Diary Week 2 – Sam Ross

Saturday 30/05/20

- Downloaded the 2.89 bullet zip from the website
- When opening the bat file kept facing a vs2010 error
- Made sure my VS was up to date and had the C++ extension installed
- Tried different earlier versions of bullet and the bullet master file
- The bat file is meant to create a vs2010 folder inside the build3 folder
- Tried installing pybullet using various methods such as using pip in the command prompt, downloading the whl file and downloading the tar.gz file and extracting it
- Kept coming up with an error no matter the method

Sunday 31/05/20

- Was previously using laptop so tried running the bat file on my desktop
- Created the vs2010 folder fine
- Copied this vs2010 folder into the build3 folder on laptop and bullet worked fine
- Tried installing pybullet on the desktop using pip and it worked

Wednesday 03/06/20

- Tried running the pybullet examples using the command prompt and found I needed to install the gym library
- Installed the gym module along with the other libraries such as numpy
- 2 of the examples worked
- The third example required tensorflow library
- Spent a long time trying to get the tensorflow library to work using various methods
- Got it sort of working with anaconda but without pybullet
- Had to reinstall python as a 64bit version is needed for tensorflow
- Realised that pybullet wasn't working with the 64bit version of python
- Reinstalled the 32bit version and pybullet worked again
- Should figure out how to get tensorflow installed to work with pybullet for the useful output

Thursday 04/06/20

- Started the useful output on how to install pybullet and some of its libraries
- Pretty much finished it

Friday 05/06/20

- Started trying to load the ar2 urdf in threejs
- Tried to implement urdf loading code from https://github.com/gkjohnson/urdf-loaders/tree/master/javascript

Week 3

Monday 08/06/20

- Set up a separate python environment using anaconda and linked vs code to it
- Installed flask in the venv using the terminal in VS code
- Run a basic flask hello world program
- Working through the flask quickstart guide

Tuesday 09/06/20

- Installed Node.js and NPM to grasp an understanding of them
- Worked on loading the urdf file into
- Started setting up NPM with threejs (https://threejs.org/docs/index.html#manual/en/buildTools/Testing-with-NPM)
- Installed mocha

Wednesday 10/06/20

- Got NPM set up with threejs
- Got a threejs file to run successfully in the flask web server
- Installed python-magic in the venv

Thursday 11/06/20

- Installed the npm urdf-loader
- Copied the files into the static folder for flask and into the flask site-packages folder
- Trying to edit the code to work with the flask webserver getting an error: 127.0.0.1/:105 Uncaught SyntaxError: Cannot use import statement outside a module
- Tried separating the threejs code from the html code
- Learnt javascript importing and exporting
- Learnt how json files work and tried importing the “main:” files form the json directly
- Realised need to target the package.json files for importing the modules as they do more than just locate the file stated after “main:”
- Still getting that error

Friday 12/06/20

- Changed the .js content type in regedit to text/javascript as it is text/plain by default
- Changed the script type to “module” as this is needed when using import and export and this fixed the import statement error
- Figured that when the script type is “module” the threejs file doesn’t load
- Need to find a way to be able to use import/export and to be able to load the threejs files as the module prevents being able have both

Week 4

Monday 15/06/20

- Trying to find a way to get threejs working as a module in html
- Able to use *import * as THREE from '../three/build/three.module.js';* to import threejs as a module without ammojs
- Trying to find a way to get ammojs imported as a module
- Leaving out ammojs until the urdf loader is working with just threejs
- Need to figure out how to get json files imported in threejs without using npm

Tuesday 16/06/20

- Running into importing errors for the urdf files
- Research on json files
- Installed npm in the venv path
- Installed the npm browserify module
- Research on implementing browserify so that npm modules can be used in a browser
- Installed urdf-loader, three and watchify npm modules
- Got basic threejs box to load in the browser by bundling using browserify

Wednesday 17/06/20

- Realised should be importing the files stated after “module:” in the json files instead of “main:”
- Finally figured out that I had to go into the src files of URDFLoader.js and change all the import paths to relative paths as they were set as npm import classes eg. from ‘three’ to '../../three/build/three.module.js'
- Did the same for the other source files and after doing so the urdf-loader methods were imported successfully
- Getting “TypeError: Cannot read property 'children' of undefined” error

Thursday 18/06/20

- Working on finding the cause of the “TypeError: Cannot read property 'children' of undefined” error and “GET http://127.0.0.1:5000/T12/urdf/T12.URDF 404 (NOT FOUND)” error. There seems to be problems locating the URDF file
- Figured out that you can check the path that chrome uses in developer tools by going to network > (file name you are trying to locate) > Initiator > Request initiator chain
- Had to change the path relative to the python folder instead of relative to the javascript file

Friday 19/06/20

- Zero errors but the robot is not visible in the scene
- Tried adjusting the robot scale to see if it was too small or too big to see
- Added orbit controls and realised it was not loading because I removed the render loop previously
- Robot loads as intended after a few adjustments such as scale and rotation
- Cleaned up the code and files so they are ready for uploading
- Started working on the useful outputs for setting up a urdf-loader on a server in Threejs
- Started learning how to use Github for uploading a repository

Week 5

Saturday 20/06/20

- Messing about in pybullet trying to figure out what code is responsible for the ar2 link6 and link5 to be movable
- Found that most of the constraints are movable
- Found that most of the constraints are movable and that “setJointMotorControlArray” is used for setting a desired control mode for one or more joint motors
- Got r2d2.urdf loaded in and trying to make sense of how the example code all fits together

Monday 22/06/20

- Started working on getting the example to work without npm modules
- Installed the npm modules used and moved them to a folder in static called “packagesMovedFromNpm”
- Editing all npm paths in source files to relative paths from that file in the static folder
- Only the html file can be in the templates folder ie. CSS files must be in static
- After researching on importing and exporting modules found that in the threejs example folder, you must make sure the modules are imported from “jsm” instead of “js”. They cannot be exported from js as they are not modules

Tuesday 23/06/20

- When following javascript import links in visual studio it brings up typescript file versions
- Got rid of a few errors
- Got 2 errors with the viewer: Uncaught ReferenceError: viewer is not defined at index.js:6
- and Uncaught ReferenceError: viewer is not defined

