Computer control of budget antenna rotators

Having the ability to control an antenna rotator remotely enables lots of possibilities.

However, the cost barrier for computer control can be prohibitive. In this article, I will describe my implementation (**Figure 1**) which enables full computer control using a relatively-inexpensive rotator available from most amateur radio stores, and an infra-red (IR) USB dongle.

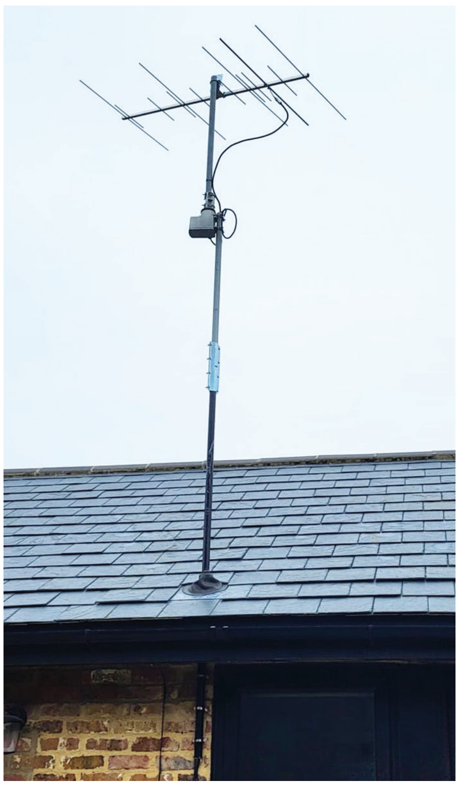


FIGURE 1: The antenna rotator mounted on a pole fixed to my roof.

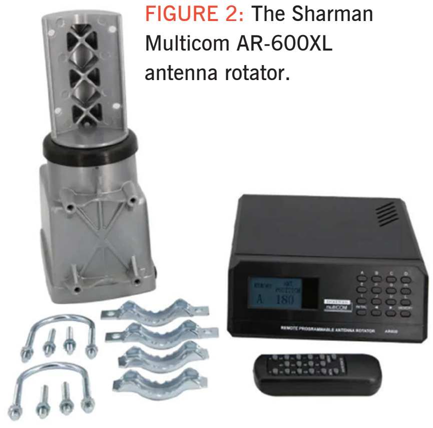


FIGURE 2: The Sharman Multicom AR-600XL antenna rotator.

FIGURE 3: The Flirc USB dongle.

Introduction

I am using a Sharman Multicom AR-600XL VHF/UHF Antenna Rotator (**Figure 2**), and the newer generation Flirc USB infrared dongle (**Figure 3**).

A block diagram of my implementation is shown in **Figure 4.** From right to left:

- the rotator is mounted on the mast which is powered and controlled via a three-core cable from a controller box within the shack; the controller box comes with an infrared remote-control;
- I have replaced the remote control with an infrared dongle and custom software that sends signals to the rotator controller from a computer; and
- amateur radio software such as the optional Web GUI (that supports rotator control), can be configured to use this custom software either directly on the same computer or via your Wi-Fi and thereby control the rotator.

In order to duplicate my implementation, you will need to be reasonably proficient with computer command-line operation, although you should be able to get by with limited knowledge. The system will work on Windows, macOS or Linux. In what follows, I give an outline of what needs to be done, and so this is not a full command-by-command explanation. Please get in touch with me by email if you need more detail.

Most software that supports rotator control will interface with HamLib 'rotctl'. HamLib is a rich set of libraries for all things to do with computers and amateur radio. I decided to make a python3 implementation that mimics HamLib, so that any software that controls rotators, and that supports HamLib rotctl, should be able to use my implementation. I doubt that it would be appropriate for me to extend HamLib, as I suspect the other rotator implementations may be more robust, but if you are a maintainer of HamLib, please let me know if you're interested.

