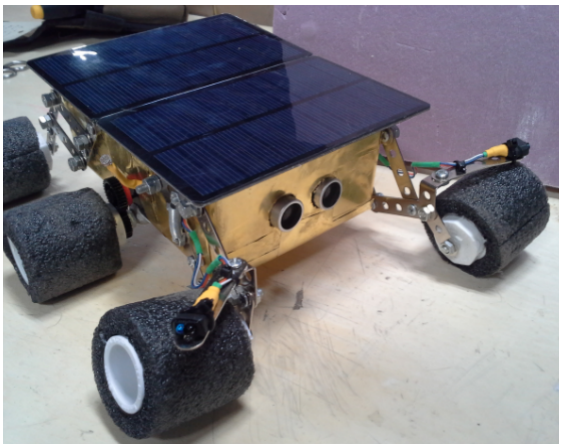
Sojourner Model

Overview:

This device consists of two components: the Rover, and the Controller. The Rover is a shared-control autonomous or telerobotic rover. The Controller is a combination communications relay/remote control.

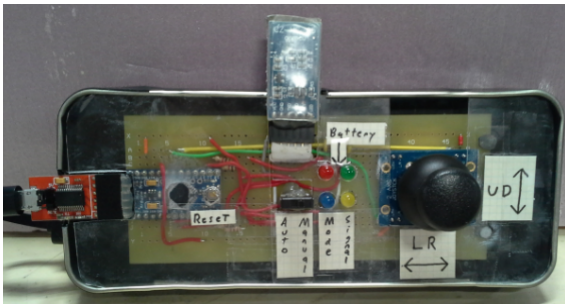
I. Rover

The rover is designed as a model of the Sojourner Mars rover, including fully functional roof solar panels, 6 wheels and suspension linkages (of which two provide drive power), and autonomous and manual control modes. The rover is powered by a 1000 mAh lithium polymer battery, which can be charged by either the solar panels or a USB connection, and navigates with the use of an ultrasonic distance sensor and two infrared distance sensors. Environmental data is collected from a combined temperature and humidity probe, as well as a light level sensor.



II. Controller

The Controller gives the user manual control of the rover (when in that mode) via a serial Bluetooth connection. Drive control is provided by a 2-axis joystick, and mode control via a slide switch. Four indicators supply the user with the status of the control mode, signal activity, and rover battery state. The controller also includes a USB-to-Serial connection for collection of data reported from the rover to a computer.



III. Software

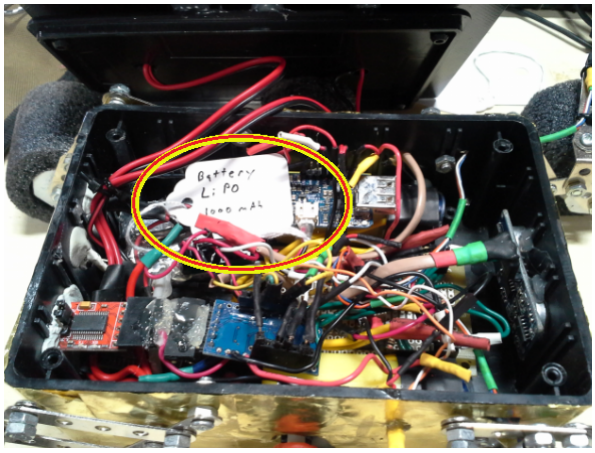
The rover implements two operational modes, manual and autonomous control. Manual control translates signals from the Controller to movement states. Autonomous control is a relatively simple avoidance and exploration routine relying on the navigational sensors to travel freely. An additonal mode, activating when the connection to the controller is lost, attempts to turn the rover to return to signal range.