Wednesday 24/06/20

- Found by using stripped basic testing that having URDFManipulator.js as a module doesn’t export the URDFManipulator class into the html file to be used by the customElement.define() method
- Realised that might have to convert all the javascript files into non-modules if that’s possible (ie. no imports or exports)
- After more testing realised can have the customElements.define() method between script tags that are of type module so can import into here
- Getting error “index.js:6 Uncaught ReferenceError: viewer is not defined”
- This error is not due to the type being module as I tested for this in a simple page and also it must be module as it is using threejs
- It is finding where the element is last used but is saying that the declaration is undefined
- Tried using querySelectorAll() to see if there are more than one use of the urdf-viewer element but there is only one

Friday 26/06/20

- Uploaded the urdfloader from last week to Github pages so it can be demoed in the browser
- Had to change the file path from ‘static…..’ to ‘/../static…..’ for it to work with Github Pages
- Trying to get the orbit controls to orbit around the centre of the robot instead of having to manually set it. This is crucial so other robots can be loaded effortlessly

Week 6 – OpenJSCAD to URDF and Motor Modelling in Threejs/Ammojs

Objectives

- Understand how OpenSCAD works and using it in normal circumstances
- Look into the OpenJSCAD src code and try to understand how and where in the code the export process takes place
- Research ways of converting STL files or other suitable files to URDF files
- Implement an “export into URDF option” directly in OpenJSCAD

Sunday 28/06/20

- Installed OpenSCAD and started working through the tutorial to try and understand how OpenSCAD works
- Made a car and exported it as an stl file
- Designed and made a house as an exercise and exported it as an stl file

Monday 29/06/20

- Working further through the tutorial
- Learnt how to use the three useful Boolean operations: union, difference and intersection
- Created wheels using the difference function and added them to the car

Tuesday 30/06/20

- Cloned the OpenJSCAD repository from https://github.com/jscad/OpenJSCAD.org
- Got the index.html file to load OpenJSCAD in the browser and got an example copied in
- The drag and drop feature doesn’t work when running it locally in the browser and neither does the export feature where it says “generate STL (ASCII)”
- Going to research converting from STL to URDF and try and work out how Ben created the URDF file from the STL CAD meshes
- Did not find anything on directly converting from CSG/JSCAD to URDF
- Think going to have to convert to STL first then to URDF
- For STL output there are 2 types: STL (Binary) and STL (ASCII) [stla and stlb]
- Not certain which one would be more suitable to our case so going to stick with STL Binary as it is the default
- conversionWorker.js in packages\web\src\io is used to convert files to JSCAD files when dragging and dropping them in
- The README says that need a proxy file to enable drag and drop on the local page
- ---------- Ignore ----------
- In index in packages/web
- Line 509 requires stlSerializer from ‘@jscad/io’
- In the source code in packages\web\dist and in file index.js on line 534 the constant outputFormatHandlers holds all the different output types that can be selected in the select box
- It converts from either CSG or CAG to the desired output

- It says that for STL Binary it converts from Constructive solid geometry (CSG) and uses the method stlSerializer
- On line 18526 stlSerializer is required from the node module using: '@jscad/stl-serializer'
- There's also a method called stlDeSerializer and they are both exported in a method on line 18538
- CSG is required from '@jscad/csg' on line 18591
- Commented out code on line 18555 that may be useful for reference
- import * as CSGToStla from './serializers/CSGToStla'
- import * as CSGToStlb from './serializers/CSGToStlb'
- export {parseSTL} from './deserializers/parseSTL'
- This STL to OpenSCAD converter is referenced on line 18860: http://jsfiddle.net/Riham/yzvGD/35/
- Line 19257 and 19301
- Line 19371+2 for STL Binary and STL ACSII
- It says in the README of the stlSerializer: This serializer outputs a 'blobable' array of data (from a CSG object) ie an array that can either be passed directly to a Blob (`new Blob(blobable)`) or converted to a Node.js buffer.
- 'node_modules\@jscad\stl-serializer'
- ---

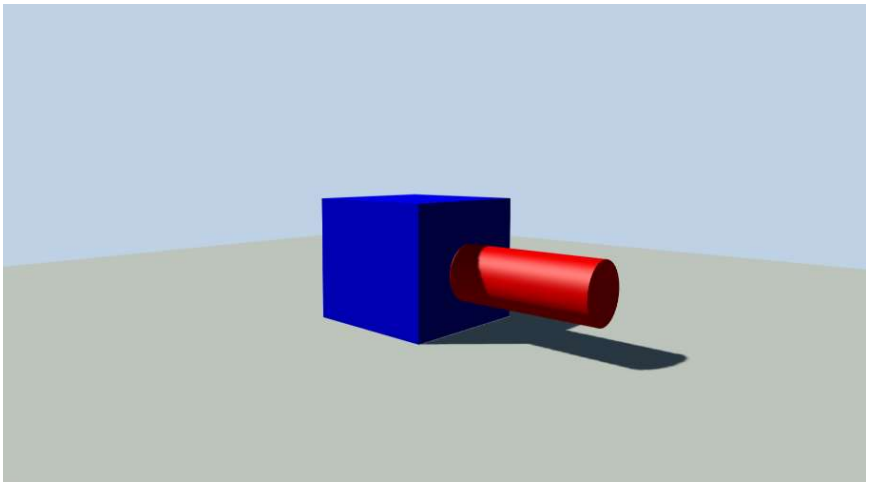
Wednesday 01/07/20

- Using OpenJSCAD I generated and downloaded an STL file in STL Binary and STL ASCII from the same model. The STL Binary file is 7 times smaller than the ASCII one and they both appear the same
- ~~My grasp on how the URDF conversion option should work is that it should first run the STL Binary conversion option then take that complete STL Binary file and do another conversion on it into URDF~~
- I'll research the other file type conversions to see if they're more suitable (X3D and DXF)
- URDF files link together separate STL meshes so not sure how this would take place in OpenJSCAD because I'm assuming that the meshes would need to be lined up exactly how they are intended to for the URDF. Also, uncertain as to how to automate the creation of links and joints when they will need to be specified manually

- Started working slowly through Matt's Motor Modelling pdf to try and understand the purpose and effect of every line written for the motors in threejs/ammojs
- Research CylinderGeometry and CylinderBufferGeometry difference
- Getting ammojs set up in the flask server so I can import modules such as OrbitControls
- Could not get ammojs to work in the local server then realised that can use the non-module version of OrbitControls which is in folder js/controls instead of the module version in jsm/controls
- After a while realised that the line taken from the Threejs website:
- controls = new OrbitControls(camera, renderer.domElement);
- should actually be:
- controls = new THREE.OrbitControls(camera, renderer.domElement);
- Got OrbitControls working on the file that I made the two blocks with the hinge joint from week 1 and the cylinders

