Hackaday Prize Entry

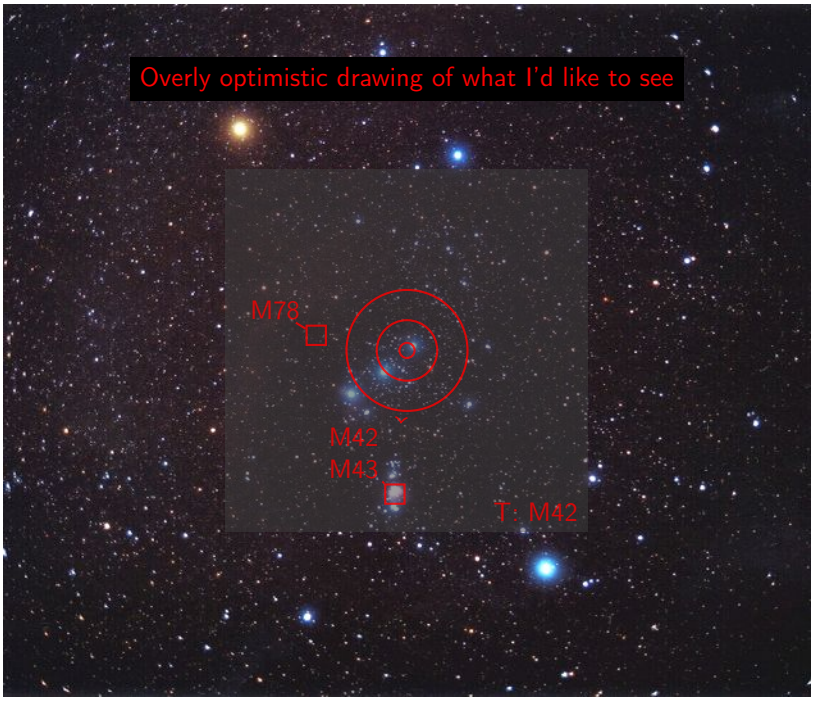
explorad

System Design Document

Christoph Redecker

August 18th, 2014

This is the "System Design Document" required to enter the quarterfinals of The Hackaday Prize 2014, for my project explorad.



Contents

1 Introduction

Amateur astronomers need to navigate through the night sky when they want to take a look at astronomical objects, such as star clusters, nebulae, and even other galaxies. To get an overview of the existing objects, astronomical charts provide location data (and other, like brightness, exact type, ...).

Getting an object into view basically requires two main pieces of information

1. which object that actually is, and
2. where the object is, and how to find it.

The first one, knowing which object should be found, is decided by the user, usually by picking one from a star chart or by recommendation from more experienced astronomers. Beginners usually start with a list of easy to find objects whose content depends on the current date and latitude of observation. Other than that, astronomical charts can provide an overview of the visible objects for the current time and location.

The second requires little helpers, patience and experience. The Great Orion Nebula, for example, is easy to find at roughly 4° "below" Orion's belt. If it wasn't such a bright and large nebula, finding it even with this information would be hard for a beginner, because estimating angular distances isn't something we learn when we are kids, and not all objects are that close to distinct features of well known constellations.

The little helpers mentioned previously can be of very different character. One solution is the use of GoTo mounts that take on object's name (or coordinates) as input, and automatically move the telescope axis to point at the desired object. They are very expensive for large telescopes. Even if money is not a problem, *they blind out some of the explorative nature of astronomy.*

A less complex solution is the "Telrad", which is a device the astronomer attaches to a telescope in addition or in place of a finder scope. The Telrad contains a screen with three backlit circles of different diameter and some optics to work as a head-up display. The diameter of these circles is 0.5, 2 and 4°, respectively. Modern astronomical charts usually also contain these circles around objects, which simplifies the process of finding them; see fig. 1. Unlike GoTo mounts, the astronomer still needs to look at a chart that provides information about the surroundings of the object, giving an incentive to explore.

