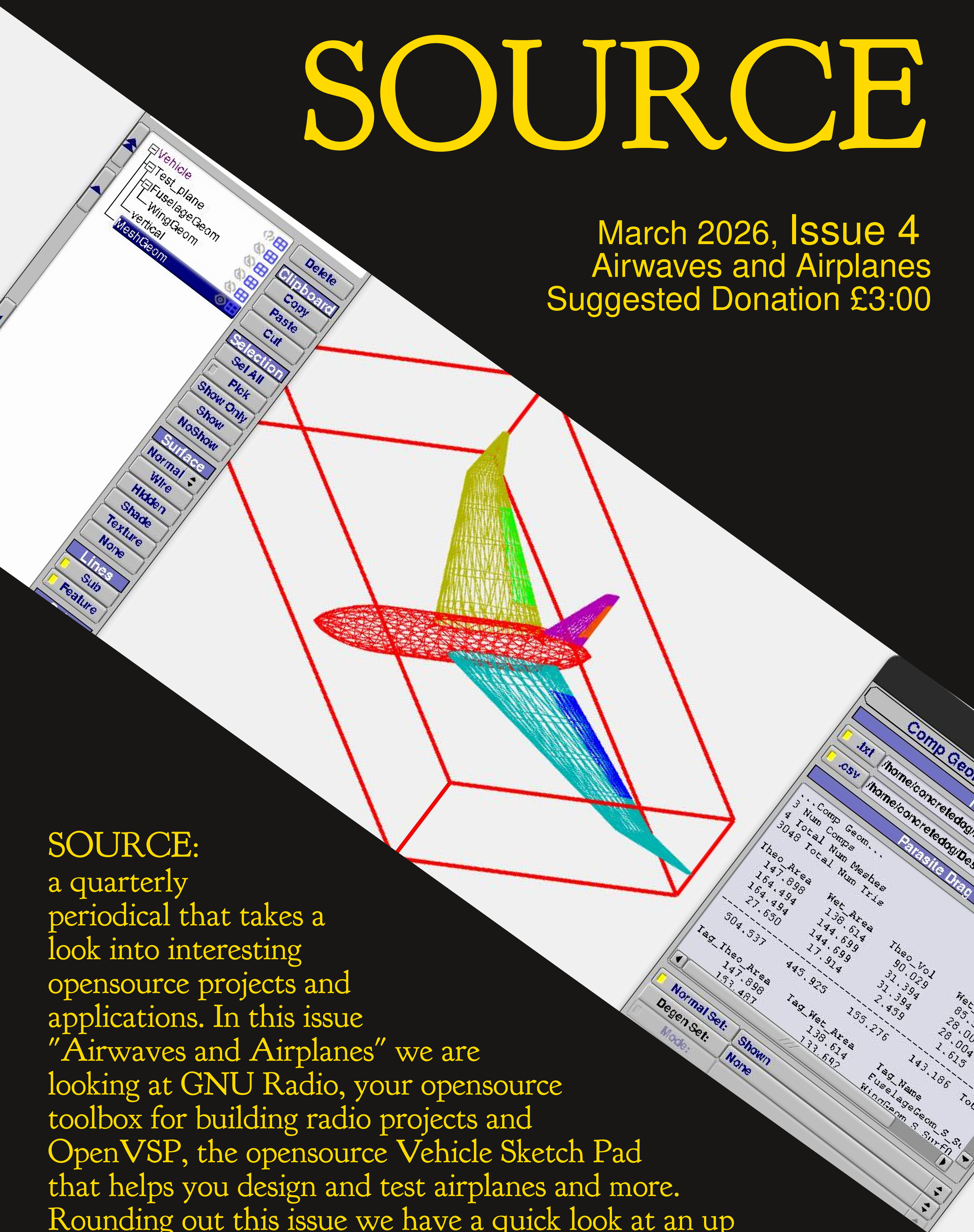
SOURCE

March 2026, Issue 4
Airwaves and Airplanes
Suggested Donation £3:00



SOURCE:
a quarterly periodical that takes a look into interesting opensource projects and applications. In this issue "Airwaves and Airplanes" we are looking at GNU Radio, your opensource toolbox for building radio projects and OpenVSP, the opensource Vehicle Sketch Pad that helps you design and test airplanes and more. Rounding out this issue we have a quick look at an up and coming FreeCAD workbench, Detessellate, which enables quick and easy reverse engineering of Mesh files.

Welcome to SOURCE

Hello, I'm Jo, AKA Concretedog, and first of all, thank you for taking interest in SOURCE.

If you've looked through earlier issues of SOURCE, you'll know that SOURCE is my place for writing content that doesn't fit elsewhere in my paid writing work. It tends to be a little more in depth in it's tutorials whilst promoting opensource.

It's pay as you feel and I'd love to see some donations come in for my time as it's a pretty big undertaking getting this all together! That said, times are hard and if you aren't in a position to donate, I really hope you enjoy reading this, tell your friends about SOURCE, and much solidarity to you.

In this issue the theme is "Airwaves and Airplanes" with two main beginner tutorials. GNU Radio is an amazing project that has spawned a million other projects. Whilst my tutorial is aimed at beginners, it really is a joyous thing to build an FM radio on your computer and hear a broadcast station with a sketch you built. Airplane design, is, of course, complex, but it's amazing that we have NASA creating open tools like Open VSP to help us create.
I really hope you enjoy this issue, as ever do share this with people, you can host it, compress it and email people, put it in your USB dead drop, or just share links with people, let's spread opensource far and wide.

Contents

Getting Started With GNU Radio

GNU Radio is an amazing opensource project. It's a software toolkit that contains a load of signal processing blocks that you can drag and drop and wire up to other blocks to create custom software radios. This tutorial is largely based on a beginners tutorial from the GNU Radio Wiki, I've simply followed it and written my own version, with some additional detail and explanation of radio terms.

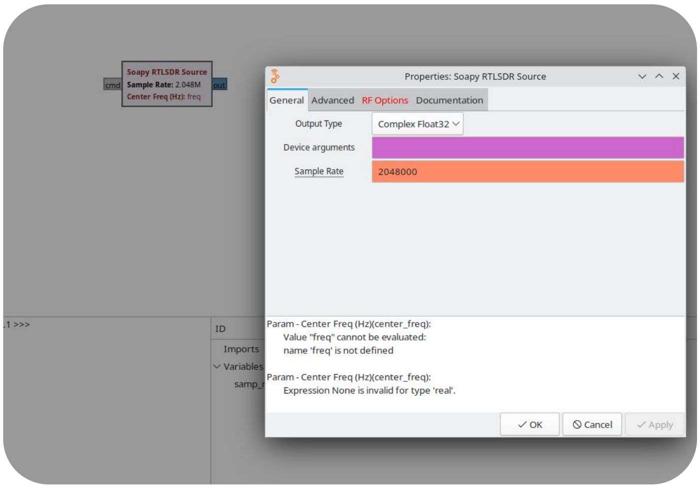
GNU Radio has a large community and there are some really well established project where GNU Radio is core ingredient, such as the global SatNOGS network developed by the Libre Space Foundation (LSF). In short the SatNOGS project is a decentralised network of ground stations that can range in scale and complexity but that all can be scheduled to turn on, tune in and record passes of satellites picked and scheduled from the web interface and a database.

The devices that from the groundstations tend to be small single board computers with the custom SatNOGS image running on them, and in turn, the majority of the SatNOGS radio software is built with GNU radio.

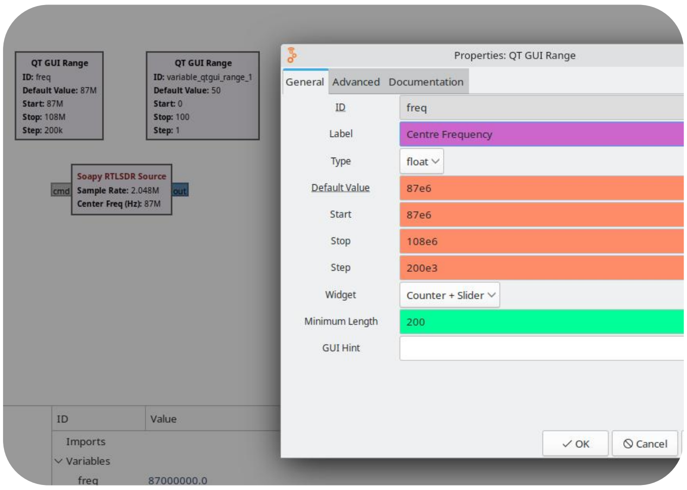
But there are dozens and dozens of interesting software defined radio (SDR) project that can be created with GNU Radio and one of the contributing factors to this is the prevalence of very affordable SDR USB dongles. Many of these were originally designed as a cheap way to pick up digital TV on a laptop but the Realtek RTL2832U chip can be used for all manner of SDR project reception. That last word is important, the cheaper Realtek SDR USB dongles are reception only, but that is no bad thing if you are new to this scene, as transmitting on frequencies is an are that may get you into trouble if you fall foul of local rules. There are numerous different variations of the Realtek USB DVB-T dongle, some more refined than others, but any of the following will do for this tutorial. RTL-SDR V2, Nooelec NE SDR Smart, or the plain old DVB-T+DAB+FM. .

