much of a difference between the two but your results may vary).*

*[WindWalkr]*

*(Responding to the suggestion of adding a small animated mesh to a scenery asset to allow control of the LOD transitions via a *.lm.txt file)*

*Don't do this. Performance will be horrible. Not only does it add the cost of animation (which is significant) and the cost of a sub-mesh (which is significant) but it removes any chance of the stitched mesh system optimising your mesh. For frequently used objects, this will increase the cost by hundreds of times (ie. the difference between showing 100, or 10000 of the same objects on the screen at the same performance level - or to put it another way - the difference between getting 60fps or 0.6fps with the same visual results.)*

*You should not be using LM files for simple scenery objects. Ever. Performance will be*

*horrible. LM is intended for objects which already have to pay the cost of animation or movement.*

*[WindWalkr]*

*(Responding to the idea of using part of the main texture map to texture a low LOD mesh)*

*There are two options which perform reasonably well:*

*1. Use a different texture for the lower LOD(s). This has the downside of preventing the higher LOD and lower LOD from being stitched together, but in practice this is often not a big concern. The higher LODs are often sufficiently detailed that stitching provides minimal benefit, and the lower LODs are potentially numerous enough that stitching is a big win there.*

*2. Use a larger section of the main texture for the lower LOD. This has the downside of "wasting" texture space when the object is at high LOD but means that the texture detail can be dropped away when only the low LOD is in the scene.*

*The thing to avoid is specifically using a small section of a large texture. This forces the game to keep the texture at high detail, adding a significant load on the GPU. (Or, alternatively, the game may choose to reduce texture detail resulting in a blurred-out low LOD.) The exact concern here is the texels-to-meters ratio- you don't want a small number of texels mapped across a large distance. The general rule that the texels-to-meters mapping should remain fairly consistent for a given asset. Small variations are unavoidable and acceptable, but large variations can cause unexpected problems in a variety of areas.*

*[WindWalkr]*

*(Responding to the idea of having a large monolithic asset containing 48 houses to improve performance)*

*SEPARATED*
*48 houses as 48 assets.*
*Camera near the row, so a few of the houses are at top LOD.*
*Let's say this LODs as follows: 2x5k, 10x1k, 20x500k, 16x50.*
*That's 31k polygons total, or around 1-2 draw calls.*

*TOGETHER*
*48 houses as a single asset*
*Camera near the asset, so the asset is at top LOD.*
*That's 48x5k*
*240k polygons total, or in the range of 10-50 draw calls.*

*Caveats:*
*LOD system has more work to do in the first example.*
*This is a worst-case scenario; if the models were always all at lowest LOD, the combined mesh would cost slightly less to set up and would not have any polygon count penalty.*

*The 48 houses have an identical arrangement every time they're placed. This can potentially get visually boring with repetition and reduces the route builder's ability to customise placement or to otherwise use the model in more restrictive requirements (eg. where only 47, or 30, or 2 houses are required.)*

*The identical arrangement applies in the height dimension as well, which means that the 48 houses cannot be used anywhere where the ground is sloped or uneven.*

*I won't say that there's no place for this kind of thing, but all else being equal, a well-built set of individual houses can perform just as well, look as good or better, and are an awful lot more flexible.*

Phew. My take on this is that you should use as few materials as possible for your model (note, not textures) and that the same material should be used for all LOD levels. This way you are allowing Trainz to control the detail of the models via the detail slider so that people can choose to reduce detail to increase performance.

You might also want to read this comment in the Trainz Developer Blog on the subject of LOD.

Having got all of that out of the way we should first consider when the best time is to build the LOD meshes and then deal with how to do it. I'd suggest completing the highest poly mesh first, certainly with as much detail as you think you need and copying this to start creating the LOD0 mesh. UV Map this model and generate the AO and normal maps, then go on to create the LOD1, LOD2 and LOD3 meshes. Baking the AO map is a good way of making sure that all the faces are pointing in the correct direction and that you don't have any doubled-up faces in there (as doubled faces or reversed ones will show up black in the AO map).

I'd also suggest putting all the LOD models on one layer but separated from each other by 5 Blender units in the Z direction. This is so that you have a direct visual comparison between the LOD models, and it also makes it easier to transfer sub meshes between the LOD levels.

Once I've finished creating the LOD meshes then I'll move them down to the correct position so that I don't have to keep moving them every time I want to export them.

