Night Rover

Autonomous Solar Harvesting Planetary Rover

David Esposito, Farzon Lotfi, Kevin Reilly, John Richardson, Roberto Pereira, Dr. Jay Summet

4/23/2012

The Intel Cornell Cup will provide a platform on which to perfect engineering designs and computing algorithms for the GT Night Rover. The GT Night Rover aims to store and utilize electrical and/or thermal energy efficiently while investigating systems for prolonging the useful mission life of a robotic planetary rover (planetary in the general sense). The final prototype will be an autonomous rover that locates sources of solar energy and continues moving through a full day/night cycle. This challenge will serve as a proof of concept for a more robust system and provide a platform for the design and construction of a rover that will survive in and provide persistent functionality over multiple day/night cycles while moving and collecting useful information. The use of the Intel Atom board will enable the implementation of more complex programming control algorithms while allowing for a clear margin of power usage in future versions of the rover, exploring novel capabilities of the Atom board and pushing the boundaries of engineering design.

Table of Contents

Introduction

Challenge Definition

The Night Rover is the initial phase for pursuing the NASA Night Rover Centennial Challenge which is to develop an autonomous robotic vehicle that can travel continuously through three Earth day/night cycles. The motivation for this competition is to improve low-cost means of implementing solar power for rovers which must survive long periods of darkness, in which they must rely on batteries, fuel cells, or complete shutdown to survive. NASA has often lost rovers through failure to restart after shutdown, and better operation of solar power storage would allow increased productivity of these lower-cost vehicles.

Missions to the moon present several problems for solar power. Due to the harmonic orbit of the Moon, it experiences a day/night cycle in excess of 28 Earth days. The period of darkness lasting over 14 Earth-days is an extremely different environment than any NASA rover has experienced on Earth or Mars - each having just over 12 hours of darkness per day.

In response to this problem, NASA has announced a Centennial Challenge open to the general public from Spring 2012 to Spring 2013 to investigate possible solutions to the problem of the Moon's extended darkness. The NASA Night Rover competition encourages industry members, research scientists, and students to build mobile systems capable of harvesting and storing energy for extended use during day/night cycles. The winning entry in the NASA challenge is measured by the maximum distance traveled over a 3 day period, while success for the Intel Cornell Cup will consist of efficient and full operation over a single day/night cycle.

Radiation hardening and power restrictions have limited NASA physically and economically in terms of computational power. Previous computational units sent into space provided up to 128MB of memory paired with a 22 MHz processor using as little as 10W. To encourage development of new computational systems in spacecraft, NASA has recently - in the past two decades - been sponsoring the use of conventional computers and microcontrollers in nanosatellites, CubeSats, and other small-scale space innovations. The computational specifications for previous NASA missions are small compared to 1GB of onboard memory, a 1.0 GHz clock speed and an average power usage of 3.3W provided by Intel's Atom board. The Intel Cornell Cup submission will be a working model with the experimental peripherals for navigation and light sensing as well as the computationally complex algorithms allowed by the Atom architecture. The power consumption of the peripherals and the atom board will be used as an upper bound while the GT Night Rover team prepares to scale its design for full competition in the NASA Night Rover Centennial Challenge.

Testing for the Intel Cornell Cup will take place on the top level of a parking deck. The area will be level and enclosed with solid walls. This will provide the largest level, low traffic, well defined open area with minimal shadows within city limits. Initial testing will consist of a 24 hours period starting at sundown, allowing the batteries to be discharged and continuing to the following sundown, allowing the batteries to be fully charged during sunlight. The rover will be monitored by team members throughout the entire experiment with minimal human interaction only when necessary.

Proposed Solution

General Overview

Team GT Night Rover will produce:

1. A functional robotic vehicle
    a. The rover shall utilize electric Propulsion
    b. The rover shall navigate using Lidar
    c. The rover structure shall be low mass
2. A system to efficiently collect and store solar power
3. An autonomous control system for the robotic vehicle

The intention of this project is to demonstrate a working model of solar energy collection and storage for autonomous vehicles that shall operate continuously over one day/night cycle. These functionalities will be exhibited in videos of full experiments and brief demonstrations of the working system seeking stronger sources of light when shaded from direct sunlight.

Solution Overview

Software Systems

This section will provide software development environments, software architecture as well as an overview of general implementation details.

Software Overview

Software functionality is simplified into low level functionality, high level functionality using a custom serial protocol for communication, debugging, and monitoring functionality using BSD sockets with an android front end and a web server with a javascript front end. Java was chosen to implement navigation and path planning for its portability and fast object oriented prototyping. Built-in Java libraries along with the RXTX library for communication provided a fast paced development environment for the high level functionality.

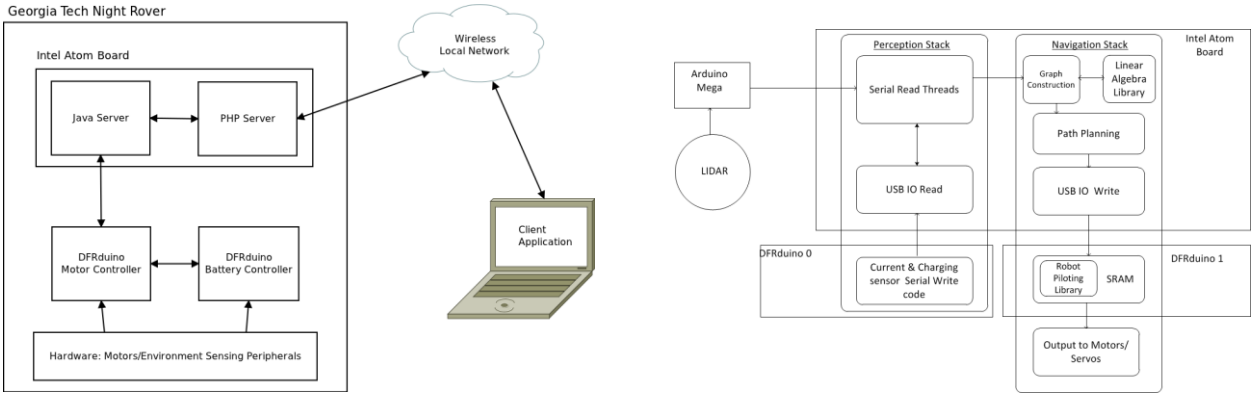


Figure 1. Diagram of the software overview, followed by a diagram of the stack with overlay on top of the hardware it overlaps.

Three Arduino compatible microcontrollers were used to interface lower level hardware. An Arduino Mega was used to interface a lidar module, while two DFRduino were used to measure analog inputs from environmental sensors as well as to control the motors for locomotion and lidar rotation.

Hardware Setup

As in the proposal, the Intel Atom board remains the primary computer for this project although cameras were not included in our final solution, thus the computational tasks required of the board no longer include computer vision algorithms. A discussion of why this was decided can be found in the "Environment Sensing" subsection of electrical design decisions. The atom board does receive and send data to microcontrollers in charge of lidar, IR, current sensing, and motor control. These microcontrollers execute tasks handed down to them by the Atom board and report results back to the board. Thus the Atom board still acts as the main computation force constructing graphs of its environment and path planning on said graphs before sending commands down to the microcontrollers.

To develop on the Night Rover the atom board startup process needed to be changed. This resulted in hardware as well as software changes. On the hardware side, a replacement for the spinning disk hard drive was needed. This was done because the nature of a moving rover makes constraints such as power consumption, shock resistance, and heat dispersion problems hard to handle. These problems offset the advantage of higher disk capacity. The first problem of mentioned of power consumption is a constraint that can't be engineered around such as heat control and shock resistance. As the night rover intends to path plan and then sleep the atom board, the atom board causes an increase of "spin up". Spin up is the process of spinning the drive up to either read or write data faster than the drive can at its current speed. Increasing the spinning speed results in more power directed to the the hard drive motor which increases overall power consumption. There are solutions to the spin up when using linux, such as calling bdflush and kupdate kernels but these solutions go beyond the scope of this project as it would need to be known when to use them. There are also utilities that can be used such as hd-idle, but that is intended for external hard drives. Second is the problem of shock control. Laptops today have active hard drive protection that parks disk heads if a machine is dropped, these laptops have a way of detecting when a laptop is about to be dropped to park these drives. There have also been solutions of

using hard drives in robotic applications that make use of a built-in accelerometers in the drives but not all hard drive possess this accelerometer[5]. Even if a disk with a built-in accelerometer was available, the night rover is using a server OS typically found in desktops which does not contain the required systems to interface with the accelerometer. Implementing a detection system ourselves would have diverted resources away from more critical code and hardware design issues, so a detection system was not implemented. Other than the problem of drive drops is the problem of disk skips which result from just moving a hard drive. disk skips cause computation to run slower, which is another reason a replacement for a spinning device was considered. Third, Spinning hard drives have a problem of adding heat to their environment, especially during spin up[6]. Since the electronics of the rover are in a closed body environment and the current models of the intel atom board already suffer from heat related issues, It was decided to look for alternative solutions. Solutions considered were replacing with a solid state drive, an sd card, or a usb drive. The solid state drive solution although they have the benefit of being faster are not more power efficient and are also much more expensive and thus were thrown out of further consideration. The next replacement idea was usb drives. These turn out to be more power efficient than hard drives, but also suffer from the same problem that ssds do of read/write power costs not being significantly greater than idle costs. Therefore if we were to scale equal gigabyte size hd to usb flash, it is possible that there might not be a power consumption decrease, since hard drives can decrease power consumption by spinning down during idle states. As this research is beyond the scope of this project it was not tested. Usb drives were used largely for testing but were not used in the final night rover product, because of the limited number of usb ports available for wifi, and the microcontrollers. an SD card was used instead. A problem the Night rover team ran into was high iowait times if the sd or usb drives had read/write speeds under 22 mb/s but this was averted by using above said speed.