In this article I will lay out the capabilities of the Sharman rotator, look at the infrared controller and other options (including building your own for pennies), go through my 'rotctlpy' implementation including using the scripts and 'webgui', and finally show you how to integrate it all with GPredict, a real-time satellite-tracking and orbit-prediction application, that can control the rotator automatically tracking satellites across the sky.

The Sharman rotator

The AR-600 has a programmable antenna controller with IR remote-control (Figure 2). It can store up to twelve antenna directions, and covers the full 360° in azimuth. It is well-suited for mounting on a permanent outdoor mast, and has an approximate rotation time of seventy-four seconds. Unfortunately, the rotator does not support an elevation drive, so the elevation is therefore fixed. Once programmed, a single press of a button will rotate the antenna to the desired direction. The rotator is powered using a three-core cable connected to the controller; the cable is not supplied, but three-core mains cable may be used which is readily obtainable.

The instructions supplied with the rotator are useful, and there are a few YouTube videos that you can watch to give you an idea of how

Steven Dodd, M0SNZ
stevendodd@outlook.com

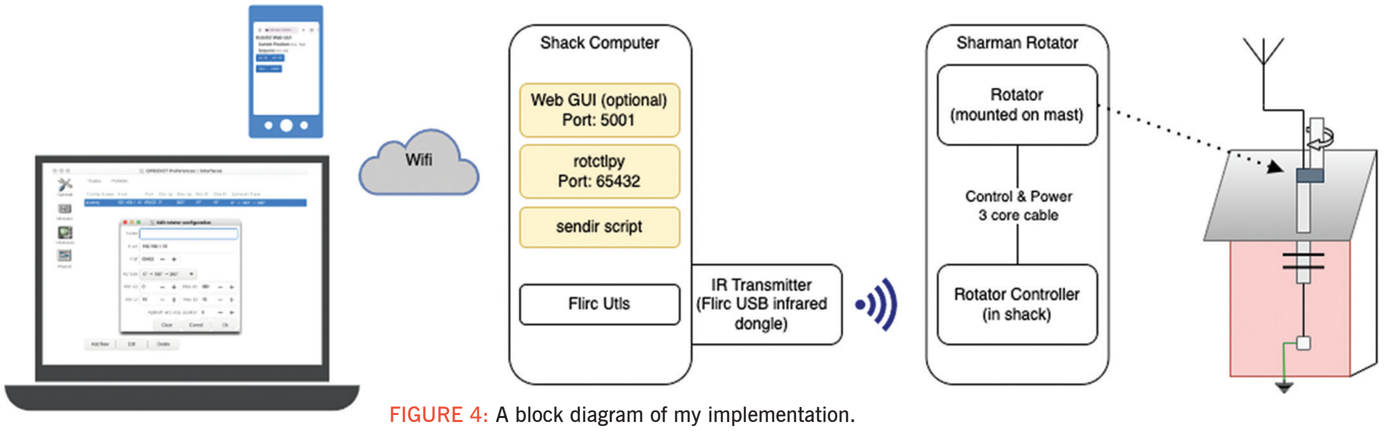


FIGURE 4: A block diagram of my implementation.

to set up the rotator and controller. There are, however, some necessary set-up tasks that must be undertaken for this implementation to work. After removing the controller from its packaging, it's best to program it and set it up before mounting the rotator on your mast. Follow the manual to see how to do this. Given that there are twelve programmable antenna directions, it makes sense to program the four cardinal directions along with two inter-cardinal directions between them. These are 0° (north), 30°, 60°, 90° (east), 120°, 150°, 180° (south), 210°, 240°, 270° (west), 300°, and 330°.

The rotator mount for the mast comes with a tongue that can slot into a groove cut into the mast, thus securing it and preventing undesired movement over time from wind and other factors. Having programmed the controller as suggested above, select 0° and mount the antenna to point towards the north.

At this stage, you should have a perfectly-good antenna rotator capable of rotating a small- to medium-sized antenna using the buttons on the main control unit or the infrared remote control.