Activation and Use Guide:

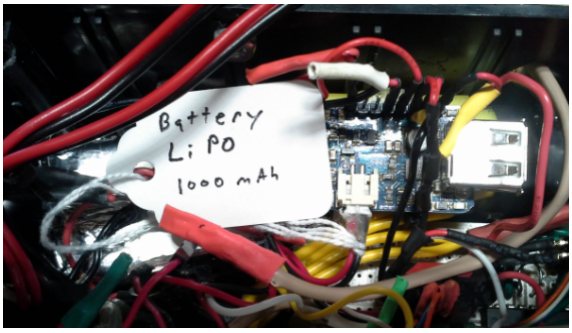
Installing the Battery:

The Rover will be shipped with its internal battery disconnected. To re-connect the battery, the solar panels should be lifted to gain access to the electronics compartment. The power switch on the back side of the rover should be set to 'Off', but the rover boots into a safety mode as well, so it won't try to take off driving if the battery is connected when 'On'.

Within the compartment, the battery is placed in its storage location, and the connecter port is tagged with a label, to allow ease of location:



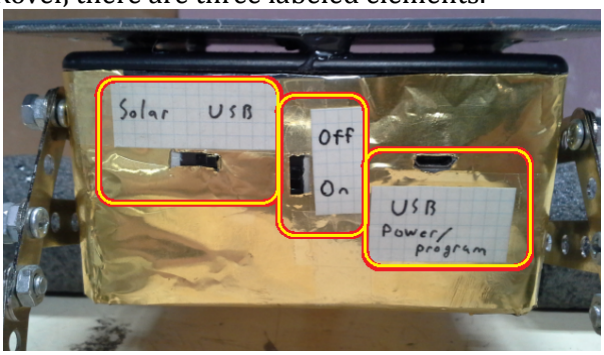
The battery plug must be connected to the power controller, as illustrated below:



The plug will only fit in the correct orientation. Once installed, the Rover can be powered on. It is recommended that the Rover be stored with the battery disconnected, to avoid leakage drains on the cell.

Rover Control Panel:

On the back panel of the Rover, there are three labeled elements.



I. Solar/USB Charge Switch

The leftmost switch controls whether the rover is set to charge its battery via the solar panels or USB connection. Solar charging rates will naturally depend on amount and quality of light. USB charging is significantly more efficient.

II. Power Switch

The middle switch is the on/off power switch for the rover. The rover must be 'On' for programming, and boots to a safety mode, which persists until the connection link between the Rover and the Controller is established. This enables more simple programming of the Rover, as well as keeping the it from driving off during initilization.

III. USB port

The rightmost item is the USB port. This serves as both the programming connection for the Rover microcontroller, and the power port for the USB charging mode.

Use of Controller:

I. Activating the Controller

The Controller is activated by pressing the power button on the battery pack on the reverse side of the device:

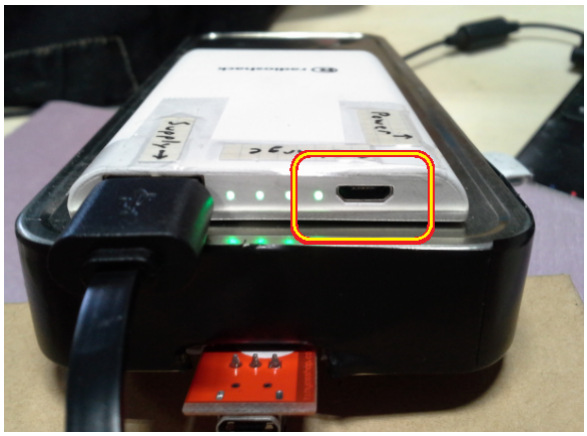
The Rover and the Controller can be independently activated at any point. The Controller makes automated attempts to connect to the rover every 5 seconds when not connected. Indicator LEDs will indicate at any point in time the connection status. Bluetooth connectivity is preset within the modules, and they will dynamically manage the connection, without further user input.

II. Power Supply Port

On the Controller, power for portable operation may be provided from the attached battery pack via the labeled supply port and the short USB cable. This cable connects to the USB-to-Serial connector on the Controller, which is also where the unit derives power when connected to a computer, though with the included long cable. The short cable is not usable as a data connection, and so should not be used for computer connectivity, except in emergency situations where battery power is unavailable.