The explorad works like a Telrad, but the three fixed backlit circles are replaced by an OLED display that can display arbitrary (well, with some limitations ...) data, like charts and other helpers, making the whole experience of astronomy much more interactive and explorative.

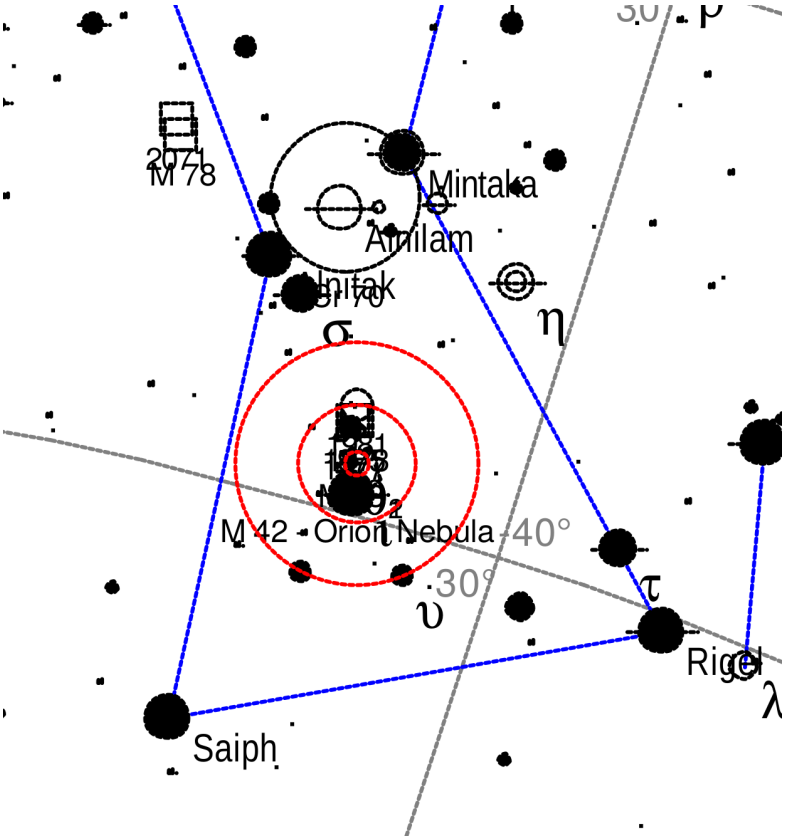


Figure 1.1: Sample astronomical chart showing the region south of Orion's Belt and telrad circles around the Great Orion Nebula

2 How the Telrad works

This chapter briefly describes how the *Telrad* works, which the explorad is based upon from an optical point of view. The Telrad is a commercially available device that acts as a head-up display with fixed information. It displays 3 circles with fixed diameter (0.5, 2 and 4°) in the viewer's field of view (FOV). It is attached to the telescope, and the optical axes of both must be aligned so that when an onject is centered in the telrad, it is also centered in the telescope's FOV. It contains some optics (namely a lens, a mirror, and a glass plate) to create the impression that the circles are infinitely far away — just like the stars, nebulae and other objects we see in the night sky. The circles also seem to "stick" to their position at the sky so that they don't move when the viewer's and the device's optical axes are misaligned.

After the principle is described in the first section, some math is done in section 2.2, followed by some ergonomic and usability considerations in section 2.3, for a telrad-like device previously built by the author, which will also influences the explorad's final design.

2.1 Principle

Figure 2.1 shows the Telrad's main components. The viewer looks at an object which is virtually infinitely far away, so the eyes are focused at infinity. A glass plate is inserted in the optical axes, resulting in the object being slightly fainter, but also acting as a nearly transparent mirror.

A movable screen with three backlit circles is placed in front of a tilted mirror, which reflects the screen's light to a convex lens. This lens is placed beneath the tilted glass plate, so that the screen appears in the viewer's optical path. In order to show the circles at infinity focus, the screen must be placed very close to the lens' focus length f. The wikipedia article on Reflector sights [1] contains more information and an animation. A selfmade Telrad is shown in figure 2.2 and the resulting projected circles are shown in figure 2.3.