To begin head over to https://wiki.gnuradio.org/index.php/InstallingGR and read up on how to install GNU Radio for your system. We installed GNU radio onto both a Debian 13 system and a Ubuntu 24.04 system using "sudo apt-get install gnuradio". Note that on this page there is a suggestion for installing GNU Radio on a Raspberry Pi, this gives many interesting options for smaller and potentially more portable hardware to experiment with and there are heaps of examples of interesting hardware builds in the GNU Radio community.

Open GNU Radio companion and on the right hand side of the screen you should see a separate area titled 'blocks'. This is the area where we can navigate, search, select and add blocks to the main area building a flowgraph. In turn, after configuring placed blocks we can wire the blocks inputs and outputs together and our flowgraph will become an FM radio receiver. If for any reason you can't see the search bar in the top of the block area then click "View – Find Blocks" to reopen this feature.

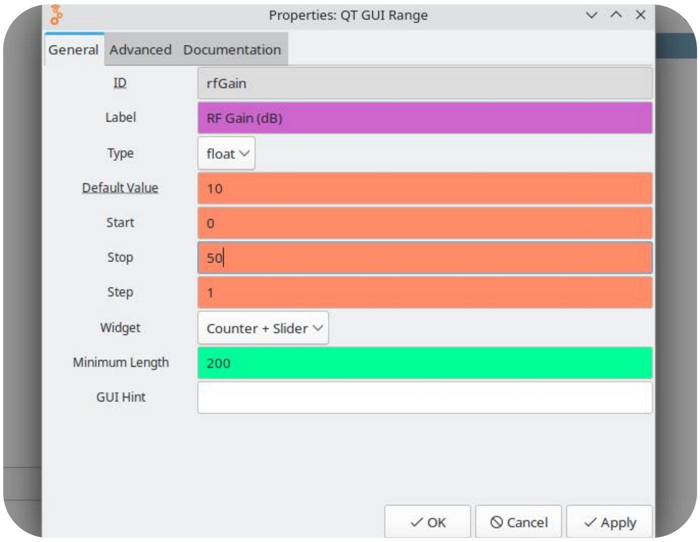


Using the search function we can add a Soapy RTLSDR Source block to our flowgraph. We can double click on block items and edit parameters using field inputs. Double click on the Soapy RTLSDR Source block and we are going to set the sample rate of the block to 2.048MHz. To do this we find the ID "Sample Rate" and change the Value to "2048000". Notice that the Soapy RTLSDR block has another parameter "Centre Freq (Hz)" this parameter represents the frequency that our FM radio will tune too. So for example we might tune to 101.4MHz if we know that is where a commercial radio station broadcasts on FM. Similarly, as with a hardware FM radio, we might want to move this value around to tune or hunt for a radio broadcast. It would become tiresome to repeatedly type in a new parameter for the centre frequency so we can use a variable and assign the variable to a GUI element block that will allow us to use a GUI slider we can move with our pointer to change the frequency. Notice before we add this block that the Value of the "Centre Freq (Hz)" is set as "freq".



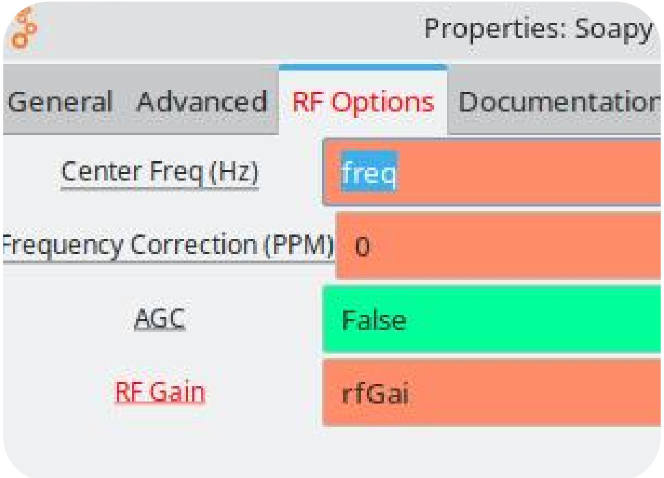
Search for "QT GUI Range" and add two of these blocks to the flowgraph area. In the first QT GUI Range block open and edit the parameters as following. Set the ID to "freq" take care that this is correct as this variable name needs to match where the variable is called in the Soapr RTLSDR Source block. For the label add "Centre Frequency". We then need to add a default value, this is the value that the freq variable will be set too when the flowgraph/project is started. If you know the frequency of a local to you FM radio station you could insert that value, however you can also make this the same as the lowest frequency we are going to allow our radio to tune too. The lowest value in this case is the "Start" value so with many regions in the world commercial FM radio broadcasting takes place between 87 and 108 MHz. So as such we will make the lowest frequency value 87MHz by setting the Start value to 87e6. As said, let's match the Start value to the default value and set Default Value to 87e6 as well. After a first successful listening session we can edit this to a station we have found perhaps. For the "Stop" value, which sets the upper tuneable limit in this example, we can add a value of 108e6.

In radio terms "Step" is a term often used in radio circles that describes the size of individual step when tuning. In a hardware handheld radio this may correspond to actual physical clicks on a hardware knob that when turned one click equals a consistant change in value. Most countries have an agreed Step amount for where radio stations or channels might exist. A common value we will use is 200Khz. This means that to move/tune a frequency by 1MHz we need to move 5 steps. Set "Step:" to 200e3.



In the second "QT GUI Range" block we are going to set up a similar slide controller for RF Gain. RF Gain controls the input signal strength on the RTL SDR dongle before the signal is digitised and processed. If we increase the RF Gain we can essentially capture more signal at the input. So why do we not set this to maximum? Well the trade of is that as well as boosting a weak signal it will also boost unwanted noise elements. Sometimes we may need to turn down RF Gain, is a station is broadcasting a powerful signal it may begin to distort as it overloads our receiver so the ability to reduce the input signal is also really useful. In the second "QT GUI Range" block set the following paramaters with the follwong values. Set the ID to "rfGain", set the Label as "RF Gain (dB). Set the Default Value to "10" a Start value of "0" a stop value of "50" and a step of "1".

Notice we used "dB" as part of the Label value. This stands for decibel and is a unit for measuring the loudness of sounds. It's a logarithmic scale, but before you panic remembering high school maths a simple way to think of it is that if a sound increases in intensity by 10dB then it is 10 as loud, if it increases by 20dB then this is 100 times as loud. Next we need to define the rfGain variable in the Soapy RTLSDR Source block properties. On the "RF Options" tab enter "rfGain" in the input labelled RF Gain. Do take care when inputting variable names, if you mismatch spelling or capitalisation then they'll return an error. If for example you spell "rfGain" incorrectly you'll notice that the label next to it will appear red in colour. If you apply the changes with an error and reopen the block you will see the error listed in the lower section of the window



Next we are going to add a couple more GUI elements to our flowgraph. Search for and then add a QT GUI Time Sink block and a QT GUI Frequency Sink block to the flowgraph area. These two GUI elements will allow us to visualise a couple of things. First we will see the Gain across a range of frequencies (this is an intuitive type of representation that allows you to spot signals that might be broadcast stations rather than just noise). Secondly we'll see the amplitude of signals change as we adjust the RF Gain value. In the QT GUI Frequency Sink block enter the variable name "freq" for the input labelled "Center Frequency (Hz)".

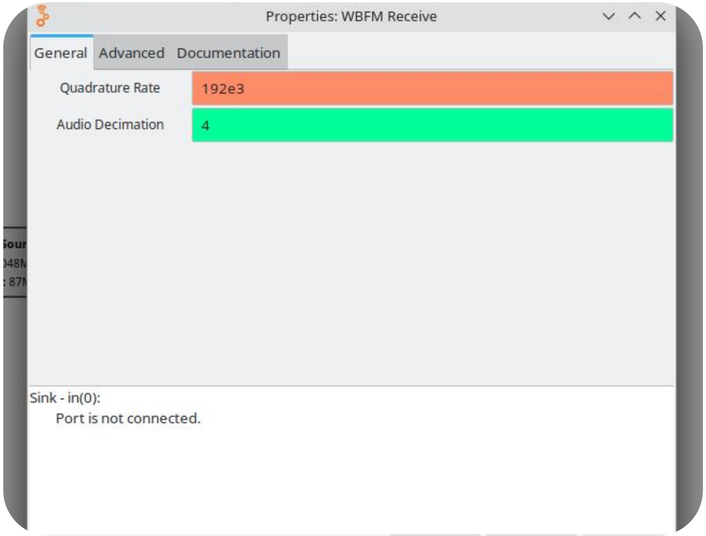
A Hz or Hertz is unit describing a repeating waveform completing one full cycle per second. So if a waveform is repeating 50 times per second you can describe it as 50Hz. A kHz or Kilohertz is 1000 cycles per second and a MHz or Megahertz is 1000,000 cycles per second.