Infrared transmitter

Computer control requires the computer to transmit the appropriate infrared codes, for which additional hardware is needed. There are several options available but I settled on the Flirc USB dongle (Figure 3) which is available for around £20 as an off-the-shelf solution. If you are feeling adventurous, it's entirely possible to substitute this dongle with an IR receiver and emitter diode, as well as the LIRC libraries, for a pound or so, and there are plenty of tutorials on-line showing you how to wire it all up.

Flirc USB dongle

At first glance, the Flirc USB dongle was exactly what I wanted; however, I did struggle with both the software installation and the use. After installing the software, I needed to raise several support cases to get the implementation to work. This is all been sorted out now, so I hope that others will not have the same problems. Flirc USB software can be installed on Windows, macOS and Linux, and the first step is to install this software, along with the included 'flirc utils' package for command-line control.

If you are using the same hardware as I am, there is no need to undertake the capture and playback steps below, as the IR codes are already captured and implemented in my software; however, they are included here for completeness. The steps will obviously be

```
+9090 -4401 +643 -453 +637 -456 +638 -456 +643 -452
+638 -456 +638 -456 +639 -456 +638 -458 +611 -1610
+639 -1588 +639 -1584 +642 -1585 +643 -1584 +638
-1585 +642 -1584 +639 -1588 +639 -456 +638 -1584 +669
-426 +612 -1615 +638 -456 +639 -456 +638 -1584 +643
-451 +643 -1584 +643 -451 +638 -1588 +639 -456 +648
-1578 +638 -1588 +638 -465 +630 -1584 +642
```

FIGURE 5: An example of a button sequence transmitted by the rotator remote control.

```
irtools sendir --raw="+9090 -4401 +643 -453 +637 -456 +638 -456 +643 -452 +638 -456
+638 -456 +639 -456 +638 -458 +611 -1610 +639 -1588 +639 -1584 +642 -1585 +643-1584
+638 -1585 +642 -1584 +639 -1588 +639 -456 +638 -1584 +669 -426 +612 -1615 +638 -456
+639 -456 +638 -1584 +643 -451 +643 -1584 +643 -451 +638 -1588 +639 -456 +648 -1578
+638 -1588 +638 -465 +630 -1584 +642" --repeat=1
```

FIGURE 6: Use of the flirc utils command to re-transmit the IR sequence.

```
Download Source code from https://github.com/stevendodd/rotctlpy/releases/latest
Extract zip file and open a command prompt in the new directory
pip install -r requirements.txt
python rotator.py
```

FIGURE 7: Installation commands.

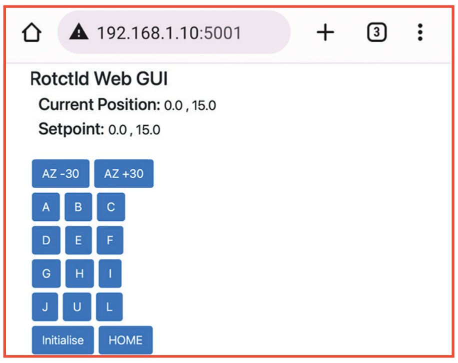
```
python rotator.py --listen_port 5001 --host localhost --port 65432 -g
```

FIGURE 8: Invoking the optional web GUI.

different if you are using different hardware.

Once the Flirc USB software is up and running, you need to enable IR debug, point your rotator remote control at the Flirc dongle, and press a button on the remote control. You should be able to capture the output in the debug log; each button has a unique sequence, and you will need to do this for all 12 rotator memory positions. The output is in some proprietary timing format and will look a bit like that shown in **Figure** 5. Make sure to capture the full sequence of numbers, and label each set with the corresponding button.

Now that you have the raw data captured from the remote control, it is a simple task to re-transmit the signal using the flirc utils, using the command shown in **Figure** 6. This command should send the infrared signal and, if the rotator controller is in range, it should engage and rotate the antenna to the desired position.