Thursday 02/07/20

- MeshBasicMaterial is for drawing in a simple, shaded, flat way and the material is not affected by lights. MeshPhongMaterial a material for shiny surfaces with specular highlights so when orbiting round the object it looks much more realistic compared to one shade of colour on the whole surface when using MeshBasicMaterial
- Difference between CylinderBufferGeometry and CylinderGeometry is that the buffer version is more efficient but less user friendly
- Need to set quaternion for both the threejs shape and the ammojs transform shape
- From here - "When a rigid body is no longer participating in dynamic interaction in the physics world, ammojs deactivates it. In this deactivated state we won't be able to apply force to it. To stop this from happening set the activation state for the rigid body to STATE.DISABLE_DEACTIVATION."
- rigidBodies.push(cylinder) adds the threejs cylinder to the rigidBodies array so it can be retrieved when we want to update objects after a physics simulation ie. without this line a dynamic object would be static
- addConstraint(p2p, true); the true parameter sets disableCollisionsBetweenLinkedBodies to true so that the linked bodies (cube and cylinder) don't collide
- Successfully added the point to point constraint by following the Motor Modelling pdf Matt created so that the cylinder and block are joined like a motor as shown below.



Week 7

Monday 06/07/20

- Working through the final rotation section of Matt's Motor Modelling guide
- Event keycode is the corresponding number to the key that was pressed/released
- addEventListener with the 'keydown' function as the first parameter listens until the user presses down any key and calls the handleKeyDown function to check if the key that was pressed down is either the up or down key
- setAngularVelocity is used to rotate the cylinder with an input resultant force
- scalingFactor is used to control the force and hence the velocity of the rotation and resultantImpulse is a vector of the direction the force should be applied
- Was getting error then realised that I had declared cylinder in the createCylinder function aswell as globally when I should only have reassigned cylinder in the createCylinder function
- The motor spins successfully when holding down the arrow keys
- The orbit controls zoom in and out slowly when holding down the up or down key so need to remove that feature
- If you go to line 76 in three\examples\js\controls\OrbitControls.js you can disable the use of the arrow keys by setting the variable enableKeys to false
- The control of the rotation of the cylinder using the arrow keys works as intended
- Started working on the short useful outputs for implementing OrbitControls to an already existing Threejs scene
- Finished the guide for when not using modules

Tuesday 07/07/20

- Finished the useful outputs guide for when you are using modules
- Started reading through Matt's Advanced Ammojs/Threejs guide
- Was working through the iterative function section which uses for loops to create multiple blocks stacked on top of each other to from a wall
- Got the iterative code implemented successfully and played around with it
- btStaticPlaneShape simulates an infinite non-moving (static) collision plane
- Worked through the invisible shapes section of the Advanced Ammojs Threejs guide

Wednesday 08/07/20

- Researching about obj file type (waveform)
- "The OBJ file format lets you store colour and texture information in a companion file format called the Material Template Library (MTL) format. (.MTL)"
- Going through the ar2 urdf file trying to understand how they work
- Create "links" from the meshes then join them using "joints"
- All the joints are of type continuous
- Visual tags seems to be for what you visuals for the scene then there are collision tags for the collision shape for collisions?
- Within the visual and collision tags are the same except collision doesn't have material and hence colour
- Link is created then a joint to pair the link with the previous link
- The inertia tags use 6 elements of a 3x3 rotational inertia matrix

Thursday 09/07/20

- OpenJSCAD allows export into STL(ASCII), STL (Binary), AMF (experimental), X3D, DXF, JSCAD, js
- Found that OpenSCAD allows export into STL, OFF, AMF, 3MF, DXF, SVG, CVG
- Tried converting from meshes from STL to OBJ on https://www.meshconvert.com/ and it seemed to work well
- Found that OpenJSCAD has an obj type deserializer but not a serializer
- Assuming must be able to drag and drop obj files into OpenJSCAD
- If able to convert obj files to OpenJSCAD files then possibly able to reverse the process
- Get the following error when trying to drag and drop obj files: The JSCAD script must return one or more CSG or CAG solids
- Found some code that might work for exporting as obj in OpenSCAD

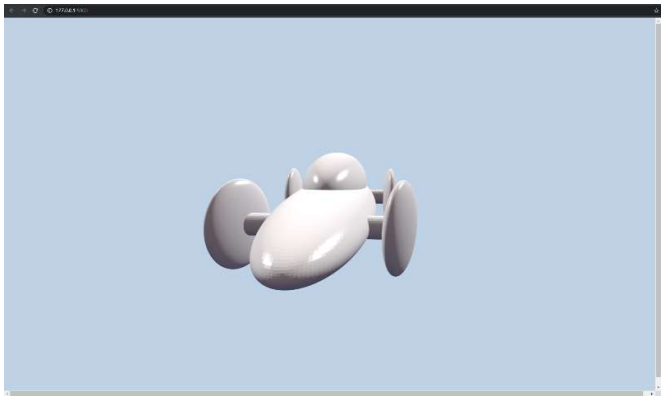
Friday 10/07/20

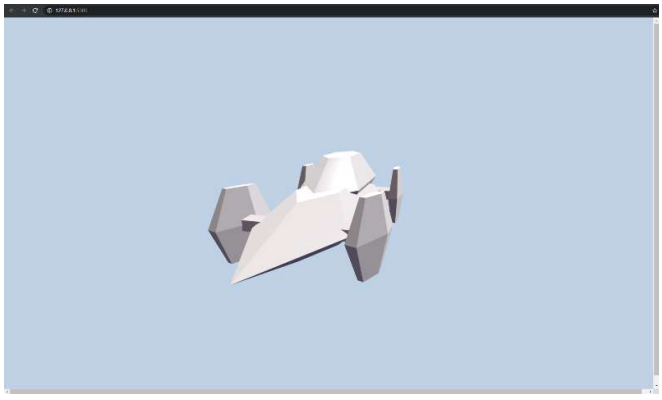
- Need to ask a few questions today as to what direction to go in so going to learn JavaScript better in the meantime
- Working through Basic JavaScript on https://www.freecodecamp.com/
- Thread found that states that there is a way to implement basic obj export: https://github.com/openscad/openscad/issues/351
- https://github.com/openscad/openscad/tree/objexport99
- https://github.com/openscad/openscad/compare/objexport99