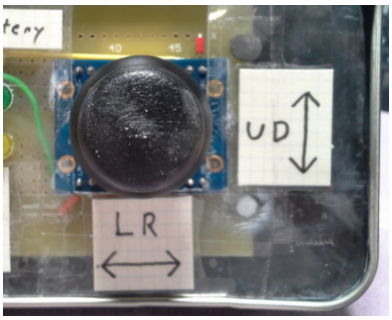
III. Charge Port

The battery pack charge port is also labeled, and can be charged by via the included USB cable. The power level is indicated between the two USB ports, as shown below:



IV. Joystick

The joystick located on the right side of the Controller allows for drive actuation of the Rover when in manual control mode. There are 8 movement modes directed by the two axis. The UD direction controls direct forward/reverse movement, and the LR direction controls skid-steering based rotation. Direct movement along one axis leads to either forward or reverse movement, or clockwise or counter-clockwise rotation. Diagonal movements along both Axis result in circular movement in the corresponding directions:



The center position, naturally, corresponds to a non-moving state.

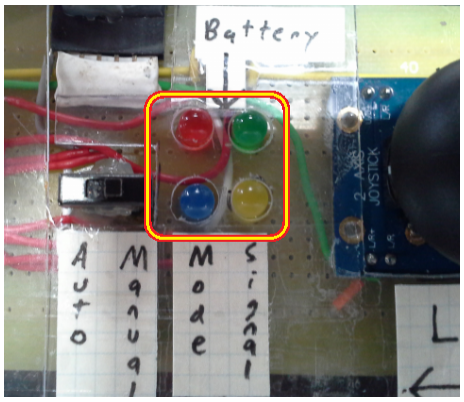
V. Mode Selection Switch

In the center of the Controller is the mode selection switch. Oriented to the left, the switch sets the Controller to the autonomous control mode, which it then transmits as part of the control data packet to the rover. To the right, it sets the manual control mode. Generally, it's preferrable to power on the Controller in the manual mode, as otherwise upon connection, the Rover will engage immediately in the autonomous navigation routine, which can be inconvenient. However, there are no technical contraindications from doing so.

VI. Indicators

The controller bears 4 primary LED indicators, with the LEDs mounted on the Bluetooth module and the microcontroller acting as ancilliary signals. The Bluetooth LED blinking rapidly indicates unconnected, unpaired status, and longer double pulses indicate that the connection has been made. The LED on the microcontroller indicates power only.

The four primary indicators illustrate the current control mode state, in blue on the bottom left corner of the four; the signal connectivity status in yellow on the bottom right; and battery status on top, with red indicating the Rover battery is low, and green indicating sufficient operating capacity thereof.

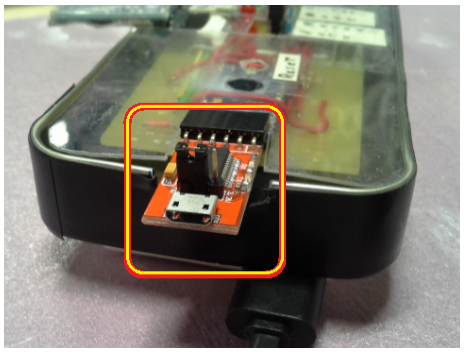


VII. Bluetooth Module

The Bluetooth module is the element enabling wireless communication with the Rover. It provides a serial connection via pins 2 and 3 on the Controller microcontroller. It has been configured to connect automatically to the module within the Rover, and needs no direct, non-software action on the user's part in standard operation.

VIII. Computer Connection

The USB port on the left side of the controller enables the connection of the Controller to a computer. When receiving data from the Rover, the Controller also passes that packet out through the serial port, and that data can be collected with a program running on a computer, or observed and copied through any serial port monitoring utility. The Arduino software (included with software package) which is used to program the microcontrollers includes such a monitor, though other clients (such as PuTTY on windows) may be used to observe and record this data.