With these two blocks added we can now make the first connections between the different block parts of our growing radio. To do this left click and hold on the "out" connector on the Soapy RTLSDR Source block then drag over to connect a wire to the "in" on the QT GUI Frequency Sink. Repeat this to place a second connection between the output Soapy RTLSDR Source block and the input of the QT GUI Time Sink block. Notice that as you make these connections the main titles on the blocks change from red to black. An none connected block basically is treated like an error and so as we make these two connections the errors are cleared and the "Execute the flow graph" button (which looks like a triangular "play" button in audio terms) becomes enabled.
Make sure that you have your SDR dongle plugged in and then you can left click the "Execute the flow graph" button to run the flow graph. If you left click this it will first prompt you to save your project, do so in the usual way.

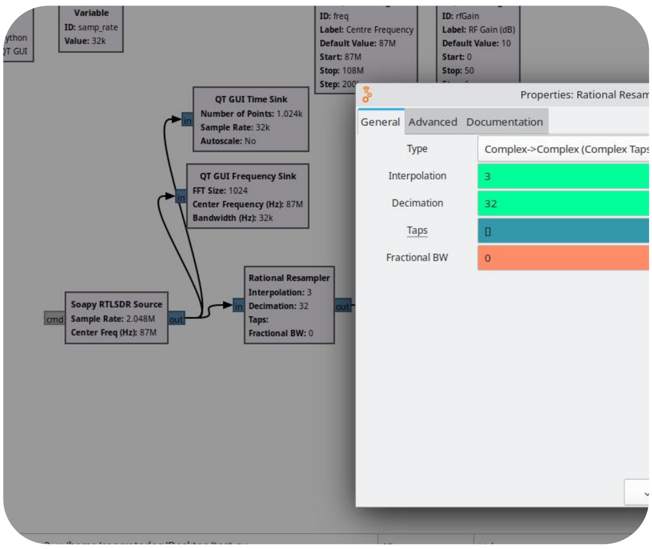
Of course, no audio will be output, but you will see the GUI elements appear in a new window and you can adjust the gain and frequency etc. Clicking the "Kill the flow graph" button (again looks like a square "stop" button in audio terms) stops activity and closes this window.

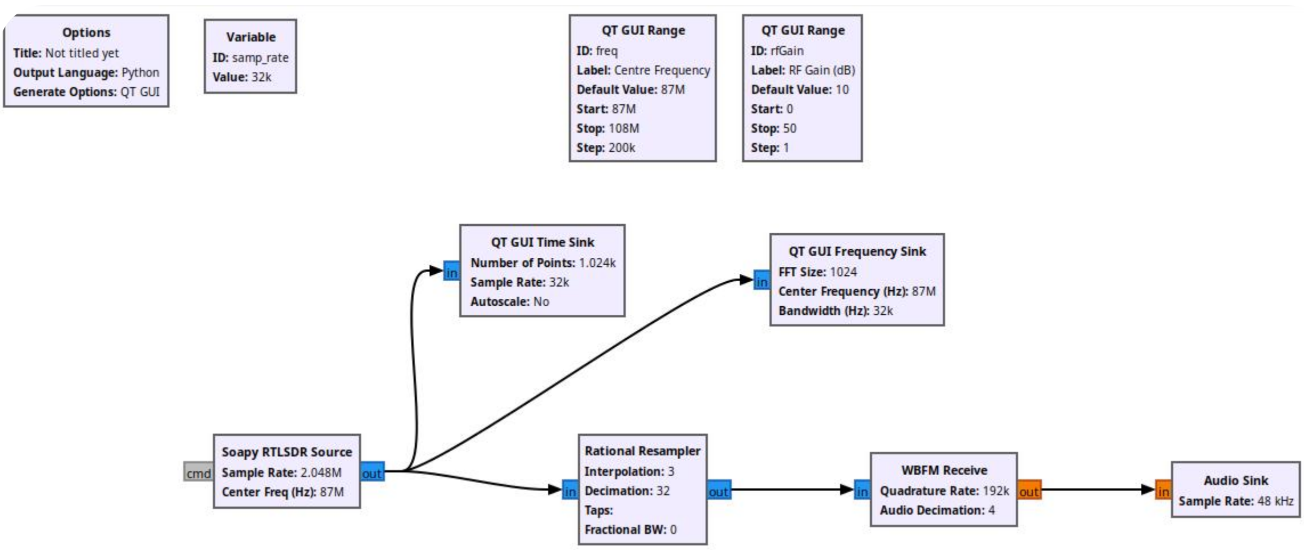
The desired outcome for an FM radio is, of course, audio output. To do this the signal will eventually reach an Audio Sink block, search for this block and add one to the flow graph area. Change the "Sample Rate" value to "48kHz".

Complex IQ sampled formats are quite a complicated subject area. However there are some excellent tutorial pages on the GNU Radio Wiki that can help you understand them.
https://wiki.gnuradio.org/index.php/IQ_Complex_Tutorial



The remaining blocks are used to take the output of the RTLSDR and which is in a complex IQ sampled format and we this to be converted to real the output samples our Audio Sink block requires. To do this we can use a WFBM Receive block. Search for this block and add it. The WFBM Receive block will essentially demodulate the FM signal and produce real output samples that it will decimate to optimise a liitle before outputting to the Audio Sink.
In the WFBM Receive block properties we need to set the "quadrature rate" to 192kHz (192e3) and the "audio decimation" value to "4". Audio decimation is a simple method where you only keep a single audio sample every multiple of the decimation in the signal. So for example we will keep only every fourth sample in the sequence of samples making the signal data smaller and more manageable. Finally dividing 192k by the decimation value of 4 results in 48k, the Sample Rate our Audio Sink Block is expecting at it's input. We can go ahead and connect the output of the WFBM Receive block to the input of the Audio Sink Block.



