OSR2 Build Instructions

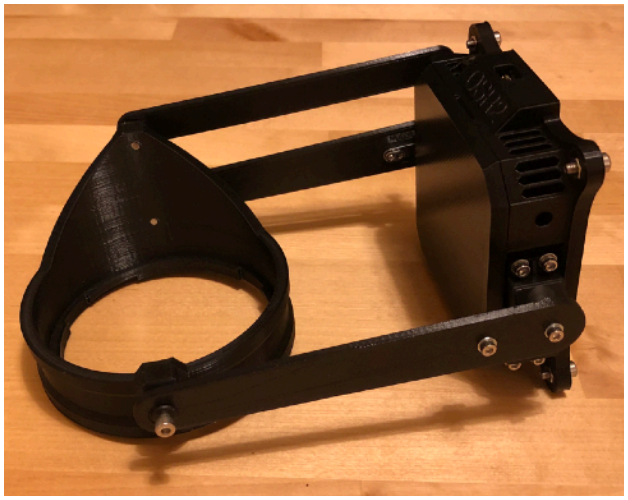
TEMPEST
TMAx
MULTI-AXIS

June 2022

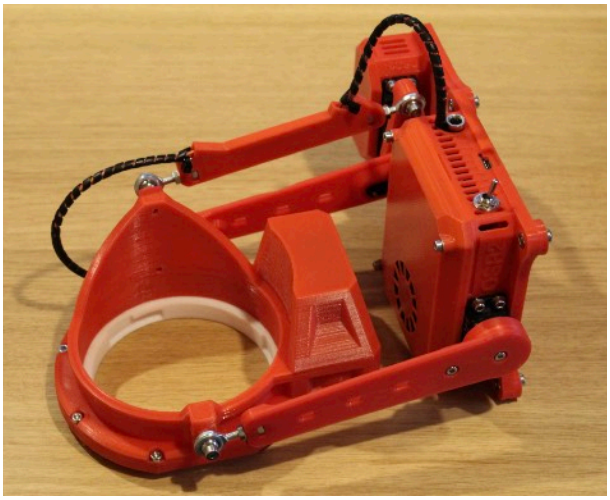
An assembly guide for the
Open-source Stroker Robot, 2-Axis

Introduction

I first released the OSR2 in February 2020, over two years ago, and in that time I have been absolutely astonished by its popularity. I don't have any way of knowing for certain, but based on the numbers of my patreon and subscribestar contributors, plus the number of free downloads of the original from thingiverse the number must now be squarely in four figures.



OSR2 FEB 2020



OSR2 MAY 2022

If I had to guess I would say that the secret to its success is its simplicity. The base 2-axis design is essentially only six 3D printed parts, some bolts, a couple of servos and a microcontroller. Even as I have expanded the design to include movement in 4-axes and variable suction, as well as some other mods not listed in this document, I have kept to the same philosophy: keep it simple, and keep it off-the-shelf. This is a toy that you can build, upgrade and maintain for yourself, and you don't even need to be a skilled hobbyist to do it.

I have created this pack as the ideal "start here" point for the new OSR2 builder. OSR2 is ideal for people who are new to homebuilding toys because it's a modular design. You can start simple, with just the basic design, and then upgrade your machine as you go along. These plans will walk you through the process step by step; they also include a complete list of the parts that you will need.

This guide includes assembly guides for the OSR2, a pitch control module (OSR2+), a twist receiver (T-wist) and the automatic variable suction valve (T-valve). My recommendation is that you work through building your OSR2 a stage at a time, but I accept that a fair few of you will just jump in with both feet and build the whole thing in one go!

I would like to thank you all for supporting my work. Even now the design, hardware and software of OSR2 continues to evolve, and it's you guys that make it possible.

Tempest, 15-5-2022

Table of contents

Record of changes

Issue	Date	Changes
OSR2 Beta 1	1-2-2020	First issue
OSR2 Beta 2	10-4-2020	Changes to beta 2 parts: • Enlarged cutouts in main arms to fit larger T25 metal servo horns • Modified base to screw servo attachment M3 bolts directly into plastic. Added to document: • Extended notes on the Romeo BLE mini (v2 & v1.1) • External Power Bus • OSR2+ (Pitcher) Assembly guide
OSR2.1 Beta 1 (Nov ’20)	16-11-2020	Changes to OSR2.1 parts: • Redesigned base and lid (inc fan lid) • Arms strengthened and cable tie holes added • Added chamfer to receiver base Added to document: • T-wist Assembly guide • T-valve Assembly guide • Extended user tips, guides, etc
OSR2.1 Feb ‘21	10-2-2021	Changes to pitcher parts • Servo mount screws now screw into M3 nuts • Pitcher enclosure size increased by 2mm • Pitcher cut-out widened Added to document: • Updated pitcher build instructions • Updated troubleshooting
OSR2.1 Sept ‘21	22-9-2021	Parts updated: • Pitcher cut-out lengthened • Twist ring gear updated to locking type • T-valve base updated to triangular hole type • ESP32 base option included Added to document: • ESP32 guide
OSR2.1 May ‘22	16-5-2022	Document & Parts updated: • New swanky cover page added • All assembly sequence images updated to show up-to-date hardware • Twist receiver hardware and instructions updated to T-wist mk4 Added to document: • Description section added • Bearing arms upgrade added Removed from document: • Hardware fix to reversed T-valve removed
OSR2 June ‘22	8-6-2022	Minor update: • Re-titled “OSR2” • Tempest Multi-Axis logo added to cover • Minor corrections to text

Description

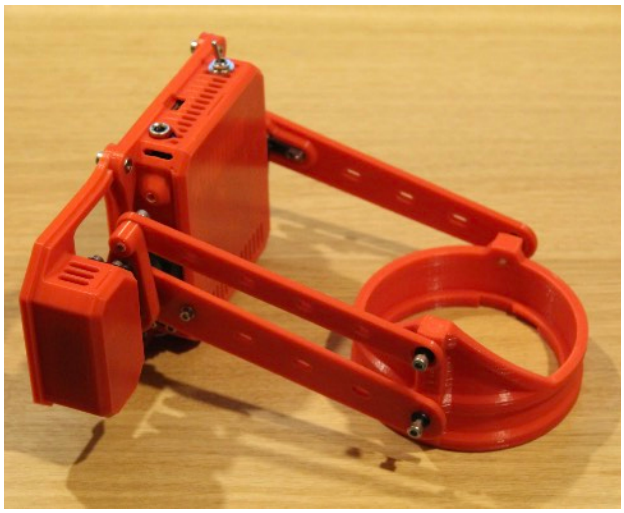
The OSR2 is a Multi-Axis Stroker Robot (MAxSR) designed to hold and move a standard fleshlight or similar toy with 2-4 degrees of freedom. It is constructed from a combination of 3D printed plastic and commercial off-the-shelf components. It is designed to be home built, modifiable, and user-repairable.

The basic OSR2 can move up and down and roll left and right. A module can be added to the side of the device to add an extra axis. This configuration is called the OSR2+ and is also capable of pitch forward and backward.

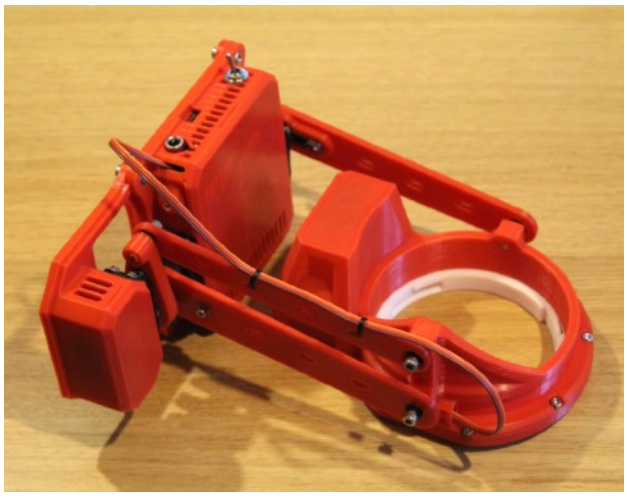
In the OSR2 the standard fleshlight is mounted into a ring-shaped part called the receiver. This part can be upgraded with a twist mechanism or “T-wist”, which adds a fourth movement axis and allows the OSR2 to rotate the mounted toy around the stroke axis.

At the core of the OSR2 are two 20kg.cm hobby servos, which provide the driving force; and additional attachments also use various other servos. The default stroke length of the OSR2 is 136mm, and the pitch and roll axes can rotate the receiver approximately ±30 degrees. The twist angles depend on the type of servo used.

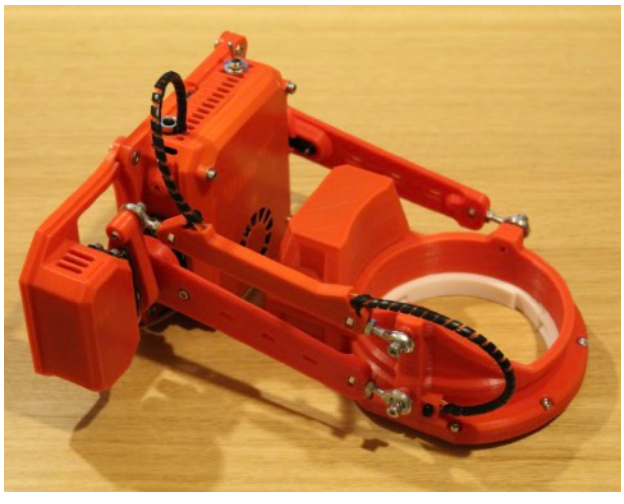
OSR2



OSR2+

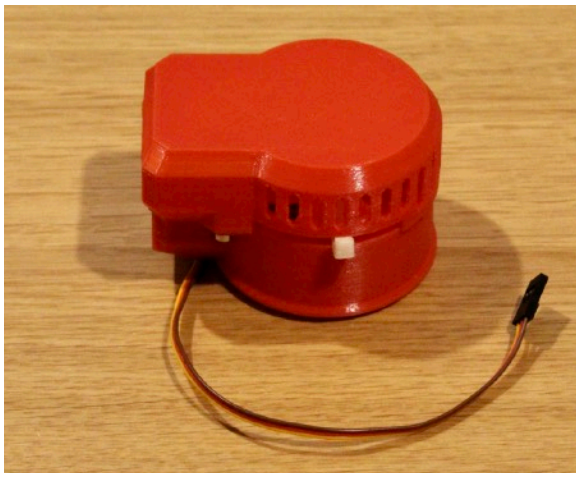


OSR2+ WITH TWIST



T-OSR2+ WITH TWIST, BEARING ARMS & FAN LID

The OSR2 can also control the dynamic suction inside the case of a standard fleshlight using an accessory called the “T-valve”. This uses a servo to automatically open and close the air gap at the top of the case.



The “OSR2.1” featured in this document is an improved version of the original OSR2 that features a redesigned base, as well as modified arms and receiver.

Control is via a generic open source USB serial protocol called T-Code. The OSR2 uses a a Romeo BLE min arduino-based microcontroller, or an ESP32 microcontroller to receive these simple commands and calculate the appropriate control angles for each of the servos to achieve the desired position for the receiver.

The OSR2 is a hands-free machine that mounts to a standard VESA 100mm hole pattern using 4x M4 bolts. This means that an enormous range of mounting options, mostly in the form of display and monitor stands, are commercially available. There are also a couple of 3D-printable desk and chair clamp options available on my Patreon/SubscribeStar/Discord platforms.

Tools

To assemble the OSR2 you will need Allen keys and/or a screwdriver. You may also find a craft knife and/or a pair of pliers useful.

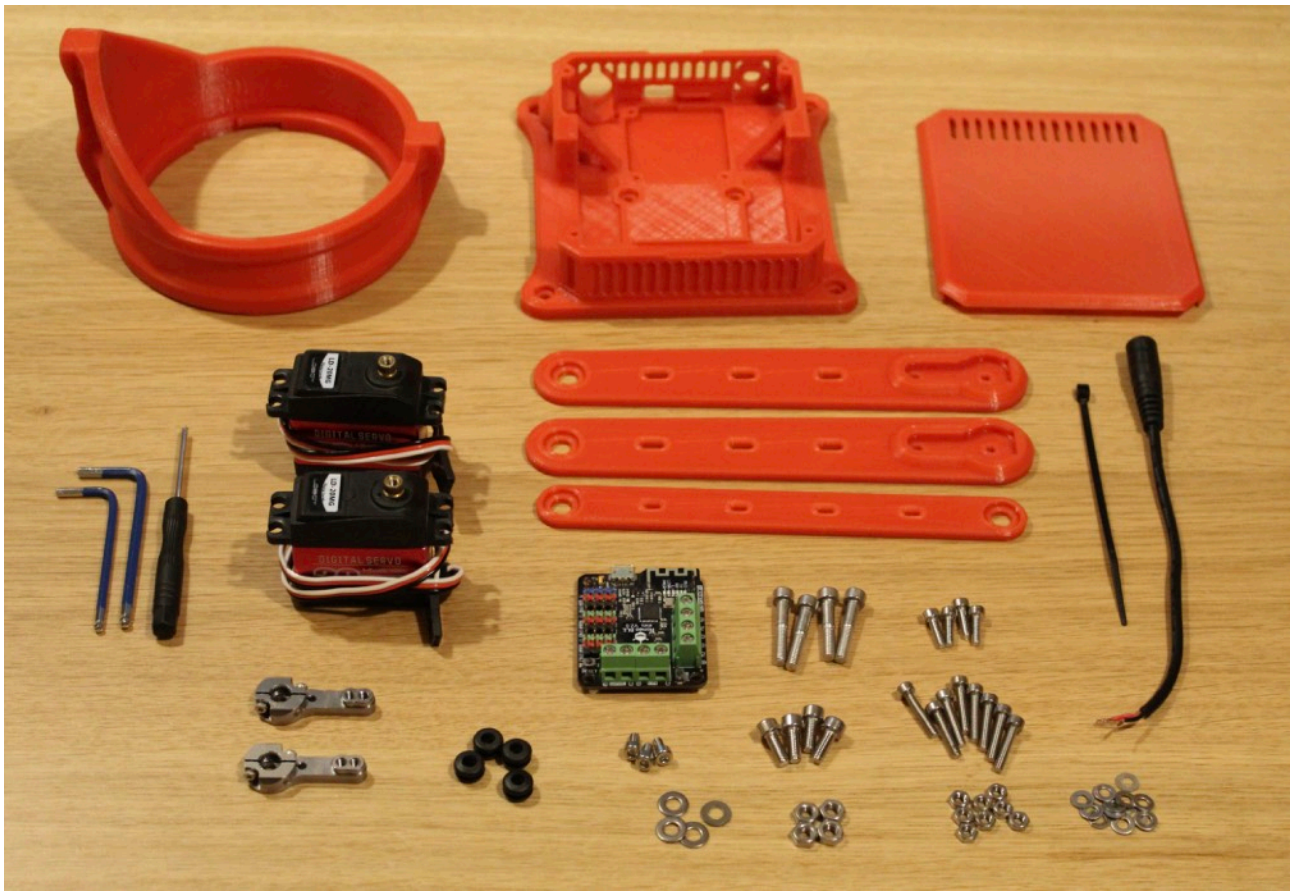
You may want to secure some parts (eg M4 nuts) with a tiny dab of superglue, but this is optional. I also recommend having a tub of Vaseline for use as a lubricant for the gears in the T-wist.

If you are using M4x30 bolts for the flexing joints you will need to use a hacksaw, and a file on the cut end. Be careful, steel can be sharp!

The external power power bus upgrade does require a soldering iron and some very basic electronics skills, but this is optional. Everything else on the OSR2 is plug-and-play.

You will need a copy of the Arduino IDE on your computer. This can be downloaded for free from www.arduino.cc.