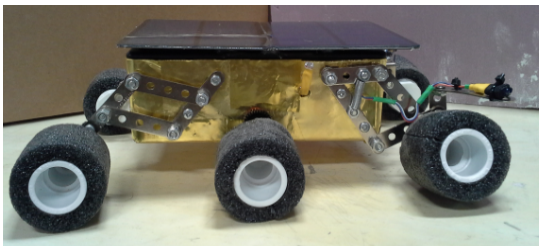
IX. Microcontroller

The central processor for the Controller is an Arduino Pro Mini. Connecting via USB allows for reprogramming of this controller with new software. Should reprogramming be a desire of the user, the original code is included with this guide to facilitate functional work, as well as return the controller to the original software package, should the need arise.

Rover Hardware:

I. Drive & Wheel Linkages

The Rover has 6 wheels. The front and rear pairs have suspension linkages, the front pair using springs, and the read pair using only weight. The middle row of two wheels lacks suspension hardware and instead has a short gear train connecting them to the drive motors. The non-drive wheels are inset loosely, allowing more free movement so that the robot can utilize skid steering from the speeds of the two drive wheels, without a need to individually articulate each wheel. This arrangement balances traction and stability at minimum of expense.



II. Motor Controller

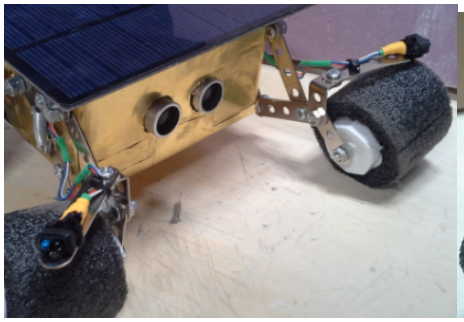
The drive motors are controlled by quad H-bridge IC, the SN754410. This chip, for each motor, has two control signal lines- one corresponding to clockwise rotation and one to counter-clockwise rotation. Each line for each motor is connected to a digital pin on the microcontroller. When one pin is high and the other low, the motor turns in the specified direction. To set speed, the active control pin is pulsed with duty cycle of 0-100% PWM to vary power supplied to the motors. In software, the scaling of this duty cycle has been equalized between motors.

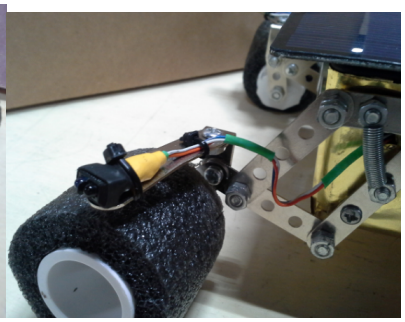
The left motor control lines are pins 9 and 10, and th right motor control pins are 5 and 6. Operation of the motors is encapsulated in software with the motorSet function.

III. Navigation Sensors

For autonomous navigation, the Rover is equipped with a pair of IR distance sensors, and an ultrasonic rangefinder. The IR sensors are mounted above the front two wheels to assist with cases in which the wheel bumps an obstacle, and the ultrasonic sensor is mounted on the front of the rover for broad obstacle detection.

The left IR sensor is connected to pin A0, and the right to pin A1. The ultrasonic sensor's data line is on pin 4.





IV. Environmental Sensors

As Dr. Zubrin says, 'there's no point in going anywhere if you don't do something when you arrive', so the Rover is equipped with a token selection of evironmental sensors. On the left side of the casing is a DHT22 temperature and humidity sensor; on the right side, a resistive light level sensor. The readings from both these sensors, along with the battery charge state, are the data which is transmitted from the Rover to the Controller.

The DHT22 is connected to the microcontroller on pin 12, and the light level sensor is on pin A3.

V. Central Processor

All activity of the rover is controlled by one central microcontroller, which is (just like for the Controller) an Arduino Pro Mini. As with the Controller, the software for the Rover is included with this packet. Using the USB port on the reverse side of the rover, it is possible to reprogram the device, as with the Controller. This can be accomplished using the Arduino IDE software (Included with software Package) and the included USB cable.

VI. Power Controller

Power management is provided using an off-the-shelf regulator/LiPO charger combination: the Adafruit Powerboost 500C. This module allows for simple implementation of the split solar/USB charge circuit, provides proper charging for the battery, and a stiff 5V regulated output capable of sustained output at up to 1A. Additionally, the module contains an enable control line, which provides the power switch functionality, and a low battery indicator which is read by the microcontroller to report when the battery is in need of charging.