Week 8

Monday 13/07/20

- Improved the loadRobot.js code by putting everything into functions so when I start working on loadSTL.js it will be much clearer to work with
- Added the loadSTL module code to the flask server and it successfully loads in the stl but it is just a white shape so you are unable to see edges of the object, just an area of whiteness
- Added a default material for objects being loaded in without a material/colour so it looks a lot more realistic
- Loaded in a few of the stl exports of the OpenSCAD models I made a couple of weeks ago and they look very nice
- Fiddling with the lighting and the camera etc. to obtain the most suitable-looking scene for the stl models





Tuesday 14/07/20

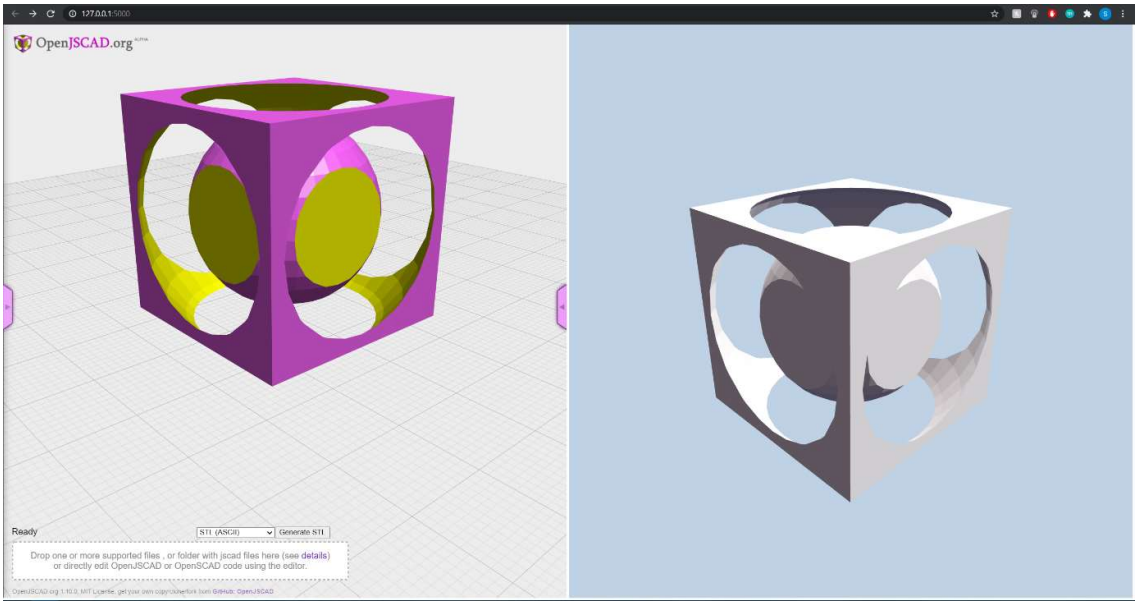
- Downloaded OpenJSCAD from the Microsoft store
- Reading through the following link figuring out the best way to implement OpenJSCAD https://github.com/jscad/OpenJSCAD.org/blob/master/packages/web/README.md
- Starting with trying to get OpenJSCAD to work in the flask server
- Had to move the index.html into the templates folder so need to change all the paths
- Diving into the packages/web/dist/index.js file to edit sources
- Line 95114 to change the left menu source
- Changes made on lines 95174, 95178, 95192, 95195
- Got the drag and drop working so it loads in my OpenSCAD files
- Changes on lines 93609 – 93616
- For the example to work, changes made on line 94820 (uncertain if necessary) and line 94995
- The auto-reload button is a bit glitchy as it keeps reloading when there are no changes made
- Spheres do not visibly load in on the sphere car
- Get the following error: WARNING: Ignoring unknown module: resize
- Sphere's load in when not using resize so need to have a look
- Get the exact same problem when trying to load the SCAD files on the OpenJSCAD website so the problem must be an already existing problem with OpenJSCAD importing SCAD models

- The OpenJSCAD app from the Microsoft store didn't seem to load any of the SCAD files never mind work with the resize function
- Works if the SCAD model is exported as stl in OpenSCAD then imported as an stl in OpenJSCAD so must be a problem with using the resize function when importing SCAD models
- The mouse drag is much slower when using OpenJSCAD on the server than on the original website
- Realised it was slower only when the chrome developer tools were open

- Ready to have a go at getting OpenJSCAD and Threejs side by side on the same page
- Then need to do more tests
- Then need to try to get the stl export download button (or add a separate option to export into Threejs) to export it into Threejs

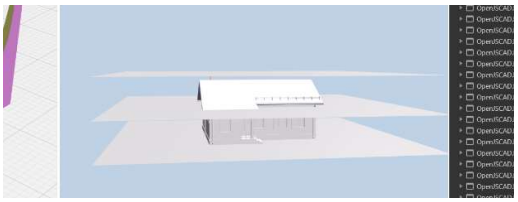
Wednesday 15/07/20

- Looking for a way to have Threejs and OpenJSCAD open at the same time
- The Threejs stl loader is on index.html and OpenJSCAD is on index2.html
- Looking at iframes for getting the two html pages to be side to side in the browser
- Made a new html page called iframes.html which only has iframes code on it
- The iframes page cannot seem to find the html pages
- After researching and trial and error I found that moving index.html and index2.html pages to the **static** folder allowed them to be accessed by the iframes.html file
- Must change all the sources again in the OpenJSCAD source file as the html files were moved
- Got them fitted nicely side by side and both are fully functional (each width set to 49.8%)
- Need to get the Threejs page to adjust as the page is resized and also to get both of the iframe heights to fit the height of the page because the current way is set manually
- Setting the iframe height to 100% does not do anything so need to figure that out
- Found that by adding a css page and setting both html and body width and height to 100% it made the body fill the whole page so setting the iframe height to 99.5% works now



- Tried adding a CSS page to get the Threejs page to stay centred in the browser when resizing but could not get it to change

- Also need to get the orbit controls to centre about the centre of the stl object loaded in instead of setting it manually
- Created a box around the stl object and found its minimum and max height to work out the center height
- The controls were not centered so I used planes at the y positions to see what was wrong
- The center height was wrong as the stl object is not centered about the y axis (max.y = 14 and min.y = -8 I had worked out the center height (11) instead of the center position (3)) so to find the real center, the centerHeight was taken away from the box.max.y (14 - (22/2) = 3)
- The correct y position centre can then be passed as a parameter to the orbit controls function, so I removed the orbit controls call from start and put it within in the loadStl function