Figure 2. SD card used to boot from.

OS installation & software setup:

The software systems started with modifications to the atom board startup process. This was done to replace the UEFI with a Bios. Using the instructions from the *BLDK Inforce User Guide v0.3_1* a legacy bios was set up. Installing ubuntu server was done using the Universal USB Installer 1.8.7.9. This

turns a usb thumb drive in a bootable drive capable of installing ubuntu server onto an sd card. The next step is to Set boot priority in the bios to usb drive. Now Ubuntu server is ready for installation. After seeing the bios splash screen the next screen will be the ubuntu installation screen. After ubuntu finishes its installation, update the software, A usb networking device might be needed. 3rd party software the atom board will need is sensors, mpstat, collectd, Java, ssh, & twisted. All of this software will be used to monitor code performance on the atom board. Sensors provides temperature information for different parts of the board, mpstat provides cpu, memory, and file io usage information. collectd is a service that gathers performance information of different parts of the board and stores them as RRD files, which is an open source data logging format. Javascript graphing user interface is then used to parse the data and display using a lightweight python based web server known as Twistd. The Java runtime environment is needed to run the night rover graph construction, path planning, serial communication, and networking code. SSH was used to directly connect to the Atom board and setup the development environment and to make changes to the software setup as needed. Detailed Instructions on how to install the bios, OS, and development environment can be found in the technical documentation below.

Software Monitoring

Robots offer a special problem in which one does not have direct access to it. Thus to come up with a solution to this problem web and socket programing were used to provide remote monitoring of the atom board and its microcontrollers. Monitoring allows for real-time visualization of computing and communication load on the Atom board and Microcontrollers. Monitoring also provides the ability to learn from the data. For example by monitoring CPU utilization dynamic power savings can be achieved. Data logged from the rover can also be profiled to see what the most hardware intensive software is. This allows for code changes or better partitioning of code to reduce computing load. One of the front ends for the network protocol the GTNR team wrote is an Android application. This android application interfaces with a multithreaded server to display lidar information, email users atom board cpu usage and temperature information. Another frontend is a web front built using php and javascript.

Figure 3. Atom performance monitoring software and Android frontend displaying lidar data respectively.

Software Communication

To communicate with the Arduino microcontrollers a serial communication protocol had to be written.

Sync byte	Op Code	Imm16	Device Byte
Byte 1	Byte 2	Byte 3-4	Byte 5

TABLE 1. Protocol format by byte

The Opcodes are FWD,REV,RHT, LFT, STOP, REQ, LOG, and OFF. FWD takes a distance value for its immediate 16 bits. After path planning the atom board would send this command to the motor controller to move the rover forward by the distance specified. REV moves the rover backwards, RHT turns the rover to the right by the degrees specified in the imm16 bits. The LFT command turns the rover to the left by the degrees specified. For readability the opcode is represented by the ascii values of F,B,R,L,S,Q, & O for opcodes FWD, REV, RHT.LFT.STOP,REQ, LOG, & OFF respectively.

The device byte will specify which device is to be written to. This byte will either be the ascii character ‘P’ for the the power microcontroller or ‘M’ For the motor controller microcontroller. The Sync byte is always -1 or 0xFF. This is used to prevent reading corrupted data, which could occur if the atom board and the microcontrollers get out of sync. software waits and serial write/read availability commands are used to prevent the occurrence of this problem, but sending an extra byte is an inexpensive redundancy step that can guarantee correctness of implementation. Detailed specification of the above mentioned operations and the byte array format can be found in the appendix.

Software Testing

The peripheral testing has largely been done by placing obstacles in front of peripherals and seeing if the peripherals properly detect them. Other testing done were to give dummy data with expected node expansions and seeing if those nodes expand. Junits tests were done to test for bugs in

the graphing algorithm. To test serial communication, microcontrollers were setup to read byte arrays and then write them back. The sent array and the received array were then compared for consistency.

Software implementations

Navigation and Sensing

The main peripherals used for navigation and obstacle avoidance for the night rover are lidar and infrared. These were chosen over our initial approach to the problem of using a kinect. A discussion on why the night rover team came to this decision can be found in “Environment Sensing” subsection of electrical design decisions. Cost concerns highlighted in the initial proposal also kept the team from using a bird’s eye view approach to solving the navigation problem with satellites. Algorithms that could have been used such as an approximate cell decomposition or a visibility graph were scrap in favor of algorithms that used onboard peripherals. Thus the path planning has simplified to be solely on local data from the rover.

Graph construction

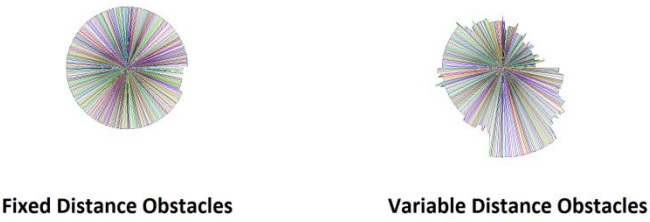


Figure 4. Example lidar output.

The night rover team considered many path planning algorithms. Eventually the night rover team plans on implementing a roadmap design, i.e. a design that will remember where it has been and construct maps based on where it has explored. As of now the algorithm is simpler, but is modularized enough that the night rover team could implement a roadmap design in the next phase of our project. The current algorithm takes the live depth reading of data from the lidar module and constructs 360 points from that using the indices of the 360 array as a degree values, It is possible to convert the depth readings into a series of reachable points (see code samples in the included folders). The next step in constructing a graph of the current state of the environment was to construct the edges which is done by treating the rover as the center point and constructing edges between the rover and the points constructed in the previous step. The final step was to construct edges between each of the 360 points This step is not currently necessary as straightest paths to a point are weighted higher by the path planning algorithm.

Path Planning

The path planning algorithm utilized first searches for the farthest depth point from the rover's current position. In finding this point it weighs points directly in front of it with three tiers of weights. The night rover team makes the assumption that the front of the rover is at 90 degrees. Thus the first tier and the highest tier of weights is applied to points in the range of 70 to 110 degrees. The second tier of weights is applied to the points between 45 to 135 degrees. The third and final tier of weights is applied to points between 0-180 degrees. the final 181-359 degrees are not weighted, The tiered weights overlap each other an increase by weights associated to their degree rather than any arbitrary weighting system. The next step is to convert the points into a series of byte arrays that can be sent to the Arduino. These byte array convert an edge into a turning instruction, either right or left and then a movement instruction, either forward or backward. A communication thread is then called and the byte arrays are sent off to the Arduinos. Rotary controllers on the wheels are used to measure how accurate the rover completed its task.

Microcontroller

Low level software was implemented using Arduino (and compatible) microcontrollers for motor control, power management and reading environment sensors. This functionality is implemented with two DFRduino microcontrollers (Arduino Duemilonova clone with on board motor drivers) and an Arduino Mega. The drive motors and lidar motors will be controlled by the DFRduinos' onboard motor drivers. This reduces the complexity of supporting circuitry required. Each DFRduino has two motors drivers which allows each Rocker-Bogie to be driven independently, enabling turning and another DFRduino to control speed of the lidar for ensuring maximum distance reading accuracy. In addition to the multiple boards needed to drive the motors, a total of ten analog values are read and logged on the atom board. Since each board supports eight analog inputs two boards are required. The lidar module supplies a constant stream of bytes at 115200 bits per second. This baud rate is far too fast for communication on a simulated serial port and requires dedicated hardware to communicate at that speed. For this purpose, the Mega was introduced since it provides four serial ports with dedicated hardware. Therefore the Mega is used to forward serial data to the Atom board for testing. The data cannot be parsed on the Atom Boards because the high rate or transmission does not allow enough computation time for smoothing of data, calculations and replies to requests for distances.

The two motors controllers functionality is split into two general responsibilities: navigation related and power related. The motor controller handles real time motor control, collision detection and light intensity measurements providing the atom board with all needed information for intelligent navigation. The battery controller logs information useful to solar harvesting, power utilization efficiency and device protection values such as temperature and current draw. The measurements from this controller provide information to the programmer and developer for further development and improvement on current algorithms and design.

Software Performance Measures

Performance in the field is planned to be performed by next weekend or April 27th. As of the time of this report all performance measures for have been performed on individual subsystems. Thus a more detailed an applicable discussion of performance evaluations will be discussed at the presentation.

The monitoring software, mentioned in the "Software Monitoring" subsection above allowed for the testing of the accuracy of the lidar and the computing load on the atom board. In addition the graphing library accuracy has been verified with Junit tests as well as well as visually verified by drawing the graph out on JPanels. Refer to the "Software Testing" subsection of software systems for further information.