Here you can see an example, LOD0 has 17903 polys, LOD1 has 8731 polys and LOD2 has 2833 polys.You should be aiming at a reduction of at least 20% between the LOD levels as otherwise the overhead of swapping the meshes over can become significant (and you'll get an error when importing the asset into the Content Manager). Better would be a 50% reduction between the levels, but this can sometimes be difficult to achieve.

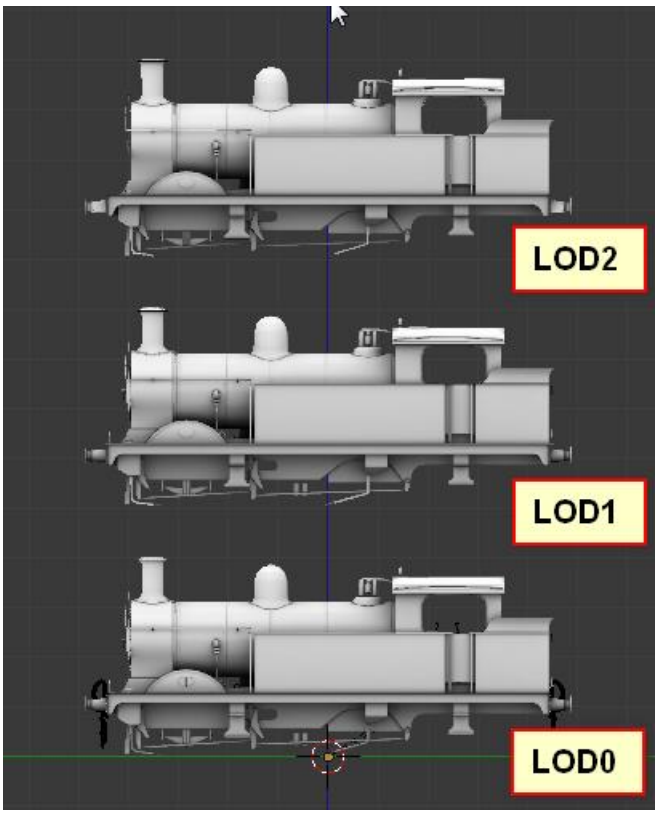



*Figure 16-1: LOD meshes in Blender*

It might be hard to believe that the LOD2 mesh only has 2833 polys, but it does, and here it is in close up. Normally you wouldn't get this close to the mesh in game.

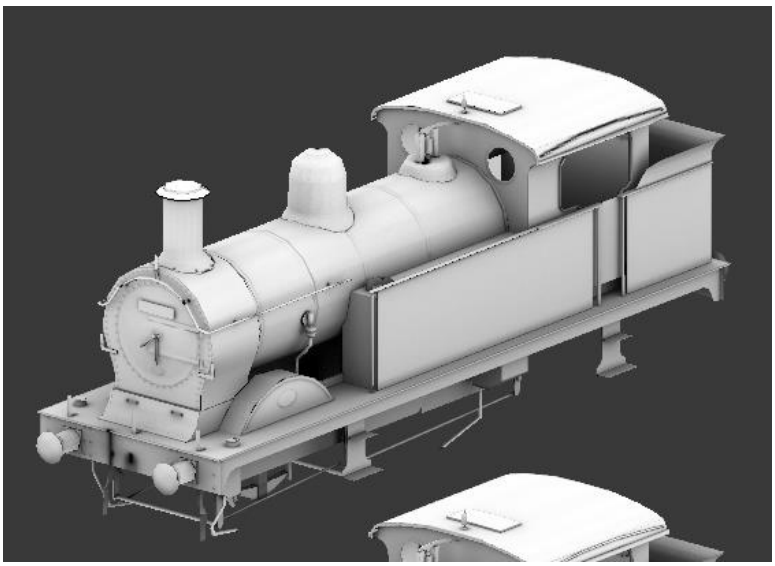


*Figure 16-2: Ex-SECR H Class LOD2 mesh (2833 polys)*

I'd do things a little differently to this example these days, I'd add a LOD3 mesh with 300-500 polys for the loco body which would also include a crude representation of the bogies, the real bogie meshes being turned off at the LOD3 changeover point. The LOD3 mesh would be visible at about 1Km away, the model being extremely small at this point. At a range of about 3Km I'd turn everything off effectively making a LOD4 mesh with 0 polys. This is vital for T:ANE as the lowest poly mesh must be under 500 polys (this includes all attached meshes as well).

The LOD3 mesh would be the second mesh I'd make once the LOD0 high detail one was complete. The relevant faces would be copied from the LOD0 mesh and moved 15 Blender units in Z, but it's only necessary to copy the faces which define the outline of the model so handrails, pipes and small details would not be included. After all, if the model is only a few pixels in size which it would be at 1Km range you don't need detail, but you will notice if the mesh suddenly disappears.