- Fixed it so orbit controls now rotate about the vertical y center of any stl objects loaded in so that it does not have to be set manually
- Put orbit controls call back into the start function and removed its argument and just added the controls.target and update the controls in loadSTL instead of calling orbitControls

Thursday 16/07/20

- Looking for a way to link the OpenJSCAD stl output to the stl loader of the Threejs scene
- Looking at the generate stl button and trying to figure out all the code behind it
- Could download the file then the src is sent to the stlLoader but I think I will need to change the stlLoader code so that it can take in the stl file straight from the OpenJSCAD export

Friday 17/07/20

- Decided going to implement a way so that the Threejs stl loader can take in the stl file straight from the OpenJSCAD export
- Thought about working on a drag and drop for the stl loader for demonstration purposes for the meeting but decided it wouldn't be a very good use of time
- Need to change the Threejs html/css code so that it auto resizes when changing the size of the browser
- Need to update the OrbitControls guide to add about automatically setting the vertical y axis target
- Uploaded the page to github.io for the meeting but getting a 404 error when trying to located logo.jscad for the example
- Updated the orbit controls guide by adding an automatic target section and also if setting up the orbit controls as a separate function
- Couldn't fix the error so had to comment out line 95003 where the logo.jscad is fetched so the only difference is that you have to press f5 on the code editor to load the example, each time you reload the page

Week 9

Tuesday 21/07/20

- After meeting, going to try to use divs instead of iframes for the separate html files
- The css for the webpages kept overriding each other so going to try just using the iframes first
- For creating global variables in JS you must use var
- For accessing the global variables created in the parent html page, you must add 'parent' in front of the variable
- So for any variable that need to be used between the two JavaScript files, they must be declared in the iframes.html and accessed using the parent call
- Need to find where the stl file is created and exported

Wednesday 22/07/20

- Opened some stl files in an editor and found that the stl ASCII is much more readable than the binary so I'm going to work with exporting in stl ASCII to begin with
- Found the last line of the export on line 19274
- Got it working so that when the export as stl ASCII button is pressed, the loadSTL code will ouput in the console "export completed"
- Trying to change the STLLoader code so that it can take in the stl as a string parameter instead of it searching for an stl file
- After lots of testing, was able to successfully edit the load function so that the code no longer looks for the file but loads it directly from a parameter
- A key point I was missing was the onload function, as the Threejs object won't load in unless this function is present
- Have to reload the page every time you want to add load another stl object so need to reset Boolean variables after loading in
- Works multiple times so you can load in multiple objects but they load on top of each other
- Tried with at least 5 objects and it works fine
- So theoretically able to drag and drop any file type that OpenJSCAD allows you to import then you can export that data into an object in Threejs
- Not perfect, was trying an export from on of the OpenJSCAD examples and some parts of the faces are much darker/lighter than they should be
- Could be something to do with the material
- Changed the export selection name to "Threejs Export - STL (ASCII)"
- Tidied up some of the variables and code

Thursday 23/07/20

- Going to start to work on the urdf converter
- To begin with I'm going to try and get two of the ar2 meshes to load into the scene in the correct position and to have the correct joint between the two meshes
- Tried using the urdf loader and realised I had edited the stl loader so need to create 2 versions
- Going to test whether the urdf loader robot is affected by an ammo object
- Need to import ammo into the page and changed the name of the edited STLLoader to STLLoaderOpenJSCAD and it works fine for importing from
- Uploaded to github pages and found there was a problem where github wouldn't upload the examples folder in the web folder so had to reference the examples folder in the one above

Week 10

Monday 27/07/20

- Starting off with trying to get ammojs working in the flask server
- Add the script tags for ammo in the threejs html page instead of iframe
- Used the code that John sent into the chat to initalise Ammo because get an error if use the basic Ammo().then(start)
- Added all the ammojs functions such as setupPhysicsWorld and updatePhysics etc.
- Added all the variables needed and got a cube with mass to drop successfully
- Going to create a basic OpenJSCAD model for testing out the transfer to threejs
- Finished an OpenJSCAD model which is a cube and cylinder, similar to Matt's threejs motor
- The model is a union of the two shapes and the cylinder is overlapping into the cube very slightly to create an object of them joint together but I don't think that will matter for the stl export as I think it is only for the surfaces
- Need to come up with a suitable way to store information for the two shapes
- Going to try to use the html parent variables to store the current OpenJSCAD code
- Found in the index.js file where the code from the page is used
- It's the source parameter on the replaceIncludes function on line 218
- Going to try storing all of this code as a global html parent variable called "openjscad_code"
- That works so the OpenJSCAD code, shown above, is logged to the console from the loadStl file

Tuesday 28/07/20

- Think I'm going to have to change the OpenJSCAD code. It says on the wiki:
  "An **OpenJSCAD** script must have at least one function defined, the *main()* function, which has to return a **CSG** object, or an array of non-intersecting CSG objects."
- I have the two shapes as a union currently when I believe they should be two separate shapes rather than one conjoined shape
- Could create an array of the objects instead of just adding them#
- Changed all the OpenJSCAD code so that the shapes are added to an array and so all the values are specified as variables at the top so they can be changed easily
- Going to get Matt's motor code implemented in the loadSTL file
- Got the motor to spin under arrow key control
- Going to make a list of all the parameters needed for creating the block and motor and for creating the p2p joint constraint between them
    - Position xyz
    - Scale xyz (Going to remove this from the threejs as the width and height etc. is enough. Also going to need to change the collision shape to depend on the dimensions instead of the scale.)
    - Quat xyz (Rotation (w stays as 1 by default))
    - Mass
    - Radius top and bottom (cylinder)
    - Height (cylinder)
    - Segments (cylinder)
    - Pivot point for cylinder
    - Pivot point for block

- Going to keep friction as set in Threejs as it shouldn't change but can be changed if necessary
- Started writing the code for reading in the number of objects and also the variable values
- Started writing the code for reading the box variables from the OpenJSCAD code
- Figured I could write a reusable method for finding an array value
- Got the method down very clean at 10 lines and it's pretty robust so it can be reused to find every x, y, and z value for each array so in this case it will be reused 21 times
- It takes a desired variable name and finds the value of the x, y and z assignments to that variable in the OpenJSCAD code