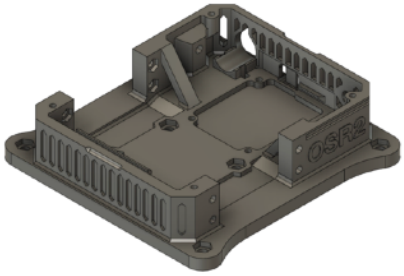
OSR2 Assembly

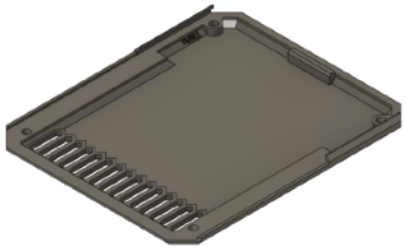
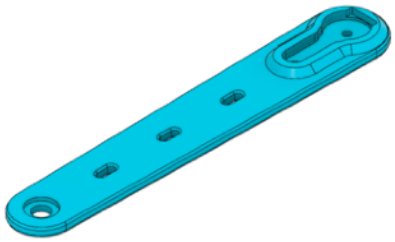
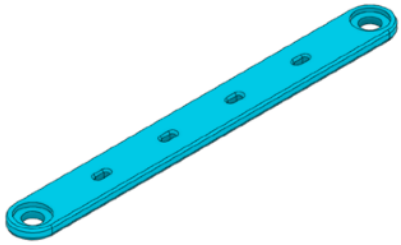


3D printed parts list

The 3D printed parts are supplied in .STL format; you can print them yourself or order them from an online 3D printing service. Unless otherwise specified the STL files are provided in the intended build orientation and are designed to be printed without supports.

The off-the-shelf parts should be easily available through ebay, amazon, local hardware or hobby shops, etc.

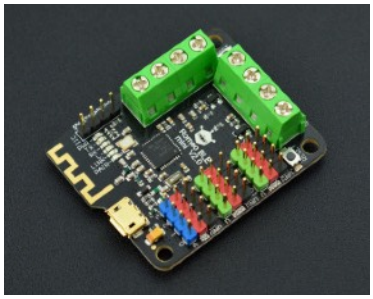
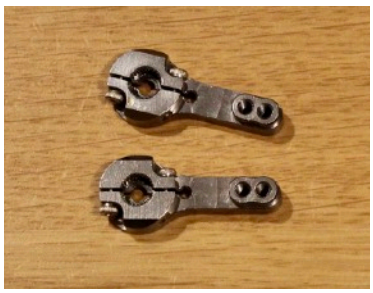
Part	Qty	Description
OSR2.1 Base 	1x	The main enclosure for the electronics and servos. It is fitted with 8x mounting holes shaped to contain M4 nuts. These are configured to interface with a 100mm VESA mount, or an AMPS pattern mount.

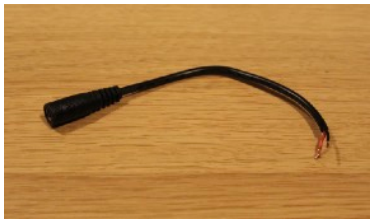
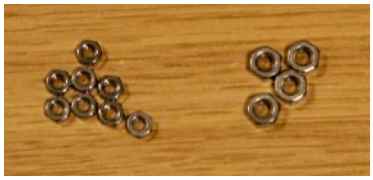
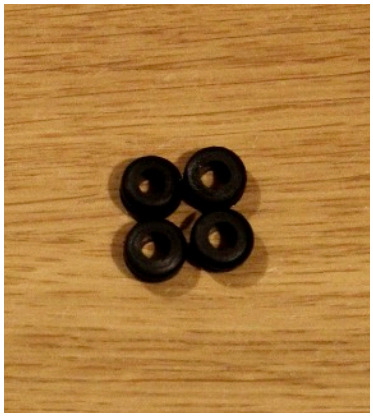
OSR2.1 Lid 	1x	The cover that clips into place over the main enclosure.
OSR2.1 Receiver 	1x	The piece that is manipulated by the arms and holds your toy. It is currently configured to carry a full sized Fleshlight.
OSR2.1 Arm 	2x	The main arms that are driven by the servos. They are designed to fit around Futaba 25T M3 metal servo horns.
OSR2.1 Support Arm 	1x	The third arm that keeps the receiver upright.

Off-the-shelf parts list

These are the off-the-shelf components that you will need to assemble a working OSR2.

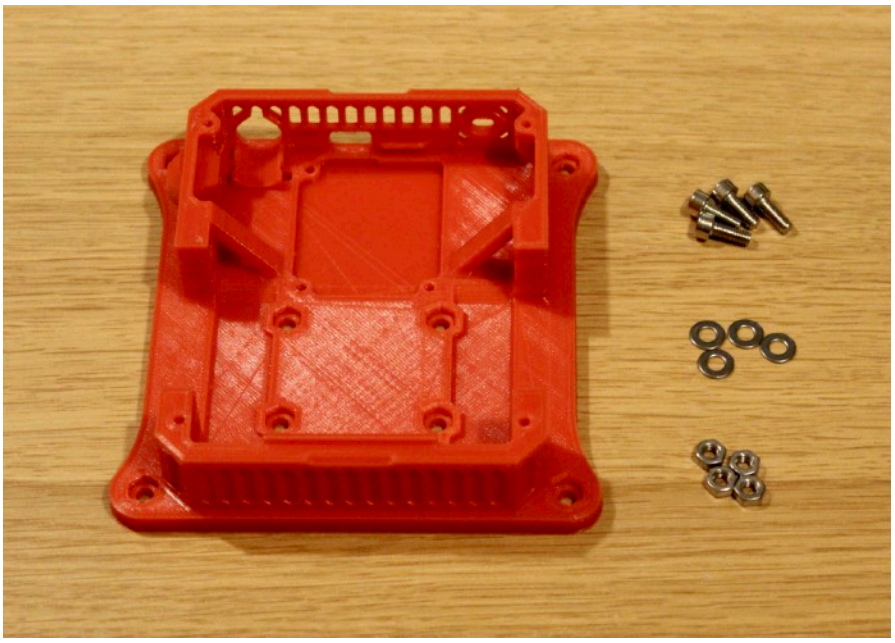
For the hardware parts such as bolts you might be able to find most or all of them from your local hardware store, but I have found ebay to be the most reliable source for this sort of thing.

	Part	Qty	Description
	Romeo BLE Mini v2 or v1.1	1x	An Arduino based microcontroller that can read incoming T-Code commands and interpret them. The OSR2 is designed around the Romeo BLE mini. This is because it is compact and offers a plug-and-play capability. See notes on
	Micro USB cable	1x	To connect to your computer.
	Power supply	1x	A power supply capable of providing 3A at 5V or 6V. Most generic power supplies come with a choice of end connectors, which is ideal. The one pictured is a Powseed Universal Power adaptor, available from Amazon.
	Standard size servo (20kg.cm or more)	2x	The driving force behind your OSR2. Standard size servos have a body size of 20x40mm. There is an incredibly wide range range of this kind of servo available on the market, so you can make your own tradeoff between performance and cost.
	Futaba 25T M3 metal servo horn	2x	The interface between the servos and the 3d printed arms. A lot of 20kg.cm servos come with these as standard, but some don't. They are very common and can be purchased separately if needed.

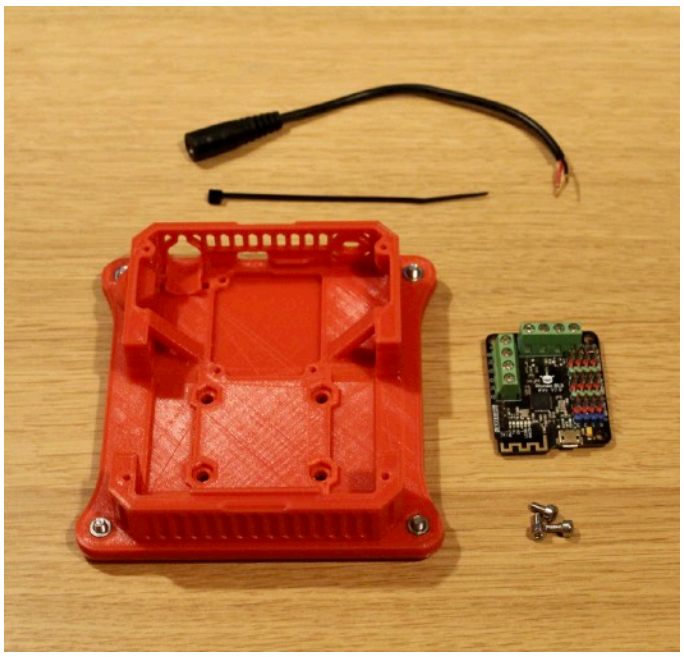
	5.5x2.1mm female barrel jack connector	1x	A female connector with a set of trailing leads. The exact model is not important as long as it is compatible with your power supply.
	Cable tie	1x	Just a normal cable tie. Used to secure the barrel jack in position
	Bolts: M3x6mm M3x10mm M3x16mm M4x10mm M4x30mm*	 3x 4x 8x 4x 4x	These are metric threaded bolts. I recommend hexagon socket cap head bolts because they are easy to work with and there are no sharp edges, but any type is fine. *The M4x30mm bolts form the joints between the arms and the receiver. These bolts have 10mm of unthreaded shaft, which makes for a fantastic smooth joint. To take advantage of this however you will need use a hacksaw to cut them down to 20mm long. If you do not want to bother with this you can substitute M4x20mm bolts.
	Nuts: M3 M4	 8x 4x*	Metric threaded nuts *Use 8x M4 nuts if you want to use both the VESA 100 and the AMPS mounting holes.
	Washers: M3 M4	 12x 4x	Metric washers, Form A.
	6mm (or 1/4”) wiring grommet	4x	These are rubber rings normally used to protect wiring as it passes through holes in metal plates. However, it also turns out that they are fantastic as a simple universal joint for a stroker robot arm! To avoid confusion: 6mm is the Outer Diameter (OD). This should correspond to an Inner Diameter (ID) of about 4mm. The best source I have found for these is ebay, where you should be able to find a pack of 25 for about $3.

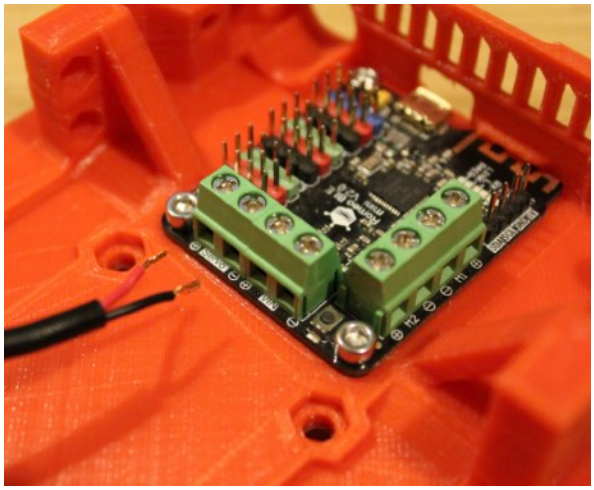
Assembly steps

1) Insert 4x (or 8x) M4 nuts into the mounting holes you intend to use on the base part. You can secure them in place with a dab of superglue if you wish. Then insert 4x M4x10mm bolts, with M4 washers into the holes from the back. These are the bolts that will be used to attach the OSR2 to your mount of choice.



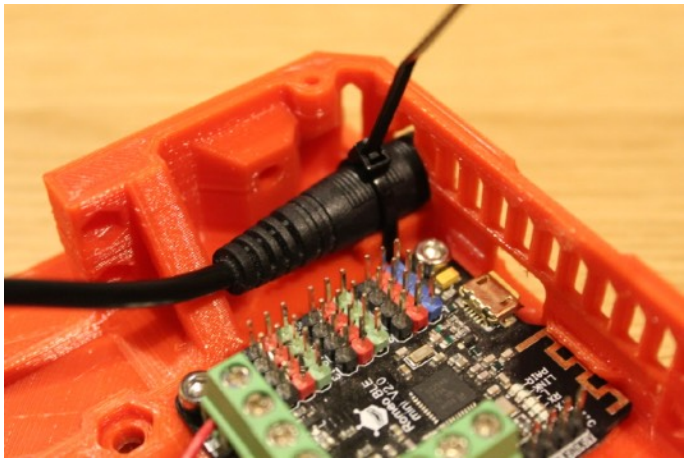
2) Install the Romeo BLE mini into the slot at the top of the base so that the micro USB port is aligned with the hole. Then secure it in place with 3x M3x6mm bolts, screwed into the plastic.



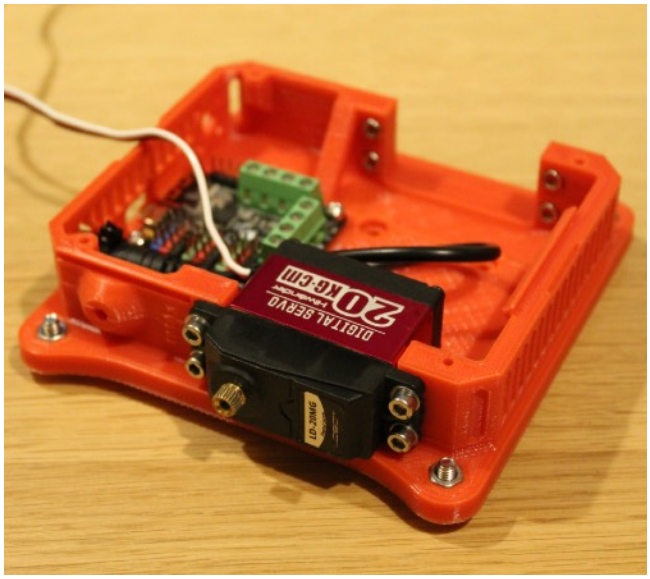


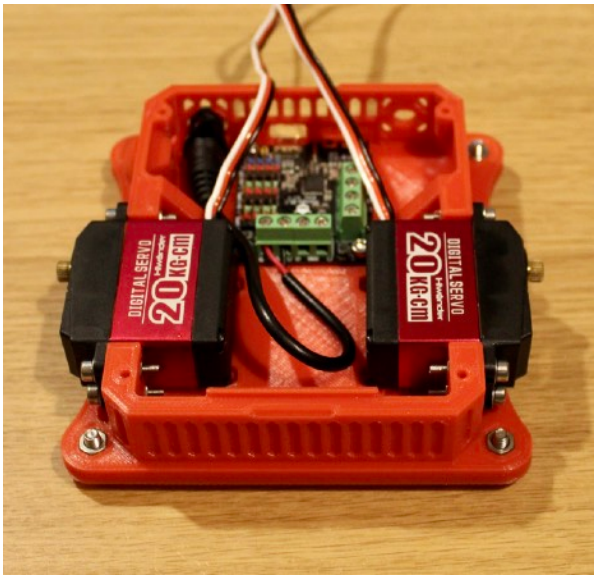
3) Install the trailing leads from the female barrel jack into the “Servo” screw connectors on the Romeo. Ensure metal-to-metal contact between the wire and the terminals and that the red wire goes to “+”, black goes to “-“.

4) Place the female barrel jack into the holder to the left of the Romeo and secure it with a cable tie.

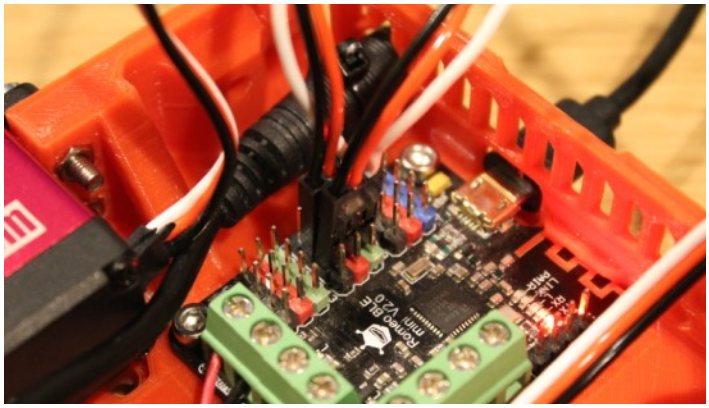


5) Install the 2x servos into the base. Ensure that the servo leads protrude from the end nearest the Romeo, then secure each servo with 4x M3x12 bolts with M3 washers screwed into 4x M3 nuts inserted into hex features on the inside of the base wall.