Technical Documentation

Bios

Material needed:

A usb to serial converter
A NULL Modem serial cable (**very important that it is a null modem)**
Putty - http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

Follow steps 5.1 - 5.1.6 BLDK Inforce User Guide v03_1.docx
In the same document follow steps 6.3.1 -6.3.11. If all goes the splash screen below will be displayed.

Figure 5. *Bios splash screen displayed on first boot*

OS

Material needed:

- 1 usb flash device to install on.
- 1 usb flash to install from or a blank cd and an external cd drive
- 1 atom board

Download the ubuntu server iso from http://www.ubuntu.com/download/server/download
if in Linux or OS X
use the default cd burner application and burn the iso or follow the usb boot instructions found here:
https://help.ubuntu.com/community/Installation/FromUSBStick

If in windows

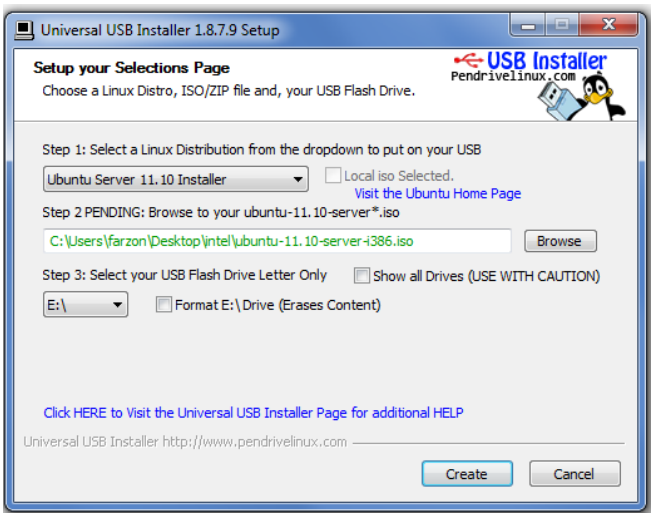


Figure 6. Universal USB installer with ubuntu server settings initialized..
and usb install selected

- Download the universal usb installer from:

http://www.pendrivelinux.com/universal-usb-installer-easy-as-1-2-3/

- Select ubuntu server from the linux distribution list
- Browse to the location of the iso, if found text will turn green, if not found the option to download will be presented
- select the install media. BE CAREFUL not to overwrite the wrong drive.
- Click create. A warning box will appear to make sure you want to proceed. click yes.
- Now wait for the program to finish

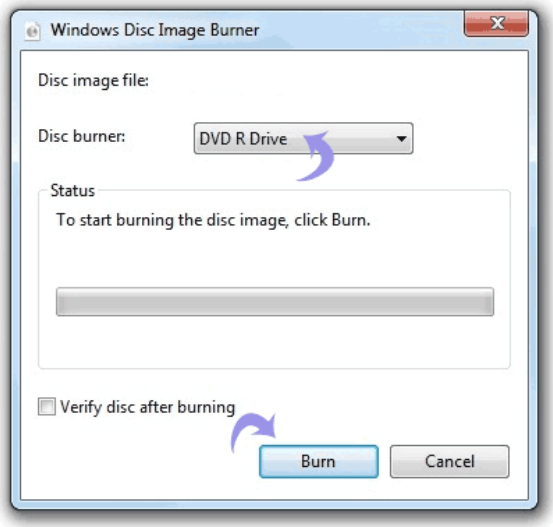


Figure 7. windows ISO image burning software.
and CD install selected

- Right click on ISO image file on computer and then click “Burn disc image”
- Windows Disc Image Burner will popup, select your burner drive from drop down menu.
- Click Burn button to start the burn process.

- Optionally click verify disc after burning to verify data is written correctly

Install Ubuntu server

Plug the USB or cd drive with cd inside into the Atom board USB port.

If boot up process does not begin reboot and set cd or USB boot in the bios boot drive settings.

Figure 8. Ubuntu installation splash screen.

Install Ubuntu using all default settings

SD card setup

After installation copy the USB installation to an SD card of the equal storage size.

Any of the hard disk cloning software found here can be used to copy to an SD card

http://www.ultimatebootcd.com/

Development Environment

To install collected for data logging, ssh for direct access, twistd for the webserver, type:

sudo apt-get install collectd-core openssh-server
python-twisted-core

To install java type the following bash commands
sudo add-apt-repository ppa:eugenesan/java
sudo apt-get update
sudo apt-get install oracle-java7-installer
if this fails to install download java directly from:
http://www.oracle.com/technetwork/java/javase/downloads/jdk-7u3-download-1501626.html
move the tar.gz file you get to /var/cache/oracle-java7-installer/

if the tar.gz file already exists delete it and replace with the newly downloaded file
then type sudo nano /var/lib/dpkg/info/oracle-java7-installer.postinst
Search for: echo "Downloading..." comment out everything below it until you see echo "Download done." It should look like:

```
echo "Downloading..."
#rm -f $FILENAME
#WGETRC=wgetrc wget $PARTNER_URL
    #|| fp_exit_with_error "download failed"
#rm -f wgetrc
echo "Download done."
```

when done.
Save the file and then type sudo apt-get install oracle-java7-installer again and it will work.
now type sudo apt-get install librxtx-java to get the rxtx serial libraries.

The code written for the project is exported as a runnable jar, thus the next step is to create a bash script that will run said code. add the following command:

java -jar gtnr.jar

to the end of the .bash_profile and the jar will startup at boot time.

Electrical Subsystems

This section will provide information, beginning with a basic overview of electrical components used, relevant equations, then a summary of key design decisions, and finally the process of implementing the complete electrical subsystem.

Overview of the Electrical Subsystem

Functionalities of the electrical subsystems include power management, component protection and sensing the state of surrounding environment to aid in navigation and path planning. This subsystem is composed of an array of analog sensors, (photocells, IC current sensors, long range IR, temperature sensors) and a lidar (Light Detection and Ranging) module from a Neato-XV11 Vacuum. Each sensor requires 5 volts for operation and provides its output to an Arduino.

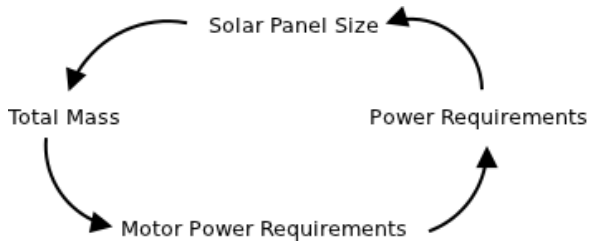


Figure 9. Power budget dependency.

Power budget dependencies are circular such that required motor torque is a function of mass which is a function of power required for electrical systems (solar panel size). Therefore a target mass was selected based on calculations of multiple dc motors in multiple settings, such as varying inclines and varying masses. The details of the measurements can be seen in the mechanical section. Selection of motors along with experimental values of the Atom board and lidar module accounted for 89% of the power budget. This allowed for the selection of an appropriate battery such that the battery would be discharged over the darkness period completely recharged during the daylight period. Given the battery capacity, the size and voltage of the solar panel could be calculated. For a preliminary power budget, the maximal current rating was used in the original calculations.

Part	**Quantity**	**Voltage (V)**
Atom Board	1	12
DFRduino	2	5
Arduino Mega	1	N/A
Lidar Motor	1	3.5
Lidar Serial Module	1	5
General Analog Sensors	8	5
Motors	6	12

Table 2. Electrical components voltage ratings.

Part	**Total Current (mA)**	**Duty Cycle**	**Power (W)**
Atom Board	800	0.1	0.96
DFRduino	20	1	0.1
Arduino Mega	N/A	N/A	N/A
Lidar Motor	70	1	0.245
Lidar Serial Module	50	1	0.25
General Analog Sensors	2	1	0.01
Motors	360	1	4.32
Total	*439mA @ 13.4V*		*5.885W*

Table 3: Electrical components current and power ratings. Note duty cycle of Atom Board (high power state 10% of time) and Arduino Mega draws power directly from Atom Board. The battery nominal

voltage is 13.4V. . Let B_C be the required battery capacity in amp hours, C_{Avg} be the average current draw, and T be time of operation during darkness in hours,

Equation 1

$$B_C = \frac{C_{Avg} T}{c}$$

Equation 2.

$$5.515Ah = \frac{0.439A * 10.2hr}{1 - 0.2}$$

Given a total wattage, the required battery capacity can be calculated. Let *c* be the lower bound battery SOC (0.2 for LiFePo batteries)

Equation 3.

$$S_p = \frac{B_v * B_c + C_{Avg} * T}{a * (24 - T)}$$

Equation 4.

$$18.476W = \frac{13.4V * 5.515Ah + 0.439A * 10.2hr}{.6 * 13.8hr}$$

The battery nominal voltage used in the above calculation is the experimental average while discharging the actual battery unit used. The solar panel wattage requirement can be calculated from this value assuming that the battery SOC will be at 20% and the electrical components will continue to draw power during charging. The solar panel needs to supply enough power to recharge the batteries while supplying enough power to operate the electrical systems. Let S_P be the power of the solar panel and B_V be the battery's nominal voltage, and *a* (~0.6) be constant which adjust for solar power drop due to average angle of the sun relative to the solar panel.

Note that this is required average power required to return the battery SOC to 100% while continuing operation.