The LOD3 mesh would also have a decimate modifier applied to reduce the poly count further. This avoids the need to create low poly versions of the high poly mesh. To show thing in action I'll use the dome of the H Class tank as an example – the LOD0 version of this has 208 polys. Here the dome has been separated from the loco body and selected in object mode.

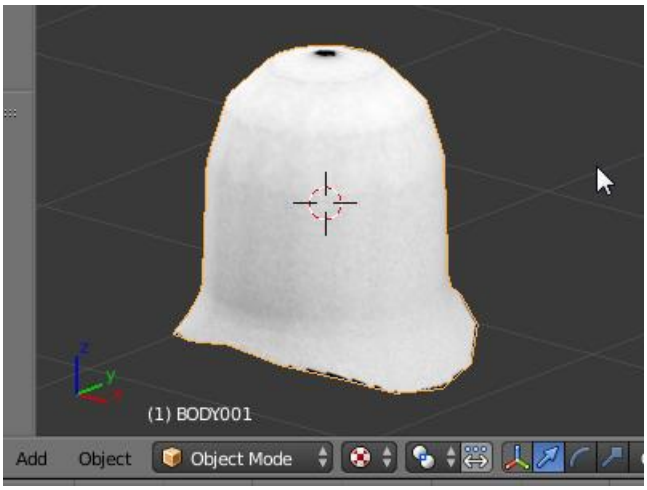



*Figure 16-3 Dome object separated and selected (208 polys)*

Make a copy of the object with ***SHIFT-D*** and then move it 5 Blender units up with ***G***, ***Z*** and ***5***. Attach the original dome nesh back to the body mesh to keep things tidy.

Now select the copy of the dome mesh and In the modifier tab select ***'Add Modifier'*** and then choose '***Decimate***' from the list.

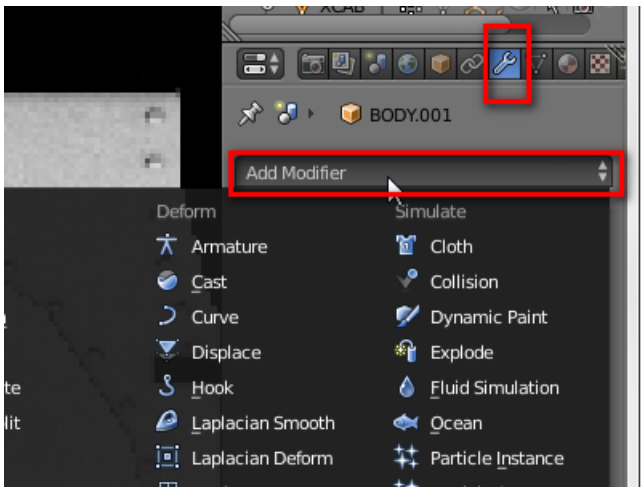



*Figure 16-4: Adding a modifier to the object*

Make sure '***Collapse***' and '***Triangulate***' are selected and then adjust the '***Ratio***' field to reduce the polys.

Fairly obviously a Ratio of 0.5 means halving the poly count.

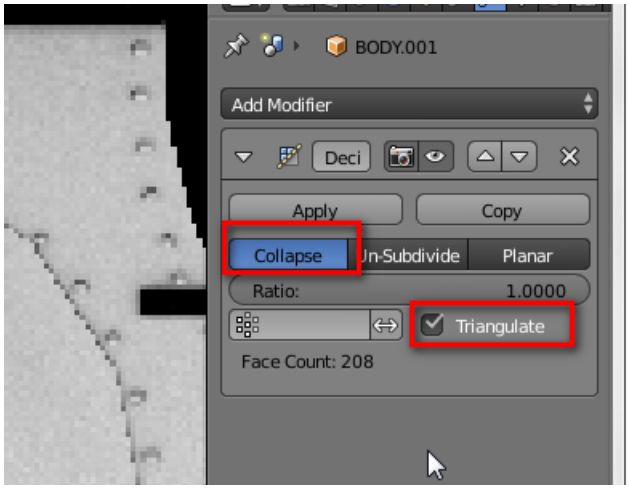



*Figure 16-5: The decimate modifier options*

Here I found that a Ratio of 0.48 gave a poly count of 98 and didn't alter the shape much.