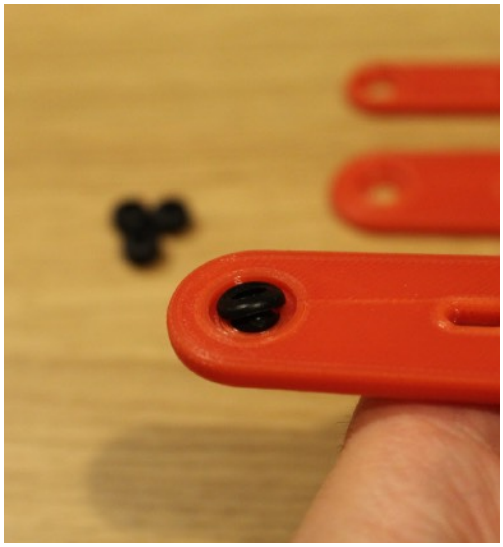


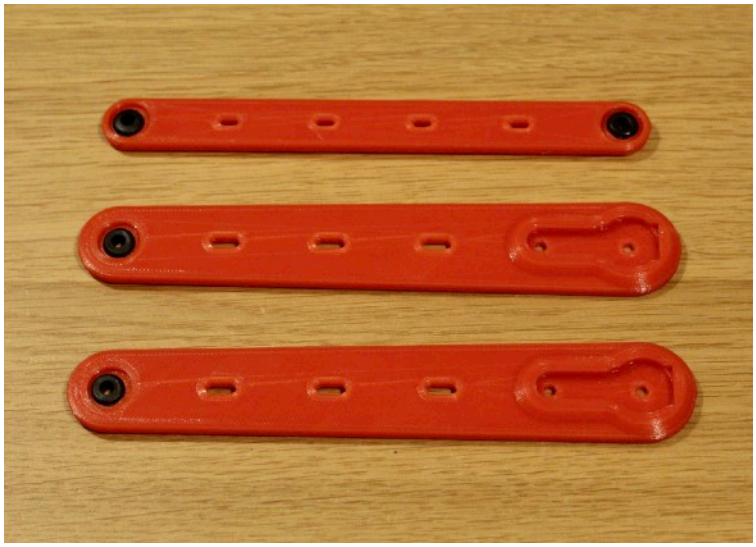


6) Plug the servos into the Romeo. Plug the left servo into the green-red-black pins marked “8”, the right servo next to it into the pins marked “3” (as of *OSR-Release2_7.ino*). Match Red to Red and Black to Black, and that will put the data wire (whatever colour it is for your servo) on the correct pin.



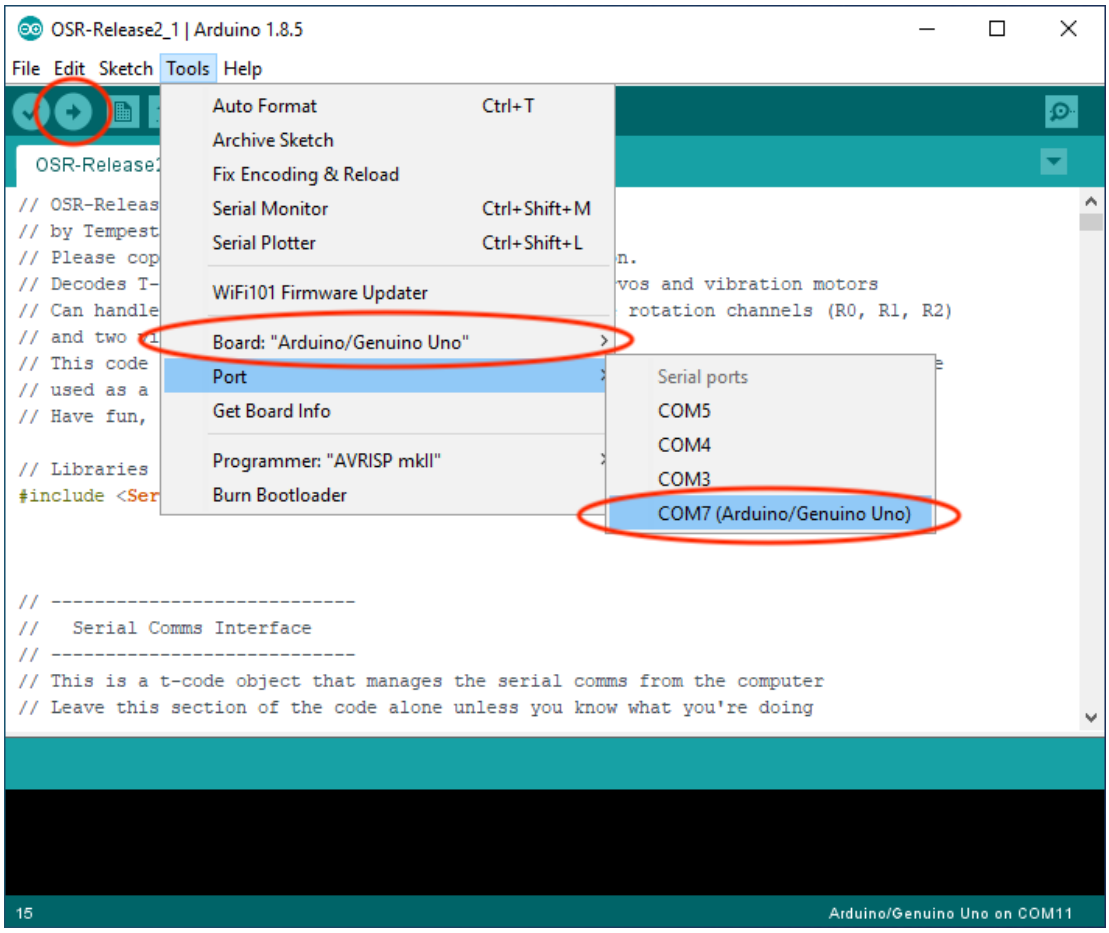
7) Install the 4x 6mm wiring grommets into the 4x 7mm diameter holes in the 2x arms and the support arm. There should be one on each arm, and one on each end of the support arm.





8) Plug the Romeo into a USB port on your computer via the micro USB cable. Connect a power supply to the power jack on the OSR2. Make sure that the power supply is set to a voltage that the servos are rated to; 5V is safe for most servos.

9) Open the Arduino IDE program and load the OSR2 sketch (.ino file format). From the Tools menu check that the Board is set to “Arduino Uno” (exact text can vary depending on your OS), and check that the Port selected is the one that is connected to the Romeo. Then click the upload arrow in the top left.



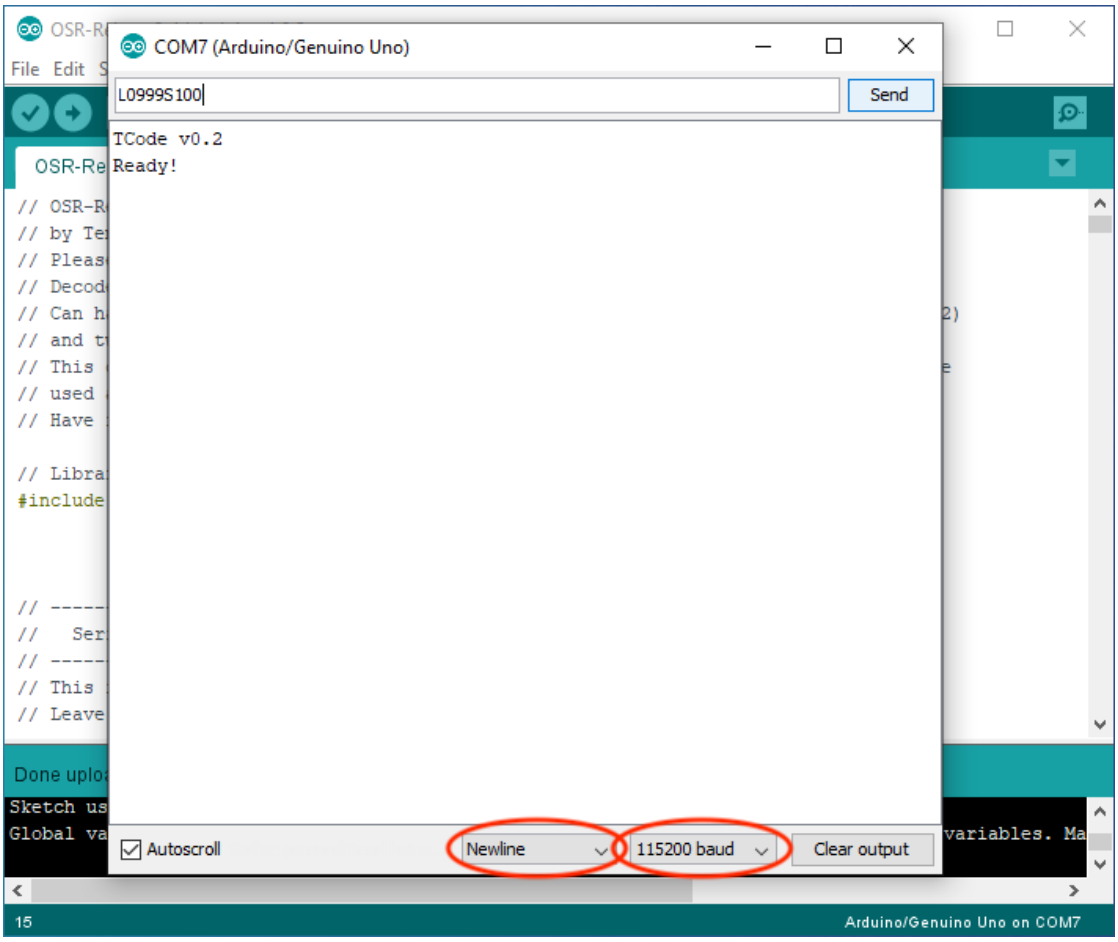
You should see lights flashing on the Romeo as the upload occurs and after about 5 seconds you should see a message saying “Done uploading”.

If the upload process is unsuccessful the first time try the upload button again. Ensure you you have the Board and Port set correctly. Disconnecting servos during upload sometimes resolves any issues.

If the upload is successful you should observe the servos moving to a set position.

10) In the Arduino IDE Open the Serial Monitor from the Tools menu. In the new window that opens ensure that “Newline” is set and “115200 baud” is selected. This should result in two messages beginning with “TCode” appearing in the window. This tells you that the sketch has been uploaded successfully to the Romeo, and that it is talking to your computer.

If you do not see anything, or the message is garbled, try unplugging the Romeo and closing the Serial Monitor, plug the Romeo back in and re-open the Serial Monitor.



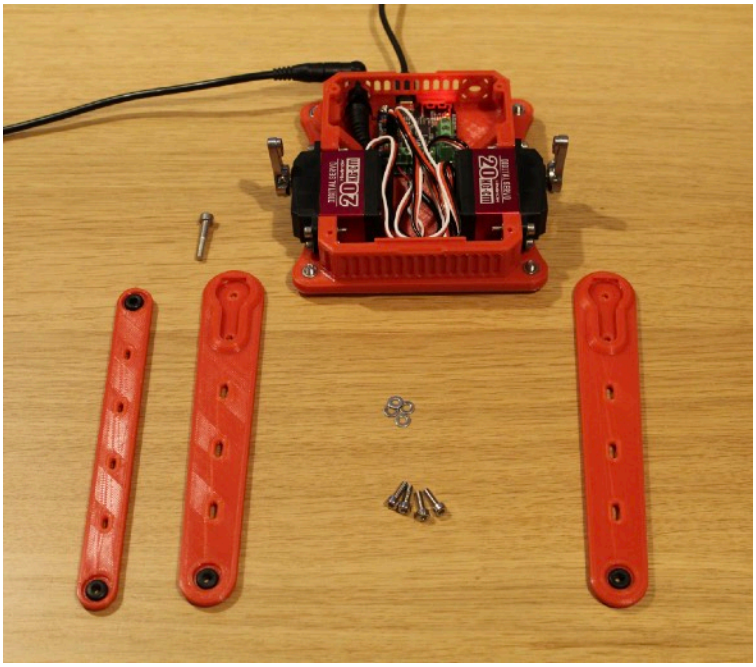
Try entering some T-Code commands into the text box and hitting “Send”:

- “L0999” should move both servos up.
- “L0000” should move both servos down
- “R1000” should move the servos in opposite directions
- “R1999” should move the servos in opposite directions the other way
- “L0999S100” should move both servos up at a slow speed.

Feel free to try different numbers. See the TCode document for a full description of how TCode works.

If the servos are moving up when they should be moving down, you have them backwards. Switch the connector positions on the Romeo, matching Red/Red and Black/Black as before.

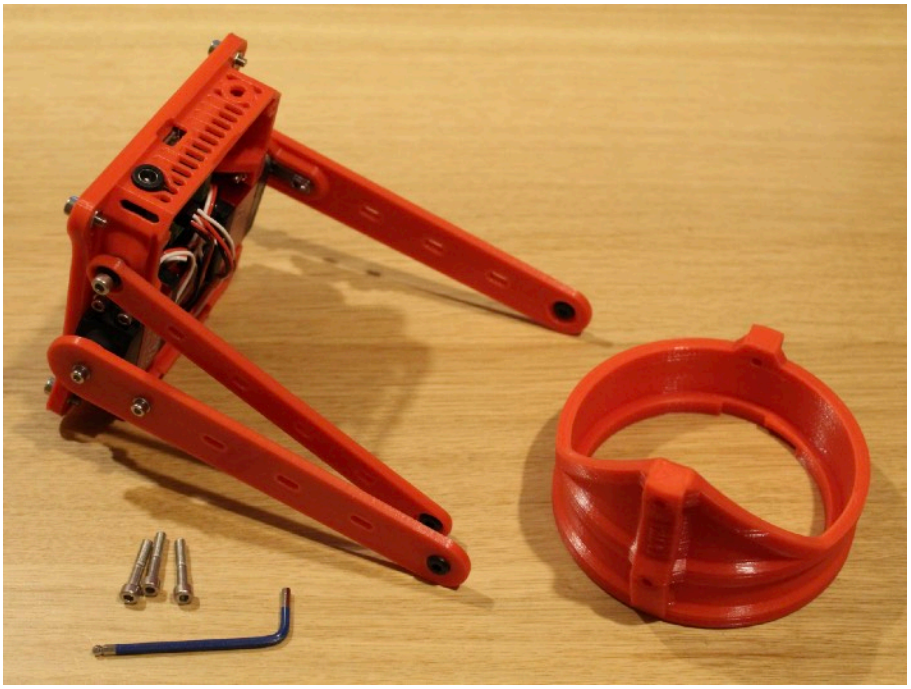
11) Close the Arduino IDE and unplug the Romeo from the USB cable. Wait a few seconds, then reconnect it. The servos should automatically return to a neutral position. With the servos in this position place the Futaba 25T M3 servo horns onto the servos so that they are at right angles to the base. Tighten the tiny screws on the servo horns in this position.



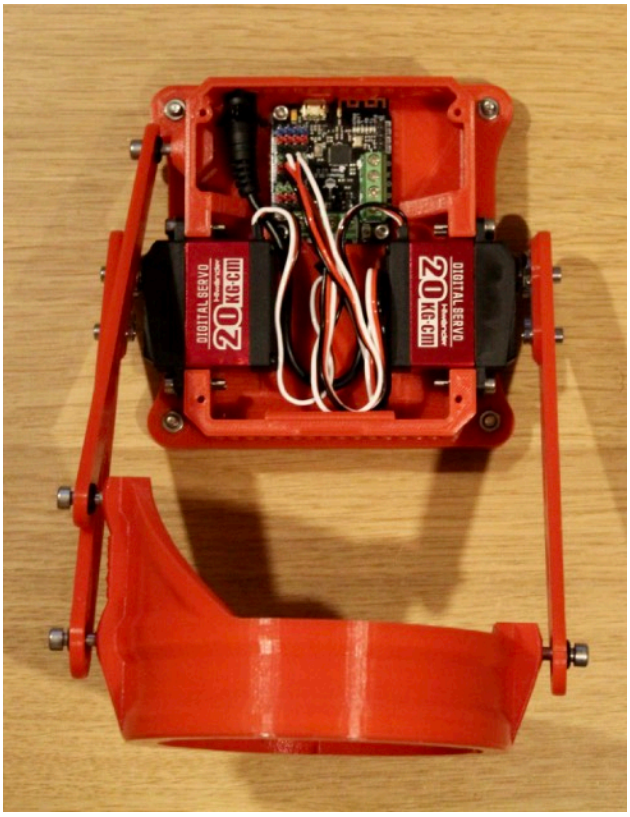
12) Install the 2x arms onto the the servo horns and secure each with 2x M3x10 bolts and M3 washers.

13) Install the support arm onto the mounting point on the left side of the base using an M4 bolt screwed directly into the plastic.

This bolt can either be an M4x20 bolt or, for better results, use an M4x30 bolt cut down to about 22mm shaft length (ie like an M4x22). The latter should leave a 10mm section of unthreaded shaft protruding through the grommet.