[1] `http://en.wikipedia.org/wiki/Reflector_sight`, as seen on August 18th, 2014

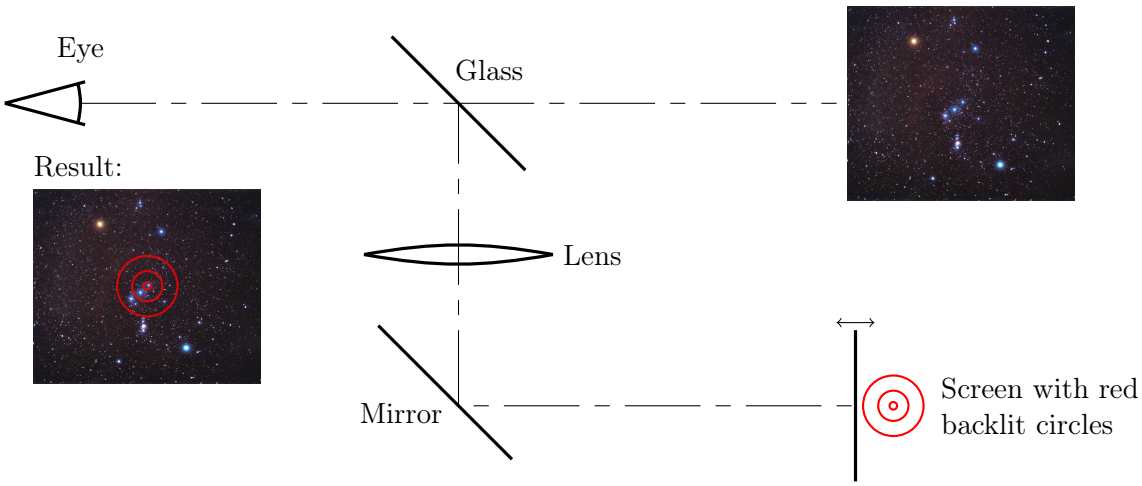


Figure 2.1: The Telrad's optical components, and how they interact to work as a head-up display. The object of interest is at the top right, being viewed through a tilted glass plate. A convex lens and a tilted mirror add the backlit circles to the image, creating the overall result shown. The screen can be moved slightly to place them at infinity focus.

Figure 2.2: Selfmade Telrad: the viewing direction is from left to right, the device is about 200 mm long and 100 mm high. The mirror cannot be seen directly in this picture as it is located below the lens; it only shows a reflection of the enclosure.

Figure 2.3: Telrad circles as seen with a selfmade Telrad. The small red dot at the top left of the circles is also at the top left of the screen, indicating that the screen image is not mirrored when viewed through the Telrad's optics.

2.2 Some math

2.2.1 Circle diameter

Given a lens with focal length f, the physical diameter d of a circle that shall appear to have an apparent size of α degrees is to be determined. The relation is simply

$$d = f \tan \alpha \tag{2.1}$$

In order to get reasonable values for d, the lens must be chosen to have a reasonable focal length f. The one used in the explorad has $f \approx 145\,\text{mm}$, resulting in diameters for the 0.5, 2 and 4° circles to be 1.27, 5.06 and 10.14 mm, respectively.

2.2.2 Glass plate size and field of view

Given a distance l between the eye and the glass plate (see fig. 2.1) and glass plate's width w, the available FOV angle β is

$$\beta = 2 \arctan \frac{w}{2l} \tag{2.2}$$

so for example for $l = 150\,\text{mm}$ and $w = 50\,\text{mm}$ we get $\beta = 18.9°$. With the given example dimensions the circles would well fit within the FOV, so the plate is large enough in this case. If the lens' diameter is roughly the same as w, we're on the safe side here.

2.3 Ergonomics

The main drawbacks of the author's homemade (and also of the original) Telrad are the small distance between the device's and the telescope's optical axes and the lack of dew protection.