Wednesday 29/07/20

- Created a new function for reading variable values instead of array values
- Trying to create the block in Threejs instead of using the STLLoader for now as the shapes are simple and in ammojs will need to create collision shapes etc. so it makes sense for now
- The y and z are different in OpenJSCAD and Threejs so for translation/positioning you have to swap y and z and make y negative
- Got box working for the Threejs code and changes successfully when you change the OpenJSCAD variables as expected
- Added the ammojs code and going to do some tests like dropping a ball on it
- Had to add a few more lines to the ammojs section to get the block to move and changed its mass to 1 and it dropped successfully under gravity
- Changed its mass back to zero and the ball drops on it as expected
- Added Threejs code for rotation and did tests
- Tested with ammojs rotation and the collision shape is as expected
- Started adding in the cylinder import code and can load it in successfully in Threejs
- For the collision shape code for the cylinder it seems to take a vector instead of a height and radius so might have to change the inputs for that
- Found a thread that suggests to use (radius, height*0.5, radius) instead of (scale.x*0.5, scal.. etc.
- Need to check the pivot points are the right way round ie. x z -y
- The cylinder is rotated around 60 degrees down in the x axis which I need to figure out why

Thursday 30/07/20

- This only happens when the cylinder rigidbodies.push line is active
- Tried changing the collision shape values and there is no difference
- It's not the interaction with the box or plane
- Realised the reason it was at that angle is because for a 90 degrees rotation in quaternions that is equal to 1
- The rotation doesn't quite rotate right as when rotating about the y axis, on OpenJSCAD it looks the same but on Threejs it looks different
- Changing the collision shape values does not make any difference
- Figured out that quaternions are complex in that you cant just enter angle of rotation for each xyz axis when rotating in more than one axis at once
- This means you can still rotate the cylinder successfully in one axis but not in two or three
- Also can't seem to rotate the box as it doesn not have the rigidbodies.push line like the cylinder has
- Added the pivot code but the cylinder slides into the cube so the position of the pivot must be wrong
- Adjusted the pivot point to 0,-20,0 and it works perfectly

- Going to try to add the move cylinder code
- Added some code so the moveCylinder() function only runs if a the constraint1 is created
- Going to create the arm in OpenJSCAD now
- Added the arm and got in working in the Threejs side
- There is a problem with the constraint between the arm and the cylinder as the arm goes through the cylinder even when it's not meant to
- Spent a lot of time trying to get the constraint working but didn't get any closer

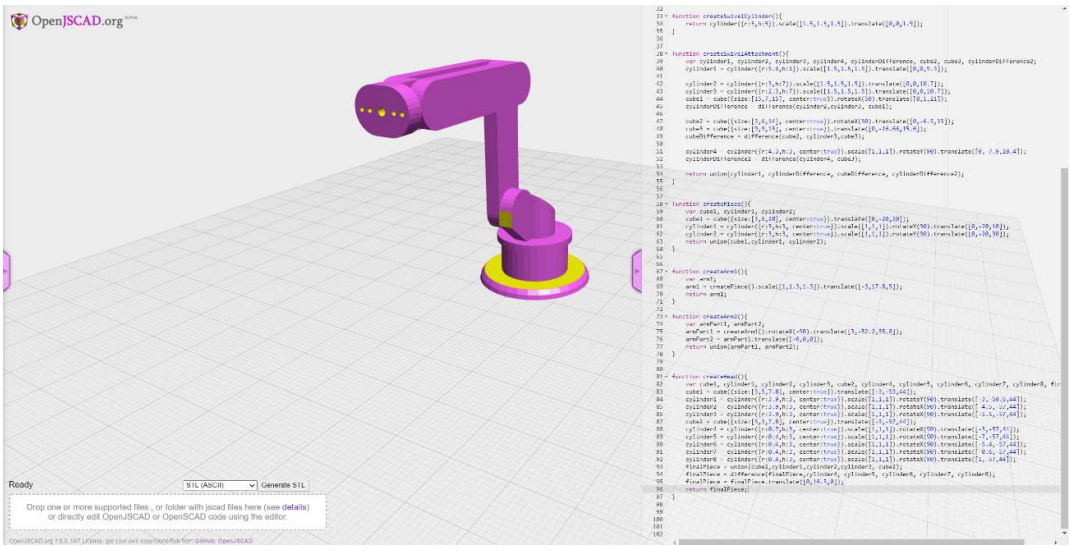
Week 11

Monday 03/08/20

- Starting off by trying to make a kuka-like robot arm in OpenJSCAD
- Also basing some bits off the AR2 robot
- Finished the base, swivelCylinder and swivelAttachment

Tuesday 04/08/20

- Finished off the robot arm as shown below



- When checking what a straight export of the robot arm from OpenJSCAD code into STL then into Threejs looks like, some of the textures aren't very well exported, as shown below

- Need to look into the source code for figuring out the best way for exporting the OpenJSCAD objects/meshes as different stl meshes
- Firstly, need to figure out how to identify meshes in the object array before exporting
- Line 19270 of web/dist/index.js for the final export of the complete STL file
- Need to create multiple STL files – one for each mesh
- Line 185 is an array containing each of the objects in the OpenJSCAD array so in this case it contains 6 objects
- Line 483
- Line 528 – goes through the else
- Line 531 a function mergeSolids2 is called and it could be a function that merges each of the 6 objects together
- Line 33960 is the location of mergeSolids2
- Line 33991 is the for loop that combines the objects using the union function
- Was able to successfully export the base alone by changing the line

    object = mergeSolids2(objects, formatInfo)

to object = objects[0];

- Line 548 key
- Line 19392
- Was able to add code so you can manually export each object by hitting f5 on the OpenJSCAD code (so it allows you to hit the export again)
- Found the code for f5 for reloading the objects (from the editor) at line 94676
- Init on 95097
- Going to try to reload the cad shapes from the editor when the Generate STL button is pressed so that you don't have to hit f5 to reload the page as a start
- Line 93434 and line 93505 for the generateOutputFileButton
- Line 93436 is the line I was looking for: that.generateOutputFile();
- This is the line which starts the whole output process so would maybe be able to follow the way through the code to find a way to get it to run through the whole objects array or start with getting it to refresh the page on click of this button

Wednesday 05/08/20

- Thinking that when the generate stl button is clicked, it needs to:
    - Reload the code
    - And generate another stl mesh
- Line 93108 is the function for generating the file
- Could maybe have a Boolean variable that is set to true when the file is created then the event listener for reloading the page waits for that file to be generated, then reloads the page
- Got an auto reload OpenJSCAD function to run after the Generate STL button is clicked by adding an event listener which is slightly delayed and checks for "mouseup". When the generate button is clicked, a Boolean variable is set to true so the auto reload function if statement runs.
- This isn't great as the generate stl button still has to be clicked for each mesh
- The delay before running the function needs to be at least 200ms
- Figured I could put a for loop around the generateOutputFile() function which is run when the generate stl button is clicked
- Staggered the executions of the functions calls, using setTimeout, to allow the stl files to be loaded into Threejs and it works