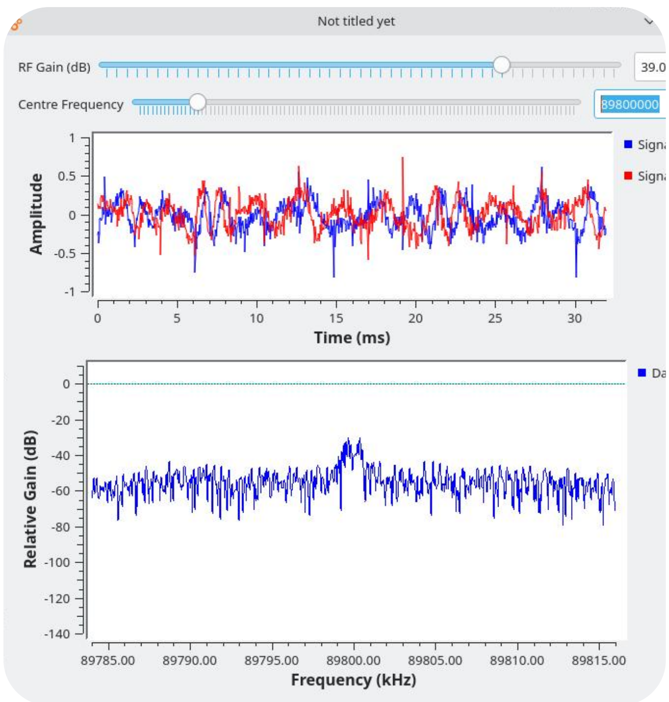


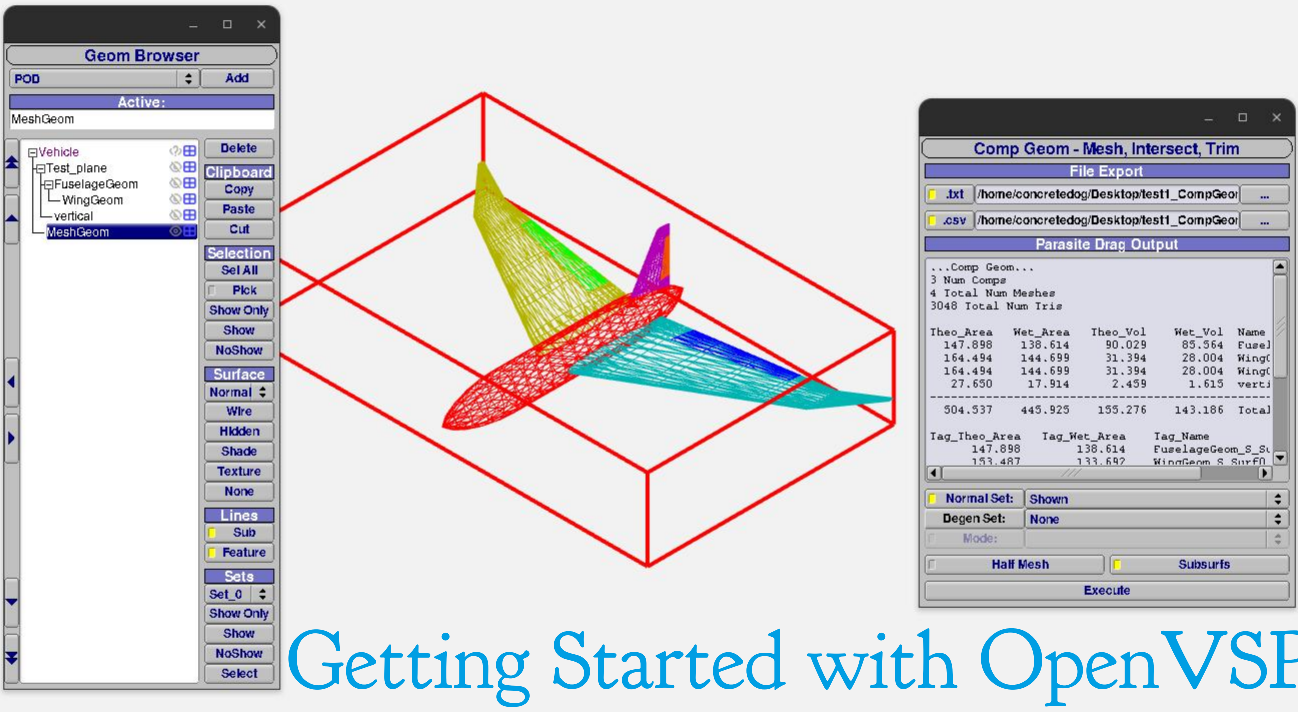
You'll notice that our WFBM Receive Block is expecting an input with a samplerate of 192k but our Soapy RTLSDR Source block is outputting a samplerate of 2.048M. This sample rate change can be expressed as a ratio of 192000/2048000 which simplifies to 3/32. As this is a rational fraction we can use a "Rational Resampler" block. Search for this block and add it to the flowgraph. In the Rational Resampler Properties set the Interpolation value to "3" and the Decimation value to "32". Finally connect the output of the Soapy RTLSDR Source block to the input of the rational resampler block and in turn connect the output of that block to the input of the WFBM Receive block.

Making sure your RTLSDR dongle is plugged into your device, run the flow graph. You should be able to hear audio coming from your computer and you can use the GUI elements we added to tune to any local FM stations!

Note that although we set the step value on the centre frequency GUI element to 200kHz we can directly type numerical values into the input box, allowing you to more accurately tune to your favourite stations.

Filled with the flushes of success building your own FM radio receiver you might quickly wonder "what next?". Well there are plenty of project examples out there in the GNU Radio community. From bluetooth sniffing, receiving weather satellites, Morse decoding, ADSB. You can also look at using different hardware than the realtek dongle, projects like the HackRF and the LimeSDR have transmit and receive capabilites and (observing local laws and rules) this opens another world of potential projects, recreating a pager system, radio control, data transmission and more. Of course, GNU Radio being opensource, you could also develop your own components and blocks. It really is a limitless environment.





Getting Started with OpenVSP

OpenVSP or Open Vehicle Sketch Pad is an opensource application that allows for fast development of 3D aircraft models using a collection of parameter driven tools. It offers a broad collection of basic geometries common to aircraft, obviously this includes fuselages, wings, vertical stabiliser etc, but also includes propellers, engine pods, and more. There are of course more advanced geometric capabilities and you can of course import component geometry created in other CAD or mesh modelling environments.

Aside from model creation there are a heap of tools that for vehicle analysis. Falling into two broad groups, aerodynamic and structural analysis these include tools that can identify centres of mass and centres of gravity, finite element analysis, flight dynamic analysis, wave and parasitic drag analysis and more. There are also numerous included meshing systems to generate various mesh types for further processing and analysis. It's really rich tool kit.

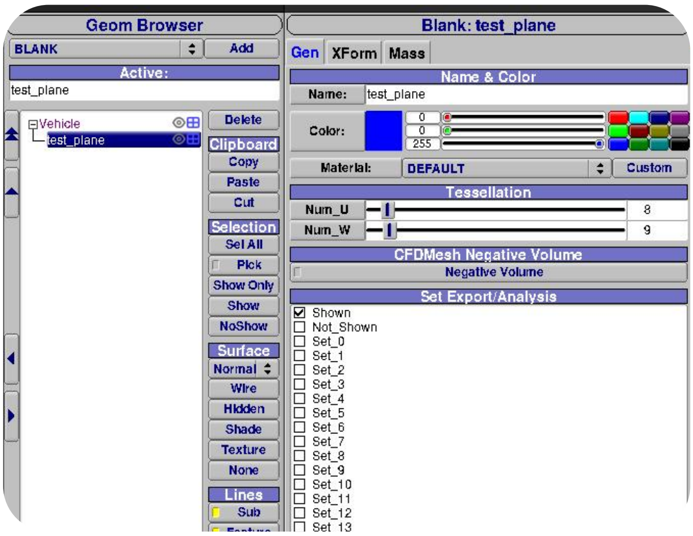
Obviously anything that allows us to design aircraft has reasonable complexity, so rather than try and tackle and entire introductory tutorial to design and build and aircraft we thought we'd look at some examples of how various parts work, link to various online tutorials we've found useful and hopefully set the stage for you to dive in whatever your level of aeronautical design experience.

At time of writing the current release of OpenVSP is 3.46.0 and there are pre built packages available for Windows, MacOS and Ubuntu. There are some build instructions for other systems linked from the readme on the main OpenVSP Github repository, we built OpenVSP from source on a Debian 13.3 machine (make sure you check your gcc version matches those in the build instructions) but as we had a Ubuntu 24 machine to hand we also simply installed the provided Ubuntu package.

There is an official track of tutorial content for OpenVSP called"OpenVSP Ground School" and has it's own website. This collection has numerous series of tutorials all in video format. They are good quality and very thorough. Certainly working through the 20 tutorials in the Fundamentals section would be a great grounding in this package. If you are keen to just have a quick taste of the workflow then this tutorial will get you started.
https://www.nasa.gov/software/openvsp-ground-school/

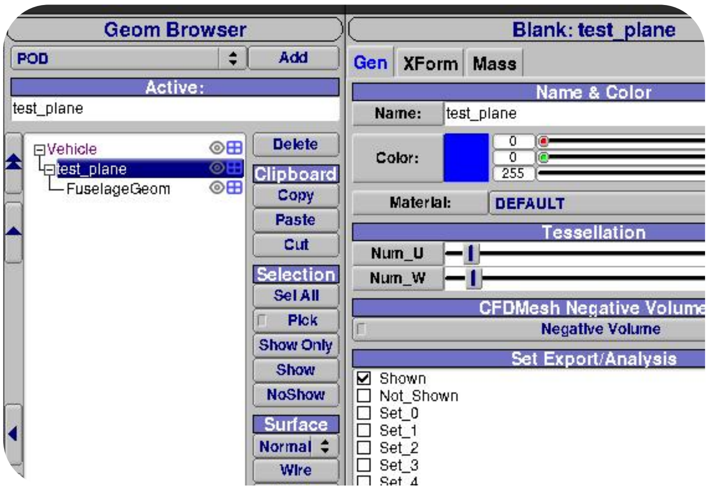
If you accidentally close the Geom Browser window you can re-open it by left clicking the "Model" dropdown menu and left clicking "Geometry...".

On booting OpenVSP you are presented with two windows. The "Geom Browser" is a floating window over a preview area and contains the contains the model tree view. This area is used to add, reorder and manipulate components as well as settings around surface finishes and model appearance. By default the geometry browser will have a main dropdown at the top of the window which will be set to "pod". If you left click to open this dropdown you will see lots of primitive components for your design listed such as, fuselage, wing, pod, ellipsoid and more. Select "blank" from the list and left click "Add". You'll see in the Geom Browser a "BlankGeom" item is added to the filetree nested under the "Vehicle" and a new window, "Blank:BlankGeom" will appear. In the new window double click in the "Name" input box and change this to a name for your aircraft design. We went with "test_plane". Close the window that was "Blank:BlankGeom" but is now "Blank: test_plane"window.