Electrical Design Decisions

Power Supplies

To ensure maximum safety during battery discharge and charging and to minimize mass LiFePo batteries were chosen as the method for energy storage. Li-Ion batteries were not considered due to the difficulty of safely and properly charging them in battery packs. According to the above calculations, the power supplies must supply at least 12V and supply at least 0.5A current draw.

The solar panel must be rated to provide at least 18.476 watts. The solar panel could be constructed from solar cells or purchased pre-assembled. A survey of available solar panels rated within the required range showed that solar panel kits were nearly ⅓ the price of preassembled panels. Kits provided the freedom of choosing construction materials in order to minimize mass. A 25W solar panel

kit was chosen to compensate for lost power in the often hazy and cloudy weather conditions within city limits. The panel is composed of 36 individual solar cells soldered in series, rated at 0.5V and 1.4A (0.7W) each.

Environment Sensing

The path planning algorithms for autonomous navigation require knowledge of the current state of the environment. The current state of the environment is defined by an array of distances to surrounding physical objects and measurements of light intensity at multiple locations on the solar panel. Path planning algorithms require continuous information for optimal path planning rather than discrete measurements. This is a requirement for further software development when more complicated algorithms are introduced.

Noting that testing will be conducted in a parking deck, on flat terrain with solid walls and no small obstacles the 3D environment could be simplified into a 2D environment. This is possible because given the rovers size, it will not fit under obstacles nor will it have to surmount obstacles. Therefore we would like to take a plane parallel to the base of the body of the rover and provide an array of distances to surrounding objects. Noting that another goal is efficient use and storage of solar power while travelling a maximum distance, we note that turning is counterproductive. Therefore providing 360 degrees of distance information would allow software to compute the optimal path without having to re-orient the rover e.g. using front facing stereo vision cameras requires the rover or a camera mount to spin to provide 360 degrees of information.

Hence the lidar module from the Neato XV-11 vacuum satisfies each of the requirements. This module is easy to remove from the vacuum system in under an hour and requires minimal supporting circuitry. The module can be interfaced using a serial connection at 115200 baud rate, supplying 5V of power to the laser and sensor and PWM power to the DC motor.

Other options were explored for environment sensing. Our original novel approach was to use input from an Xbox Kinect. However communication with computer vision researchers revealed the Kinect's sensitivity to direct sunlight i.e. distance sensing would work at night but would provide little to no accuracy during the day due to the infrared interference caused by the sun . Using the Kinect would require another solution to sense distance during the day.

The second consideration was using stereo vision along with Open CV libraries to build a 3D model of the environment. Any camera chosen would provide a limited range of view, requiring the cameras to be re-oriented. This would either waste power, turning the entire rover with fixed front facing cameras or increase design complexity, requiring an extra motor to rotate the camera. Furthermore, the cameras would not be able to provide useful information during hours of darkness which would require another depth sensing system.

To preserve power in accordance with the project definition the lidar module will be used solely for path planning and navigation. The module is powered down while the atom board is not reading the serial data; hence the rover does not incorporate real time collision avoidance using lidar. Long range infrared sensors were used by the Arduinos for real time environment sensing. Using a state machine, the Arduino can halt motion to avoid collisions and notify the Atom board when objects are sensed. USB interrupts notify the Atom board to sense the environment, path plan and continue motion with a new

list of commands. Alternative solutions included sonar and lidar sensing. Sonar was not used in view of goals for the NASA competition because sonar would not work in space. Lidar was not used in this application because of the laser's large power draw during operation.

Light, Power and Safety Sensing

Path planning algorithms will use light intensity readings to know when the rover has crossed into a shadow or if the sun is setting to allow for multiple power settings based on battery SOC (state of charge) and available solar power. Light intensity can be measured by a drop in the available current from the solar panel. This will happen during sunset but could be confused while traveling through a shadow. To allow two points of failure and more robust sunset detection, four photocells placed at each corner of the solar panel. These measures will allow the rover to accurately detect a uniform or isolated drops in solar intensity measurements relating to sunset or entering a shadow respectively.

Sensors were used to protect vulnerable electrical components such as the atom board and batteries. It is well known that LiFePo (Lithium Iron Phosphate) batteries will be damaged if discharged below a certain threshold. Therefore it is necessary to monitor the battery SOC and disconnect the batteries if once they near the discharge threshold. In addition, overcharging of batteries can be avoided by increasing current to electrical components e.g. increasing motor speed. Charging and discharging LiFePo batteries can generate heat and cause fire or explosions. As a safety measure temperature sensors were placed on the battery pack which allows batteries to be disconnected if in danger of heat damage. Similarly, experience shows that Atom boards have a tendency to overheat when placed in a small enclosure with poor circulation. Placing a temperature sensor on the Atom board allows the battery controller to determine when the temperature is too high and cut power to the Atom board. After a period of cool down, the controller can provide power to the board and resume normal operation.

To ensure efficient use of power, analog current sensors were used to monitor current provided by the solar panel, A_S, as well as total current pulled from the combined power supplies (solar panel and batteries), A_B. The values were read intermittently and logged on the Arduinos and Atom board. Let V_B be the battery voltage (13.5V) and note the following relationships

Equation 5.

$$\sum (A_{si} * V_{Bi})$$

where *i* is the logging index

is the total power provided by the solar panel.

Equation 6.

$$\sum ((A_{Bi} - A_{Si}) * V_{Bi})$$

where *i* is the logging index

is the total provided by the battery separate from the solar panel.

Note that if this summation is negative over the subsequence of measurements then the batteries were using solar power current for charging. A summation to zero at the end of a daylight period implies the solar panel provided enough power to fully recharge the batteries.

The current sensor uses a shunt resistor with a known resistance to measure the current by calculating the voltage across the shunt. The value of the required resistor can be found by the following equation. Let V be voltage and R be resistance, and I be current,

Equation 7

$$R = \frac{V}{I}$$

where V=5V and I = max current.

The current can then be deduced by using the same equation but solving for I. The resistance of the shunts required for this particular application were found by taking the maximum voltage that can be safely input to the Arduino, 5V, and dividing that number by the maximum current supplied possible of the particular components being measured.

The battery controller will be able to estimate SOC using the following equation. Let Δt be the time in seconds between log entries and B_C be the battery capacity in Ah

Equation 8.

$$SOC = 1 - \frac{\Delta t}{B_c * 3600s} \sum (A_{Bi} - A_{Si})$$

Note that this equation provides a value from [0,1] representing the estimated percentage charge based.

Electrical Component Details

Power Supplies

A survey of available LiFePo batteries showed similar prices across online and local distributors for comparable products. The batteries chosen are rated to supply 3.2V and maximum of 6A discharge. Four batteries will be placed in series to supply the required power. Experimental measurements show that the charged batteries provide up to 14.5V with an average voltage of 13.5V during a complete discharge. The custom constructed solar panel is rated at 25.2W. Experimental measurements show that it provides 20V with no load and 2.2A in direct sunlight.

Lidar Module

Using the Neato XV-11 lidar module allowed fast and easy integration of environmental sensing through a generic serial port parsed by the Atom board. The module provides a constant stream of serial data with the following protocol:

Byte	1	2	3-4	5-8	9-12	13-16	17-20	21-22
Description	Sync	Rotation	Motor Speed	Distance 1	Distance 2	Distance 3	Distance 4	Checksum

- The sync byte will always be 0xFA.
- Rotation Byte ranges from [0xA0, 0xF9]. Note that A0 contains information for degree 0,1,2 and 3, 0xA1 contains information for degree 4,5,6 and 7, etc.
- Motor speed should be maintained in the range of [0x49XX, 0x4AXX] with PWM.
- Distance 1-4 are decomposed as follows
    - The most significant bit in the first bit is the error bit.
    - First two bytes are distance in mm.
    - Last two bytes are reflection information.

TABLE 4. Lidar data description.

The module provides distance readings from 0-16 meters with accuracy degrading around 12m. Greatest accuracy is achieved with the dc motor running at 3.4V (~3000 motor RPM).

Environmental Sensors

Minimal supporting circuitry is required for a number of the analog sensors. Temperature sensors require pull up resistors on data output. Current sensors require a low ohm (< 2ohm) shunt resistor. Photocells require a pull down resistor on data out. IR sensors require capacitors across the +5V and GND lines to reduce noise in the output.

Electrical system verification

In order to make sure that all the electrical subsystems were performing as expected, each component was verified with a digital multimeter as it was integrated into the system. If the component passed the multimeter test, then it was validated again by using individual routines in the Arduino controller designed to test each specific element. If the component did not pass one of the tests, it was debugged by using a digital multimeter to check different nodes across the connection to find possible sources of conflict. After a component successfully passed both test, the next component was integrated and put through the same procedure.

Mechanical Subsystem

This section will provide information, beginning with a basic review of mechanical properties, concepts, and relevant equations, then a summary of key design decisions, and finally the process of

Introduction to the Mechanical Subsystem

The boundaries of the mechanical subsystem are defined by the interaction of the physical components of the rover. This includes tasks such as the design of structural components, the selection of materials, the construction of parts, and the assembly of the rover. One measure of the mechanical subsystem in a gross sense is the project mass budget. The mass budget gives a measure of the durability both required and provided by the system structures. In the case of this rover, the most important actuators are the motors used to turn the wheels.