192.168.1.10:5001

RotctlD Web GUI

Current Position: 0.0 , 15.0

Setpoint: 0.0 , 15.0

FIGURE 9: The optional web GUI.

Rotator controller software 'rotctlpy'

The last piece of the puzzle is to create an implementation that can interface between the hardware and readily-available amateur-radio software. I decided to create a python3 implementation of HamLib rotctl, as most software that controls antenna rotators will support this interface.

The implementation is made up of three components indicated by the yellow boxes within the shack computer as shown in Figure 4. These are the rotctld-type server, a script to send IR commands, and an optional web GUI interface. The web GUI can be used from your mobile phone, or indeed from any other web browser. It was originally written by Mark Jessop, and I adapted it for my use case, and updated it for use with python3.

Once the server is running, it will accept rotctl commands on port 65432. If asked to rotate the antenna, it will invoke the script as a sub process to send the IR command. Please note that I have not tested it for use with multiple clients so, for example, if you are using GPredict and the supplied web GUI, you should only connect one client at a time to the server.

Installation

Having installed python3 on your system, the set of commands shown in **Figure 7** may be used to effect the following steps:

1. Download the rotctlpy source zip file, and the optional web-gui zip file, from https://github.com/stevendodd/rotctlpy/releases .
2. Extract the release source zip file onto your file system.
3. Extract the optional web-gui zip file into the supplied, and unzipped, rotctld-web-gui folder made in step 2.

Running the implementation

In the top-level folder, there is a script called 'sendir.sh'. It takes a single command line option, namely the button you want to 'press' on the remote. You can invoke it as follows:

```
./sendir.sh A
```

This will transmit the sequence corresponding to button A on the remote control, using the Flirc command line 'utils' to trigger a rotation manually. If you are running on a UNIX-like operating system, this can be invoked via an SSH command remotely without having the rotctld-like server running. If you are using Windows, you can use the supplied batch file instead: 'sendir.bat'. If you use any other IR software, this script is the only place you will need to edit with your corresponding infrared commands.

The heart of the implementation is a rotctld python3 implementation. It can be started by simply running:

