Lunabotics Mining Competition
Systems Engineering Paper

McGill LunarEx Robotics
McGill University
Montreal, Canada

April 22, 2013

Nicholas Speal

Mechanical
Juan Pedro Celis

Patrick Abouzakhm
Paul Albert-Lebrun
Matei Anghel
Samantha Baribeault
Spencer Boone
Hyder Ally Cadersa
Shawn Fontaine
David Villegas Gonzalez
James Michael Harrison
Nate Xiao He
Thomas Hoekman
Sanchit Jain
Ricardo Jordan
Michael Elliot King
Dorothy Laura Lee
Matthew Mayers
Todd McKenzie Scrimgeour
Charles Joseph Turner
Marc Wang
John George Willes

Electrical
Matthew Johnston

Nicholas Athanasoulias
Russell Buchanan
Jérôme Colomb
Jean-Sébastien Déry
Salman Hashmi
Seif Kurdi-Teylouni
Jordan Litwin
Hamza Riaz
Benjamin Suire

Software
Ernest Oppetit

Jordan Benabou
Michael Lagacé
Sébastien Lemieux-Codère
Srinivas Nadella
Brian Nolan
Christopher James Reny
Hadi Eddy Sayar
Nguyen Khoi Tran
Mathieu Wang
Yang Wu
Guang Yang

Faculty Advisor

I have read and reviewed this paper prior to its submission to NASA.

Peter Henry Radziszewski
Associate Professor
McGill University

Contents

Abstract

McGill LunarEx Robotics is a team of fifty undergraduate students from McGill University, competing in the 2013 NASA Lunabotics Mining Competition. This report documents the team's progress through the systems engineering process, focusing on the design and integration of Mechanical, Software, and Electrical systems. The Lunabot's design is centered on an innovative auger excavation mechanism and complete autonomy, with features such as deformable wheels, independent 4-wheel drive and steering, and an active suspension system. The implementation of a software system on top of the emerging standard framework, ROS, is discussed.

Introduction

Purpose Statement

One of the most fascinating aspects of robotics is its multi-disciplinary nature. Building a successful robot relies on effective integration of techniques and processes in Mechanical, Electrical, and Software Engineering. This report details the Systems Engineering process that was followed by McGill LunarEx Robotics throughout the development of an autonomous lunar rover for the NASA Lunabotics Mining Competition. As such, emphasis is placed on component integration to design a complete robot, rather than the details of how each system works on its own.

Problem Definition

Mining resources on the moon will be a necessary step in the evolution of space travel. Access to raw materials in space without having to spend energy to overcome Earth's gravity would be a tremendous boost to facilitate development in space and continued space exploration.

The substantial economic and scientific benefits of lunar mining motivate the development of technology to accomplish this task. Environmental limitations require that such a system should be able to operate in a vacuum, on dusty terrain, and with limited communications. It should be able to avoid obstacles and craters, and minimizing the mass would greatly reduce the expense of sending it to the moon. The challenge for McGill LunarEx Robotics is to design and manufacture an autonomous lunar rover capable of meeting these criteria.

Design Philosophy

With such an open-ended task at hand, it was necessary to clarify the objectives from the start. The design philosophy is intimately related to our motivations for coming together as a team in the first place: to develop professionally as an engineer, and to gain exposure to cutting-edge technologies. With this in mind, many design decisions were made to be as innovative as possible, sometimes in light of safer alternatives.

An important distinction between the Lunabotics competition and a NASA mission is the level of risk aversion. Indeed, the consequences of failure are so great for a full-scale space mission that NASA cannot afford to take chances without investing years into developing confidence in the technology. For LunarEx, the notions of pushing boundaries and acquiring skills prevail throughout the design, from the choice of an unconventional excavation system to the implementation of fully autonomous control. Optimizations like weight and price can come with further development as the technology readiness level progresses.

An important goal involves determining which designs paid off and which did not, to gain knowledge for the team's future participation in the Lunabotics competition.