Item	Quantity	Length(mm)	Width(mm)	Radius(mm)	Mass(g)	Thickness(mm)
Atomboard	1	132.08	172.72	N/A	400	63.5
DFRduino	2	101.6	83.82	N/A	60	25.4
Protoboard	1	114.3	177.8	N/A	150	25.4
Solar Panel	1	787.4	609.6	N/A	3775	6.35
Batteries	4	90	100	N/A	90	40
Lidar Module	1	114.3	101.6	N/A	200	38.1
Total Mass						**4945**

Table 5: Mass budget for summary for rover and components.

These motors, as shown in figure 10, must be chosen based upon projected and measured component masses such that the torque provided is sufficient to move the rover. More in-depth analysis is possible to investigate the mass limits for motion at various levels of velocity or acceleration or power consumption. This aspect, as well as the interaction with sensor components, represents areas of the mechanical subsystem which overlap with the electrical subsystem.

An Overview of the Mechanics of Materials

In designing components, it is necessary to consider how the components will perform under various stresses. Materials need to be thick enough such that they do not bend or break under the loadings they will experience. Given an isotropic elastic material subjected to an axial load of tension or compression, the deformity will be described by Equation 9. Let *F* be force magnitude, L_0 be original length, E young's modulus, A_0 the original cross-sectional area and be the deformation

Equation 9:

$$\Delta x = \frac{-FL_0}{EA_0}$$

As can be seen above, increasing the cross-sectional area of a component increases the resistance of that component to tensile or compressive forces. However, as will be noted later, some elements of the rover design are subjected to conditions that might induce bending. Bending is the deformation of an isotropic, homogeneous beam under a transverse load. This deformation is described by equations 10, 11, and 12 where M(x) is the bending moment, E the young's modulus, I the area moment of inertia and dx the deflection of the neutral beam axis.

Equation 10:

$$M(x) = -\frac{EId^2\delta(x)}{d^2x}$$

Equation 11:

$$Q(x) = -EI\frac{d^3\delta(x)}{d^3x}$$

Equation 12:

$$q(x) = EI\frac{d^4\delta(x)}{d^4x}$$

These equations do not account for component failure - buckling - nor do they account for complex materials. However, for some simple layered composite beams, these equations can be applied to each section and then added for a final result. Such mathematics is necessary for a detailed review of the structural properties of a system.

Mechanical Design Decisions

Motor Selection

One of the most important considerations for the GT Night Rover team in considering the mechanical design was how well the rover could propel itself using a set of six electric motors. This is a function of the operating torque a motor can provide, the wheels used, and the mass of the rover. By using the "axial thrust" property of the electric motors, maximum rover masses were estimated for various safety factors. We assume that a motor with 3 kg-force axial thrust at sea level could push a 3 kg-mass at an acceleration of 1 meter per second per second. Further, we assume that 1 m/s/s is a necessary minimum acceleration. Therefore, with 6 3-kg-force motors, an 18 kg-mass rover could be pushed at an acceleration of 1 m/s/s with a safety factor of 1. To have a safety factor of any significance given motor failure, the rover would then need to have a mass of less than 18 kg. For an acceleration of

1 m/s/s, a safety factor of 2 is given with a rover mass of 9 kg-mass and motor axial thrust of 3 kg-force per motor. Similarly, a 6 kg rover would have a safety factor of 3.

Micro Metal Gearmotor Features:

- 100:1 Gear ratio
- 120rpm @ 6V
- 30mA @ 6V
- 420mA stall current @ 6V
- 13 oz inches torque @ 6V

More complex analysis of a single motor type with multiple wheel sizes on various inclines is also possible in order to predict the efficacy of a motor in various circumstances. Spreadsheet "MotorCalculations.xls" details the analysis of the properties of the micrometal gear motors with several different wheels and inclines.

While early in the design process the use of one or two electric bike motors and a chain system to turn wheels, this design was dropped for powering individual wheels with smaller robotics motors. Additionally, while the original designs for the rover had simple bike wheels and only two axles, the final design saw the use of a rocker-bogie suspension system for stabilization and the handling of obstacles. Using a 3-wheel rocker-bogie design, this led to 6-independently powered wheels.

Figure 10. Motors used for the rover

Wheel Selection

Another factor in structural design was the wheel size and type. Wheel size impacts the ability of the rover to turn, and is driven in part by the size of the components that hold the motors. Limitations on wheel choice are compatibility with selected motors and the selection commercially available products. The use of robotics wheels led to an initial choice of 44 millimeter Pololu robotics wheels. However, the large wheel base did not allow the rover to turn effectively. Omni-directional wheels replaced the two front wheels. This replacement allow the rover to swing the front wheels as the rover pivoted around a point in the center of the four back wheels. The back wheels remained small to allow for greater torque while accelerating and turning with a smaller wheel diameter.

	
Original 49mm Wheels	Vex Omnidirectional Wheels

Figure 11. Motors used for the rover

Rocker-Bogie Suspension System

The rocker-bogie was selected as the suspension system after internal discussion regarding the rover design. While initially it was thought that using larger bicycle equipment might be able to simplify and/or reduce system cost, preliminary research determined that parts cost would actually be higher. Therefore a suspension system appropriate for smaller robotics parts was necessary. In addition to allowing a vehicle to mount small obstacles easily, the rocker-bogie design has been used for several Mars rovers successfully. Using the resources, a simplified version of the rocker-bogie system was envisioned to accommodate the motors and programmatic constraints of the GT Night Rover team. The night rover team has specializations in computer science, electrical engineering, and aerospace engineering, but the aerospace engineers - tasked with much of the mechanical end of the project - would not be available to help construct the final product.

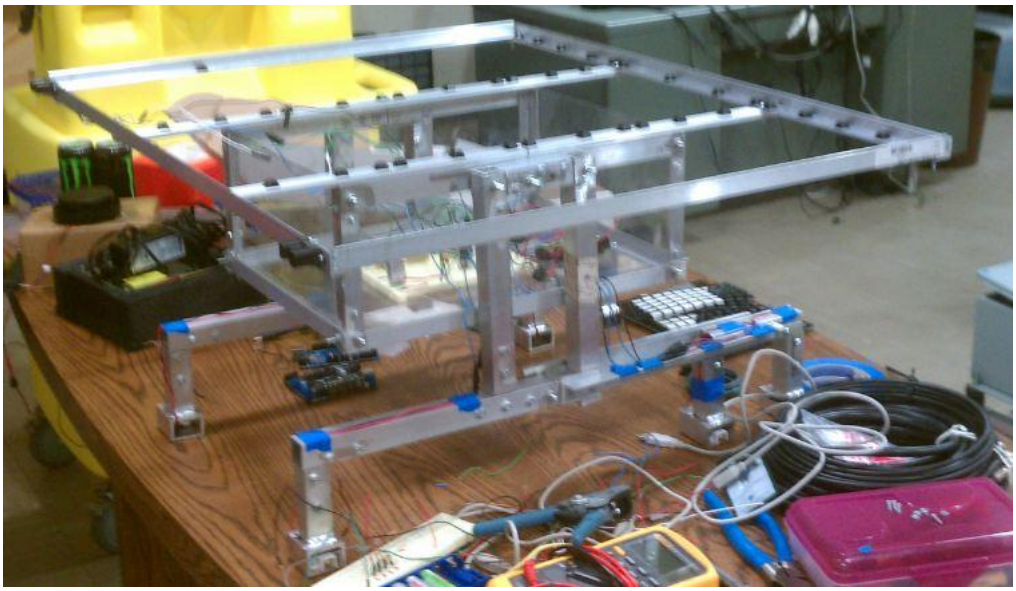


Figure 12. Completed structure minus the mounted solar panel.

Therefore the design needed to be sufficiently simplistic such that it would be easy to construct without many revisions during the build process. While some rocker-bogie systems use diagonal members that flex apart from each other to provide a suspensions effect, the Night Rover design eliminates the need for springs and hard-stops in a suspension system by using rigid vertical and horizontal members. Besides the wheels, there were two moveable joints planned in the Night Rover rocker-bogie - the bogie itself and the axle between the rocker and the main body of the rover. The problem of stabilization of the main body axle was solved by fixing the axles - essentially fixing the rocker for minimization of parts - i.e. springs, hard-stops, restraints, etc. - needed to maintain level-pointing of the body while climbing obstacles. The simplified "rocker-bogie" legs then provide housing for the individual motors as well as the clearance height necessary for the LIDAR unit carried underneath the body.

Performance Evaluation

Due to lack of experienced personnel and structural expertise team members were unable to carryout structural tests. However, while researching build materials team members received advice for material dimensions and thicknesses which would provide more than enough structural rigidness. The 0.125" aluminum thickness for square beam and angle aluminum as well as using steel right angle brackets will provide strength beyond the needs for this project.

Project Management Summary

Schedule Discussion

The original team consisted of two computer science majors, one electrical engineer and two aerospace engineers. During the month of January, un-expected events caused one of the AE majors to leave the group, and the other AE was accepted to study abroad in France. In late January another electrical engineer signed on to help. Due to the difficulties of communication with the team mate located in France, the group lost a great deal of mechanical engineering experience. The team of two computer scientists and two electrical engineers completed the vast majority of the work.