Placing the Telrad's optical axis too close to that of the thelescope means that it can sometimes be hard to place an eye on the Telrad's optical axes especially when other hardware is attached to the telescope. This is a very common situation, so the explorad's optical axis should be a bit higher. On the other hand, placing the optical axis too high would require a larger glass plate and a wider enclosure to acommodate it.

The circles displyed by the telrad are red. This, of course, has a reason: The human eye is least sensitive to red light, and must stay adapted to the darkness during observations. Red is the color that least disturbs this adaption and is therefore the color of choice.

Dew protection is necessary to avoid dew settling on the optical parts as they cool down during observation. When they cool down below the air's dew point due to radiative heat transfer between a surface and the (very) cold sky, the water will start to settle on the surfaces as little drops and blur the view. This can be avoided by adding a "dew cap" that catches all the dew as it cools down, leaving less humid air around the optical parts.

3 Explorad hardware overview

From an optical point of view, the explorad works just like a Telrad (which was described in the previous chapter), with the following modifications:

- The fixed screen with 3 backlit circles is replaced by an OLED display;
- Hardware is added to drive the display and to provide user input;
- Sensors are needed to determine the mount's orientation. Only "Dobsonian", or *altitude-azimuth mounts*[1], are considered for now. This is in line with the overall idea that this is a device for beginners, who usually don't have equatorial mounts[2].

The following sections describe the display, microcontroller, storage, sensor, wireless communication and user input choices that were made.

3.1 Display choice

The display chosen for the explorad is a 128×128 OLED display as sold by adafruit[3]. Its main characteristics relevant to the explorad are its OLED'ness, resolution, pixel pitch, overall size and form factor of the board, hardware interface and the integrated microSD card socket.

Why OLED OLED displays don't need a backlight, and the user can choose to display only red pixels. As described in section 2.3, this ensures that the display does not have a negative effect on the eyes' adaption to the darkness.

Resolution The resolution of 128×128 pixels offers enough opportunity for some detail in the displayed data, but is not too large for a decent microcontroller to manage.

Pixel pitch The pixel pitch 0.21 mm. For the chosen lens (see section 2.2) this results in a diameter of 6 pixels for the 0.5° circle, enough to make it look like a circle. The largest circle's diameter is 48 pixels, small enough to be displayed entirely *and* to add some information around it.

Overall size and form factor It's not larger than it needs to be, and is almost square. This imposes very little restrictions on the surrounding design.

[1] `http://en.wikipedia.org/wiki/Altazimuth_mount`
[2] `http://en.wikipedia.org/wiki/Equatorial_mount`
[3] `http://www.adafruit.com/products/1431`

Hardware interface The display controller has an SPI and needs only 5 I/O lines including all control signals.

Integrated microSD card socket See section 3.3 why an SD card is useful for this device. Already having a microSD card socket somwhere is just convenient.

3.2 Microcontroller choice

The microcontroller was chosen based on some preliminary experience with AVRs, Arduino, and the fact that these projects usually end up having more features than planned. A first test with the display had shown that an AVR (ATmega32U4) could drive the display at 5 fps, while doing nothing else. That was way too slow.

The display uses 16 bits per pixel, so for smooth 25 fps it would need an SPI clock frequency of

$$\begin{aligned} f_{SCK} &= 25\,\text{fps} \times 128 \times 128 \times 16 \\ &\approx 6.55\,\text{MHz} \end{aligned} \tag{3.1}$$

Given the fact that some SPI time would also be needed for reading data from the SD card, the SPI would need to run at about 12 MHz or even faster. An ATmega32U4 cannot provide that. Because the Teensy 2.0[4] from PJRC was used for previous projects, and because the support for it was so awesome, the next upgrade available from PJRC was considered, which is the Teensy 3.1 based on a Cortex-M4. After writing a display driver that uses DMA for the SPI transfers, the Teensy 3.1 could drive the display at 25 fps and about 10 % CPU load. The Teensy 3.1 also has enough other interfaces such as $I^2C$and several UARTs.