14) Install the receiver onto the end of the three arms using the 3x remaining M4 bolts. These should form the same joints as described above.



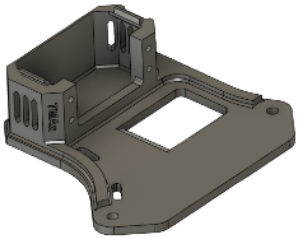
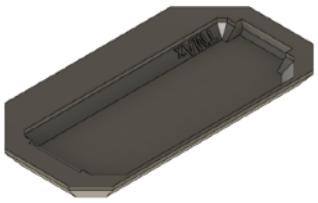
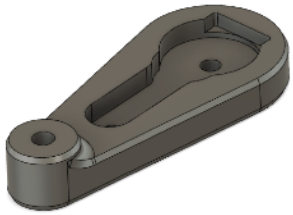
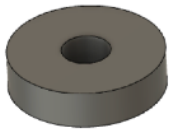
15) Collect up the servo wires so that they are inside the base enclosure and install the lid. The lid should clip securely into place on the top of the base.

The OSR2 is now ready for use!

Pitch Module (OSR2+) Assembly

The OSR2+ is an OSR2 with a simple bolt-on modification called a pitcher that adds a second rotation axis by employing a 3rd servo.

Pitch module parts list

Part	Qty	Description
OSR2+ Pitcher 	1x	This fits between the OSR2 base and the VESA mount to hold the 3rd, pitcher servo. It should be fitted on the left side of the base.
OSR2+ Pitcher Lid 	1x	The cover that clips into place over the pitcher enclosure.
OSR2+ Pitcher Arm 	1x	This arm mounts onto the pitcher servo's servo horn and attaches to the support arm.
OSR2+ Spacer 	2x	These sit between the OSR2 base and the VESA mount on the two bolts that don't go through the Pitcher.
Bolts, Washers & Nuts 	2x M3x10 Bolt 4x M3x12 Bolt 6x M3 Washer 4x M3 Nut	Used to assemble the servo into the pitcher, and the pitcher arm onto the servo horn.

Pitch module assembly steps

1) Install the servo into the pitcher part and secure it with 4x M3x12 bolts and M3 washers. These screw into 4x M3 nuts inserted into hex features on the inside of the pitcher housing.

2) Wrap the servo lead through the holes in the pitcher plate so that it's out of the way.

3) Clip the lid onto the top of the pitcher enclosure.

4) Plug the servo into pin 9 on the Romeo

5) Power up and reset the OSR2. This will drive the pitcher servo into the neutral position.

6) Now install the servo horn and pitcher arm with M3x10 bolts and M3 washers. The arm should be parallel with the pitcher plate when the servo is in the neutral position.

7) Power down the OSR2 and unplug the pitcher servo.

8) On the OSR2 remove the M4 bolt holding the support (3rd) arm to the base.

9) Turn the bolt around and attach the support arm to the pitcher arm. The bolt should be screwed into the plastic far enough so that the pitcher arm is free to move forward and back without colliding when the pitcher is in position.

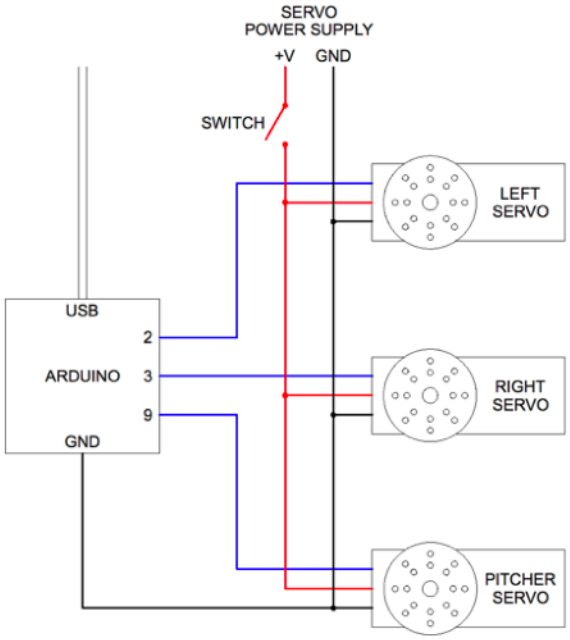
10) Mount the pitcher to the OSR2 by sandwiching it between the OSR2 base and the VESA mount using the 4x M4 mounting bolts. Two of the bolts should pass through the pitcher; the other two bolts should pass through spacers.

11) Thread the pitcher servo cable into OSR2 enclosure and plug in to pin 9 on the Romeo. The OSR2.1 enclosure has a hole in the lower left corner that can be opened for this purpose.

Power Bus

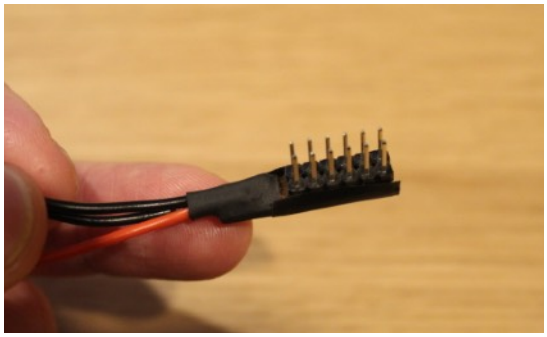
In this section I will show you how to make a power bus. This is useful if you want to avoid putting too much power through the Romeo or if you want to use another type of Arduino. It also allows you to add a power switch so that you can turn the power to the servos on and off at will, which is a useful feature.

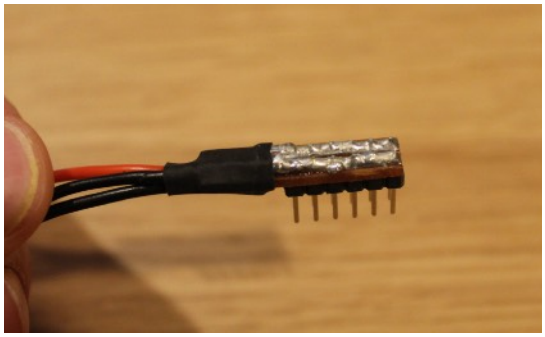
Servos essentially require three connections: power, ground, and signal. With this in mind, the wiring diagram for the OSR2+ is shown below.



Note that the servo power is completely independent of the Arduino. Also note, however, that because the servo signal wires connect to the Arduino, it is necessary to connect the servo ground wires to ground on the Arduino in order to complete the circuit.

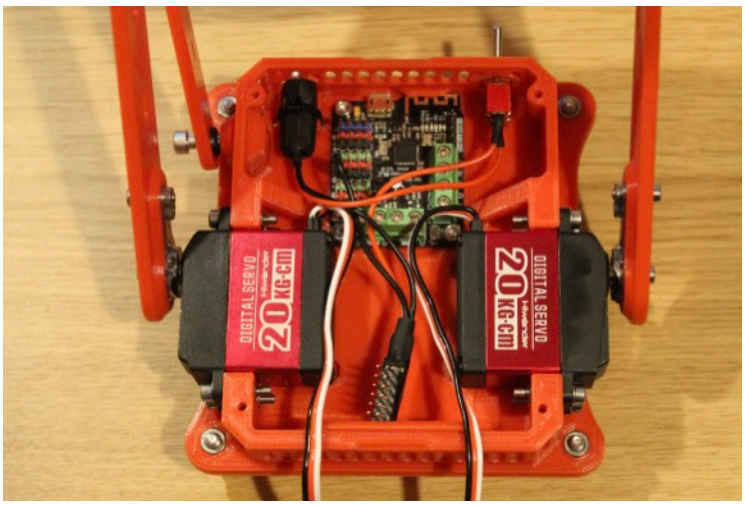
The advantage to using the Romeo is that it's plug-and-play; all of the wiring is built into the board. It is, however, very easy to assemble an alternative to the Romeo's servo pins using a piece of copper stripboard, some 0.1" header pins, and multi-stranded jumper wire. The jumper wire you use should be rated for the power rating on your power supply.



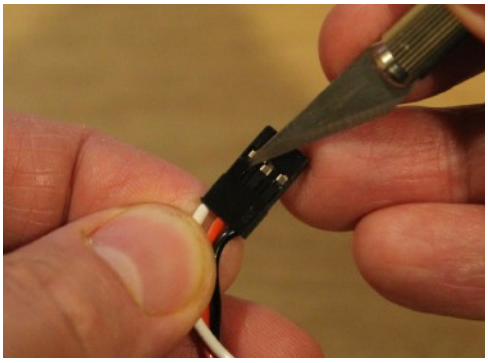


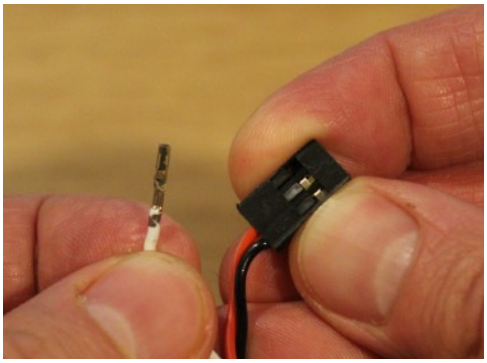
Cut out a piece of stripboard that is two strips wide, so that one can be used for positive and the other for negative. The header pins can be soldered into the board in two rows, effectively providing power rails that the servos can be plugged into. The jumper wires to connect the power bus to the Arduino and to the power supply should also be soldered into the positive and negative power rails.

If you want to add a power switch it should be located on the power wire between the jack plug and the power bus. There is a hole for it in the top right corner of the OSR2 base.

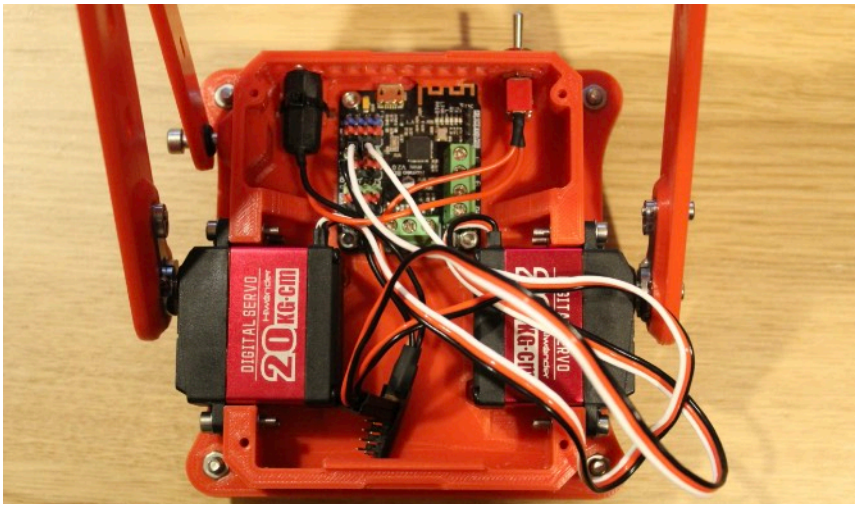


The signal cables need to be plugged in to the appropriate pins on the Romeo. To do this you can remove them from the connector housings by using a craft knife or similar tool to lever up the plastic clip that holds them in place. I recommend wrapping a piece of electrical tape around the bare metal connector fitting so that it does not foul on adjacent connectors.





The servo connector, now containing only the power and ground cables, can be plugged into the power bus. Be sure to plug these in the right way around! Also make sure that the exposed metal pins on the bottom of the bus are not shorting on anything metal, especially if your servos have metal bases. Again, you can use tape to prevent metal-to-metal contact.



If you would like more details I have now produced several tutorial videos for this modification, which are available on my Patreon page.

T-wist Module Assembly

The T-wist is a modification for the OSR2 that adds an additional movement axis. The new axis is 180° or 270° rotation (±90° or ±135°) of the fleshlight case around the long axis.

T-WIST 4

The twist movement is achieved by means of a rotating ring mount inside the receiver, which is driven by a standard sized servo. This is made possible by the use of a simple gearbox that transfers the servo movement to the ring with a 1:1 ratio.

Previous versions of the T-wist have used a continuous rotation servo and depended on some kind of feedback system. Versions 1 and 2 used a potentiometer located at the top of the fleshlight case in a T-valve housing; version 3 used a specific servo type, the Parallax 360, which has a feedback wire. Version 4 therefore represents a welcome simplification!

T-WIST 2

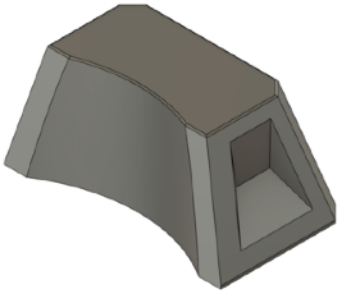
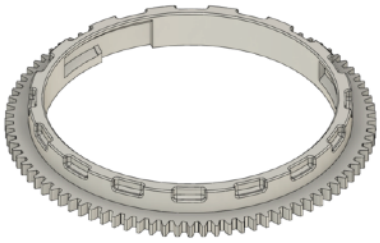
T-WIST 3

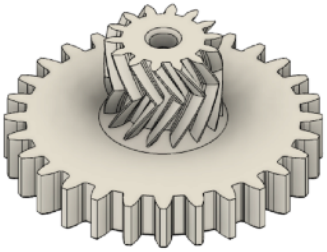
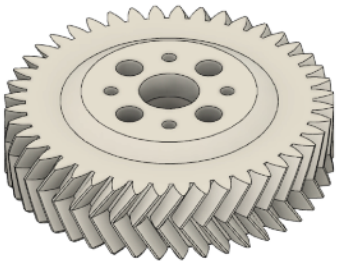
T-wist parts list

This is a complete list of the parts that comprise the T-wist.

The 3D printed parts are supplied in .STL format; you can print them yourself or order them from an online 3D printing service. Unless otherwise specified the STL files are provided in the intended build orientation and are designed to be printed without supports.

The off-the-shelf parts should be easily available through ebay, amazon, local hardware or hobby shops, etc.

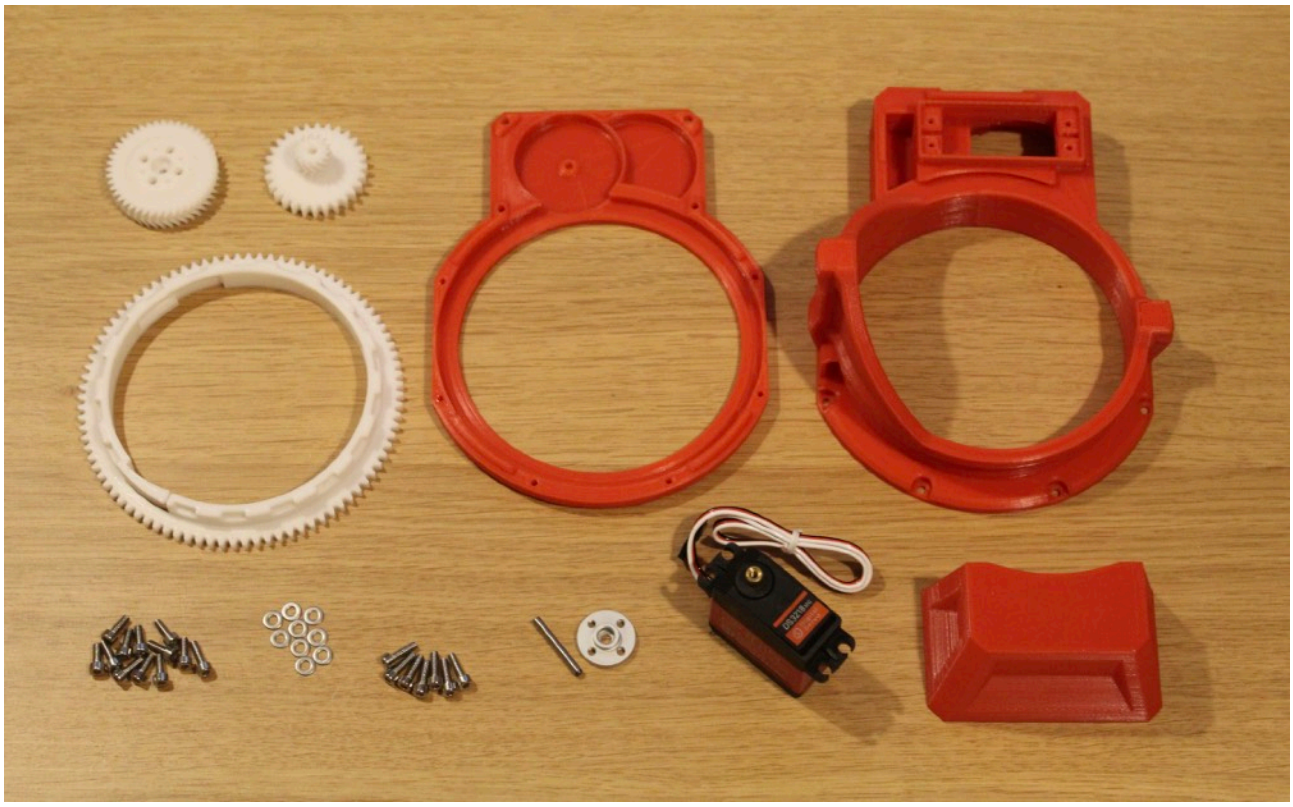
Part	Qty	Description
T-wist4 OSR2 Receiver Base 	1x	This is the base of the receiver. The corners of the gear enclosure are supported during the print by small break-off tabs.
T-wist4 OSR2 Receiver Body 	1x	This is the main receiver body with attachment points that match the standard OSR2 receiver. There is a break-off support under the gear enclosure that ensures that the transfer gear axle hole forms correctly.
T-wist4 Receiver Lid 	1x	This is the clip-on lid that sits over the servo on the receiver body.
T-wist Ring Gear 	1x	This is the rotating mount for the fleshlight case. This part is backwards compatible, ie it is interchangeable with the clip ring from the T-wist 3.