Requirements

There are various constraints within which the team must operate. The requirements represent a synthesis of the Lunabotics competition rules and the McGill LunarEx Robotics design philosophy. They can be broken down into five categories:

Functional – What functions are necessary to accomplish the objectives

- The Lunabot shall excavate, store, and deposit lunar regolith simulant
- The Lunabot shall operate on loose terrain in the presence of rocks and craters
- The Lunabot shall communicate wirelessly with the control room
- The Lunabot shall fit into a box of dimensions 1.5m x 0.75m x 0.75m

Performance – Quantitative indicators of how well the system performs its functions

- The Lunabot shall deposit more than 50 kg of lunar regolith simulant
- The Lunabot shall operate with no human intervention
- The Lunabot shall complete two cycles of operations within 10 minutes

Verification – Assurance that requirements will be met

- System Requirements, Preliminary Design, and Critical Design reviews shall be conducted prior to manufacturing
- All functional and performance requirements shall be met by May 16, 2013

Interface – Benchmarks for how subsystems should interact

- Power systems shall be able to accommodate the loads at full bucket capacity
- No systems shall interfere with each other
- The communications system shall interact with the NASA network and not interfere with competitors

Human Factors – Operator requirements

- The Lunabot shall be teleoperated by one user with one Xbox controller
- In autonomous mode, sufficient information shall be relayed to the control room for the operator to be aware of the state of the system.
- A safety button to kill all power shall be readily accessible

Systems Engineering Life-Cycle

Systems Engineering Phase	Objectives	Timeline	Key Events
Pre-Phase A: Concept Studies and Initial Design Decisions	Determination of high-level design	Summer 2012	
Phase A: Concept and Technology Development	Trade-analysis and functional allocation	September, 2012	September 29: System Requirements Review
Phase B: Preliminary Design	Subsystem Development and Interface definition	October, 2012	November 3: Preliminary Design Review
Phase C: Final Design and Fabrication	Detailed Design Development	November, 2012	December 5: Critical Design Review
Phase D: System Assembly, Integration and Test, Launch	Iterations based on testing results	January - May 2013	March 10: Lunabot Assembly Deadline
Phase E: Operations and Sustainment	Competition	May 20-24, 2013	February 7: CSA Progress Update
Phase F: Closeout	Learning from Competition, Recruit for next year	Beyond May 24, 2013	May 24: Closing Ceremonies

Pre-phase A: Concept Studies and Initial Design Decisions

In July 2012, this year's effort started with two team members, a well-equipped workspace, an excavation auger prototype, faculty support, and some rollover funding. The first move was to test the auger to find out two key parameters: collection rate and beam trajectory. The mechanism proved to be successful on both parameters, collecting 20kg of sand in 15 seconds with an average horizontal range of approximately one meter at full speed.

Figure 1 - Testing the 2011 auger prototype

The test results indicated that an auger should be used if its weight could be brought down. The team decided to redesign a lighter auger and to build the other mechanical systems around it. In addition, spurred on by the fact that no team had ever achieved full autonomy and due to its increased importance in the 2013 scoring scheme, the team decided to avidly pursue full autonomy. Throughout the project, the allocation of resources was centered on these two key goals.

Phase A: Concept and Technology Development

Concept of Operations

From an outside perspective, the "black-box" Lunabot will:

- Autonomously complete two full cycles of operation within 10 minutes. In a cycle the Lunabot will localize in the arena, plan a trajectory, drive through the obstacle area, avoid craters and rocks, mine and store 50 kg of regolith, go back to the starting position and deposit the regolith in the lunabin.
- Be optionally teleoperatable in case of unexpected circumstances
- Measure and report power consumption during competition runs.
- Behave as it should on the moon – with a clean operation that minimizes dust.
- Have two operating states based on the auger mechanism requirements: travel and mining. The travel state is when the robot is high with respect to the ground level. It is implemented for localization, driving, avoiding obstacles and dumping. The mining state is when the auger is lower than the ground level and the Lunabot is both excavating and storing regolith simulant.

Functional Allocation

The process of choosing the technology to satisfy functional requirements is discussed.

Mechanical System

Excavation

The auger excavation system is a single degree of freedom rotating screw that operates similarly to a conventional snow blower. However, the LunarEx auger design has only one stage as opposed to the two stage (screw-fan) traditional design. The regolith is pushed towards the center and shot out of the casing by the screw only. The 1 DoF condition makes the mechanism simple and reliable compared to a conventional bucket chain system with many moving components that add probability of failure on the moon. In addition, a high mass flow rate of regolith is maintained. The previous prototype showed to be very effective but extremely heavy (22kg) and bulky (72cm length). Thus, focus was placed on the most critical problem: weight. The team aimed to make the auger lighter (12kg max) by studying the design parameters and exploring new manufacturing methods while maintaining operational reliability and effectiveness.