3.3 Data Storage

As the explorad is supposed to display information about what the astronomer could have a look at, this information must be stored somewhere. An SD card provides enough space to store information for thousands of objects, a number of different object catalogues, and bright alignment stars.

3.4 Sensors

As long as the optical axes of the explorad and the telescope are aligned with each other, it should be allowed to rotate the explorad around its optical axis, because sometimes it's just not possible to simply mount it upwards: the finder scope or other hardware might be in the way, and mounting would also need to be exact to a very small angular error. The sensors must therefore supply information about azimuth, altitude, and rotation of the device.

[4] `https://www.pjrc.com/teensy/`

3.4.1 Azimuth

The original idea was to use an electronic compass to sense the azimuth. However, it turned out that getting reliable and reproducible readings from such a sensor was almost impossible. It would have required extensive calibration on the actual telescope, and even wearing a wrist watch led to insane errors.

Not yet built and implemented: The azimuth will be sensed with an optical encoder mounted to the telescope base. Pulses are evaluated by a small microcontroller (ATtiny85 or similar) and sent to the main device via a serial connection.

The current implementation of azimuth sensing is simply a potentiometer connected to an analog input of the microcontroller. This serves as a dummy azimuth sensor on the breadboard until the real hardware is built.

3.4.2 Altitude and rotation around the optical axis

Altitude and rotation around the optical axis are sensed with a 3D accelerometer. The one used in this project is the LSM303D on a prototyping board made by Pololu[5] and it is connected via I^2C. The sensor also includes a 3D magnetometer, see 3.4.1 for why this choice was made initially. Future hardware might just include the 3D accelerometer.

3.4.3 Location and time

Earth rotates. While this is essential for life, it complicates the device described in this document.

Location and time (UTC) information is necessary to display the data for the current location, time, and viewing direction. This information is collected from a GPS module that is enabled during the device's startup procedure. Astronomers usually don't move that far after they've set up their equipment, so the GPS module can be switched on and off using a MOSFET switch for reduced power consumption. After the GPS is switched off, time is tracked with a CPU timer.

3.5 User input

An external input device is used to control the main device. It provides 5 buttons arranged in a cross arrangement (left, up, right, down, and enter) and also the power supply for all the electronics. The device is powered by a single LiIon or LiPo cell, including a step-up converter and charger circuit with USB input. The battery can thus be charged without being removed, simply by plugging the input device into a USB port. The controller used in the input device is an ATmega32U4 (Teensy 2.0)[6].

All user input is passed to menus shown on the display.

A touch wheel might be added later to make things easier.

[5] `http://www.pololu.com/product/2127`

[6] `https://www.pjrc.com/teensy/index.html`

3.6 Wireless communication

A CC3000 breakout board[7] was added to get the explorad online. The plan is to use it for reporting usage statistics, such as which objects were viewed, and to download updated object catalogs. Updated catalogs might be necessary to provide information about solar system bodies (planets for example) because their celestial positions change rapidly[8] and it might just be more convenient to grab that data from the web on startup.

Currently, the CC3000 is not used due to the fact that it communicates over SPI and therefore shares that interface with the display and SD card. While this is no problem from a hardware point of view, the CC3000 driver provided by adafruit uses the SPI during an interrupt service routine triggered by an external signal from the CC3000, and that breaks communication with the two other peripherals. teensyduino 1.20[9] provides a workaround, but the current software uses teensyduino 1.18 and needs a lot of porting before these features can be used together.

3.7 Teensy 3.1 communication interface usage

This chapter outlines which communication interfaces provided by the Teensy 3.1 are used for which purpose, and what data is transferred through them.

3.7.1 SPI0

SPI0 (the only fully available SPI port) is used for

- the OLED display, 6.55 Mbps,
- the SD card, some smallish amount of data certainly smaller than that for the display,
- planned: CC3000, also not much data.

The SD card is used whenever the viewing direction relative to the fixed sky coordinates (right ascension and declination[10]) changes. This happens either when the telescope is moved or when a certain amount of time elapsed. Information about which objects are in view a then loaded from the SD card.