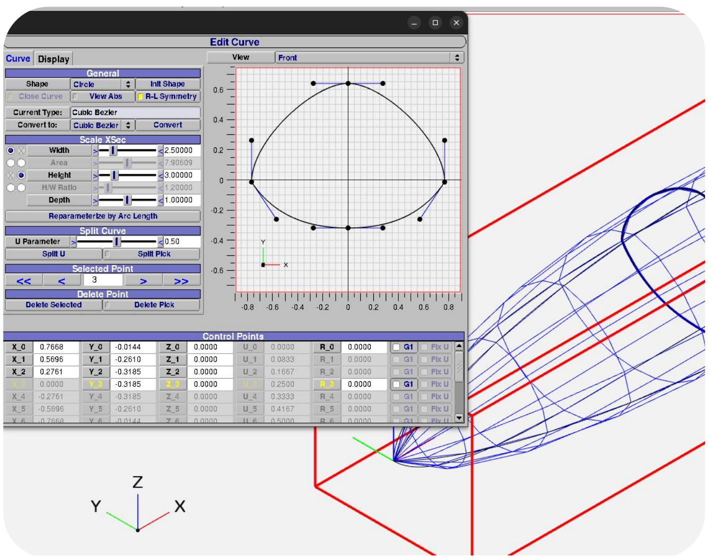
Next highlight the "test_plane" item in the model tree, and then in the Geom Browser change the uppermost dropdown menu to "FUSELAGE" and click the "Add" button. You should now see a fuselage item appear both as a wireframe model in the main preview area, and see an item "FuselageGeom" appear in the model tree nested under the "test_plane" item.

Throughout OpenVSP you'll notice that it never specifies units. Using relative units like this makes it pretty simply, but you must keep thinking in the same unit system!



Adding this fuselage component you should now see a "Fuselage:FuselageGeom" window open. These Geom windows for each component in the file tree can also be opened by double left clicking on the item in the file tree. In theses windows you can heavily edit and customise the component they refer too. They offer a lot of complexity and each component can have hundreds of adjustable parameters. In the Fuselage:FuselageGeom window switch to the "Design" tab.

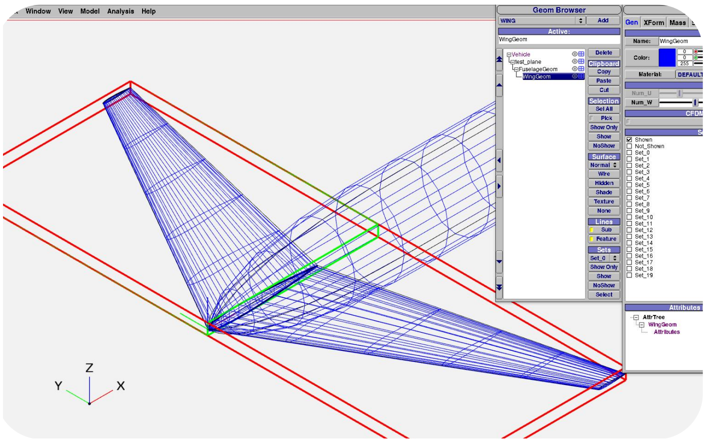
Notice at the top of the design tab you can click use a slider to increase or decrease the length of the fuselage component. Close the Fuselage window and with it still highlighted in the model tree use the geom browser window to select "shade" under the "surface" tab to make the fuselage more visible. Click F8 to move the preview to an isometric view. Note that using F5 to F8 will swap the view of your project. Double clicking "fuselage" in the model tree reopens the fuselage dialogue to make adjustments to the fuselage design.



Moving to the “XSec” tab we can use the arrows to move selection of each cross section in our fuselage. We can flip between cross section 0, 1, 2,3 and 4. If we select cross section 1 we can then use the “Choose type” option towards the bottom of the window to change the type of cross section for further manipulation. From the list we can select “EDIT_CURVE” Under the “General” tab we can click the “Circle” dropdown and select “Ellipse” and then click the “Edit Curve” button to the right. This opens a new window “Edit curve”, an editor where we can directly manipulate the shape of the cross section design. By default any nodes that you drag to change the cross section shape should be applied symmetrically across the right and left but of course you can explore removing this symmetric constraint should you need to. You’ll see as you make changes that changes to this cross section alter the entire geometry of the fuselage, if at any point you want to reset the cross section shape you are editing then you can, still in the “Edit Curve” window click the “Init Shape” button at the top of the “General Section” to return to the base shape selected.

Some quick navigation tips for the main preview area are left click to orbit, right click to pan, with a centre click or mouse wheel click you can scroll to zoom in or out.

Next we are going to add a wing. As the wing would often attach to the fuselage on a plane we highlight “FuselageGeom” in the Geom Browser and then change the dropdown menu to “WING” and again click “Add”. By default your wing will be added at the very front tip of the fuselage. When you added the wing object a “Wing: WingGeom” window should have opened. Select the “XForm” tab and you can use the “XLoc” slider in the “Coord System:” section to adjust the position of the wing relative to the fuselage. For the sake of this getting started tutorial we are just placing items in a way that they look reasonable rather than using any design approach or underlying engineering. There is a huge amount of options again for wing design in OpenVSP. IN the Wing:WingGeom window you can experiment with the planform of the wing using the “plan” tab. In the plan tab you can use the various “Total Planform” sliders to adjust many aspects. Play with adjusting the “span” (the tip to tip dimension of the wing), the “Chord”, the length of the wing at its widest point in the centre of the fuselage. You’ll notice that some of the sliders are linked so as you vary the dimensions of the wings the “Area” slider value is calculating the area of the wing. As wing area is often a critical design consideration you can also manipulate the area as an input.

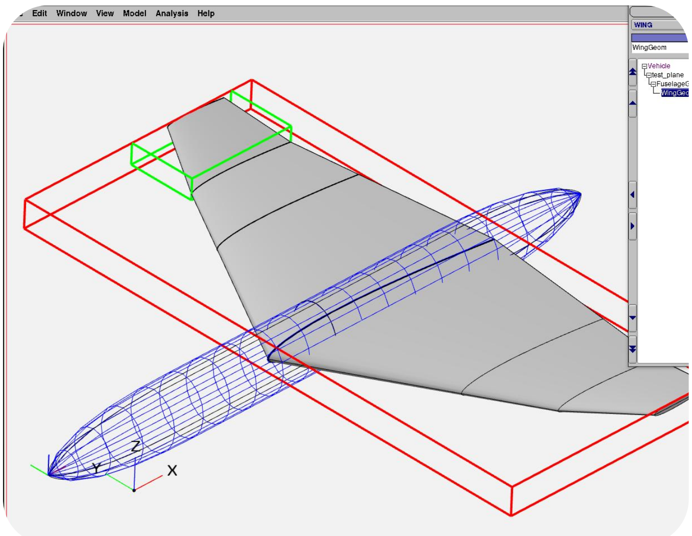


Under the “Airfoil” tab of the Wing:WingGeom window we can select different airfoil profiles. There are many series of airfoil profiles built in to OpenVSP and it’s also possible to import and use additional airfoil files should you need to.

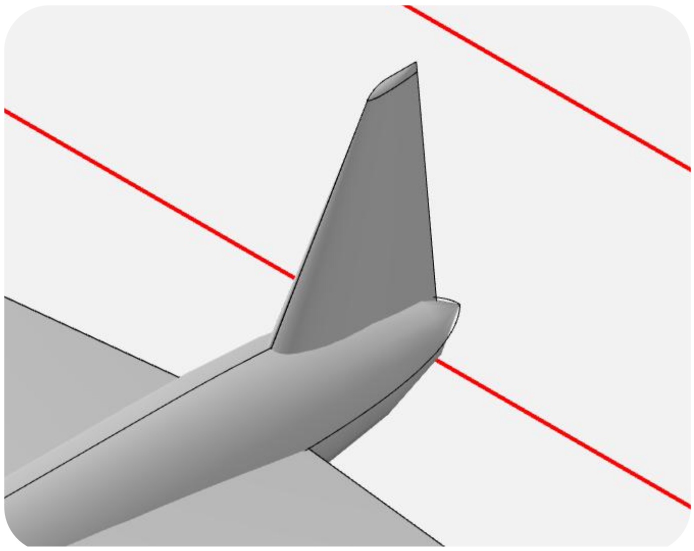
Moving to the “Sect” tab we can add and manipulate different areas of the wing by splitting the wing with additional cross sections points. Currently there is only one wing section per side. Click the “Split” button under the “Wing Section” area. You’ll notice that another cross section is added and a green bounding box appears on the wing. If you then use the left and right arrows to move between the wing sections (at the top of the “Wing Section” area”) you can highlight each section of the wing. This means you can now edit individual sections of the wing and any changes will be symetrically applied to both wings. Select the outermost section of the wing (the tip) and click the “Split” button once again. Again move using the arrow buttons to the outer tip sectionanf then at the bottom of the Wing:WingGeom window you can adjust the “Dihedral” value. Dihedral is the angling up of the ends of wings which can increase stability and self righting of airplanes.

It can get a little hard sometimes to see changes using the default Wire view for components. In the main Geom Browser window you can click "Shade" to shade in specific components.