A single stage 4-bar dumping mechanism with a storage bucket design was chosen based on the characteristics of the auger dust beam capabilities. The system requires

lifting the bucket above 0.5m and angling it enough to allow the regolith to slide out. A sealed door keeps the regolith inside until the upright position is reached. The bucket needs to be both light and able to hold a volume equivalent to 100kg regolith. The 4-bar mechanism was evaluated and considered the best option due to its simplicity to perform both required motions (lifting, angling) with a single actuator and minimal space requirement. In addition, it reduces the regolith release process to a single stage that can be rapidly performed. Alternative mechanisms such as a scissor lift or a pulley system were also considered but were rejected due to complexity.

Locomotion

From several types of robot mobility systems summarized in the literature (Nildeep Patel, 2010), the conventional 4-wheeled locomotion system was selected in combination with an active suspension mechanism. Other taxonomic alternatives such as tracks, discrete motion and a hybrid system were considered but rejected. LunarEx developed a deformable wheel design with an internal cellular structure. The deformation of the wheel allows a greater contact area on the ground, improving traction. This innovative concept alternative to pneumatic tires in a lunar environment has the advantage of tracks without the failure modes. The cellular structure also provides a lightweight design. The number of wheels was chosen in order to reduce control and manufacturing costs while keeping a stable base.

Independent steering and drive for each wheel was decided for the locomotion design. The additional complexity of multiple motors and degrees of freedoms on the mechanisms was justified by the addition of several modes of operation for traveling and mining states. For example, a continuous circular mining path is possible. The active suspension provided several advantages to the overall mechanical system. This mechanism removes the need for an additional mechanism to lower and raise the auger, simplifies the auger bucket connection, adds additional height reach to the dumping mechanism, reduces the risk of running directly into obstacles and works as a fail safe to obstacle detection errors. The dependency on an active suspension for a successful operation of the Lunabot highlights the importance of resources allocated towards a comprehensive locomotion design analysis.

Software System

Software Framework

To interface the robot's hardware sub-systems through the use of drivers, and the software sub-systems through common operating-system services, the team set out to find a suitable framework on which to develop the Lunabot's software.

The Robotic Operating System (ROS) was swiftly chosen due to its rapidly growing body of open-source software and its emergence as a standard in the robotics industry. ROS is supported on Ubuntu Linux, and provides two key sets of functionality:

• Packages: a multitude of user-generated software modules (called nodes) to implement a collective functionality. For instance, the driver we used to interact with our LiDAR is one such package, called hokuyo_node.

• Common operating system services such as hardware abstraction and message-passing between processes. This is done through ROS topics, which provide a steady stream of data from a source node to its destination(s), or through ROS Services, where a requesting node queries another node to obtain some data.

The Raspberry Pi and the BeagleBoard microcontrollers were both considered to run the software, following cost/performance analysis and the prospect of mishaps from developing on Intel-based laptops for an ARM-based device, the team settled on a dual core Toshiba netbook.

Communications

A client-server protocol was required to facilitate the sending the "go" command, robot status feedback and, as a backup, teleoperation. The client-side GUI was implemented in Java and Swing because of cross-platform compatibility, past familiarity, and socket library availability. The server was written in Python because it interfaces nicely with ROS and JSON.

Navigation

A crucial part of the robot's autonomous operation involves taking in a current and goal pose and determining what path to take based on the map.
Two alternatives were considered to implement this function:

- a custom A* algorithm written by the team, to provide path-finding based on a map of nodes
- The black-box ROS Navigation stack which takes in parameters such as a map and goal pose, and performs dynamic path-planning to output safe velocity commands (ROS)

Table 1 - Functional allocation decision matrix for the navigation function

	Navigation Stack	A*	Weight
Well integrated in ROS?	Yes, especially given the output format of the Map_builder node	No	0.1
Coordinate transformation provided?	Yes	No	0.2
Development time?	Short: its operation involves only setting configuration files	Long	0.1
Easy to test?	Yes: "Stage" simulation in the ROS simulation environment	No, requires complex adaptation of data types to test in Stage	0.3
Difficulty?	Medium – Hard: some configurations are difficult to set	Medium	0.2
Documentation Status?	Poor	Good, the algorithm is widespread and has many resources available online	0.1
Score	**0.7**	**0.3**	**1.0**