3.7.2 UART0 (Serial1 in teensyduino)

Used by the GPS during startup. It might be possible to have the GPS and the azimuth sensing controller share a UART since they are not needed at the same time — the GPS is used during startup and the azimuth is only needed later — but that seems more complicated than necessary and currently an extra UART is not needed.

[7] `http://www.adafruit.com/products/1469`
[8] In astronomical terms, that is
[9] `https://www.pjrc.com/teensy/teensyduino.html`
[10] `http://en.wikipedia.org/wiki/Equatorial_coordinate_system`

3.7.3 UART1 (Serial2 in teensyduino)

Currently these pins used by UART1 are occupied by display signals but that will have to change to make room for the controller that reads the azimuth encoder (see section 3.4.1).

3.7.4 UART2 (Serial3 in teensyduino)

Used for the input device. The input device basically just sends information about button press events, i.e. which button was pressed and which is released. As it contains a microcontroller, it can also provide battery status information (voltage and if it is currently being charged).

3.7.5 I^2C

The I^2Cinterface is used for the LSM303D, currently no other sensor is connected via I^2C. Data is read at 50 Hz and filtered for a smooth ride.

4 Current development status and plans

This chapter outlines the current development status, what features don't work as expected (yet), and what would be desirable. Explorad is made up of three main parts: The main device, the input device and the azimuth sensor.

4.1 Main device

The main device consists of the microcontroller, Display, SD card, GPS, Accelerometer and CC3000 (WiFi). It is the part that is subject to most changes, and currently only built on a breadboard, as shown in figure 4.1. What works:

- The display is updated at 25 fps. The display is drawn in pages so as to use less memory for drawing;
- A small set of simple widget classes is used to create the graphics;
- The dummy mount (LSM303D plus azimuth potentiometer) is used to determine azimuth, altitude and roll of the device;
- A (currently fixed) location and startup time (updated at runtime) is used to calculate right ascension and declination;
- According to right ascension and declination, the currently visible objects are read from a catalogue on the SD card;
- The object catalogue is based on the Revised New General Catalogue and Index Catalogue by Wolfgang Steinicke[1] and contains 13957 objects (galaxies, nebulae, star clusters and so on);
- The celestial sphere is divided into 1280 triangular patches to speed up loading of visible objects from the catalogue. The patches are loaded from the SD card on the fly. The most crowded patch contains 267 objects, that's why not all can be displayed at a time even if they are organized in patches;
- Up to ten (the brightest) visible objects can be displayed and labeled on screen;
- A target hint can be displayed that currently points to south (azimuth) at the horizon (altitude), but it correctly follows changes in azimuth, altitude and rotation;
- Telrad circles can also be displayed if required;