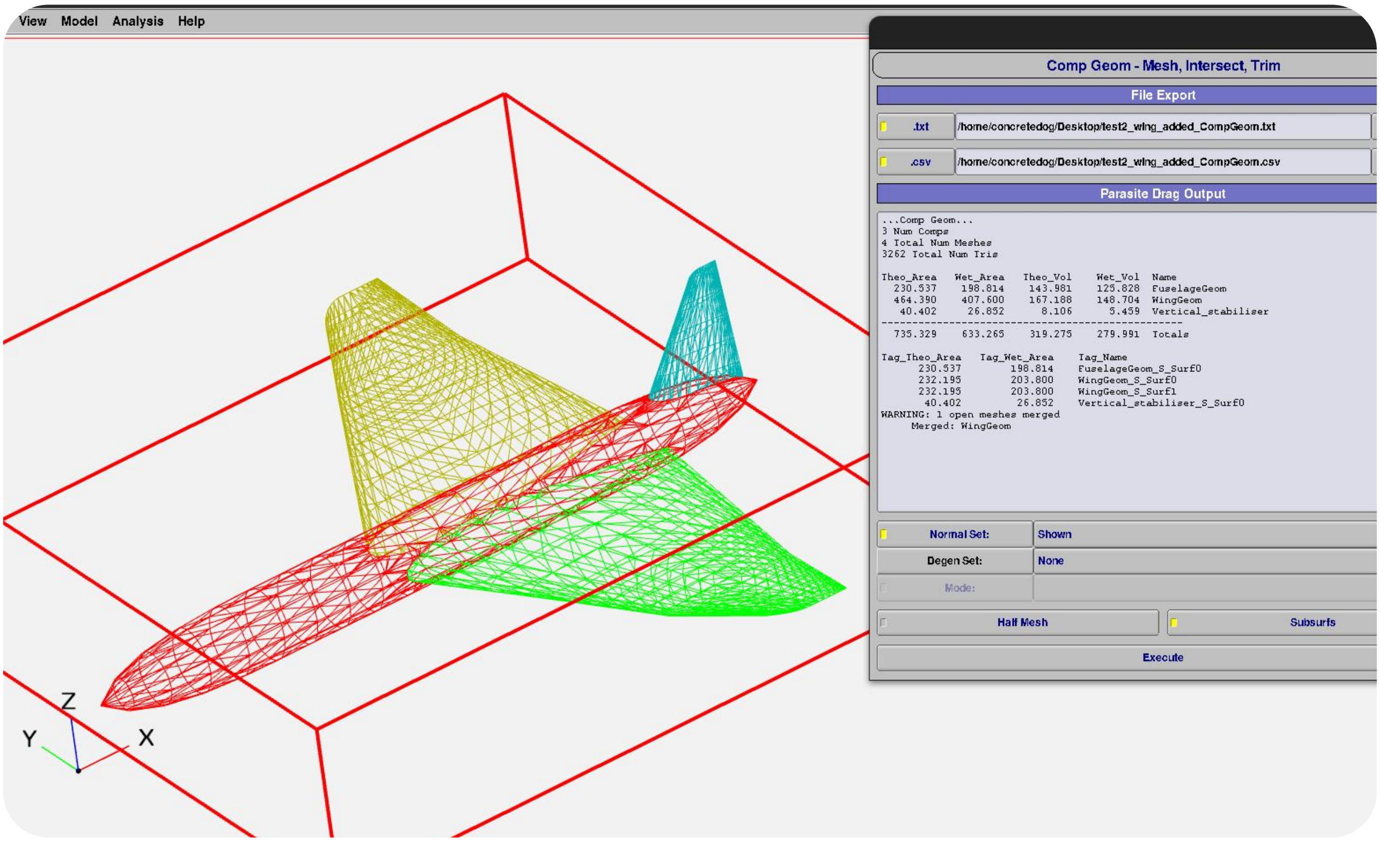
Before we leave the Wing:WingGeom window note that back on the "Plan" tab you can use the "Tip Treatment" section to add nice tips to the ends of your wing design. You can change the "Tip Cap Type" from "None" to various options. Switching to "Round Ext TE" we founf that the tips looked wuite small even when the "Length" value slider was set to the initial maximum of 2.000. Note that in OpenVSP you can click the < symbol to the left of a numerical value input box and then this allows the slider to have a larger range. Alternatively you can also type values directly into these boxes. We ended up increasing out wing tips to 6.75 for our design.



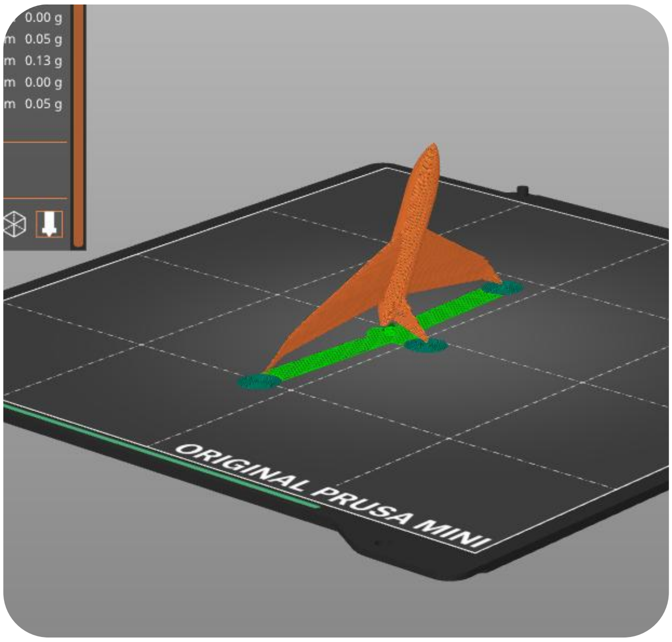
For our final component in this basic, but hopefully fun, project we are going to add a vertical stabiliser section at the rear of the fuselage. Again, on an aircraft this would be attached to the fuselage and so highlight the fuselage object in the model tree. Then switch the dropdown to WING again and click add to add a second wing. This wing again will appear at the front of the fuselage. First of all we don't need a symmetrical wing so in the Wing:WingGeom window move to the "Xform" tab and under the "Symmetry" section click to toggle the "XZ" button in the "planar" section to off. You should then only have one small wing section showing. Next in the same window in the "Transform" section you need to change the "XRot" value to "90" this then sets the small single wing vertical relative to the fuselage. Finally, move the "Xloc" slider to place the vertical tail section at the rear of the fuselage. Of course, the processes we covered earlier for the main wing can also be played with in terms of the vertical stabiliser. As a final tip it's worth noting that in the main Geom Browser window both the wing and the vertical stabiliser appear as "WingGeom". If you select the vertical stabiliser and in it's Wing:WingGeom on the "Gen" tab you can change the name under the "Name and Color" section. Changing this to "Vertical_Stabiliser" or similar can help you navigate the model tree and is good practice for when you create more complex models.



We are going to nearly end this tutorial here as, if you have got this far, you have learnt enough basics to start to go and find other tutorials around the OpenVSP community and to play with other features. Some common next steps are to explore using the "Sub" tab on "Wing:WingGeom" windows to add sub surfaces to wings, this means you can add flaps, ailerons, elevators rudders and more, to your designs. You can also explore the "File – Export" options where you can find lot's of 2D and 3D export options including Step files, STL files, and also SVG options to render multiview plan drawings of your design.