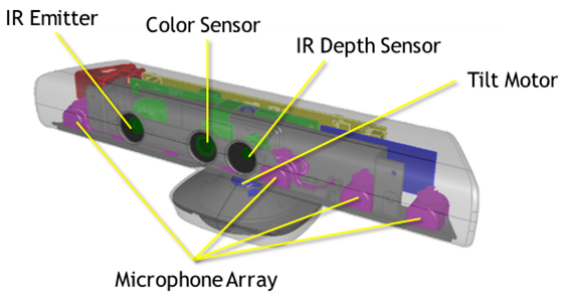
Localization

Prerequisite to the goal of autonomous operation of our Lunabot is that of localization in the arena throughout the run, with the two following important considerations:

- The two potential starting poses (position and heading) of the robot and the exact dimensions of the arena are known prior to the competition
- Due to slippage, wheel-based odometry may prove unreliable

Using a vision system to find the distances between the robot and the four walls and triangulate its pose was immediately viewed as the best option to localize.

Given the availability of a Hokuyo LiDAR UTM-30LX (Hokuyo, 2012), the choice of which sensor to use for localization was straight-forward. LiDAR is an optical remote sensing technology that can measure the distance to a target by illuminating it with light. This concept is then expanded to multiple points by reflecting the source light onto a rotating mirror to perform a sweeping motion.



Obstacle Avoidance

In order to autonomously navigate in the arena, obstacles (rocks and craters) must be identified and added to the map used by navigation. Since the LiDAR scans on a horizontal plane, it would not be able to "see" the craters; hence another sensor was needed for the purpose of obstacle avoidance. Again, the hardware choice was simplified by ownership of a Microsoft Kinect. The Kinect transmits invisible near-infrared light and measures its "time of flight" after it reflects off the objects.

Given its ability to scan in 3D, one might wonder why the Kinect was not used for both localization and obstacle avoidance; the reason is threefold: (i) the LiDAR is more precise; (ii) Hector_mapping, a ROS node (described in Phase B) which greatly simplified the localization task, is only available for LiDARs; (iii) having two sensors for makes it easier to divide work.

Map integration & Control

For navigation purposes, it is necessary to have a single, global map containing the arena's walls, and the positions of the obstacles. Hector mapping provides the former and through its map output and the obstacle avoidance node outputs the latter. A custom node called Map_builder is created to integrate the sensors.

A central ROS control node – called Command_Node – was identified as the best way to send commands to the navigation and motor control systems, based on data gathered from the vision systems.

Electrical Sub-system

Power Circuit

The starting point for motor development was the 2011 LunarEx robot, which featured the Hacker A50 brushless DC motor powering the auger, and large Grosschopp motors for motion. The Grosschopp drive motors are too heavy for the 2013 design, but the Hacker motor performed well with the last auger, so it stayed in the new design.

The 2011 robot used LiPo batteries. They performed well in the past and have high energy density, so they were chosen over the alternatives of NiMH, lithium-ion, and lead acid.

Control Circuit

The Hacker X80 motor controller continued to work well with the auger's motor and so it was not changed. The team also decided to implement a power monitoring system to determine the total energy consumed during the run.

System Hierarchy and Team Structure

This year marked a substantial change to the team structure. Past teams were typically very small, but this year approximately 50 people were recruited. One of the advantages of a small team is a genuine sense of responsibility for the outcome of the project that is shared by a few members. With this in mind, the whole team was broken up into groups of less than 6 people, each of which was responsible for a subsystem. Each leader kept their sub-teams engaged and on-task, while coordinating at the system level within the Mechanical, Electrical, or Software team. The system hierarchy, which also dictated the team's structure, is shown in Figure 2.