- So each mesh loads into Threejs successfully one by one
- No need for the “mouseup” method for autoreloading anymore so need to tidy up the index.js and iframe.html code then work on Threejs methods
- Cleaned up index.js and iframe.html so ready to start adding Threejs methods for the meshes tomorrow and to look at a transferring a second set of data across to Threejs for the physics values

Thursday 06/08/20

- Added a little rectangle block in union in the middle of horizontal arm (mesh 5) so the two sides are joined together
- Added a textarea in the left panel where the js dictionary text should go
- Added a method for Threejs to resize with the browser window
- Trying to figure out how to create custom collision shapes in ammojs
- Looking at all the examples and on forums
- https://stackoverflow.com/questions/59665854/ammo-js-custom-mesh-collision-with-sphere
- The forum on the link above uses Ammo.btTriangleMesh and Ammo.btBvhTriangleMeshShape for creating the collision shape for the ammojs object
- https://stackoverflow.com/questions/39441459/rigid-body-shape-in-bullet-ammo-js-from-a-mesh-in-three-js
- This forum says to look at the source code for physijs where a concave collision shape can be created

Friday 07/08/20

- Added code for the orbit controls so that it centers the height out of all of the objects instead of the last one
- Also added code so that it centers in the x and z direction too
- Going to try look through the physijs source code to try to understand how it creates custom collision shapes
- The geometry.faces.length line didn’t seem to work so when looking the in the loadSTL source code it seems for the parseASCII it doesn’t use geometry.faces whereas parseBinary does.
- Going to try to get parseBinary working with it in a separate file
- It didn’t work either then I realised that I should be adding the code to the three module because the linked code is in the physijs module and the faces array is local to the three module
- Tried to implement different methods into the loadSTL and threejs.module file but I think the face array was always undefined so not sure if the geometry that is created from the loadSTL file has that
- Uploading to github pages (note line 95039 and 94845 for future)
- Should have a look at the urdf loader to see how it creates rigid bodies and constraints

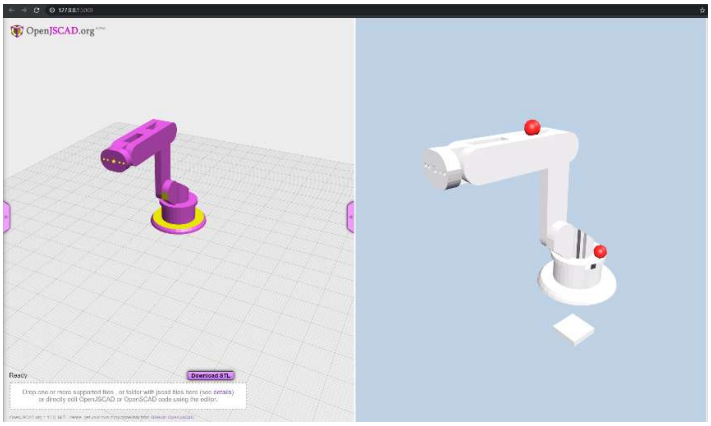
Week 12

Monday 10/08/20

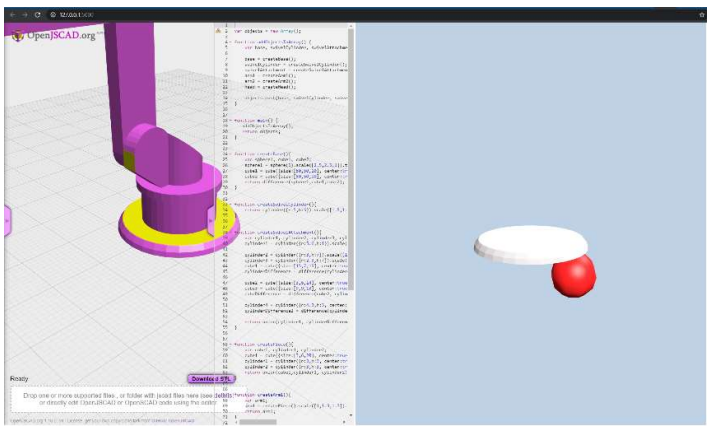
- Working on creating the collision shape for the Threejs objects
- Able to use objectArray[parent.counter].geometry.attributes.position.array[index] as a point of a triangle for each mesh
- Got to a point where it theoretically should work
- Tested the ball drop and it dropped through the robot arm
- Set the mass of the arm to 1 and all the parts drop
- Created a plane 20 units below origin and the parts all drop then land in the perfectly in the robot arm shape which is strange
- Going to add a function for shooting balls using the mouse to test the collision shapes

Wednesday 12/08/20

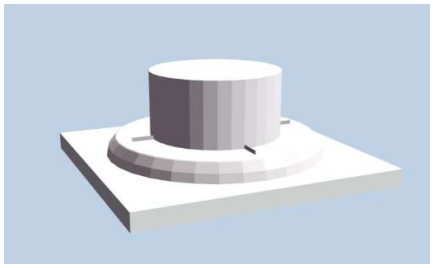
- Realised I had the set the vectors to THREE.vector3's instead of Ammo.btVector3's
- Tried dropping the balls on just the base mesh and they rolled off just as expected so the collision shape seems to be set right
- All the collision shapes seem to be perfect in the all the parts



- Seems to be a problem with the inertia/center of mass in the parts. Shown below, the base part drops down and lands on the ball but gets stuck like this when it should fall due to gravity



- Got a fixed constraint to work between the base and the cylinder part



- Need to work on the rest of the constraints then I'll know what data I'll need for every constraint when adding in json/js dictionary

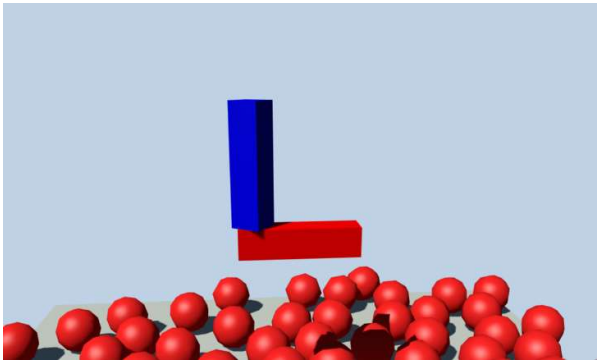
Week 13

Monday 17/08/20

- Added planes to figure out the exact position that the next constraint should be at
- Tried adding the constraint but the mesh slides to the side
- Added a cube4 difference to remove a protruding edge at the bottom of the swivelAttachment (part 3)
- Trying to figure out how to get the hinge constraint to work
- Realised the part needs to be moved a bit laterally to sit as intended