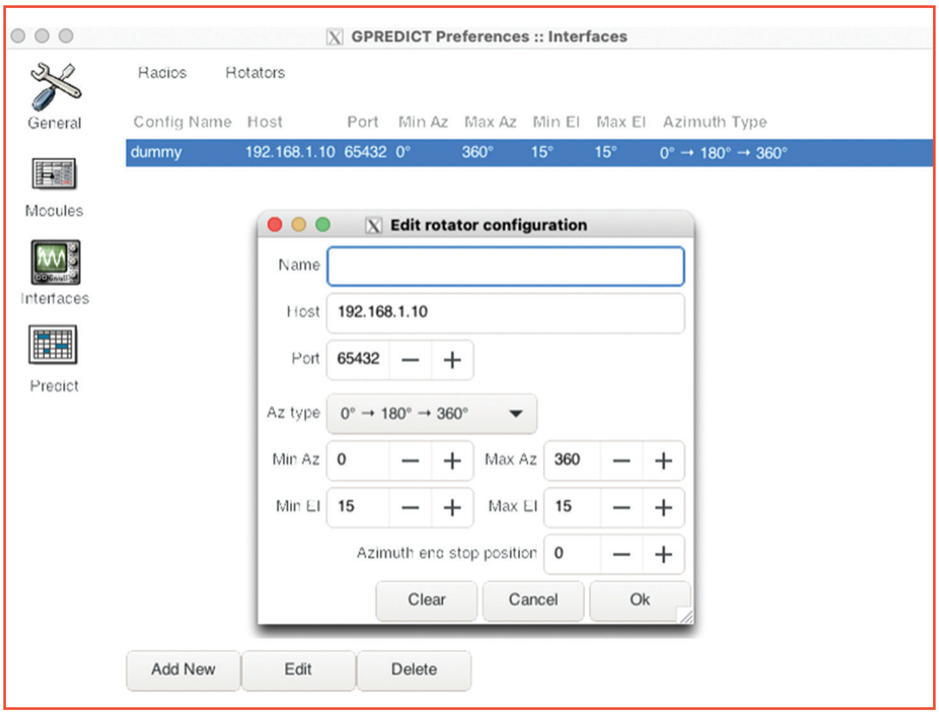
```
python3 rotator.py
```

Once invoked, the server will listen for a TCP/IP connection on port 65432, and will accept standard rotctl commands to operate, and report on, the rotator. Obviously, without an interface to the antenna controller, other than the ability to send an update-position IR signal, there is no way to guarantee that the antenna is indeed at a particular position. The server will send an initialise command on start-up, and then attempt to maintain the antenna position in memory. If, for any reason, the position is incorrect, simply restart the server, which will in turn re-initialise the antenna.

Please note that this python script invokes the sendir script. If you are using alternative commands to transmit the IR signals simply update the relevant sendir script for your operating system.

Web GUI

I wanted a simple interface to the server using a browser and, rather than write my


GPREDICT Preferences :: Interfaces

Config Name	Host	Port	Min Az	Max Az	Min El	Max El	Azimuth Type
dummy	192.168.1.10	65432	0°	360°	15°	15°	0° → 180° → 360°

Edit rotator configuration

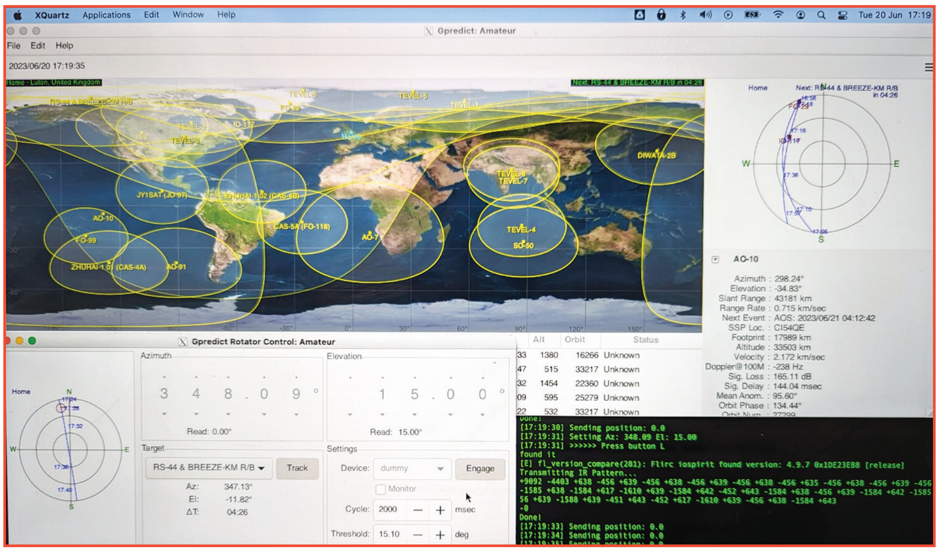
Name:
Host: 192.168.1.10
Port: 65432
Az type: 0° → 180° → 360°
Min Az: 0
Max Az: 360
Min El: 15
Max El: 15
Azimuth end stop position: 0

FIGURE 10: GPredict rotator configuration.

own, I included a previously-written rotctld web gui python implementation (see **Figure 9**). To start the GUI, simply move to the top-level directory, and invoke the command shown in **Figure 8**. The web GUI will then connect to the server, and you should be able to get access to it at the URL:

```
http://localhost:5001
```