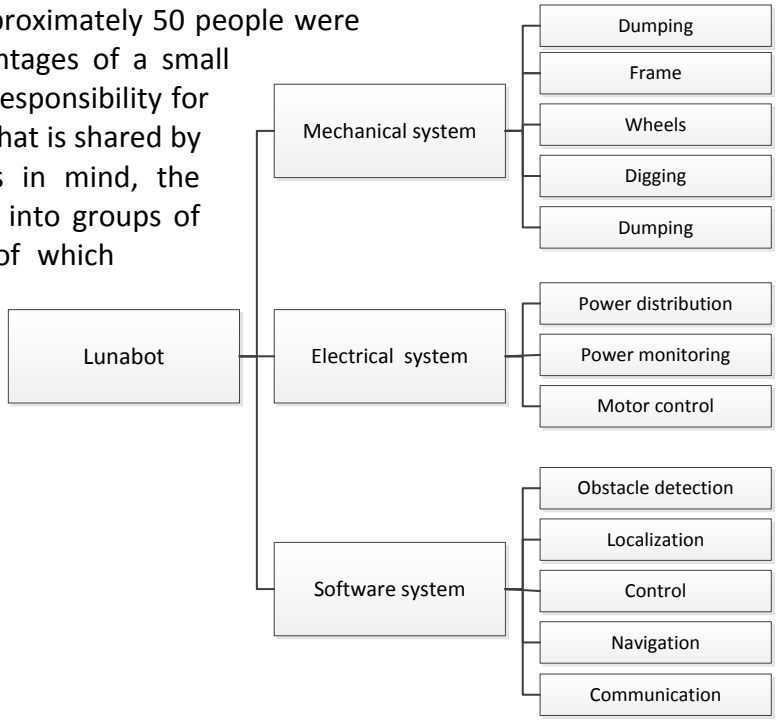


Figure 2 - Sub-system and team structure

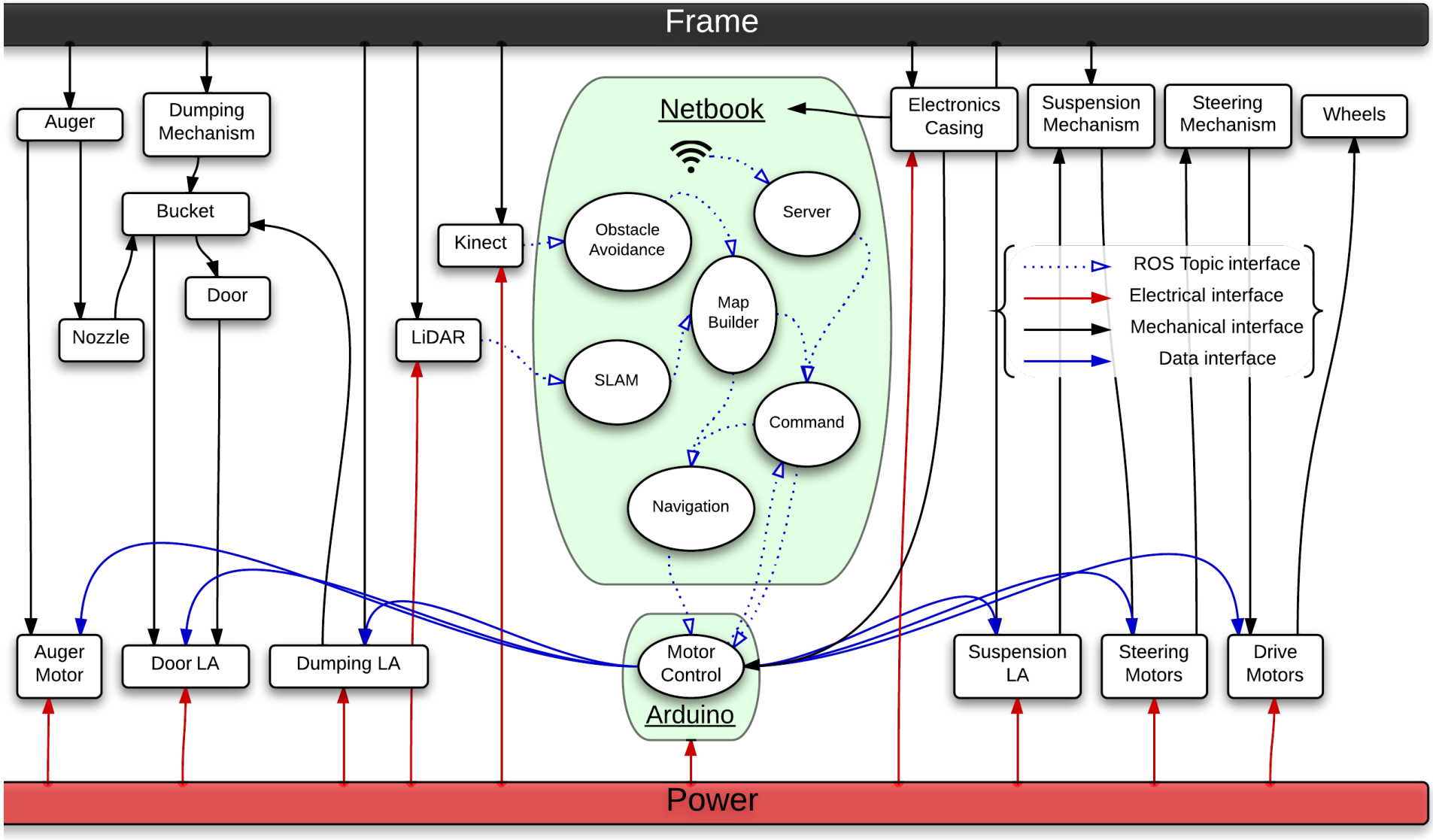


Figure 3 - Lunabot high-level interfaces

Phase B: Preliminary Design

Interfaces defined

A key consideration in systems engineering and team management is the definition of detailed and unambiguous interfaces, to ensure everyone is on the same page regarding what is expected of their sub-system, and what they can expect from others'. The Lunabot's high-level interfaces are summarized in Figure 3.