Time spent learning design software, build materials properties and machining techniques proved be time consuming for inexperienced members. The learning curve required constructing an aluminum structure took inexperienced students far more time to complete than was provided in the schedule. The time spent to construct the body took time away from other aspects of the project such as software, electronics and report writing. Building a platform from scratch was a mild concern from the beginning of the project which eventually came to fruition.

Each sub-team remained on time with the electronics team a week ahead of schedule through mid March. Once the final design structural design was decided on, one of the software programmers was assigned to take care of the structural component of the project. Progress was slow through the

rest of the semester with the team falling nearly two weeks behind schedule due to time spent on structure construction. At the time of the final report the team had completed the structure and electronics. Electronics have been tested for correctness and the software team has code ready for implementation and testing once the rover is complete. It is the hope that fast progress will be made over the next week providing the team with conclusive results to present at the final competition in Disney. A complete schedule can be seen in Appendix E.

Financial Update and Discussion

The team remained under budget through careful planning and support from campus electronics labs with supplies and tools. The major design decision which influenced overall budget was scaling down the original design. Original projections for the initial budget easily exceeded the provided funds. Hence reduced dimensions and mass translated to less expensive mechanical components.

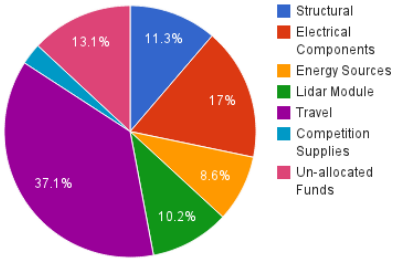


Figure 13: Financial spending breakdown.

Appendices

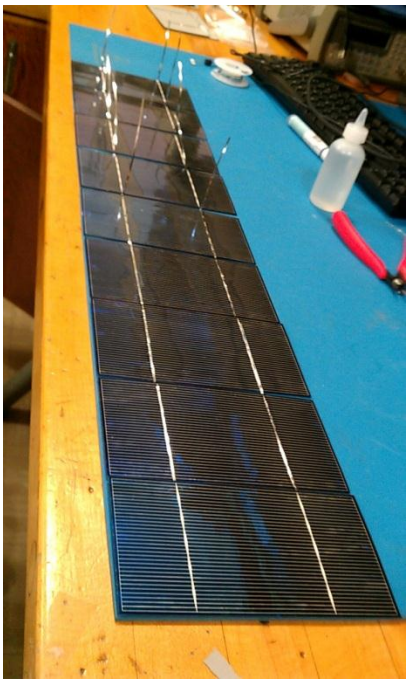
Appendix A: Neato XV-11 Lidar Module Extraction

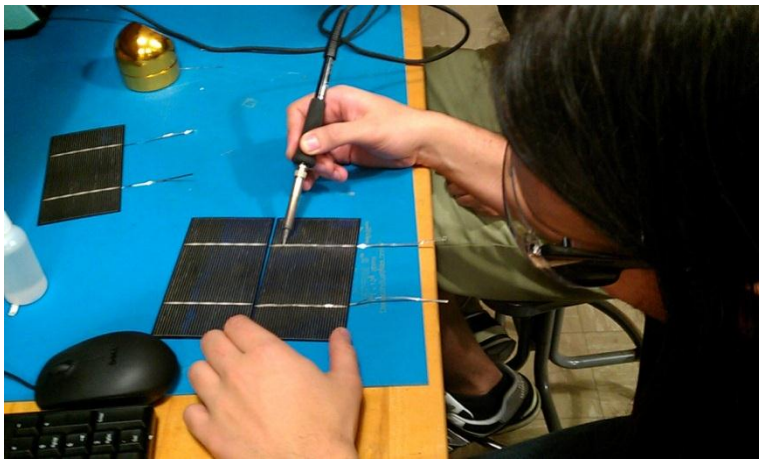
Figure 14: Lidar module being removed from the Neato XV-11 vacuum.

Appendix B: Solar Panel Assembly and Construction

This appendix outlines the steps necessary to assembly the solar panel.

1. Solder the solar cells into series, top (blue) of cell is positive and bottom of cell is negative. These should be soldered into four columns of nine cells. Since each cell is rated at .5V, this combination will produce the required voltage of 18V to charge the batteries and supply power to the rover.





2. Connect the four columns of cells into series in a similar manner.

3. Test solar panel output to make sure all connections are secure. Attach a multimeter to each end of the source to measure voltage which should read approximately 20V in direct sunlight with no load. Note that a load is required to measure current so high ohm, high watt resistors should be used. Once the resistors are in place, the current can be measured by the ammeter by placing the meter in series with the broken circuit.

Figure 16. Multimeter showing 19.72V with open circuit.

4. Cut 0.125” sheet of glass to 24”x31” dimensions. Use extreme caution while cutting glass! This step should be completed before the next step since the solar panels are so fragile.

5. Using rubber cement, glue the photocells' blue sides to the glass sheet. Place upside down, blue side down, in a safe place to allow glue to dry for 24 hours. There should be a small weight placed on the solar panels to ensure they stay flush with the class but be cautious as to not put too much weight since the solar cells are very fragile.

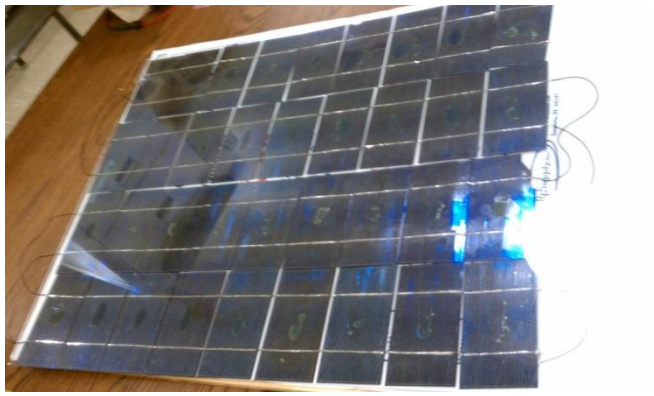


Figure 17. Solar panel with glass cover glued on the front.

Appendix C: Electrical Assembly and Construction

This appendix outlines the steps necessary to connect the electrical circuit.

1. Study the supplied schematic in file "schematic 3.0.JPG".
2. Solder break out boards for easy mounting and interfacing analog peripherals. Be sure to include:
   pull down resistors for the photocells
   pull up resistors for the temperature sensors
   capacitors across power for the IR sensors to stabilize the input data

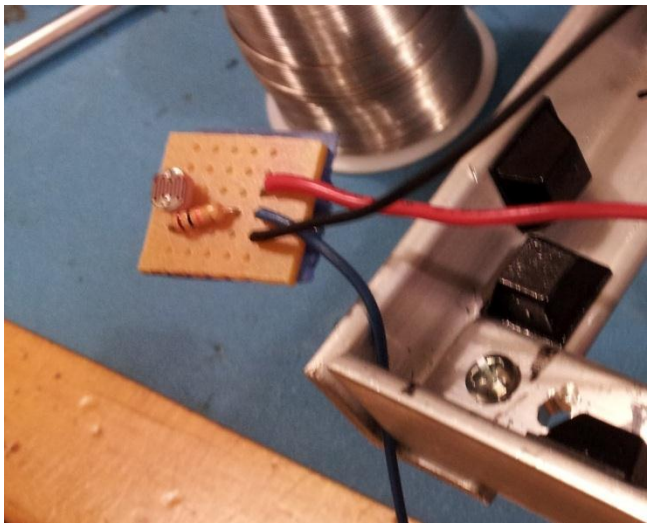


Figure 18 Light sensor soldered onto board for easy mounting.

Optional: Solder plugs to allow for easy assembly and disassembly. Use BD9 connectors for data and motors, and a 2 pin molex connection for solar panel power. Each rocker-bogie should use 6 pins, the solar panel peripherals can be fit into 8 pins if a common power bus is shared.

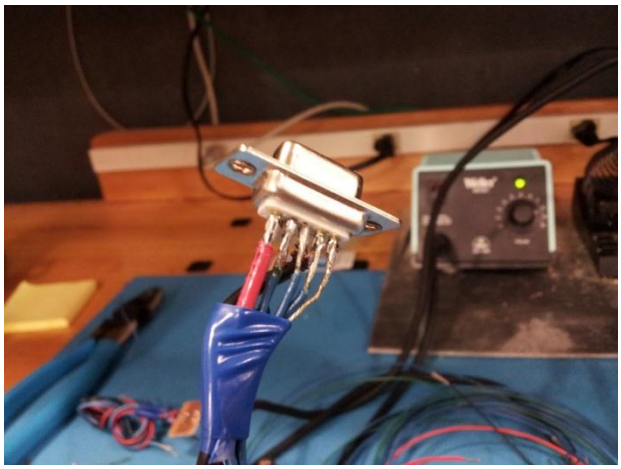


Figure 19. Pin for left side which includes motor and wheel encoder power and data. Right is the same.
Optional: Solder wire inputs to an intermediate "connection" board to organize wire and create more secure connections as wire can be easily pulled out of Arduino headers. This step will produce less stress on the wires connected to the arduino which could potentially be pulled out if enough pressure is applied. Solder wires coming out of intermediate board to header pins for easy connection and removal to arduino. This will make for easier removal of the components in case something needs to be tested individually or replaced.