[1] `http://www.klima-luft.de/steinicke/ngcic/rev2000/Explan.htm`

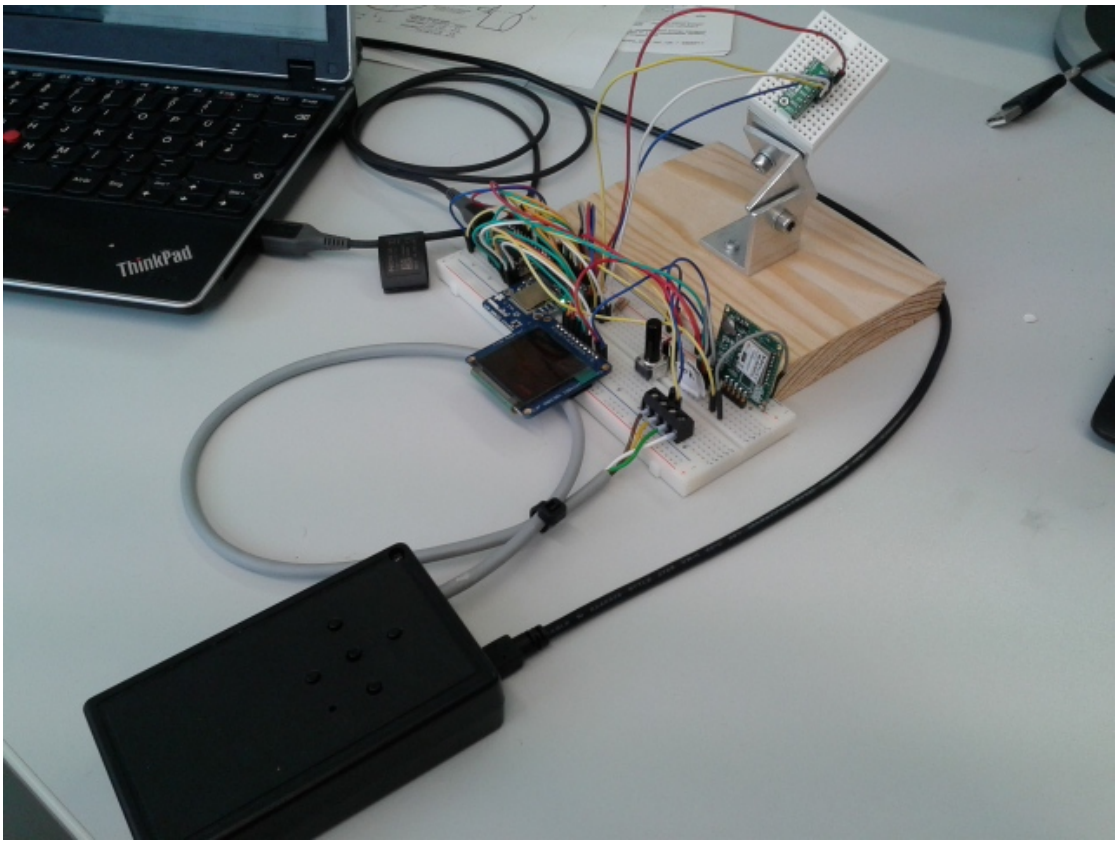


Figure 4.1: Main device built on a breadboard, the LSM303D is attached to a small dummy mount on a wooden base. The Teensy 3.1 is hidden behind that colorful mess of wires, but a USB cable connecting to it can be seen. The input device is also included in the picture (black enclosure with 5 buttons).

- Display brightness can be adjusted.

The most important plans are:

- Porting to teensyduino 1.20. This will make it possible to use the CC3000 together with other devices on the same SPI bus. That's very important;
- Currently the GPS is not used, but parsing NMEA sentences works well. There is also no menu to grab the GPS location or manually enter coordinates, but both GPS and manual input will be added. Same for time;
- Alignment with a set of bright stars. As the optical azimuth sensor (to be added) will only provide relative data, it must be aligned with the real sky in order to give sensible results. A good GUI will need to be created. Alignment can only be done with visible stars, so this highly depends on the current location and time. Alignment will also slightly improve the readings from altitude and rotation;
- Searching for an object to be displayed as the target is not yet implemented;
- When the CC3000 works, code must be written to download missing data about solar system bodies and possibly upload usage data;
- Build the hardware to hold the electronics, mirror, lens, and glass plate to actually turn this into a usable device. As the author has already built a Telrad, this is known to be simple and doable within two days.

Less important:

- A better way to place the object labels on screen. They currently overlap in crowded regions, and a force-directed approach is well within the available CPU power *and* would be awesome;
- Long term: Get rid of the GPS. Theoretically it's possible to use star alignment alone to calculate startup time and location;
- When we're already connecting to the net, some kind of multiplayer game could be added to this so that two ore more users could play against each other. Pong comes to mind, of course, but also some sort of space invaders — there are many possibilites.

Good news: A closer look at the Telrad revealed that the image displayed by the optical setup does not mirror the image displayed on the screen (see 2.3 again). That means that no mirroring code has to be implemented, which could be quite hard given that font drawing and rotation compensation already work well.

4.2 Input device

The input device is partially finished, including an enclosure (see figure 4.1 again), but still lacks the battery, charger circuit, and voltage regulator required for mobile usage. Other than that, it happily sends button press and release events to the main device. Battery, charger circuit, and voltage regulator already work well on a breadboard.