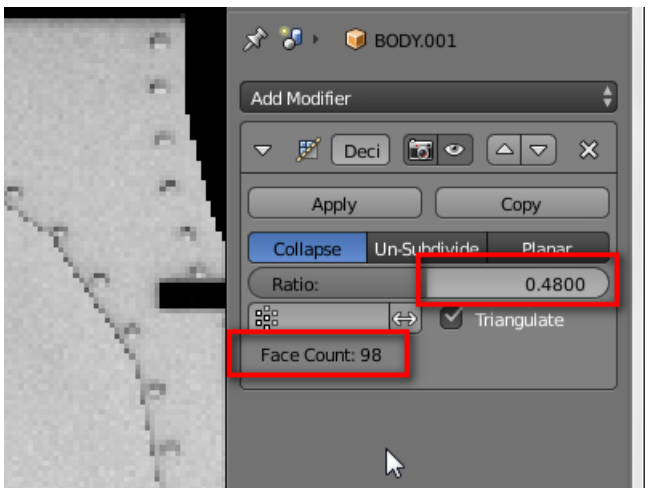



*Figure 16-6: Ratio 0.48, poly count 98*

As you can see here.

*Figure 16-7: Result with 98 polys*

The options for the modifier can be adjusted as much as you like until it has been applied, which it has to be to make the poly reduction permanent. It can also be applied to the whole body mesh in one go, but this is likely to distort important parts of the mesh. If you go too far with the poly reduction you could end up with something like this:

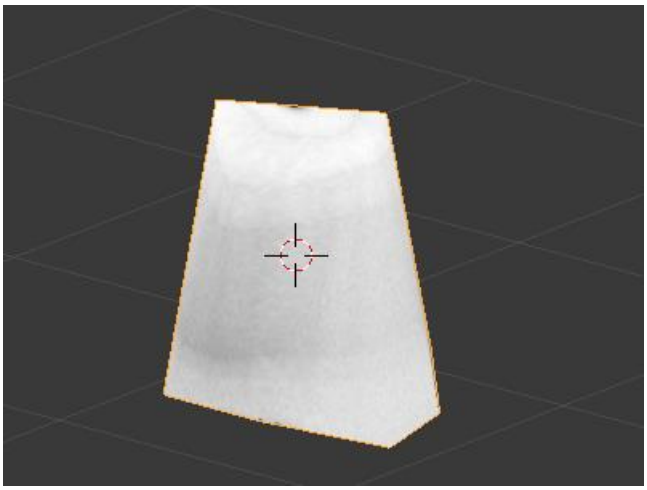


*Figure 16-8: Result with only 14 polys*

It's worth experimenting though, in the case of the LOD2 mesh of the H Class tank I was at over 5000 polys and the decimate modifier knocked off over 2000 polys without making changes that might be visible in game.

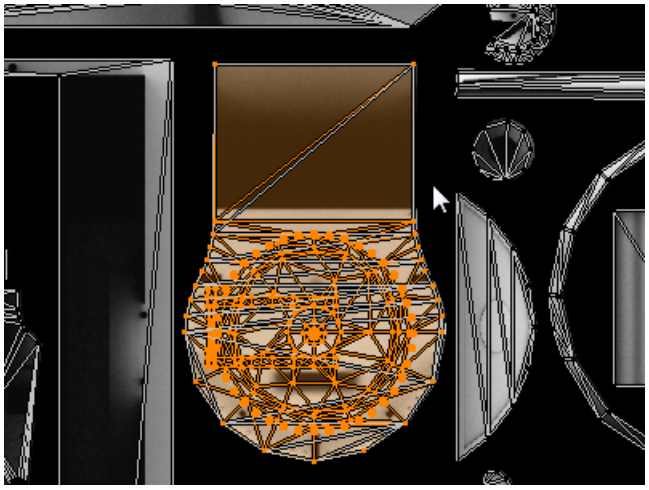
The more alert among you many have also noticed that although the dome now has far fewer polys the mapping has been retained which saves a huge amount of work and makes the LOD transitions much harder to spot in game as the AO map of the high poly model has been applied to the low poly mesh.

This means that creating the LOD meshes is much easier than it is in GMax where you'd need to create low poly meshes and try and map them to the high poly AO map manually. There is no real excuse for not making LOD meshes any more.

You need to take a lot more care over the LOD1 mesh though to avoid visible LOD transitions. Here I'd firstly remove any surface detail that could be baked into the AO map such as rivets and nuts and bolts – this alone may give enough of a reduction for this LOD step although as ever you should remove everything you can if it won't be noticed.

One thing you need to watch when removing detail is that you need to replace it with something in the texture. If you just remove the rivets from the LOD1 mesh you'll be left with black spots where they were for example, so you need to bake the rivets into the texture. In the highest LOD mesh the 3D rivets will cover the baked versions.