VII. USB/Serial converter

Device programming access is granted through a USB-to-Serial converter, via the port on the back side of the device. Programming and USB charging both route through this module.

VIII. Solar Panels

The top of the Rover is covered by a pair of 6V nominal output, 1.5W solar panels, for a total solar power production potential of 3W. The panels are integrated with the charging circuit, and can provide either supplemental power or charging. Solar charging is approximately 1/6 as efficient as USB charging, presuming good lighting.

Controller Software:

The software on the controller unit serves two functions- managing the data flow between Controller, Rover and, optionally, computer, and interpreting control and status signals for the manual control mode.

The format of a data packet transmitted from the controller is:

```
& LR value, UD value, control mode $
```

The & and $ symbols indicate the start and end of a complete packet, respectively.

I. Control Signals

The Controller reads in two inputs for control- the joystick, which controls the motion of the Rover in the manual mode, and the mode select switch, which determines whether the Rover is in autonomous navigation mode or remote controlled mode. In both cases, the Controller and Rover regularly transmit data packages to one another. The packets from the Controller contain the position readings from the joystick (which the rover converts to movements) and the control mode selection.

II. Data Signals

The Controller also reads in, whenever a packet is observed arriving from the Rover, the sensor readings and Rover state variables. It stores these reading and later transmits them over the non-Bluetooth serial port, allowing collection of the data on a computer via serial monitor or other connection method (such as a script running on the host machine). Of this data, only the battery state has a direct impact on the user-side operation of the Controller, as it is indicated on the green and red LEDs on the controller indicator panel.

III. Connection Status

The Controller also keeps a timer which measures the time since last receipt of a packet of data from the rover. Should this duration exceed 5 seconds, the Controller notes a loss of signal and thereafter attempts to reconnect with the Rover every 5 seconds.

The signal connectivity is indicated by the yellow LED.

Rover Software:

The Rover software can be broken up into three main components: the manual control segment, the autonomous navigation segment, and the utility activity segment.

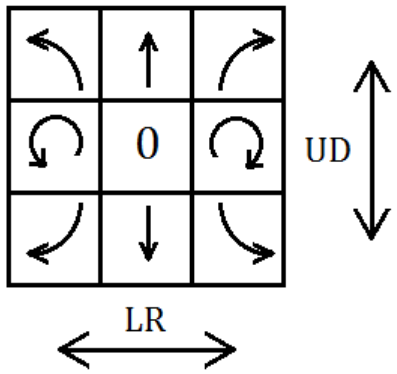
The manual control and autonomous navigation portions are both included within a selecting 'if' block which is governed by the control mode flag read from the controller transmission. Within the autonomous section, a switch/case governed state machine manages the operation of the avoidance algorithm. This mode can either be switched by the controller, or when the Rover loses contact with the Controller. When loss of signal occurs, the Rover it enters autonomous mode with a special 'return' function to attempt to return to signal range before transferring to the usual exploration routine.

Within the utility segment, outside the control portion of the software, the rover takes nagvigation and sensor readings, manages the transmission of data to and from the Controller, and checks connectivity to same. When the Rover boots, it starts in a neutral holding state where it executes neither control routine, in anticipation of receipt of signal from the controller, switching to the mode set thereupon when contact is made.

I. Manual control

The manual control mode is much simpler than the autonomous mode, consisting of only a logical transform which converts the readings from the joystick into motor control signals, using the controlToSpeed and setMotors functions.

There are 9 control profiles, corresponding to the 8 general positions of the joystick and the central, neutral positon. Cardinal directions either move the Rover in a forward/reverse direction (for UD only movement of the stick) or an in-place turn (for LR only movement). Joystick positions along diagonals of the LR and UD axis instruct the Rover to make arcing turns combining forward and turn movement:



II. Sensor Readings

The Rover has two different sensor groups; navigational sensors and environmental sensors.