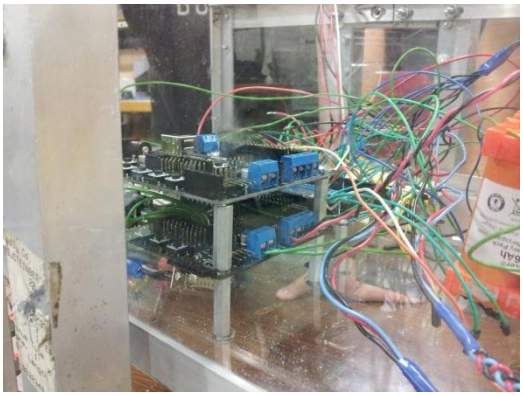
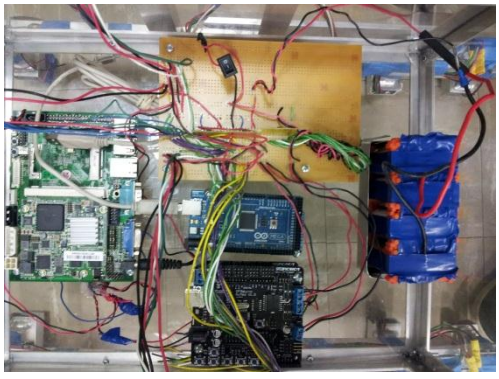
Before Connection Board	After Connection Board
	

Figure 20. Before and after soldering the wires onto the breakout board. It is clear that this new design is much neater and would be easier to debug.

3. Connect the correct wires to the following pins on the DFRduinos

Battery Controller	**Batter Controller**
Motor Power Terminal - Connect to 12V Line Motor 1 Terminal - +/- to Lidar DC Motor Motor 2 Terminal - USB Port - 0 - Tx Motor Controller 1 - Rx Motor Controller 2 - 3 - 4 - M1 Direction Control * 5 - M1 Speed Control * 6 - 7 - 8 - 9 - 10 - 11 - 12 - 13 - Status LED * A0 - Current Solar Panel A1 - Current 12V A2 - A3 - A4 - Temperature Atom Board A5 - Temperature Battery A6 - A7 -	Motor Power Terminal - Connect to 12V Line Motor 1 Terminal - +/- to Left Rocker-Bogie Motor 2 Terminal - +/- to Right Rocker-Bogie USB Port - Cable to Atom Board 0 - Tx Atomboard (USB) * 1 - Rx Atomboard (USB) * 2 - Tx Battery Controller 3 - Rx Battery Controller 4 - M1 Direction Control * 5 - M1 Speed Control * 6 - M2 Speed Control * 7 - M1 Direction Control * 8 - Right Encoder A 9 - Right Encoder B 10 - Left Encoder A 11 - Left Encoder B 12 - 13 - Status LED * A0 - Photocell0 A1 - Photocell1 A2 - Photocell2 A3 - Photocell3 A4 - IR Sensor 0 A5 - IR Sensor 1 A6 - A7 -
Arduino Mega	
USB Port - Atom Board 0 - Tx Atomboard (USB) * 1 - Rx Atomboard (USB) * 2 - Tx Lidar Module (Orange Out) 3 - 4 - … * No need to hook up.	

TABLE 6. Pin assignments for arduino.

4. Pull use +5V and GND from the motor controller DFRduino to power all analog sensors. Use +5V and GND from the Arduino Mega to power lidar module.

5. The +5V rail will be obtained from the output of the arduino.

6. The +12V comes directly off the battery which will be around 13.7V

7. Current sensors- The connections for the current sensors can be seen in the table below. The numbers correspond to the pins on the current sensor.

Solar Current Sensor	Power Consumption Current Sensor
1 - Positive lead from current sensor shunt resistor 2 - GND 3 - GND 4 - NC(not connected) 5 - VOUT to 6 - 5V+ 7 - GND 8 - Positive lead to battery bus shunt resistor	1 - Positive lead coming from battery/solar panel shunt resistor 2 - GND 3 - GND 4 - NC 5 - VOUT to 6 - 5V+ 7 - GND 8 - Positive lead to 12V bus shunt resistor

TABLE 7. Connections for pins to be connected to the current sensors.

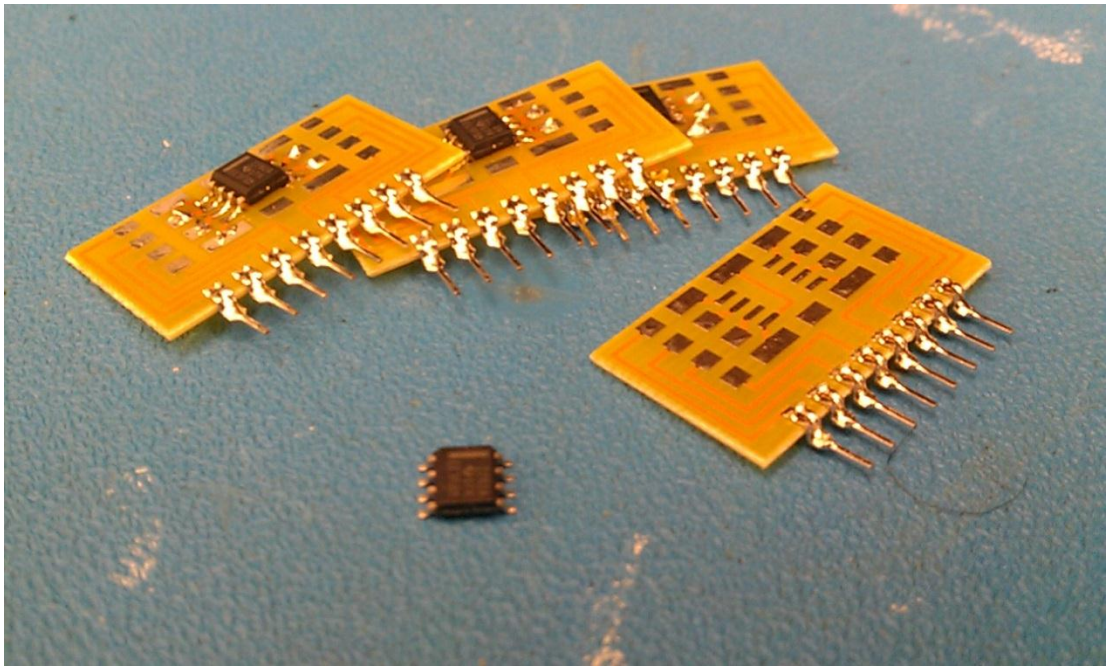


Figure 21. Current filters before and after being soldered onto the breakout board.

8. A main power switch for the solar panel should be installed by breaking the hot wire coming from the solar panel with a switch. This switch will be used to cut the power to the solar panel when there is not enough light out to provide any charge to the battery.
There should be a second switch placed on the 12V rail which will cut the power to everything after the the battery. This would be done by breaking the positive lead coming out of the battery going to the current regulator.
The final switch should be placed on the positive lidar sensor lead to cut the lidar off when not in use.

9 Double check all connections then connect battery to input of analog sensor 2.

Appendix D: Mechanical Assembly and Construction

This appendix outlines the steps necessary to build the rover structures.

1. Cut the required lengths of aluminum 1” square, 0.125” thick pipe:
   - 8x 3” lengths
   - 2x 5” lengths
   - 2x 9” lengths
   - 4x 12” lengths

Figure 22. Aluminum tubing after being cut for the rover.

2. Cut the required lengths of aluminum 1”x1.5” rectangular, 0.125” thick pipe:
   - 6x 2” lengths

3. Cut the required lengths of the 0.25” aluminum plate
   - 4 x 2”x12” inch pieces (or order similar pieces)
   - 2 x 1”x25” inch pieces (or order similar pieces)

3. Cut the required lengths of aluminum 0.75” angle aluminum (0.125” thick)
   - 2 x 16” lengths
   - 2 x 12” lengths
   - 4 x 16” lengths
   - 2 x 25” lengths
   - 2 x 32” lengths

3. Cut the required lengths of aluminum 1” angle aluminum (0.125” thick)
   - 2 x 16” lengths

4. Cut the required sizes of lexan sheets
   - 1 x 16”x12”
   - 2 x 16”x6”
   - 2 x 12”6”

5. Assemble body case and solar panel support

   Assemble the solar power. Build a rectangle using the 25” and 32”. The 1”x25” sheets should be spaced by a 16” gap which is centered respective to the 32” angle aluminum.

Figure 23. Constructed frame for the solar panel.

Construct the aluminum box using the 16” and 12” angle aluminum as a base and the 6” angle aluminum as the walls.

Figure 24. Structure for the center of the body

Screw the lexan sheets into place.