T-wist 4 Transfer Gear 	1x	This is the intermediate gear that transfers power from the drive gear to the ring gear I advise printing this part in high resolution for the best performance.
T-wist 4 Drive Gear 	1x	This is the gear that is mounted on the servo horn. It has 4x counterbored mounting holes for M3 bolts to mount to a metal servo horn, and 4x holes for M2 bolts for mounting to a plastic servo horn. I advise printing this part in high resolution for the best performance.
Servo 	1x	The T-wist 4 works with a standard sized servo. I recommend that you use a 270° servo for best range of movement, but a 180° servo will also work. A 20kg.cm servo has been pictured, however this is likely overkill and a servo in the 5-10kg.cm torque range is probably more than sufficient. I look forward to the community feedback on this!
Circular Metal 25T Servo Horn 	1x	I recommend that you use a metal servo horn. These will almost certainly not come with your servo and will have to be sourced separately. If you can't get hold of a circular metal servo horn you can substitute the circular plastic one that often comes with standard sized servo.
M3x25 Dowel Pin 	1x	This is a metal pin that is used as the axle for the transfer gear. An M3x25 dowel pin (3mm diameter, 25mm length) is a standard part that should be easy to get hold of. If you have trouble sourcing one a length of 3mm metal rod or tube cut to length will work just fine.

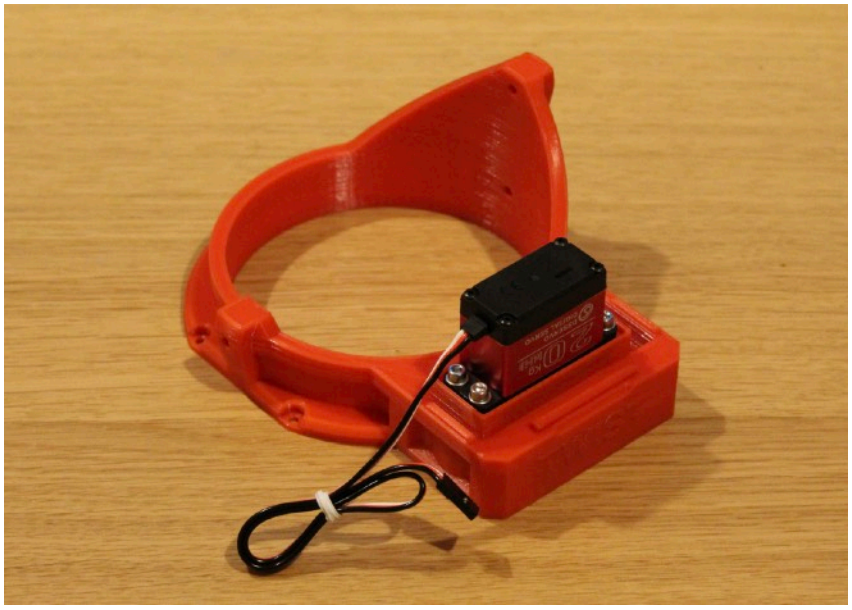
Bolts, washers, etc 	7x M3x8 Bolt 12x M3x10 Bolt 9x M3 Washer *4x M2x6 Bolt *4x M2 Washer *1x M3x16 Bolt	The two parts of the receiver are held together using (6x) M3x8 and (4x) M3x10 bolts, which screw directly into the plastic. Likewise the servo is held in place with (4x) M3x10 bolts, with M3 washers. 4x M3x10 bolts with M3 washers mount the metal servo horn to the drive gear. An M3x8 bolt and washer mounts the horn to the servo. *If you are using a plastic servo horn you will instead need (4x) M2x6 bolts and washers, and a single M3x16 to attach the drive gear to the servo.
Servo lead extension 	1x	This is to connect the servo back to the OSR2 body via the arms/links. The optimal length of extension from the T-wist back to the main enclosure is 600mm
Spiral cable binding & cable ties 	-	These are used to tidy up the cables once you have everything working.

You will also want your OSR2 to hand so that you can check that the mechanism is working.

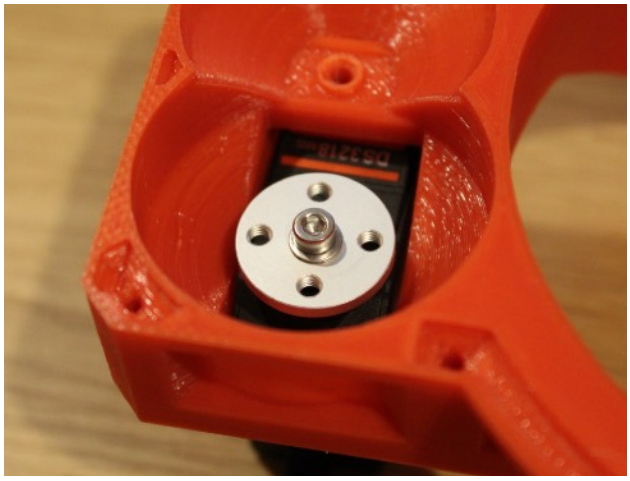
T-wist assembly steps

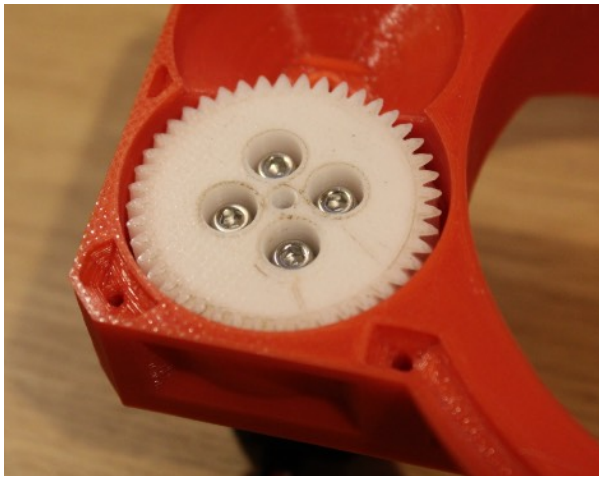


1) Mount the servo into the receiver body and attach it using 4x M3x10 bolts and M3 washers. Do not fully tighten the bolts at this stage: allow some freedom of movement to the servo.

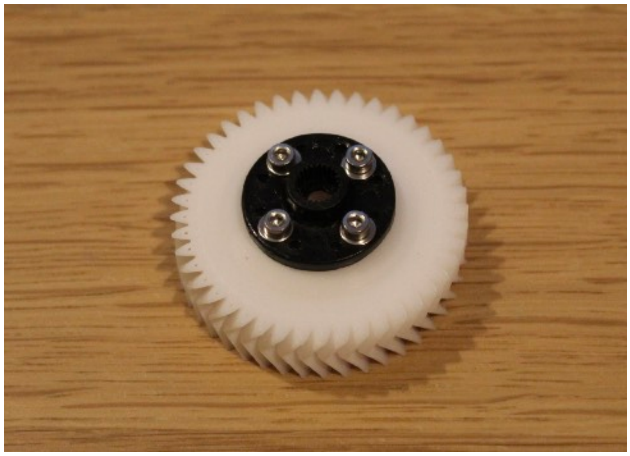


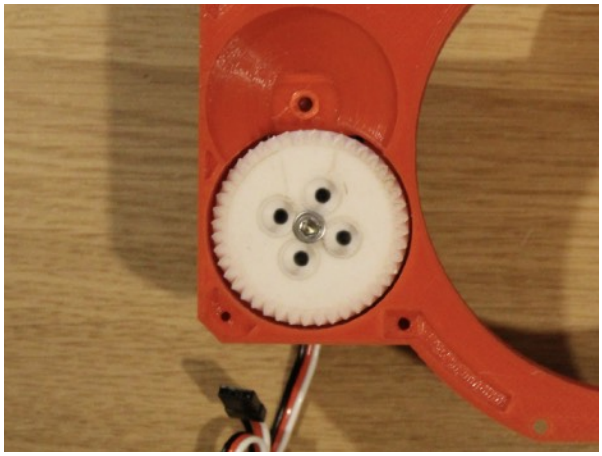
2) If you are using a metal servo horn (recommended), place it onto the servo and secure it in place with an M3x8 bolt and M3 washer. Then install the drive gear onto the servo horn using 4x M3x10 bolts and M3 washers.





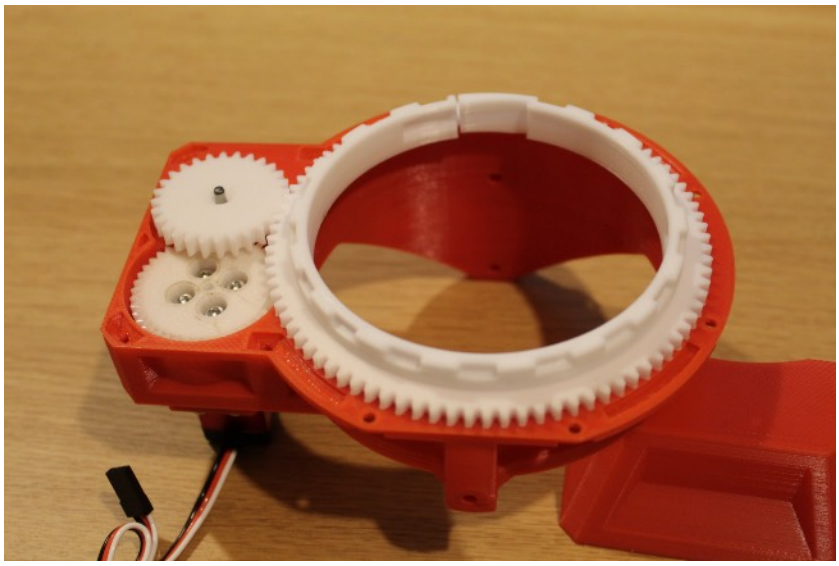
2A) If you are using a plastic servo horn mount the servo horn to the drive gear first using 4x M2x6 bolts and M2 washers. You may need to enlarge the holes in the plastic servo horn to do this. Next install the horn and gear onto the servo using the M3x16 bolt and an M3 washer.





3) Install the transfer gear into the receiver body and insert the M3x25 dowel through the gear so that it sits in the hole on the receiver body.

4) Place the ring gear on the receiver so that it intermeshes with the transfer gear.



5) Place the receiver base onto the receiver body, making sure that the M3x25 dowel sits in the appropriate hole in the receiver base. Temporarily secure the receiver base in place using 2x M3x10 bolts.

6) Check that the mechanism is working correctly by turning the ring gear by hand. You should be able to feel and hear the servo rotating in train with the gears as you do this.

7) Move the servo into a position where you can feel the drive gear smoothly engaging with the transfer gear and tighten the 4x M3x10 bolts to fully secure the servo in place.

8) At this point an optional but highly recommended step is to remove the base and liberally coat all of the mechanism contact surfaces with vaseline. This makes an enormous difference to how smoothly (and quietly!) the mechanism will run.

9) Close up the two halves of the receiver fully using 6x M3x8 bolts and 4x M3x10 bolts, screwing directly into the plastic. This completes the assembly of the T-wist mechanism.

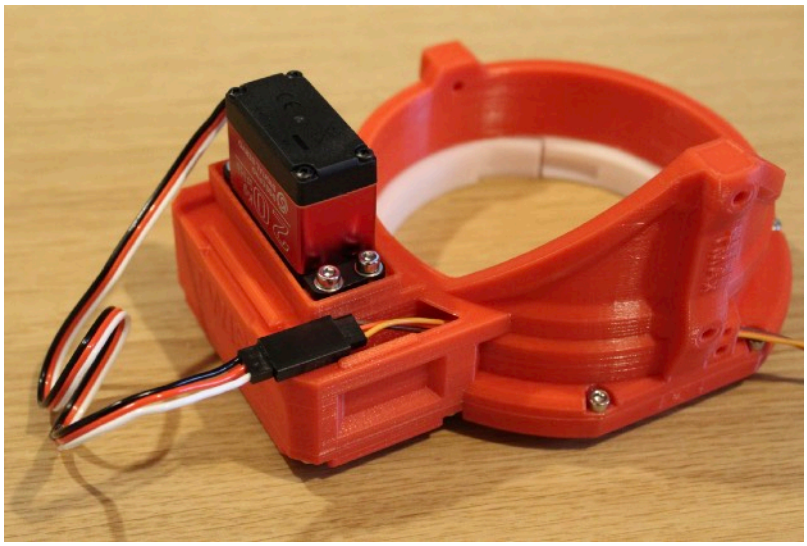
10) Make sure your OSR2's Romeo BLE mini is running *OSR-Release3_4.ino* or later. (*SR6-Alpha4_ESP32.ino* or later if you're using an ESP32)

11) Hook up the servo to pin 10 on the Romeo in the OSR2 (*D27 on the ESP32*).

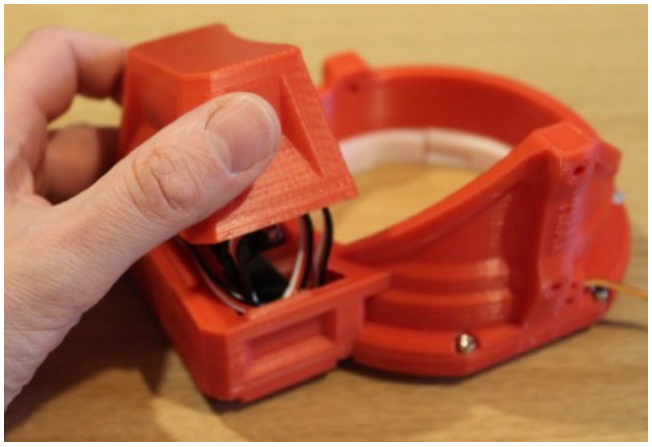
12) When you power on the OSR2 the T-wist should now respond to R0 T-Code commands. You can test this by using issuing these commands using something like the Arduino Serial monitor (as detailed in the OSR2 build instructions) or the MOSA web app. https://trymosa.netlify.app/.

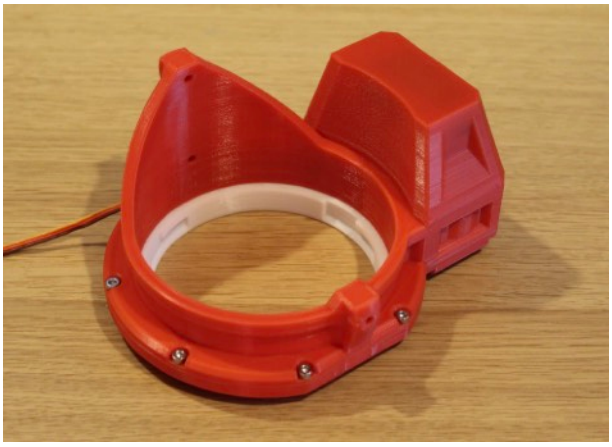
With the T-wist functioning you can now install the receiver onto the OSR2 in place of the old receiver.