The navigational sensor group includes the IR sensors and the ultrasonic sensor. These sensors are read from during the autonomous navigation routine, immediately prior to execution of the state machine code. The function getIR, which reads the IR sensors, prepares two sets of readings- one which is a digital thresholded value, and one which is an analog scaled value. Both are immediately available, as they are convenient in different circumstances throughout the code.

The environmental sensors are the DHT22 Temperature and Humidity sensor, and the photoresistive light sensor. Data is collected from these modules outside of the control blocks, based on a timer measurement, which also determines when data packets are transmitted to the controller.

Additionally, the battery, though not a proper sensor, is read from and reported within the same segments as the other sensors.

III. Communication

Communication is handled, as with the Controller, over a serial connection to the Bluetooth module via pins 2 and 3. Control readings from the Controller are taken in whenever there is data detected in the serial buffer, and outbound transmissions are made whenever a sensor data packet is collected. The Controller is constantly broadcasting all control signals, even when the Rover is in autonomous mode through the Rover naturally does not make use of those signals when not in manual mode. This allows for uniform data packet size, simplifying the communication protocol greatly, and increasing robustness.

The receipt of a packet also sets a connection timer as in the Controller, such that after 5 seconds with no data received, a loss of signal is indicated. When this occurs, the Rover is triggered to enter a special return-contingency mode, which attempts to set the Rover on a course that will re-enter signal range.

Operation of data transmission is controlled within a separate operation block from control algorithms, using an independent timer. This allows for independent contact and operation of the Controller and Rover transmissions. The format of a packet from the Rover is as follows:

```
& temperature, humidity, light, battery, autonomous state$
```

As for the Controller, the & and $ symbols represent start and end of packet transmission.

IV. Manual control

The manual control mode, given that the control data reception acts in parallel with the control blocks, is very simple. It consists entirely of a series of logical operations to convert analog readings from the Controller joystick into motor control signals in the form of -100% to 100% speed values.

These logic operations are created as a compact representation of the illustrated control protocol above.

V. Autonomous Navigation

The autonomous control segment is significantly more complex than the manual control section. It is operated as a state machine which is activated when the Controller indicates a switch from manual to automatic mode, or when communication with the Controller is lost. Autonomous navigation thus implements two features- explore and avoid behaviors relying on the distance sensor readings; and a short routine which, on loss of signal, attempts to turn the Rover 180 degrees.

The assumption for the latter behavior is that the only loss of signal event the Rover can make a sensible navigational response to is one wherein it has traveled outside communication range. Therefor, turning 180 degrees should roughly put it on a relative course to return to signal range, given the lack of information about the operating environment. Once the turn is completed, the rover switches to normal explore and avoid behaviors.

The default autonomous state is forward movement. In this base state, there is a continuous check of the navigation sensors for detection of obstacles. When detecting an obstacle with any sensor, the state machine progresses to an intermediary state which selects an avoidance response based on which sensors observe the obstacle, as indicated below:

1. Obstacle seen by ultrasonic only:
   Avoidance Routine: Direct reverse, followed by timed turn
2. Obstacle seen by ultrasonic and left IR:
   Avoidance Routine: Rightwards reverse
3. Obstacle seen by ultrasonic and right IR:
   Avoidance Routine: Leftwards reverse
4. Obstacle seen by all sensors:
   Avoidance Routine: Direct reverse, followed by timed turn
5. Obstacle seen by left IR only:
   Avoidance Routine: Direct reverse, followed by timed turn
6. Obstacle seen by right IR only:
   Avoidance Routine: Direct reverse, followed by timed turn
7. Obstacle seen by left and right IR but not ultrasonic:
   Avoidance Routine: Direct reverse, followed by timed turn
8. Obstacle seen by no sensor:
   This is a vacuous state to cover all 8 potentialities, nominally causing a return to the defult state, but not actually a valid obstacle case.

Additionally, all avoidance maneuvers include a timer, which runs until the maneuver is completed, meaning the absence of a detected obstacle. If this timer exceeds 5 seconds, the Rover performs a timed turn in place, based on the assumption that since the maneuver is not working, the Rover may be stuck. This stop-gap measure enables the Rover to regularly free itself from awkward positions, with a little fidgeting.