**[Update needed to cover low to high poly AO baking…]**



I applied a planar map to all the faces so that they were all in the correct position relative to each other.

*Figure 16-9: Planar UV map of smokebox front*

For the smokebox front I baked these faces using the '***Mesh > Sort Elements > Cursor Distance***' trick described earlier to ensure that the overlapping faces were baked in the correct order.

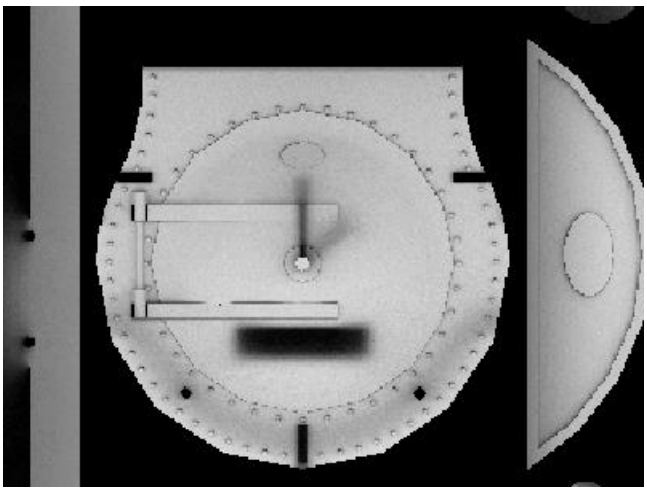


*Figure 16-10: AO map of smokebox front*

Here's the result in 3D:

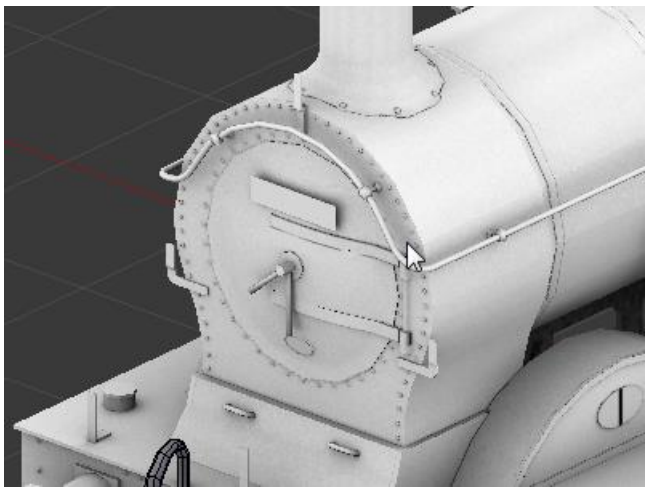


*Figure 16-11: The baked smokebox front on the 3D mesh*

I wouldn't make a hole in the smokebox front plate these days, this makes things harder when doing the LOD meshes and makes for more polys. For this loco (which is a very old model, over ten years old at the time of writing) I made a second smokebox front plate without the hole for the LOD2 mesh. If you are thinking straight when building the mesh, you won't need to do this step.

The LOD2 smokebox front plate is completely flat and very low poly.

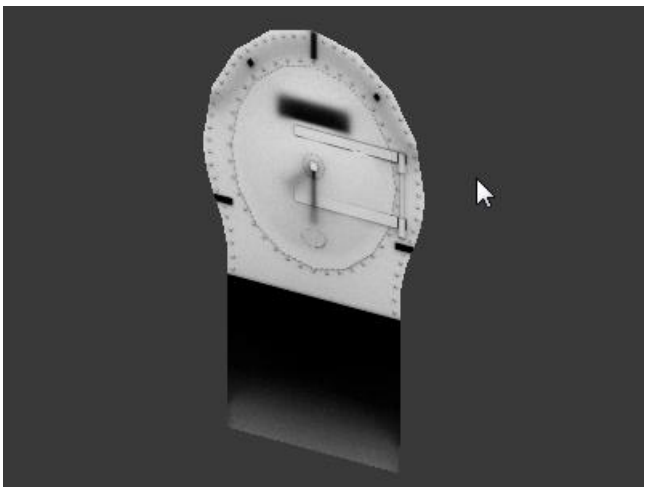


*Figure 16-12: The LOD2 smokebox front plate (19 polys)*

All the LOD meshes should be complete and AO baked before starting to paint and weather the skin as it is harder to match the existing weathering on new parts of the mesh – or to put it another way it's much easier to weather the whole mesh in one go rather than doing it piecemeal.