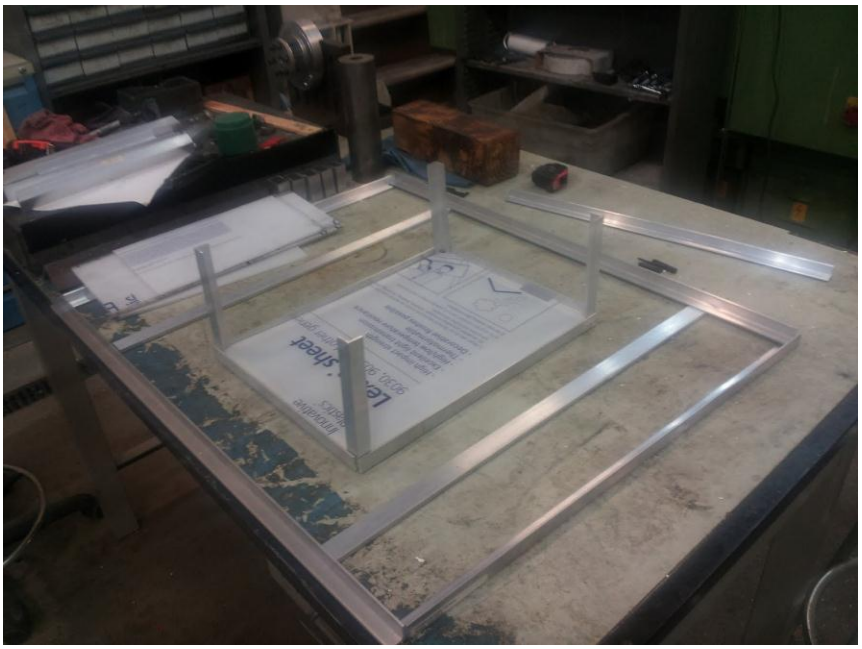


Figure 25. Structure for robot.

Screw the 1” angle aluminum to complete the top of the body frame. The 6” legs should fit into the right angle of the 16” piece. This will provide a flat platform for the 1”x25” pieces to be bolted to. This will allow the solar panel to be mounted securely adding structural strength.

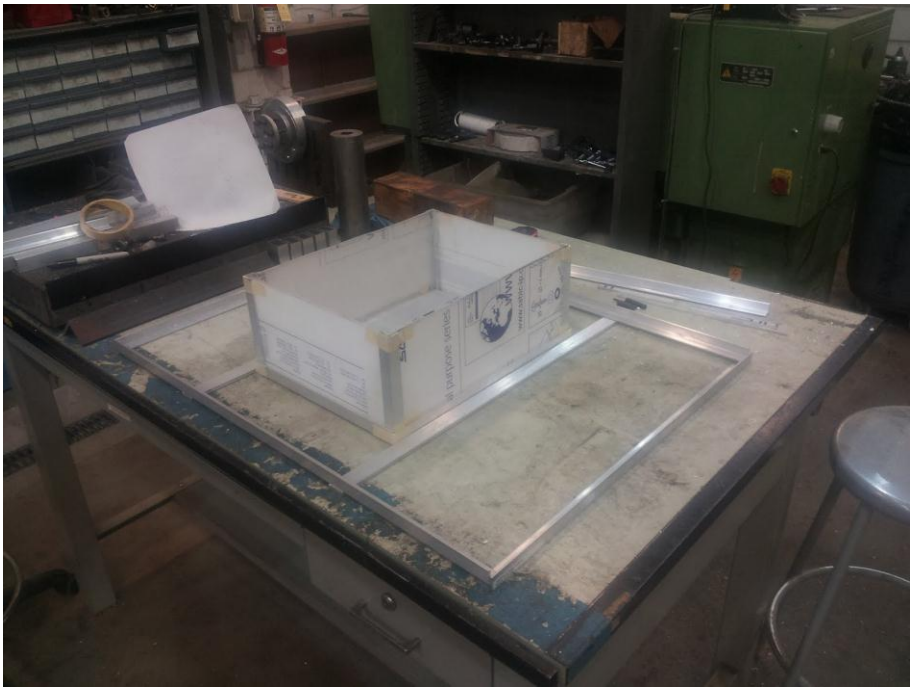
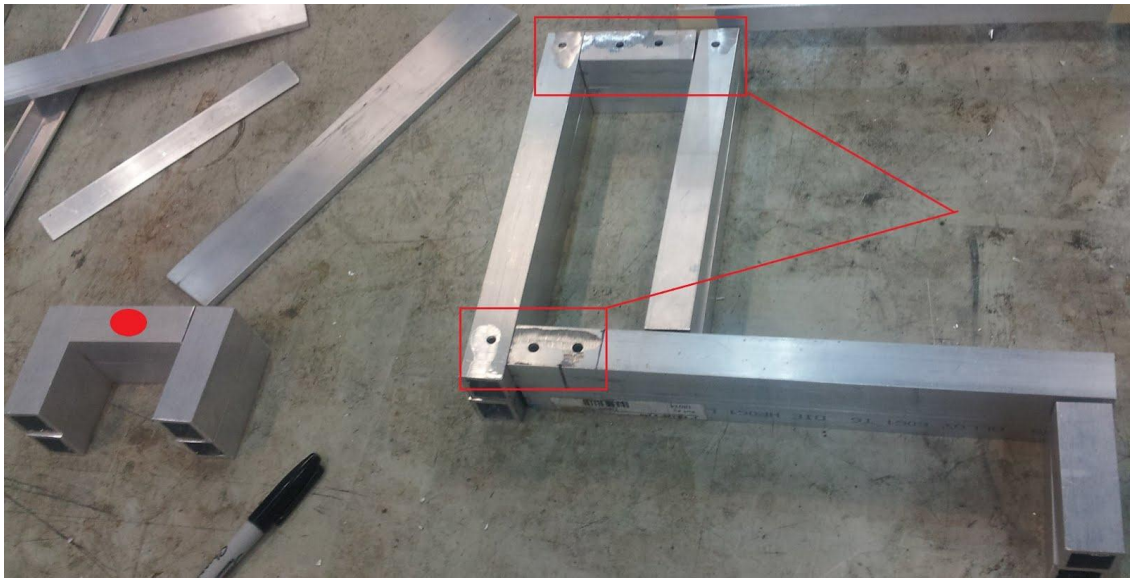


Figure 26. Center body structure with solar panel frame attached.

6. Assemble the Rocker-Bogie

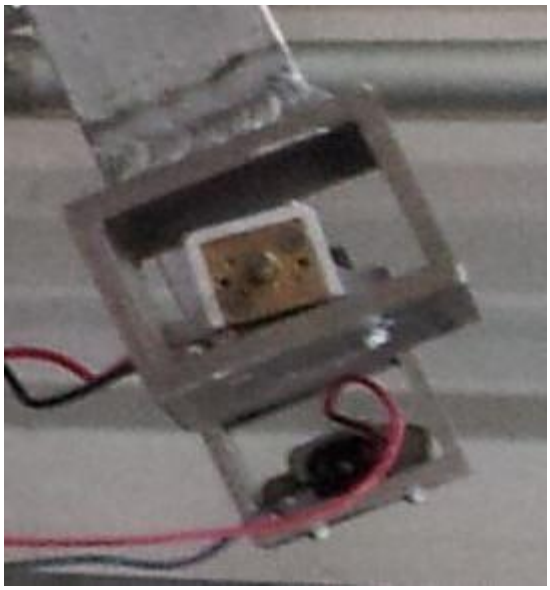
Two options for securing joint include the use angle brackets or welding. To reduce mass welding is the better option but aluminum can be extremely difficult to weld. In the current implementation angle brackets were used.

Drill the following holes for mounting the 2”x12” plates which sandwich the bogie and mount to the top frame of the body. Evenly space the holes for the greatest structural integrity.



7. Mount motors

Using the micro metal brackets, measure that required hardware mount location such that the wheels clear all aluminum when fully mounted to on the motors. Note that when mounting the motors, the provided bolts easily slip out of mount. An easy solution is to tape or super glue the screws such that pressure can be applied while mounting the motors without dislodging the bolts.



Apendix E: Complete Schedule

Task	Projected Completion Date	Notes
Submit Structural Proposal	1/1/2012	Rocker-Bogie suspension with closed body
Electrical Systems Proposal	1/1/2012	Lidar for perception
Submit Power Budget	1/10/2012	
Finalize Power Budget	1/20/2012	
Order Electrical Components	1/21/2012	
Finalize Structural Dimensions	1/24/2012	Body: 24" x 12" x 4" and Solar Panel: 25" x 30"
Mid-Term Review	1/26/2012	
Atom Board Development Environment Setup	2/3/2012	
Atom Board Communication with Lidar	2/10/2012	Serial Arduino data dump
Solar Panel Construction and Charging Circuit	2/10/2012	
Temperature and Daylight Detection Circuits	2/17/2012	
Motor Control Circuit and Real Time Collision Detection	2/24/2012	
CAD and Machine Parts	3/9/2012	Work distributed among
Electrical Testing Phase I: Motors, Charging, Sensory	3/9/2012	Report with known bugs and missing functionality
Atom Board Navigation Libraries Phase I	3/9/2012	Base Functionality
Construction of Chasis With Integrated Electrical Systems	3/16/2012	
Atom Board Navigation Libraries Phase II	3/23/2012	3D (or 2D) Mapping and Data Logging
Power Controller Logging System	3/30/2012	
Functionality Testing	4/6/2012	
Data Collection	4/18/2012	
Final-Review	4/23/2012	
Final Competition	5/4/2012	Disney

TABLE 8. Schedule for rover.

Appendix F: Software Architecture Diagrams

Please reference the folder “Software_Architecture”. Lower level Java application.