Tuesday 18/08/20

- Going to try to get a similar basic hinge working in a separate scene so I understand better what is going wrong
- Got the hinge to work where the blue cylinder spins about the y axis, as shown below



- Cylinder1 has a radius of 5.6*1.5 = 8.4
- If I can find the bounding box width in the x direction I can work out the size of that sticks out to attach to the arm so then I can work out the position change for finding the center of the cylinder
- Turns out need to work out the z change instead of the x change
- 12.16 – 8.4 = 3.76
- 12.16 + 8.4 = 20.56 (the total width in the z axis
- So the current center will be half that, at 10.28
- 10.28 - 8.4 = 1.88
- 10.28 - (3.76/2) = 1.88
- 8.4 – (10.28 – 3.76) = 1.88
- Therefore the center is off by 1.88 in the z axis
- After trying that out realised that the wrong face is lining up in the center
- Testing out without the hinge with the balls to test the faces
- Thought maybe the base and second part became one object but that was proved wrong
- Trying just the cylinder1 part of the object and it seems to be off center too
- Might add code so balls can be fired using the mouse to help figure out the way the hinge moves

Wednesday 19/08/20

- Going to start working on the write up for the subproject
- Starting by cleaning up all of the code
- Commented almost all of loadSTL.js

Thursday 20/08/20

- Finished commenting loadSTL.js
- Deleted all irrelevant files from the static and templates folders
- Changed the balls so they drop only after the last part has been added in
- Changed the camera angle to a more suitable angle
- Started the write up for the sub project

Friday 21/08/20

- Uploaded to Github successfully and created a folder in my Github Pages for the OpenJSCAD-Threejs project and another for the Threejs-URDF-Loader project
- Line 95029 94854 94835 for changing the examples and their src for uploading the Github
- Working more on the guide and wrote the “getting everything set up” section

Week 14

Monday 24/08/20

- Started working on the “Understanding the code” section of the writeup
- In this section going to go through each of the 7 most important files and try to explain in the best detail all that is going on and useful information
- Got the first two out of the seven finished

Tuesday 25/08/20

- Completed the third section for the OpenJSCAD.html page

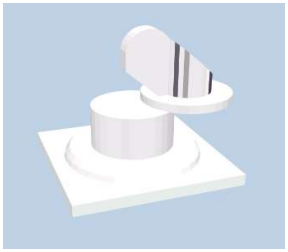
Week 15

Saturday 29/08/20

- Completed the fourth section for the Threejs.html page

Sunday 30/08/20

- Started working on the hinge again and getting the same measurements for the pivot point
- Part 3 seems to slide off when the hinge is constructed, as shown to the right



- Going to add keyboard input so that part 3 can have an angular velocity applied to it so that I can get a better grasp at how the hinge is working
- Got it to rotate about the y axis when using the arrow keys but it seems to rotate about that position
- Trying different positions then finally got it. Found that by having both the pivots set to (0,0,0) worked for some reason and the hinge rotates perfectly as expected
- Going to try the next hinge and going to set up different key pairs for each hinge
- Changed the movement keys to Q and A then the next pair will be W and S etc.
- Got the hinge between part 3 and 4 working close to perfectly
- Adding so that both the hinges can be moved using different key pairs
- Added a method for moving the hinge 3 laterally because, without it, part 4 doesn’t rotate with part 3
- When the method for moving it laterally is added, then the other hinge 3 method doesn’t work
- Going to leave it for now and try to add the hinge between 4 and 5
- Got the hinge 4 working perfectly
- Going to try adding the final hinge
- Got the final hinge working perfectly so all the hinges now are in place

- Arm pieces can pass through each other but they don’t pass through the block which is strange
- Need to figure that out and how to get them to rotate with the hinge 2 also
- Then need to create parameterizable functions so that they can be reused for every hinge
- Trying to work out how to rotate every part with hinge 2

- Tried setting a new key for rotating part 4 but then the usual rotation doesn't work, similar to last time
- Got the whole rest of the arm to rotate with hinge 2 but when using the other hinges again, they rotate at crazy angles
- Need to find a way to calculate the rotation angle from where it is currently rotated at
- Found a method called setAngularFactor so I created a function where all the relevant parts have their angular factor set to only about the x-axis but it doesn't seem to work

Friday 04/09/20

- Going to try to add text that informs the user of the controls for the hinges
- Looked at the soft body cloth ammojs example and implemented the code for the text at the top of the screen
- Tried using setLinearFactor but then the hinges seem to stop being attached to all the limbs
- Realised the vector for setLinearFactor and setAngularFactor need to be opposites
- So, hinges work again but not as bad as without the two methods
- Need to figure out a way to change the axis of the hinge as the robot arm is rotated round
- Going to upload the current stage of progress to github pages
- Uploaded successfully and could look at using the mouse for dragging the meshes with respect to the hinges

Saturday 12/09/20

- Setting up the project to work on my laptop by following the Threejs OpenJSCAD Robot Arm guide
- Found various problems that need to fix
- Stuck with a mimetype error

Wednesday 16/09/20

- Finally fixed the mime type error by firstly adding in some code to the app.py file then changed went into the regedit and changed the content type from **text/plain** to **text/javascript**
- Going to update the guide to explain how to do this as I couldn't find any other way to get it working
- Realised this is a flask issue so it shouldn't be too much of an issue as it will work on Github regardless
- Need to update the write up to the current stage then I will finish explaining the final two files
- Commented more of loadSTL.js so that it is up to date and cleaned up the code a bit
- Added in more useful information that I learned when setting up the project on my laptop
- Started writing up the explanation of loadSTL.js

Thursday 17/09/20

- Finished the loadSTL.js section
- Wrote the STLLoaderOpenJSCAD.js section
- Wrote the index.js section
- Wrote the What's Next section

Friday 18/09/20

- Tidied up the write-up document and code

Saturday 19/09/20

- Working on figuring out the hinge problem
- Broke it down into two small cuboids
- Realised they weren't acting like ragdolls either so changed the second pivot to 0,0,0 and it started sort of normally
- Need to test more