13) Feed the servo cable extension down the conduit on the side of the receiver and connect it up to the servo.



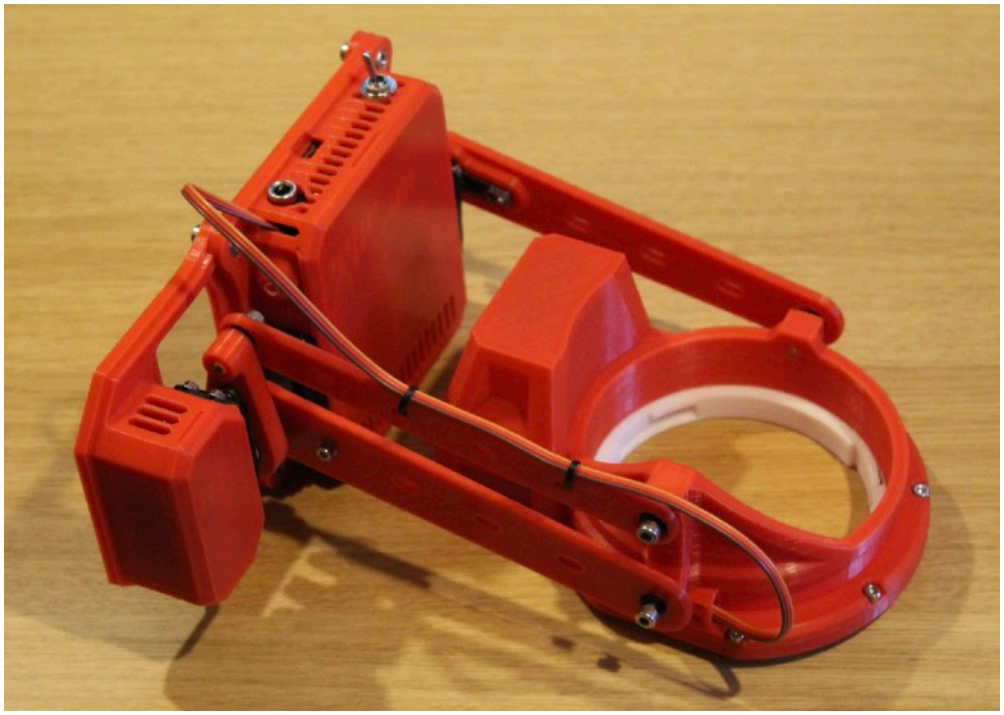
14) collect up the spare cable and tuck it inside the lid, which you can then clip down.





15) Install the twist receiver onto your OSR2 in place of the original receiver.

16) Use cable ties and spiral cable binding to route the cable back to the OSR2 enclosure. Use gentle curves to avoid putting excessive strain on the cable during use.

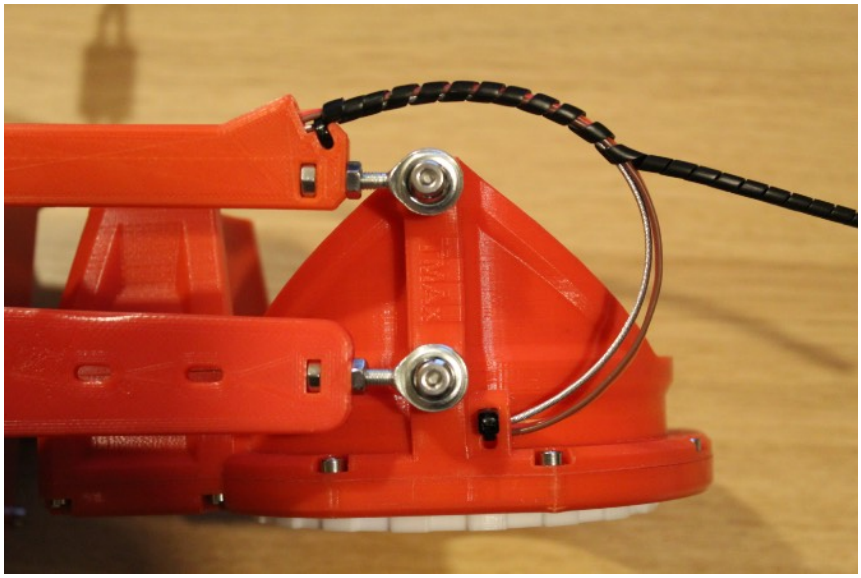


This completes the T-wist assembly process.

T-wist cable routing

A frequently occurring problem with previous versions of the T-wist was wires going to the servo on the receiver becoming damaged by the movement of the device. This was compounded by the usual point of failure being where the wires met the servo itself, with the specialised parallax servo being particularly expensive and difficult to get hold of.

This new version of the twist receiver addresses this problem in two ways. First, it uses a generic servo and the servo wires themselves do not see any bending. It's just the servo extension cable that bends and this component is a lot less expensive to replace.

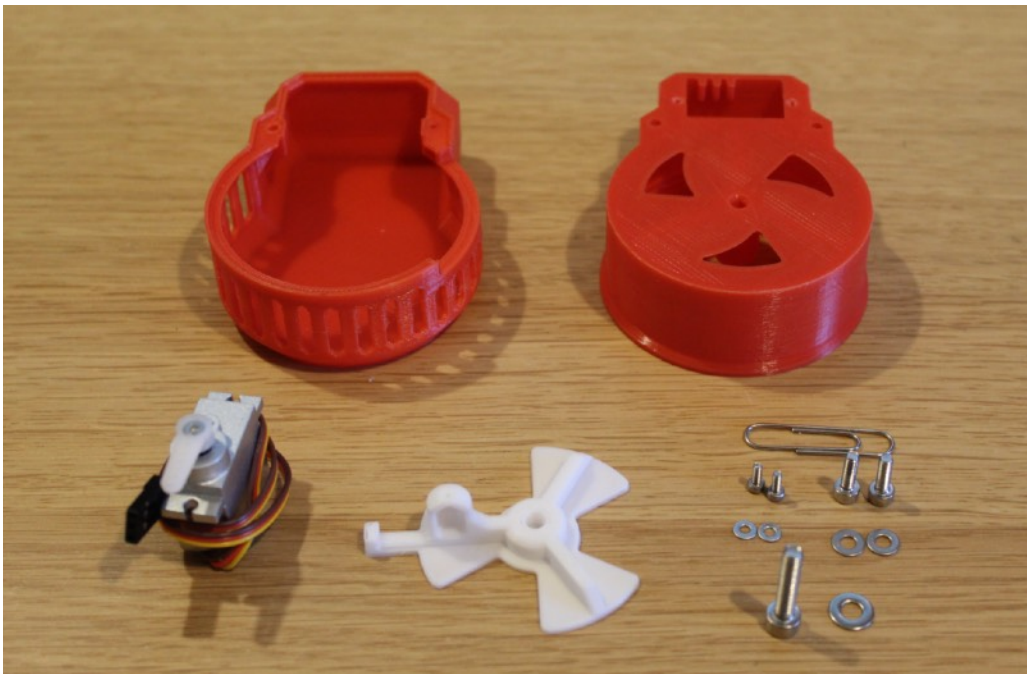


Secondly, the new receiver routes the extension cable out of the front of the receiver. This is so that the bend radius for the cable is nice and gentle and therefore in theory it should not see any harsh localised bending. Added to this the best way I have found to protect the extension is to use spiral cable binding to wrap it up with a tougher piece of wire, for example a length of bicycle brake cable.

T-Valve

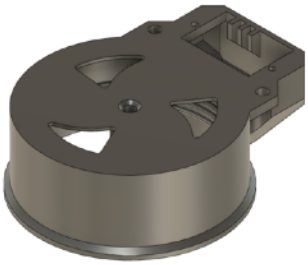
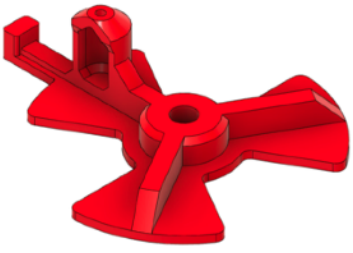
The T-Valve is an automatic valve that controls the level of suction inside a standard fleshlight case. It screws on top like a regular cap, but it uses a 9g micro-servo to constantly adjust the air resistance in and out of the case during stroking. It means your T-Code device can change the level of suction from stroke to stroke!

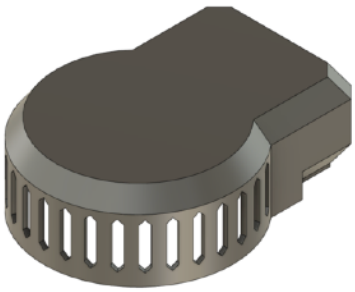
T-valve parts list



The 3D printed parts are supplied in .STL format; you can print them yourself or order them from an online 3D printing service.

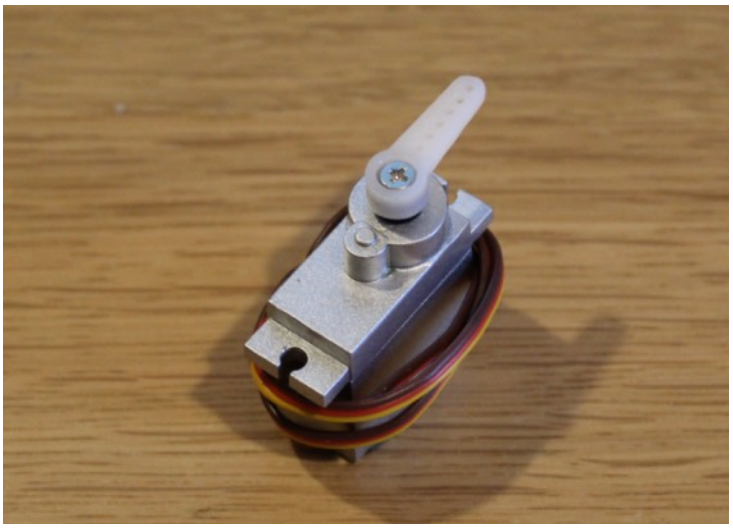
The off-the-shelf parts should be easily available through ebay, amazon, local hardware or hobby shops, etc.

Part	Qty	Description
T-Valve Base 	1x	This is the main body of the T-valve and is designed to screw onto a standard fleshlight case in place of the usual screw cap.
T-Valve Blades 	1x	This is the moving part of the T-Valve, used to control the aperture size. It includes a lever that indicates the valve position and allows the angle of the blades to be set manually when the servo is not powered.

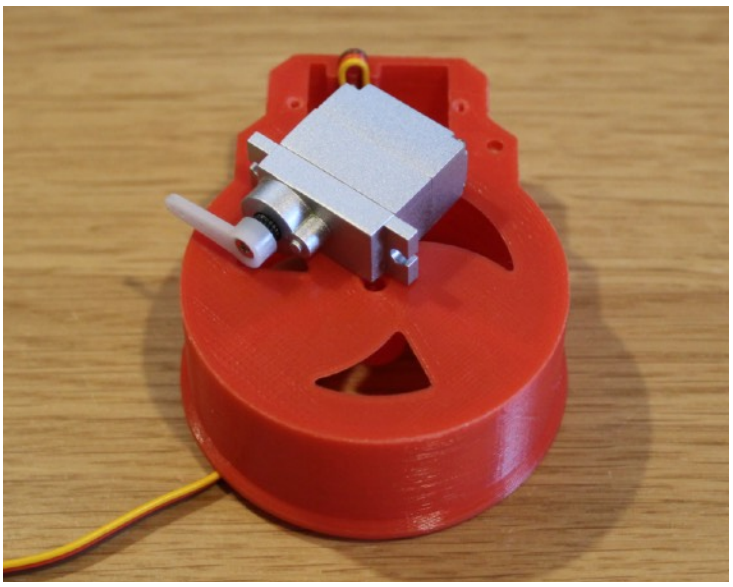
T-Valve Lid 	1x	This part is an optional lid that sits on the top of the T-Valve to conceal and protect the working parts.
Micro (9g) Servo 	1x	Very little force is needed to move the T-Valve blades, so a small servo is required. For example a Sparkfun Small Servo (ROB-09065). (In the build images I have used a DILWE 9G RC metal servo, which is more expensive but very quiet!) In theory any “9g” micro-sized servo (base dimensions ~11.7 x 23.2mm) could be used.
Paperclip 	1x	A large, sturdy, paperclip. This is to form the drive linkage from the servo to the blades. If you want to be fancy you can use a bit of model makers’ brass rod instead.
Bolts & Washers 	1x M4x20 Bolt 1x M4 Washer 2x M3x10 Bolt 2x M3 Washer 2x M2x6 Bolt 2x M2 Washer	The fasteners you’ll need. Used to assemble the servo into the pitcher, and the pitcher arm onto the servo horn.
Servo lead extension 	1x	This will allow you to extend the short lead on the micro servo all the way to the pins on your microcontroller. For the OSR2 I recommend a length of at least 300mm.

T-valve assembly steps

1) Take the micro servo and a servo horn with a hole 12mm from the centre. Install the horn onto the servo with the horn in the 12 o'clock position with the servo centred. An easy way to do this is to temporarily plug the servo into pin 2 or 3 on an OSR2, which automatically centres the servos connected to these pins at startup.

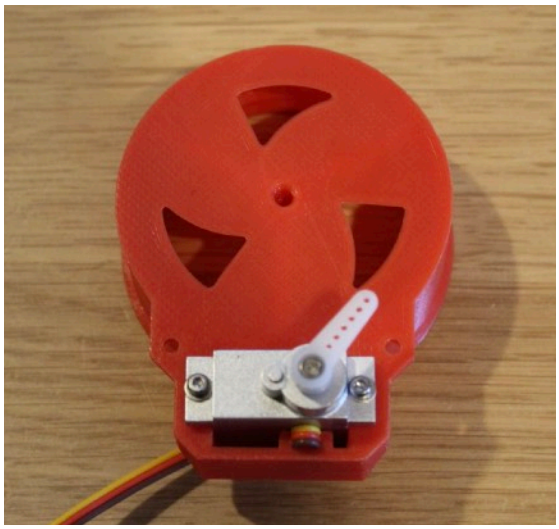


2) Take the micro servo and feed the lead down through the hole at the base of the servo enclosure on the T-Valve Base. Then place the servo on the top of the base so that you can feed the lead up and then down inside the grooves inside the enclosure.



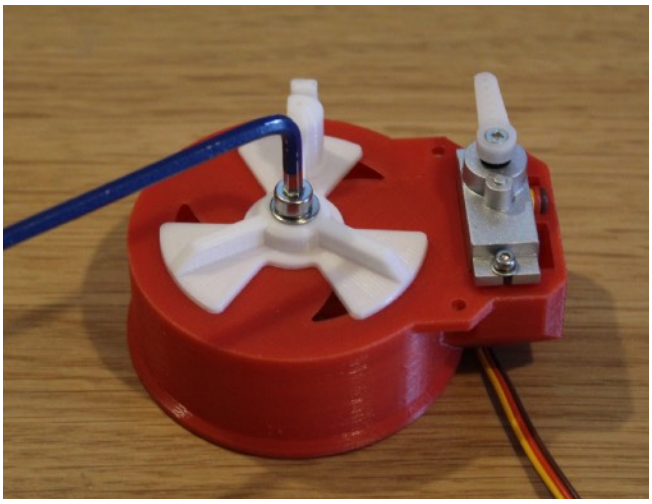
3) Gently slide the micro servo into the enclosure ensuring that the servo lead is seated comfortably in the up and down grooves. Installing the lead in this manner prevents any tension applied to the lead during use from causing damage to the electrical connections inside the servo.

4) Screw the micro servo into position with 2x M2x8 bolts with M2 washers, screwed directly into the plastic.