One thing to watch out for is not to create lower LOD meshes for boilers and saddle tanks in the first LOD level (LOD1). This is because the shading variations caused by the different smoothing of the lesser number of polys can be obvious. This applies to any relatively large, curved surfaces.

I'd suggest only being aggressive with the LOD on curved surfaces for the lowest (under 500 poly) version – when this model is on the screen it should be very, very small and only the general colour and the silhouette is important.

# 17. Exporting the Model

**[To be expanded…]**

## *Exporting to FBX*

For TS19 and later it is necessary to export the meshes as FBX files, as this supports PBR materials which look much better than the Trainz legacy materials. It is well worth doing. this as the difference is like night and day and it is not as difficult as it might seem.

To export as FBX you don't need to install any extra add-ons as support for this is built into Blender. You do need, however, to edit the system preferences as shown here.

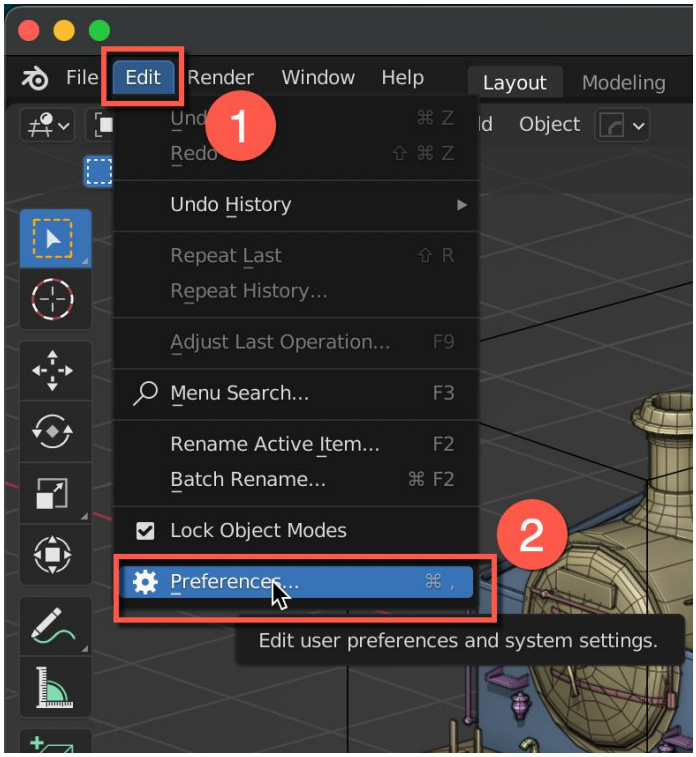



*Figure 17-1: Editing the system preferences*

Activate the add-on by selecting the options shown below.