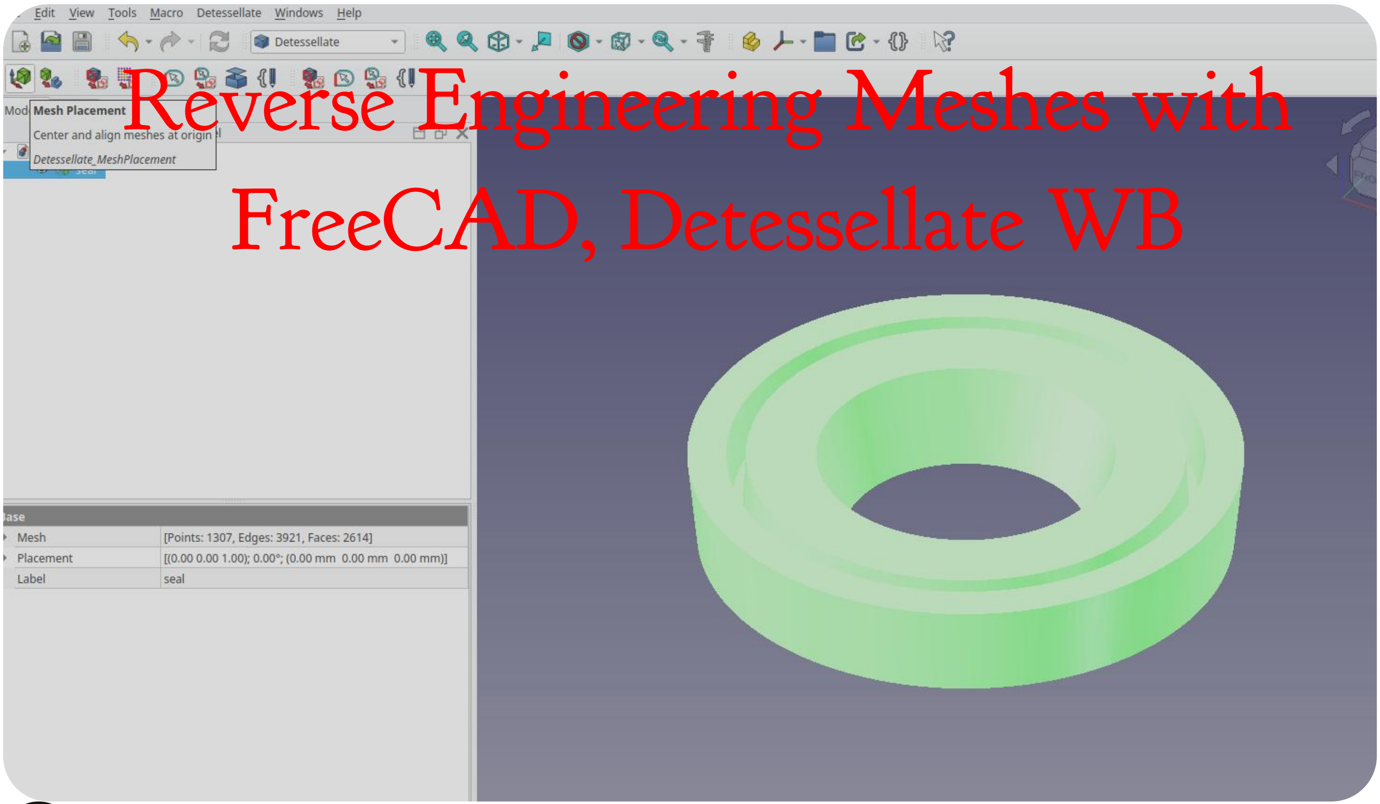


Finally another huge area that is incredibly useful is under the "Analysis" dropdown. There are lots of options and functionality. As a simple first step with this menu is to click the CompGeom... option and then in the resulting "Comp Geom -Mesh, Intersect, Trim" window click the "Execute" button. In simple terms this is a function that combines the different mesh element of the plane design and stitches them together and removes overlap so you have a watertight mesh with no internal geometry, such as you might do in any CAD program when creating object for 3D print. When clicked you'll see the plane design swap to a new coloured wireframe and in the Comp Geom -Mesh, Intersect, Trim window you will see a Parasitic Drag report. This handily reports the total wet area's and volumes of each section of your aircraft design.

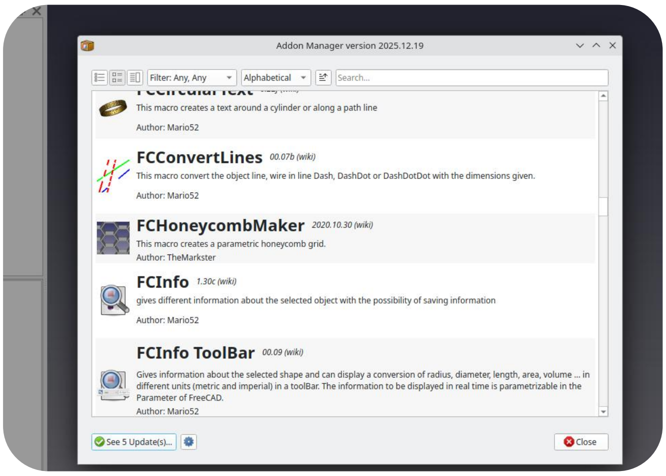


Hopefully you've enjoyed this overview of OpenVSP. There are huge amounts of resources online and probably in your local library about plane design should you want to take this further. A good book for Model Airplane design from more heuristic or "rules of thumb" is "Designing Model Aircraft" by Peter Miller, slightly more advanced, but still approachable is "Flight without Formulae" by A C Kermode, who also wrote the excellent "Mechanics of Flight".

Reverse Engineering Meshes with FreeCAD, Detessellate WB



One of the amazing things about FreeCAD is you can install add on workbenches, full of all manner of tools, from the built in Addon manager. There really are an amazing array of workbenches, from Fasteners which creates single click to create any specific nut or bolt, through to Airplane design, shipbuilding and a myriad more.

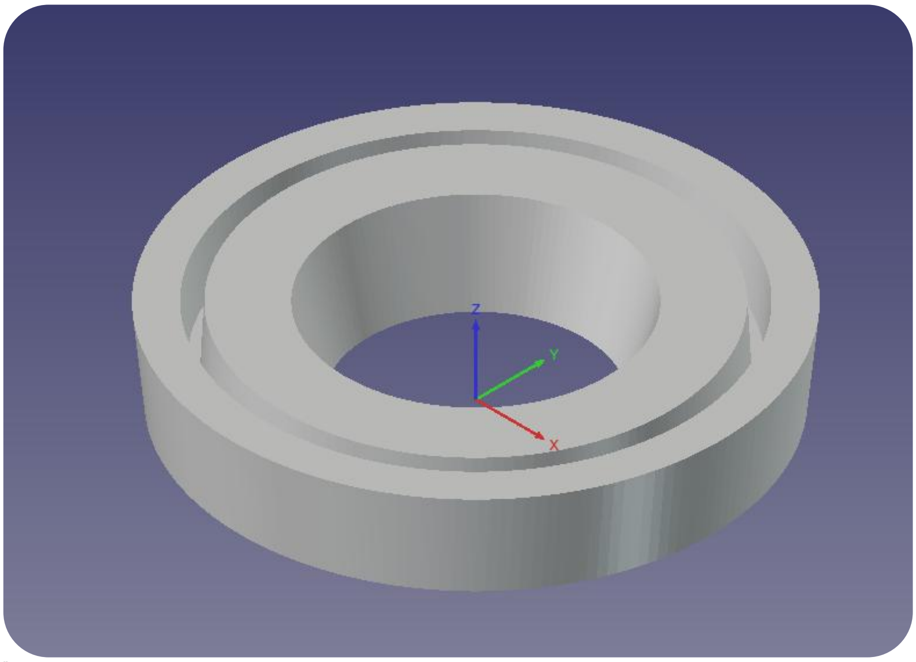


Beyond what is accepted and included in the Addon manager there are also workbenches that are in development that might not quite be ready for the prime time inclusion. One that spotted my eye and has proved incredibly useful in a recent project is the Detesselate workbench. It's essentially a workbench that provides quick workflows for converting mesh items to solids and then detecting geometry and creating sketches. This means it's a superb tool for reverse engineering when only stl, obj or 3mf mesh files are available. Let's take a look.

Firing up FreeCAD version 1.1 release candidate 2 we need to first add the Detessellate workbench github repository to the Addon manager preferences. To do this go to "Edit – Preferences" and go to "Addon Manager – Addon Manager Options". On this tab you need to click the "+" button under the "Custom Repositories" input box. Paste "https://github.com/DesignWeaver3D/Detessellate" into the "Repository url" input and in the "Branch" input box type "main". Click "OK" to add the custom repository to the Addon manager list. Next open the Addon manager "Tools – Addon Manager" and then find the Detessellate workbench in the list. Click to install the workbench, wait for it to install and then close the Addon manager. You'll be prompted to reload FreeCAD and do so.

Once restarted let's click "Empty File" to create a new empty project. Remembering that Detessellate is a new and experimental workbench it's a good idea to try a simple project first so we targeted a fairly simple stl file for a small rubber seal that we wanted to reverse engineer. In the new project use the workbench dropdown to move to the Detessellate workbench. Next use "File – Import" to import your target mesh file selecting the "STL Mesh (Mesh)" option in the resulting popup dialogue.

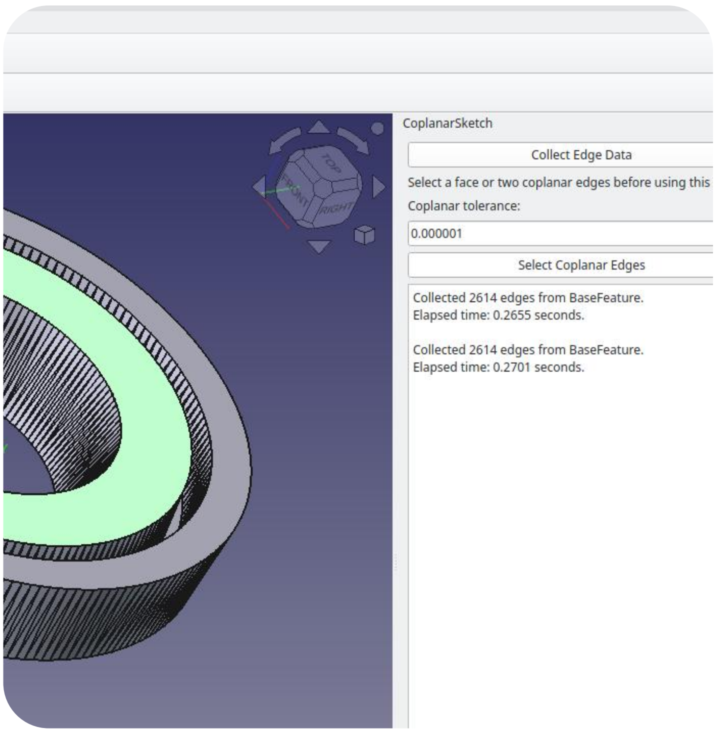
Sometimes when you import a mesh file it may be placed a long way from the origin point. This can create real complexity when we get to perhaps recreating sketch elements etc. Detessellate has a specific tool to help recentre and or place the mesh at a useful position relative to the origin point. With your imported mesh highlighted in the model tree view left click on the "Mesh Placement" tool icon. A docked window will open on the right hand side of the screen and you can use the various buttons to place the imported mesh relative to the origin point. The uppermost button is "Center XYZ" and we can press this first to centre the mesh. However for our simple seal example it's potentially more useful to have the base of the seal sat on the Z axis origin so then having Centred XYZ we pressed the "Align Bottom" button to place the seal on the origin.