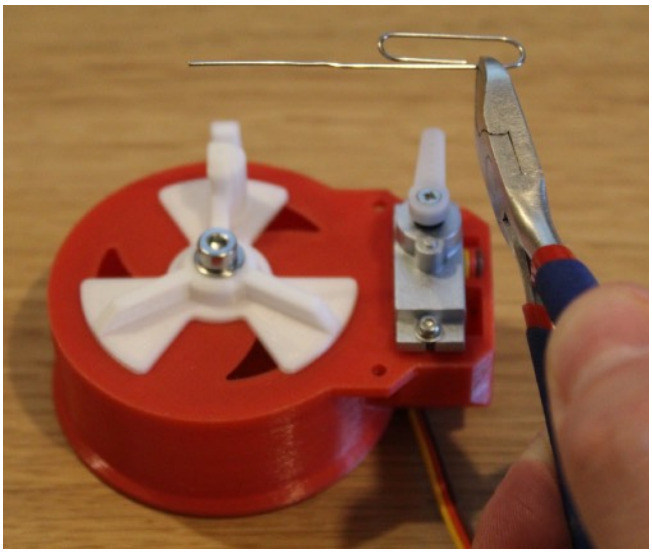


5) Place the T-Valve Blades onto the top of the base and screw them down using an M4x20 bolt with an M4 washer. Note that the washer is under the bolt head; the blades should ride directly on the top of the base.

6) Tighten the M4 bolt to the point where the blades are unable to rotate, then turn the bolt back slightly so that the blades are still held in contact with the base, but are able to turn freely.

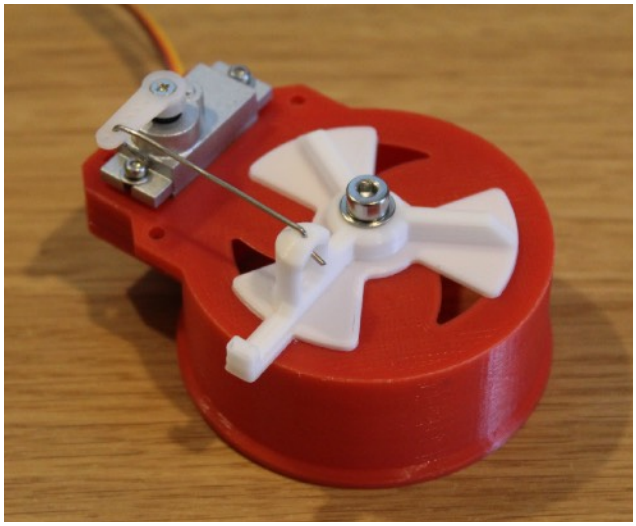


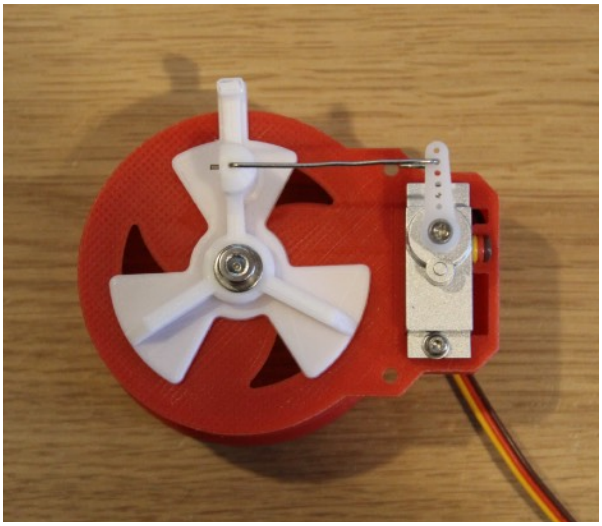
7) Rotate the blades and the servo horn so they are both in the 12 o'clock position. Straighten out the paperclip and cut off a straight piece to span the distance between the two holes, with approximately 10mm extra on each end.



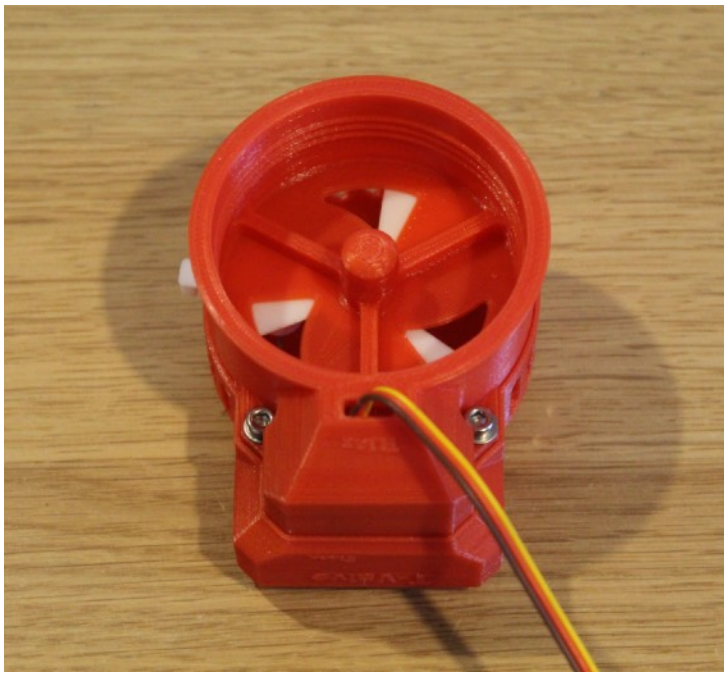
8) Bend the paperclip rod so that it can be inserted into the 12mm-off-axis hole on the servo and the drive hole on the valve blades. If the bends are the right distance apart both parts should be in the 12 o'clock position at the same time. This might require a little bit of adjustment.

9) When the rod is the right length, bend the ends over so that the rod is held captive. Note that the rod at the servo end must be bent back inward toward the valve blades. This will prevent it fouling on the lid when it is installed.

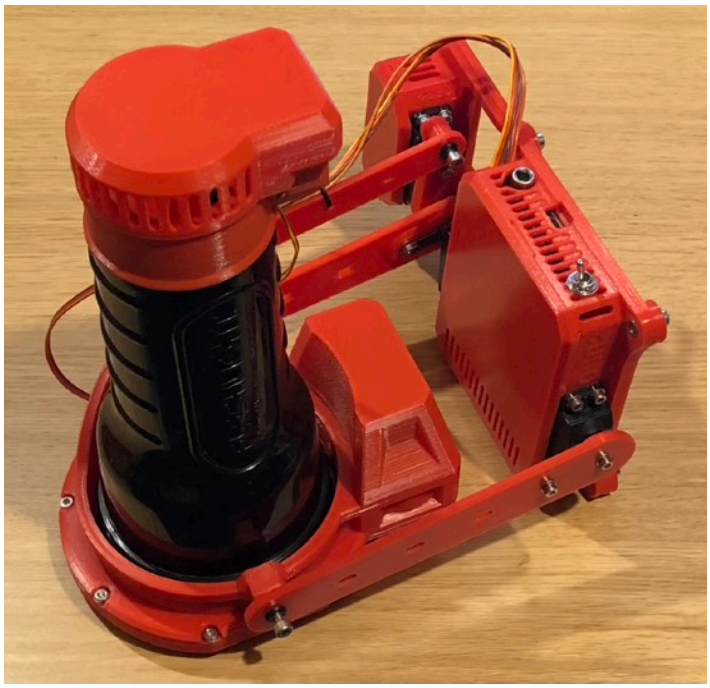




10) Install the T-Valve lid onto the top of the base, covering up the servo and blades. Fix it into place from underneath using 2x M3x10 bolts with M3 washers.



11) The T-Valve can now be installed onto the top of a standard fleshlight case. Unlike a standard cap, it should be screwed down all the way so that the valve now controls all of the airflow into and out of the case.



Using the T-Valve with the OSR2

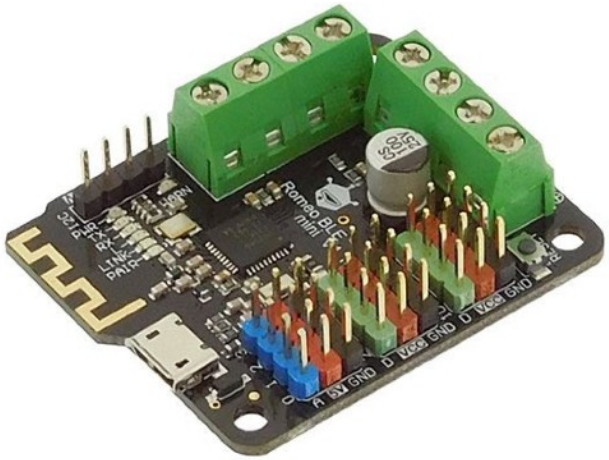
The T-Valve should be connected to digital pin 12 on the Romeo BLE Mini inside the OSR2. This requires a servo lead extension, but otherwise the micro servo should be hooked up in exactly the same way as the OSR2's other servos.

The T-Valve requires that the Romeo is running *OSR-Release2_3.ino* or later. By default the firmware drives the valve automatically, opening the valve on the down stroke but closing the valve to 50% for the up-strokes.

As of T-Code v0.3 (*OSR-Release3_0.ino* or later) the valve is now controllable via the A0 or A1 channels (0 = fully open, 999 = fully closed). A0 directly sets and fixes the valve position, whereas A1 allows a suction level to be set, with the valve moving to the commanded position, but automatically opening on down strokes. The OSR2 will adopt the mode of the channel on which the last command was sent.

The Romeo BLE mini

The Romeo BLE mini is a wonderful little board for putting together a device like the OSR2 because it offers plug-and-play, without the need for soldering or DIY electronics. it is therefore ideal for the novice homebuilder.



Remember to set the board type to “Arduino/Genuino Uno” when uploading to the board from the Arduino IDE.

v2 vs v1.1

The current version of the Romeo BLE mini is the v2, and it is this version for which the OSR Arduino sketch is written. The OSR sketch can run just fine on a Romeo BLE mini v1.1 however, with only a small modification.

You will know that you have a v1.1 because the digital pins, into which you plug the servos, will be labelled “2,**4**,**8**,**9**” as opposed to “2,**3**,**7**,**8**” on a v2. You may have discovered this because you plugged everything in as you should, but the arms aren’t moving correctly.

This is not a problem. All that you need to do is go into the Arduino Code and change the pin assignments. Look for this section in the Arduino sketch (*OSR-Release3_4.ino* at the time of writing):

```
// Pin assignments
// T-wist feedback goes on digital pin 2
#define Servo1_PIN 8   // Left Servo (change to 7 for Romeo v1.1)
#define Servo2_PIN 3   // Right Servo (change to 4 for Romeo v1.1)
#define Servo3_PIN 9   // Pitch Servo (change to 8 for Romeo v1.1)
#define Servo4_PIN 12  // Valve Servo
#define Servo5_PIN 10  // Twist Servo
#define Vibe0_PIN 5    // Vibration motor 1
#define Vibe1_PIN 6    // Vibration motor 2
```

Change these lines:

```
#define Servo1_PIN 7
#define Servo2_PIN 4
#define Servo3_PIN 8
```

Vibration channels

The Romeo BLE incorporates a dual motor controller. These are fully functional through the TCode V0 and V1 channels. You can hook up vibration motors to the M1 and M2 screw fittings, powered from the “Vin” connector, and they will respond to TCode commands.

Power Users

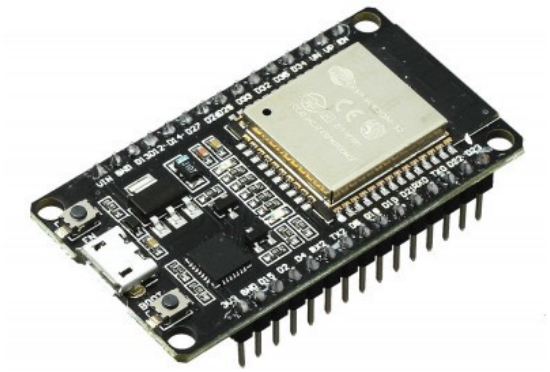
The Romeo is designed for simple hobby robots and that means that if you start pulling serious amps through the board by using high powered servos at high speeds you run the risk of burning it out. If you consider yourself to be a power user, so to speak, then I recommend that you create an external power bus for the servos so that the main power circuit does not pass through the Romeo.

Alternatives

In theory any Arduino compatible board such as a Nano or UNO could be used as an alternative to the Romeo to drive a stroker robot. If you want/need to substitute another board type I would recommend building yourself an external power bus to distribute power to the servos. Remember to set the board type appropriately, and update the pin definitions if you need to.

The ESP32 DevKit v1

The ESP32 is a microcontroller that can be used in place of the Romeo BLE mini to control the OSR2. Its main advantage is that it is substantially more powerful, which allows it to command servos at frequencies up to 330Hz, as opposed to the 50Hz offered by the Romeo. It has also found favour with a large proportion of the user base because paradoxically it is often a lot cheaper and more available than the Romeo.



The downside to the ESP32 is that it is not plug-and-play in the same way as the Romeo. You will definitely need to use an external power bus with the ESP32.

If you are using expensive, high performance servos the higher command frequency offered by the ESP32 will make a considerable difference to the smooth running of the servos, which makes it a very desirable upgrade.

The firmware is uploaded to the ESP32 from the Arduino IDE in the same way as with the Romeo. You will need to install the ESP32 add-on; you can find a useful set of instructions for this here or here. Before uploading select “DOIT ESP32 DEVKIT V1” as the board type.

In an OSR2 the ESP32 is actually used with the latest SR6 firmware. You will have to switch this over to operate in OSR2 mode by altering this line near the top of the .ino file.

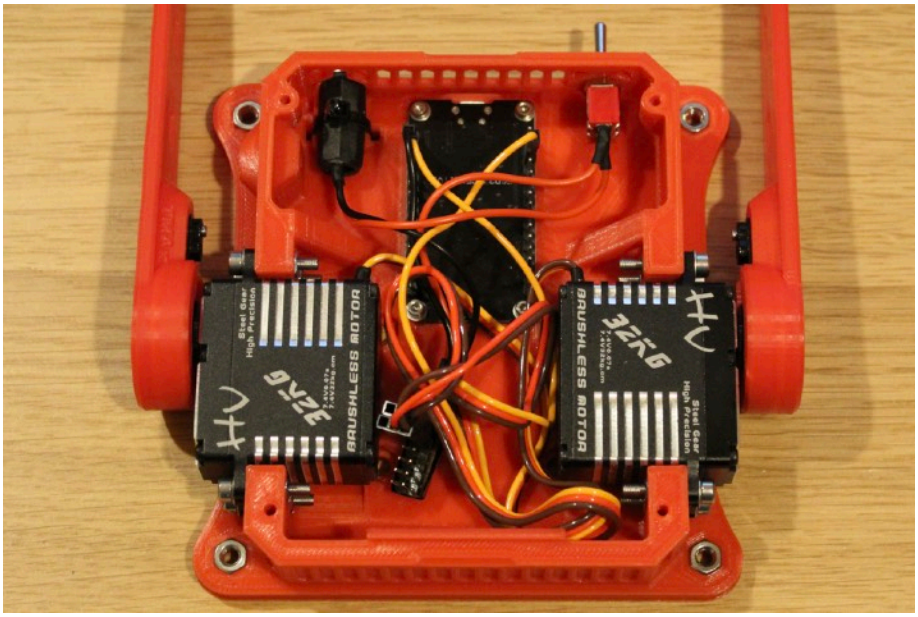
```
#define OSR2_MODE false // (true/false) Switch servo outputs to OSR2 mode
```

Change the word “false” to read “true”.

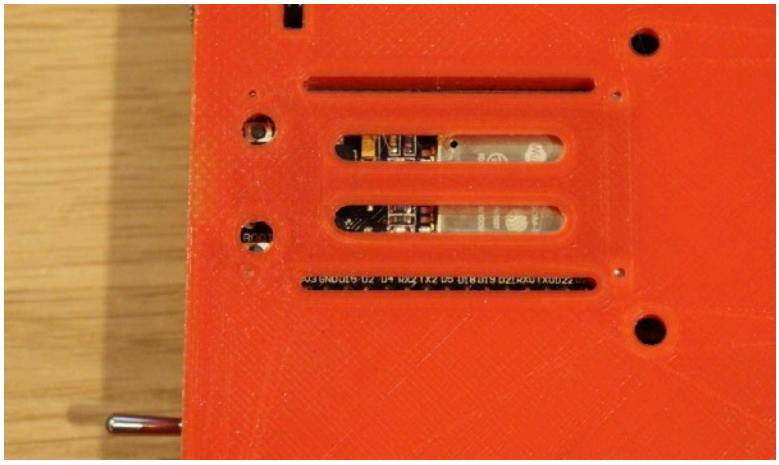
The pin assignments for the ESP32 are listed in the .ino file, and the servo signal pins are connected to the appropriate “D” pins as on the Romeo. The defaults are:

Left Servo: D15
Right Servo: D13
Pitch Servo: D4
Twist Servo: D27
Valve Servo: D25

As with the Romeo, the ESP32 must also be connected to ground on the power bus through one of the “GND” pins.