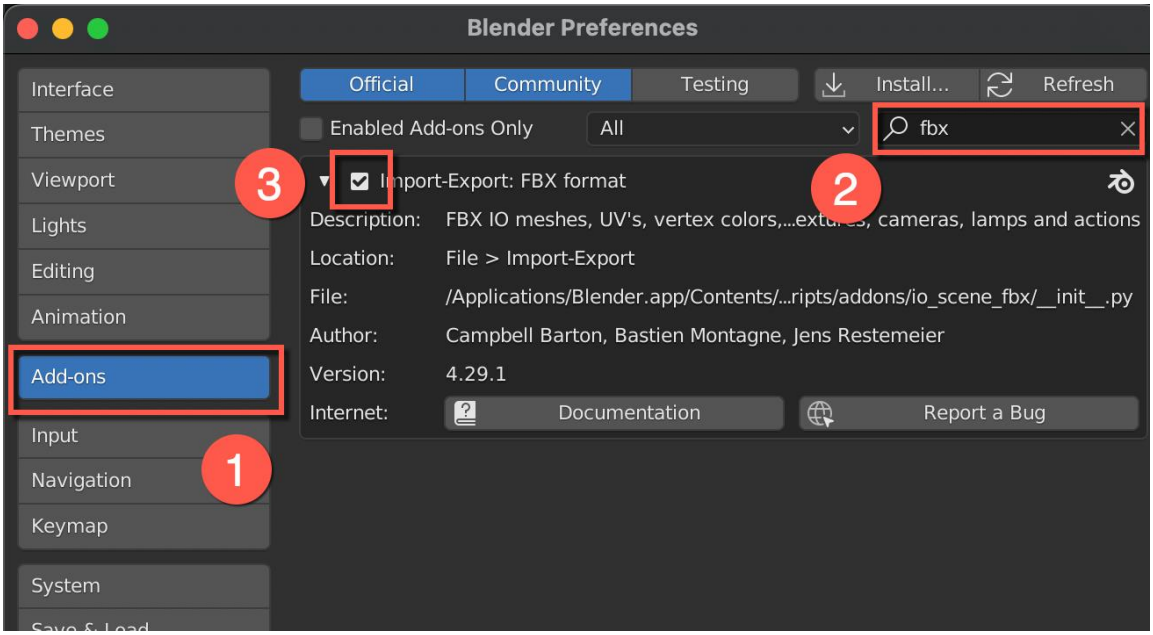



*Figure 17-2: Activating the FBX export add-on*

Once this has been done you will have the option to export to FBX by choosing '***File > Export > FBX (.fbx)***' as shown here, but before doing this it is best to select the objects that you want to export.

Hopefully you have organised your project with nice collections for each asset. That you want to export to Trainz, in this case you can just hide the collections that you don't want to export and then select all the visible objects by choosing '***Select > All***' or by pressing '***A***'.'