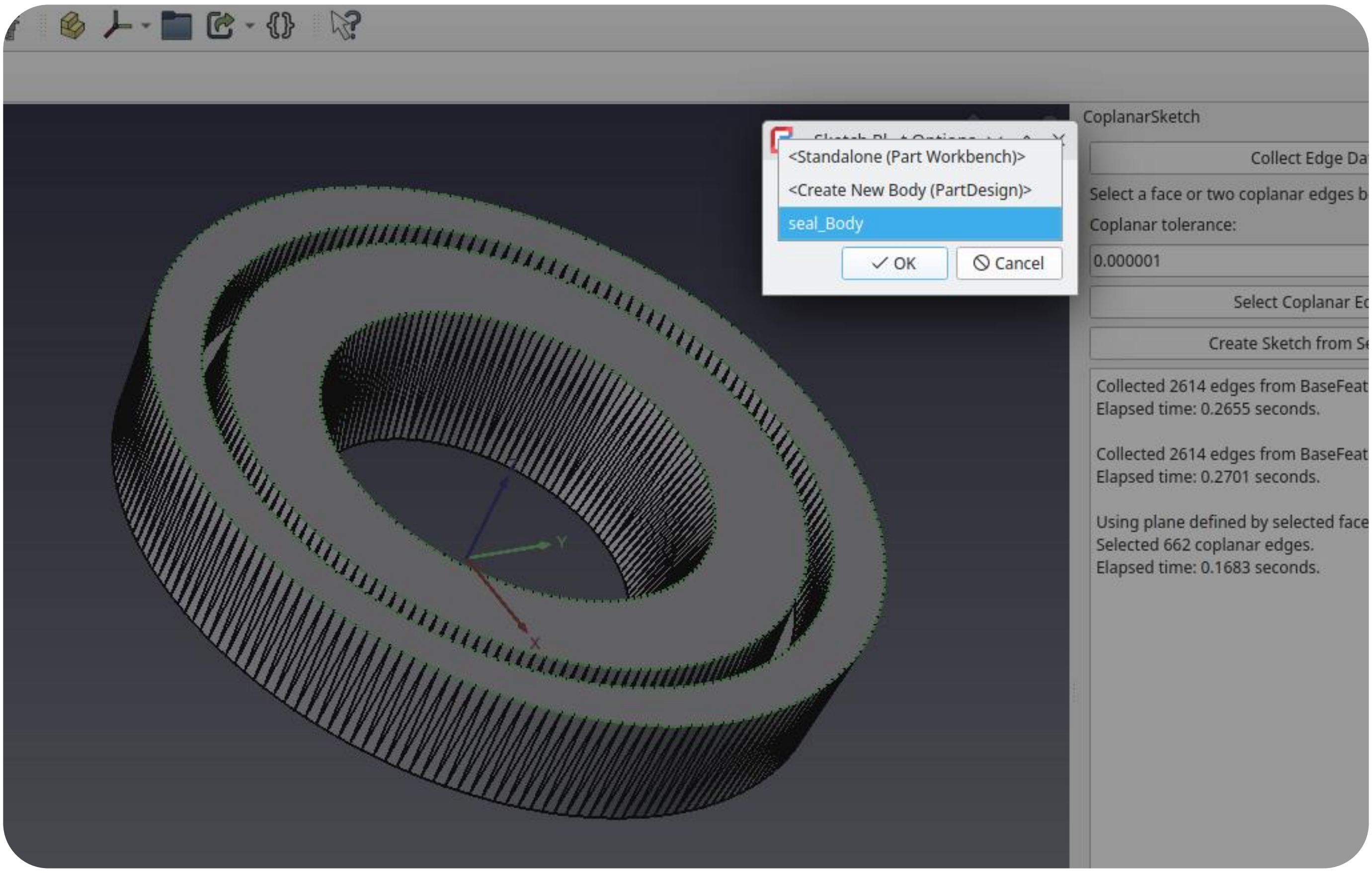


In this tutorial we always use the "Tool Tip" names for tools that appear when you hover over tool icons. This means you might have to search for the tool, but you might also discover other tools whilst you do so!

With the imported mesh centred to our satisfaction we can close the Mesh placement docked window on the right hand side. Next, with the imported mesh highlighted in the model tree, we can click the next Detessellate tool icon, the "Mesh to Body" tool. This tool performs a couple of actions. First of all it creates a simple solid from the mesh, places it in the model tree and toggles it to transparent. It also creates a new body, and places the solid version created from the mesh inside the body as a basefeature. In the following instructions where we create sketches from points detected from coplanar edges or faces we can either work inside the body using the basefeature, or we can delete the body and work with the simple solid that has been created. This leans into the idea that the Detessellate workbench can work with Part, Part Design or other FreeCAD workflows.

We'll work using the basefeature inside the body. With that basefeature highlighted next let's click the "Coplanar Sketch" tool. Again a docked window will appear on the right hand side. There is currently only one button "Collect Edge Data" so proceed to click it! Once the tool has run and edge data has been collected another button will appear "Select Coplanar Edges".





We then left clicked the centre face of the top of our seal model in the preview window. Despite only selecting the centre section when we then click "Select Coplanar Edges" it will detect all the face geometry/edges on that plane, so the entire of the top of the seal. Whne clicked you'll see green highlights around the geometry on that plane in the preview window and a new button appears "Create Sketch from Selection". Clicking this button will launch a small popup that is concerned with sketch placement. There are 3 options in our example, "<Standalone (Part Workbench)>", " <Create New Body (PartDesign)>", or "seal_Body" which is the newly created body already in the model tree where our solid basefeature is already placed. We opted for this third option to create the sketch in the existing seal_Body.

You'll shortly see a sketch created in the body in the model tree. If you double left click on this item it will launch the sketcher workbench and you'll see a (in our case quite large!) collection of construction geometry. As our seal design is all made from circles that are concentric to the origin point we can delete most of the construction geometry as we only need one point on each circle feature to use the circle tool to create an exact sketch replicating this planes geometry. There is, as part of the Detessellate workbench a "Sketch Reprofile" tool which automates conversion of construction geometry to sketch geometry elements, however we've had mixed results using this tool and it's often been easier to manually redraw sketch elements using the construction geometry as guides or point to constrain too.

We aren't going to fully explore how we reverse engineered the seal item in this tutorial, but, of course, you can re run the "Select Coplanar Edges" tool for the different planes of the sketch design which then resulted in 3 separate sketches correctly placed on the Z axis. We then used both additive and subtractive loft tools to recreate the seal, but not before making the dimensional changes we needed that prompted this entire reverse engineering!

We look forward to seeing the Detessellate workbench continue to develop and potentially become available directly in the Addon Manager. Kudos and thanks to the developer for their work.

Thanks for Reading!

Thanks so much for reading SOURCE issue 4. When I first started SOURCE I commited to doing 4 issues and seeing where we went from there. So I've kind of fullfilled my commitment! This last issue has been hard, other personal commitments, some family bereavement and other stuff has really scupprered me getting this one out in a timely fashion. However, I do enjoy the feeling of putting this out there still. There's a few options moving forward. I'm definitely keeping going, but I think I'll aim for 2 slightly larger issues a year with more varied content. I'd really like to get some hardware projects in here, and also perhaps reviews of opensource products potentially. I've had a couple of tentative offes for contributors, but, that adds another layer of complexity that I am not sure I am prepared for! I've also had a couple of people in the community suggest I do a print run. I like the idea, but theres obviously a cost, and also there's stuff like distribution etc. One thought was perhaps compilling multiple issues into a sort of bumper annual issue! Watch this space!

Take good care all.
Jo AKA Concretedog.

Visit the SOURCE magazine website to download other issues.

Pay what you feel via Paypal here

PayPal

Pay what you feel via KoFi

Support me on Ko-fi