A touch wheel might be added later, because it would for example make entering numbers a lot more comfortable. A slightly smaller enclosure would probably work if a real PCB is designed. This seems to be a reasonable plan as the input device can actually be used to navigate through all sorts of similarly structured user interfaces, so hardware-wise it's the most generic part of explorad.

4.3 Azimuth sensor

As mentioned previously in section 3.4.1, the sensor hardware has not been built yet, so this is completely something that still needs to be done.

5 Bill of materials

The bill of materials is divided into tables for main device, input device and azimuth sensor. It might change a lot when real PCBs are designed.

5.1 Main device

Optics and mechanics (really just a guideline):

Amount	Name	Description
1	Lens	convex, $d \approx 50\,\text{mm}$, $f \approx 150\,\text{mm}$
1	Glass plate	thin, flat, about $50\,\text{mm} \times 70\,\text{mm}$
1	Mirror	thin, flat, about $50\,\text{mm} \times 70\,\text{mm}$
a few	plastic plates	to create an enclosure
a few	screws	to hold the plastic plates
a few	more screws	to create an alignable mirror

This is really not meant to be exact information, as it's possible to build this device in many, many ways. Especially the hardware needed to adjust the display's position to be at infinity focus is not yet included.

Electronics:

Amount	Name	Description
1	PJRC Teensy 3.1	Freescale mk20dx256 breakout board
1	Adafruit # 1431	128×128 OLED display, SSD1351 controller
1	Pololu # 2127	LSM303D breakout board
1	Adafruit # 1469	CC3000 breakout board
1	Navilock NL-507ETTL	GPS breakout board, TTL levels
1	microSD card	any size, one GB is already more than enough

The GPS module can be replaced with any module that automatically starts to transmit NMEA sentences after power up, at 9600 8N1 and TTL levels. The NMEA sentences do not necessarily have to contain valid data immediately, the code will simply discard empty sentences. The point is that such a module doesn't require any startup code.

5.2 Input device

This is basically complete unless a touch wheel gets added later and/or a real PCB is designed.

Amount	Name	Description
1	PJRC Teensy 2.0	Atmel ATmega32U4 breakout board
1	MCP73832-2ACI-OT	LiPo battery charger chip, SOT-23
1	IRLML6401	P-Channel MOSFET, SOT-23
2	SOT23 breakout boards	
1	SB 120	low leakage, small voltage drop schottky diode
1	10 kΩ resistor	Used to set charging current
2	4.7 µF Tantalum cap	
1	1 µF Tantalum cap	
1	Pololu # 2119	5V step up/down regulator
5	Push button	whatever type you like
1	PCF8574A	I^2CI/O expander
1	100 nF	decoupling for above IC
2	4.7 kΩ resistor	I^2Cpull-ups
1	Slider switch	For on/off
1	Enclosure	Comfortable to hold
1	Piece of perfboard	
1	Piece of 4-wire cable	Connection to main device
1	3.7 V LiPo or LiIon battery	To power the whole thing

How long the device operates obvisouly depends on the battery's capacity. However, what battery size fits into the enclosure depends on the enclosure and the available battery types and the PCB layout (even if it's a perfboard). So simply pick the largest one that fits.

5.3 Azimuth sensor

Very preliminary, but it will surely contain the optical encoder, some mechanics, and a microcontroller. So let's make an educated guess.

Optics and mechanics:

Amount	Name	Description
1	Shaft	
2	Ball bearing	
1	friction wheel	
1	Housing	for the above parts
1	Spring	
a few	other things	we'll see

The plan is to use the spring to pull the housing (which holds the bearings, shaft and wheel) against the telescope's round base plate.

Electronics:

Amount	Name	Description
1	Avago HEDS-5540 H06	Optical encoder
1	ATtiny85	A small microcontroller
1	100 nF	decoupling cap
1	Piece of 4-wire cable	Connection to main device