I use my mobile phone to connect to my computer by substituting 'localhost' with the computer's IP address, which then allows me to operate the antenna via my Wi-Fi network from anywhere in the house. The GUI allows movement of the antenna incrementally 30° at a time through the full 360° rotation, or to reset to the initial position, and it reports the antenna position as registered in the server.



```
AO-10
     Azimuth : 298.24°
   Elevation : -34.83°
 Slant Range : 43181 km
  Range Rate : 0.715 km/sec
  Next Event : AOS: 2023/06/21 04:12:42
    SSP Loc. : CI54QE
   Footprint : 17989 km
    Altitude : 33503 km
    Velocity : 2.172 km/sec
Doppler@100M : -238 Hz
   Sig. Loss : 165.11 dB
   Sig. Delay : 144.04 msec
  Mean Anom. : 95.60°
 Orbit Phase : 134.44°
```

```
Target: RS-44 & BREEZE-KM R/B   Az: 347.13°   El: -11.82°   ΔT: 04:26
Device: dummy   Cycle: 2000 msec   Threshold: 15.10 deg
```

```
[17:19:30] Sending position: 0.0
[17:19:31] Setting Az: 348.09 El: 15.00
[17:19:31] >>>>>> Press button L
found it
[E] fl_version_compare(281): Flirc iospirit found version: 4.9.7 0x1DE23EB8 [release]
Transmitting IR Pattern...
+9092 -4403 +638 -456 +639 -456 +638 -456 +639 -456 +638 -456 +635 -456 +638 -456 +639 -456
-1585 +638 -1584 +617 -1610 +639 -1584 +642 -452 +643 -1584 +638 -456 +639 -1584 +642 -1585
56 +639 -1588 +639 -451 +643 -452 +617 -1610 +639 -456 +638 -1584 +643
-0
Done!
[17:19:33] Sending position: 0.0
[17:19:34] Sending position: 0.0
```

FIGURE 11: **Gpredict using rotctlpy to track satellites automatically.**

GPredict satellite tracking with rotator control

By now, you should have the ability to adjust the antenna manually by running the sendIR script locally or remotely, along with a web-based interface if you prefer. In addition, you should have a server running that can be connected to readily-available amateur-radio software, such as GPredict, and in this section I will show you how to connect GPredict to the server for automatic satellite tracking.

In the GPredict 'Preferences', you can set up an antenna rotator in the interfaces section (see **Figure 10**). Give your rotator a name which is used within the rest of the GPredict interface. In the host input box, add the IP address of the computer running the rotctlpy server. Select port 65432. The 'AZ type' should be set to 0° - 180° - 360°, with a minimum AZ of 0° and a maximum AZ of 360°. Set the minimum and maximum EL to 15; this is not relevant as the elevation is fixed, but my script by default sends back 'in EL of 15°'. Save your rotator, and exit the preferences.

To track a satellite, open the rotator-control dialogue box within Gpredict, select a target satellite, and press the 'track' button. In the settings section of the rotator-control dialogue box, select the name of the rotator you set up on the previous step. Choose and a threshold of 15.1°, and a cycle of 2000 ms; this will update the antenna position every two seconds. The threshold indicates when GPredict needs to send an updated target position to the rotctl server. Since our pre-programmed settings are 30° apart, 15° is the halfway point which will trigger movement to the next predefined direction.

Once these settings have been applied, press the 'engaged' button to connect to the rotctlpy server; you should see the antenna position plotted on the satellite polar view, along with the reported assumed position, whenever GPredict asks the antenna to rotate. See **Figure 11**.

Improvements

Future enhancements could include adding a solar-powered compass to the rotator mast, adding the ability to obtain true directional readings. The Raspberry pi zero 2 W, with the 'Sense Hat' that was for developed for use on the international space station https://www.raspberrypi.com/products/sense-hat/ , would be a good choice. I have also integrated the solution into 'Home Assistant' with a clickable graphical compass as a replacement for the web GUI.

This implementation is not a commercially-polished application, and requires a little bit of tinkering to get working. However, it works a treat and will provide you with low-cost integrated computer antenna control. Have fun!