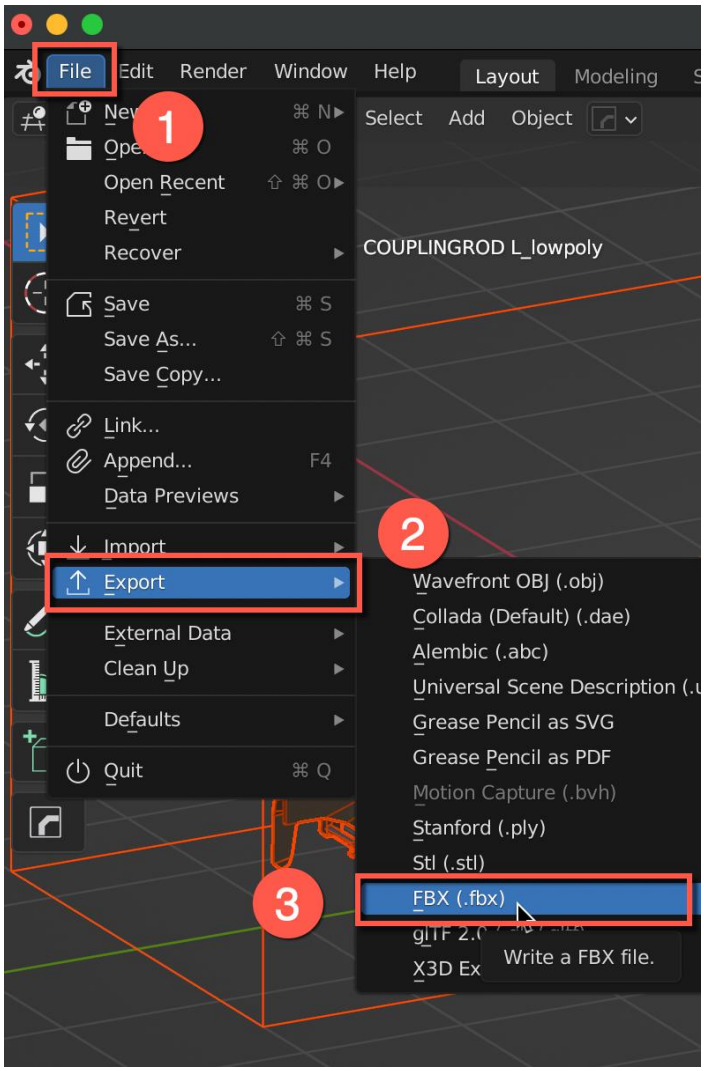



*Figure 17-3: Selecting the type of file to export*

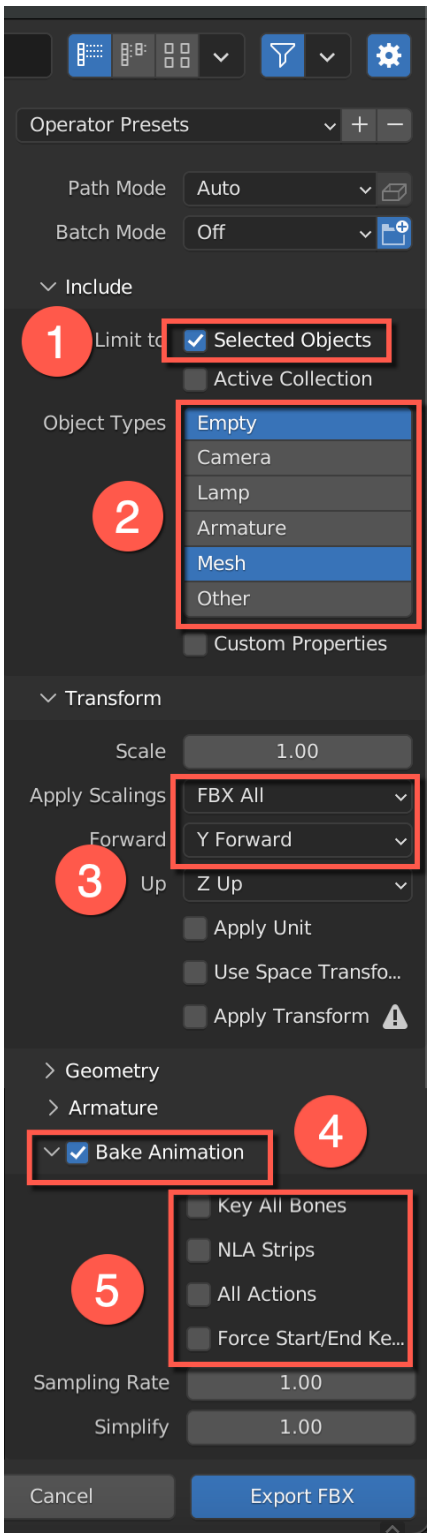



*Figure 17-4: The settings for a FBX export to Trainz (TS19 and later)*

After choosing the FBX file export option you will need to set up a few things in the export dialog box. Firstly choose '***Limit to Selected Objects***' (1) – if you don't do this then all of the objects in the Blender file will be exported which is definitely not what you want for a loco. Next, set the '***Object Types***' to '***Empty***' and '***Mesh***' (2).

Then set '***Apply Scalings***' to '***FBX all***' and '***Forward***' to '***Y Forward***'.

Select '***Bake Animation***' and deselect '***Key All Bones***', '***NLA Strips***', '***All Actions***' and '***Force Start/End Keyframes***'.

All other settings can be left at default.

This seems like a lot to remember, but once this has been set up the settings can be saved as pre-set for the next time.

Give the pre-set a meaningful name, as you may need others for exporting to Substance Painter or Quixel Mixer.

Annoyingly you must select the pre-set again each time you start Blender as it doesn't save the pre-set internally…

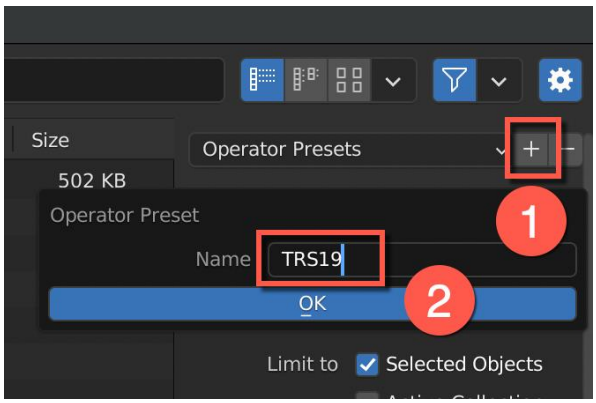



*Figure 17-5: Saving a pre-set for an export*

There are also a couple of things to note about exporting bogies – the *.kin file for a bogie must be named '***anim.kin**', so you can avoid a bit of work by exporting the LOD0 mesh as anim.fbx although annoyingly the *kin file will be named '***anim.scene.kin**' when Trainz imports this.

Because of this you have to commit the bogie asset in the Content Manager, open it for Editing and then rename '***anim.scene.kin**' to '***anim.kin**' every time you load the asset into the Content Manager.

## *Exporting to OBJ*

## *Importing to Trainz Content Manager*

A few notes on things that can go wrong when importing the asset into the Trainz Content Manager:

**Error Zero UV Coordinates** – This is a very common error that is caused by having more than one UV map on one or more meshes. This can happen without you realising it as Blender assigns a UV map called '***UVMap***' to primitives such as Cubes, but at other times a UV map called '***UVTex***' by default. If you then join two objects that have UV maps with different names, the resulting mesh will have two UV maps.

Additionally, merging meshes from other blend files can cause extra UV maps (for example '***UVMap.001***') to be created if the UV map has the same name as an existing one.

It is a very good idea to check the names of any UV maps before joining meshes. To fix the problem it is necessary to delete any extra UV maps leaving just one for reach mesh.

**Error No Ear Found** - Caused by NGons that are too complex, the solution is to add a Triangulate modifier to the mesh that is causing the problem.