An alternate OSR2.1 base design is available to accommodate the ESP32 DevKit v1 in place of the Romeo BLE. Unlike the Romeo the ESP32 is mounted upside down, but windows have been cut into the base to make the pin numbers visible.



The ESP32 has wifi capability, though this is not at present supported by the official firmware. If you are interested in exploiting this feature I highly recommend that you look up Khrull and his alternate SR6/OSR2 firmware.

The ESP32 is mounted using 4x M2x6mm bolts, with washers, which are not included in the parts list.

Bearing Arms

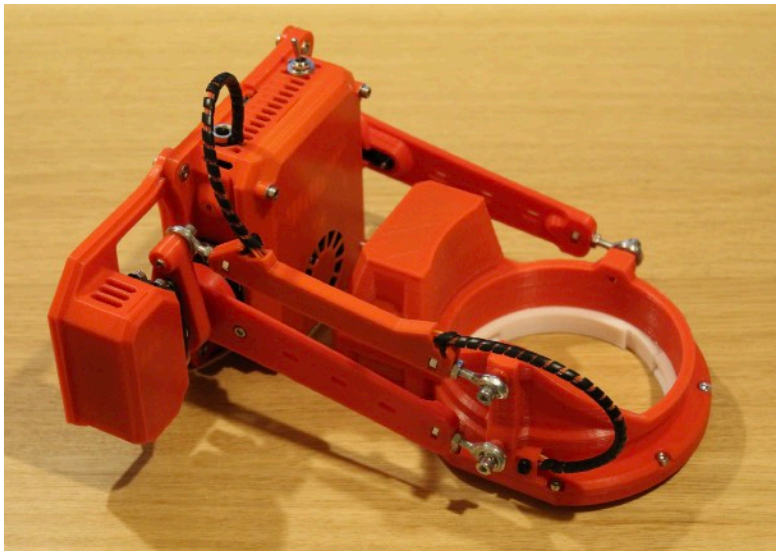
An alternate arm option is available for the OSR2, designed around rod end bearings rather than the wiring grommets used by the default design. These have the benefit that they allow a lot less play in the flexible joints. The bearing parts can however be more difficult and expensive to get hold of.

To build these arms you will need 4x 4mm rod end bearings. Sometimes called “rose joints” or “fish eye bearings”. The type used should have a threaded shaft with an M4 metric thread, and a 4mm hole. Each bearing is held in place using 2x M4 nut, meaning 8x nuts in total will be needed.

To install a rod end bearing first insert an M4 nut into the slot on the end of the arm or link. Next screw a nut onto the bearing shaft. The bearing shaft can now be inserted into the hole on the end of the arm or link and screwed down into the desired position. Finally, to secure the bearing in place, the nut on the bearing shaft can be tightened against the arm or link using a spanner (wrench) or a pair of pliers.

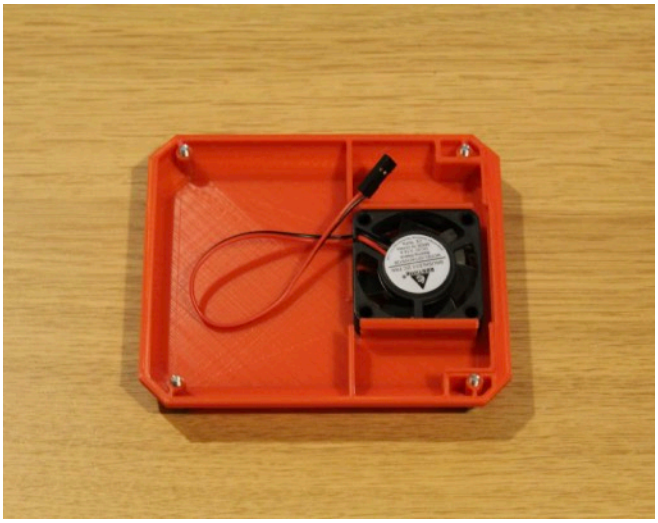
To allow full freedom of movement for the 3x bearings that attach to the receiver must be attached via a 3D printed 4x3mm spacer. These sit between the receiver and the link with the narrow end pointing toward the bearing.

This release comes with a bearing arm design and two bearing link designs: a standard design, and one with a conduit for routing cables.



Fan Lid

The OSR2.1 has been designed with an alternate lid option, which can hold a 40mm 5v PC cooling fan. This kind of active cooling is probably unnecessary for most users, but this option exists as an upgrade if you want to increase the airflow through your OSR2.



The fan lid is held in place by 4x M3x16 bolts. For power I recommend plugging it in to one of the +5v power pins next to the Romeo's analog (blue) pins. This will draw the small amount of power needed to drive the fan from the USB 5v supply that powers the Romeo, rather than the servo power.

If you are using an ESP32 you can plug the fan into the "Vin" and adjacent "GND" pins. This will power the fan from the USB connection, meaning it will continue to run even if the servos are powered off.

Servos

The question of which servos are the best to buy to use with an OSR2 is hotly debated amongst OSR2 users. There are people who like the solidity and nuance of a PowerHD LW-20MG, many who swear by the speed and power of a JX BLS-HV7132MG, and others who are willing to shell out for a set of high quality Hitec D954SW servos.

The advantage to the OSR2 is that it can take any combination of any type of standard size type servos you care to choose, and of course you can always upgrade later.

The best advice I can give to you when it comes to choosing your servos is that in my opinion you should avoid buying cheap 20kg.cm servos. It may seem really tempting to only have to spend $12 each on a set of red generic servos from China, especially when you have to buy three of them. The experience from the community however is that these servos have a nasty habit of overheating and dying with very little use. Spending a little bit more will generally guarantee better quality and probably save you money in the long run.

Your servos may or may not come packaged with a Futaba 25T metal servo horn, so be sure to check when you buy them. If they do not, then you will have to buy them separately. They are widely available through outlets like eBay or amazon for a few dollars each in a wide range of colours.

The Tempest Discord Server

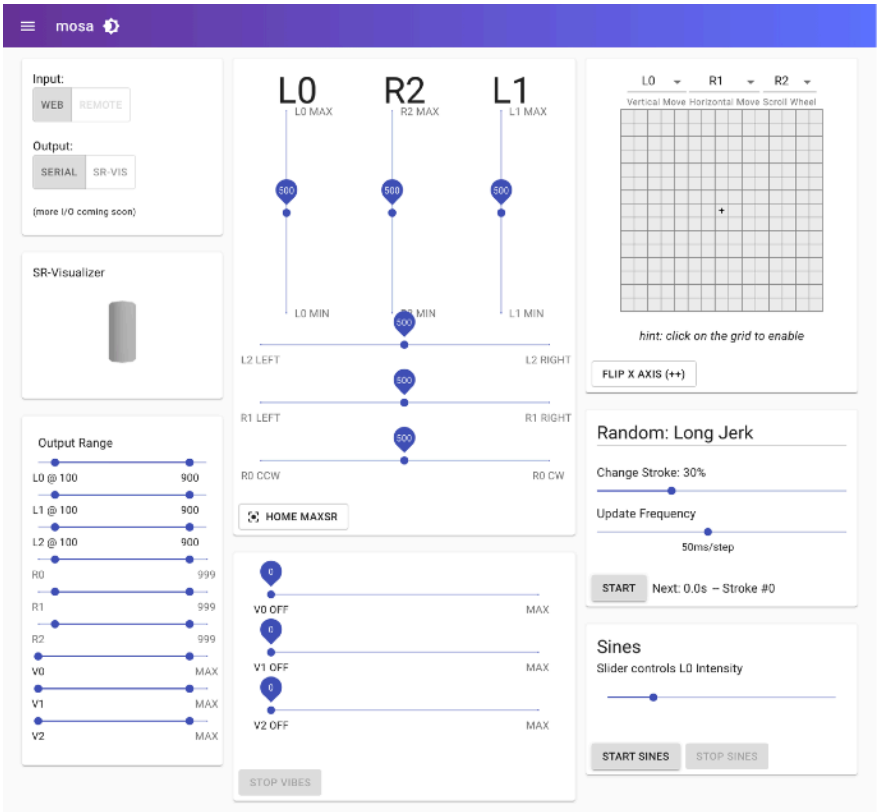
If you are supporting me on either of my Patreon or SubscribeStar platforms then you are entitled, and encouraged, to join the Tempest Discord server. This is a message board and chatroom for OSR2 users to swap content, tips and ideas. The server also also hosts an archive of all of my work to date, including documents and STL files.

If you have any questions or need help this definitely the best place to get an answer quickly.

To join the server all you need to do is add you discord account to your Patreon or SubscribeStar account and you should be added automatically. If you're having trouble getting this to work feel free to contact me directly through Patreon or SubscribeStar and I can add you manually.

The MOSA test app

A discord member called tnxa has created a web-based app called MOSA. This is a T-code controller application that is superb for testing the function of your OSR2, and I highly recommend you take a look.



You can find it at https://trymosa.netlify.app/.

Troubleshooting

Over the last two years or so I have helped a lot of people sort out problems with their OSR2 setup. I have listed below some of the most common problems they have encountered, along with the most common solutions.

For the best help troubleshooting an issue with an OSR2 I highly recommend that you join the Tempest discord server. I am available on there most of the time, but even when I'm not there will be someone else from the community who can help you. Most likely someone will have seen your issue before.

Receiver going up/down in the wrong direction

Check:
- Left servo - pin 8
- Right servo - pin 3

(This has been labelled incorrectly in some issues of the firmware)

The usual cause for this problem is the left and right servos being plugged in the wrong way around, ie left-to-right, right-to-left. Some servos work backwards out of the box. In either case, swapping the two leads (ie pin 8 to pin 3, pin 3 to pin 8) should resolve it. In some rare cases this is a software issue, in which case it can be resolved by re-uploading the Arduino sketch.

Arduino sketch not uploading

The usual cause of this is a quirk of the Romeo in which under certain circumstances some of the USB power is sent to the servos, which drains power from the board causing it to reset itself. If you are unable to upload the sketch to the Romeo this might be happening. You can resolve it by powering or unplugging the servos when you attempt the upload.

Problems with the Arduino environment can be addressed by checking out generic Arduino IDE tutorials and troubleshooting guides; remember that the Romeo BLE is an Arduino Uno as far as the software is concerned.

No lights showing on Romeo

There are small LEDs on the Romeo. We particularly care about the red one that indicates that it is powered, and the two green ones that will flash to indicate that data is passing between your computer and the Romeo.

If the red light does not come on when you plug the Romeo in to your computer the most likely reason is that there is something wrong with the board, or more likely the cable. A replacement cable usually does the trick.

Sometimes the red light will come on but the green lights will not. This might be because you are using a mobile phone charger cable which contains power wires but no data wires. Again, the first thing to try is a replacement cable.

If you are sure that the cable works but the program that you're using to send T-code (VaM, XTP, JFP, etc) doesn't appear to be sending data check for another program that might also be trying to use the serial link. Note Arduino IDE does not need to be open to run the OSR2 and the serial monitor should be closed when you're using the OSR2 with another program.

Computer cannot identify the Romeo

When the Romeo is plugged in to your computer it should appear as an accessible device in the Arduino IDE. If it does not you should check your device manager to see if it appears under the list of USB devices. If it doesn't appear on the list it may be a problem with the cable or the Romeo. Searching for generic Arduino IDE troubleshooting tips might help. The Romeo BLE is identical to an Arduino Uno as far as your computer is concerned.

Blue light showing on Romeo

This is a rare error in which the Romeo is unresponsive to T-code and a blue LED is lit up on the board. If this is happening it is probably due to the Romeo connecting via bluetooth to a mobile device, and thus ignoring the serial connection. Check for nearby devices and make sure that none of them are connected to the Romeo.

Servos not working

If powered the servos should all automatically centre themselves and hold position when you power up the Romeo. If the servos don't move this is usually a problem with the power supply to the servos. Check that you have wired up the servo power to the "Servo" connectors rather than the "Vin" connectors. If you are using an external power bus make sure that the power bus is grounded to the Romeo.

If you notice the servos getting very hot unplug the power supply immediately.

Single servo not working

If a single servo is not working the usual reason is that it is not plugged in to the correct pin. The usual reason for this is that your Romeo is a v1.1, not a v2 and you will need to reassign the output pins as I have outlined in a previous section. Also make sure that you are using the most recent sketch (*OSR-Release3_4.ino* at the time of writing) as the pin assignments have changed slightly over time.

Servo arms don't line up

The Futaba 25T servo horns have splines with 25 teeth on them, which means 25 possible mounting angles for each arm. This will mean that it's likely that once installed the two main arms will not line up with each other perfectly when the OSR2 is in the neutral position.

If you want to fine tune the arm zero positions you can change these in the Arduino sketch, which you will then have to re-upload to the Romeo for the changes to take effect. Look for this part in the code:

```
// Arm servo zeros
// Change these to adjust arm positions
// (1500 = centre)
#define Servo1_ZERO 1500  // Right Servo
#define Servo2_ZERO 1500  // Left Servo
#define Servo3_ZERO 1500  // Pitch Servo
```

By changing the "1500" value to a higher or lower value you can change the centre position for each servo. Mathematically an increase of 80 should result in 1/25th of a turn, so it should not be necessary to change the number by more than +/- 40 (ie 1460 to 1540).

T-Valve working backwards

As of *OSR-Release3_2.ino* the T-valve should move to the commanded (A1 channel) position, 0 = fully open, 999 = fully closed, except when the receiver is moving downwards, when it should be fully open.

Some models of micro servo seem to set up to work backwards. If the valve is doing the opposite of the above (0 = fully closed, 999 = fully open, fully closed on the up stroke) you probably have one of these servos.

The easiest solution to this problem is to reverse the valve direction in the firmware. Look for this line near the top of the .ino file:

```
#define REVERSE_VALVE_SERVO false // (true/false) Reverse T-Valve
direction
```

Change the word "false" to read "true".

Complete parts list

This is a complete list of all the non-3D printed parts to build an OSR2+ with a T-wist and T-valve.

Part	Qty	Notes
Romeo BLE Mini	1	v2 or v1.1
Micro USB cable	1	
Power supply	1	
Female barrel jack connector	1	5.5x2.1mm
Standard size servo	3	20kg.cm or more, 180 degrees
Futaba 25T M3 metal servo horn	3	
Standard size servo	1	5kg.cm or more, 270 degrees for best results
Circular Metal 25T Servo Horn	1	
M3x25 Dowel Pin	1	
Servo extension lead	2	60cm
Micro servo	1	
Paperclip	1	
Bolt M2x6	2	
M2 Washer	4	(Metric, 2mm, Form A)
Bolt M3x6mm	3	
Bolt M3x8mm	7	
Bolt M3x10mm	20	
Bolt M3x12mm	4	
Bolt M3x16mm	8	
M3 Washer	31	(Metric, 3mm, Form A)
M3 Nut	12	(Metric, 3mm)
Bolt M4x10	4	
Bolt M4x20	1	
Bolt M4x30mm	4	Cut down to 20mm. Can use M4x20mm instead.
M4 Washer	5	(Metric, 4mm, Form A)
M4 Nut	4	(Metric, 4mm)
6mm wiring grommet	4	1/4” also works
Spiral cable binding		
Cable ties		

When buying hardware parts such as bolts, washers, nuts etc I highly recommend over-buying on quantity. It’s always better to have 10 too many than 1 too few.

A word about personal safety

The OSR2 is an adult toy, and that means you need to be an adult and make sure that you play safe. This is not a finished consumer product, this is a piece of experimental tech that you put together yourself. Electrics can overheat, motors can exert a lot of force, materials can be sharp, microelectronics can be unpredictable, etc, so be careful!

I suggest you take some time to understand how the OSR2 can move before use. Make sure that you remove any sharp edges that may be created by your 3D printer with a craft knife or file. Do print and use the lid, as it will protect you, or rather important parts of you, from coming into contact with hot servos! I also highly recommend that you don't leave any hardware unattended whilst powered up, especially if you are using "budget" components.

Overall I take no responsibility for any actions that you may take after reading this document. It is your responsibility, and yours alone, to use any information that you have received here in a way that does not risk your personal safety.

Have fun; be safe!

And finally…

Congratulations on building your own OSR2, and welcome to the future!

Keep an eye on my Patreon or SubscribeStar page for the latest updates.

Enjoy!

Tempest