Mechanical system

Initial mass estimate

The following mass constraints were provided as a guideline to the individual sub teams:

	Excavation	Dumping	Frame	Electronics	Actuators	Wheels	Locomotion	Total
Mass [kg]	12	7	10	7	15	8	7	**66**

Table 2 - Initial Mass Estimates

Software system

The software system's interfaces are shown in more detail on the ROS Graph (Figure 4) below, where each ellipse represents a node and the arrows are topics or services.

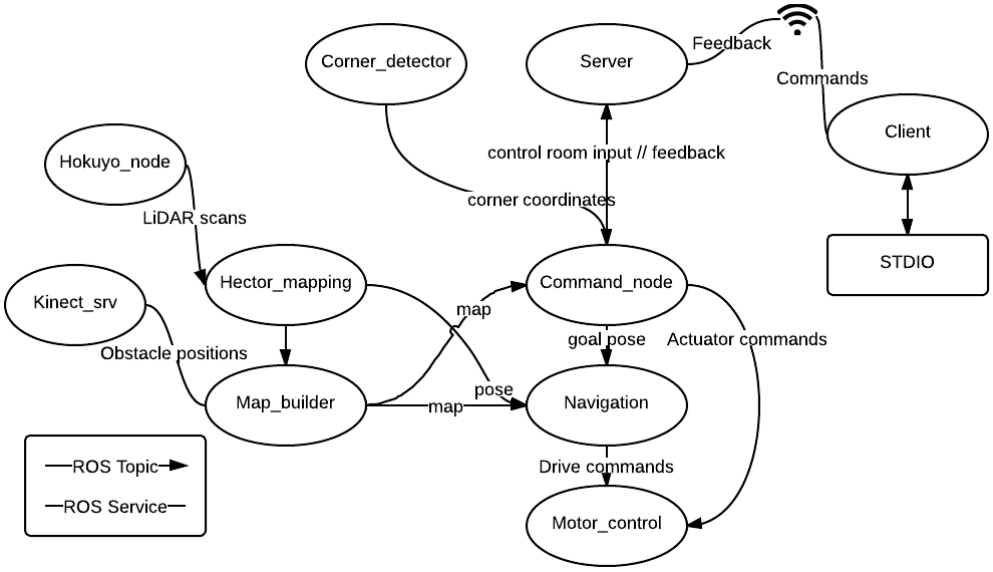


Figure 4 - ROS Graph, showing the software system's interfaces

This system was designed to ensure that all functionalities outlined in the Lunabot's concept of operations were being met, keeping in mind the functional allocation presented in Phase A. Each node shown in Figure 4 performs a clear goal, distinct from the other nodes in the system. The nodes can be grouped by functionality:

- Communications: `Client` and `Server`
- Navigation: `Navigation` and `Motor_Control`
- Localization: `Hokuyo_node`, `Hector_mapping`, and `Corner_detector`
- Obstacle Avoidance: `Kinect_srv`

- Map integration & Control: `Map_builder` and `Command_node`

Certain ROS node functionalities are clarified below, particularly for nodes which were not presented during Phase A.

- `Hector_mapping` is an open-source package that provides a map of the LiDAR's surroundings and the change in the LiDAR's pose.
- `Corner_detector` is a ROS Service that complements `Hector_mapping`;
- `Kinect_srv`, the node responsible for obstacle-avoidance. It uses `Libfreenect` drivers to obtain Kinect data and output the locations of the obstacles

Electrical system

Power Budget

Device	Current	Voltage
Wheels	4x 7 A	44.4 V
Auger	1x 80 A	22.2 V
Actuators	3x 17 A, 2x 5 A	11.1 V
5V BEC	2x 15 A	11.1 V
LiDAR	2 A	11.1 V
Kinect	2 A	11.1 V
Arduino	1 A	11.1 V

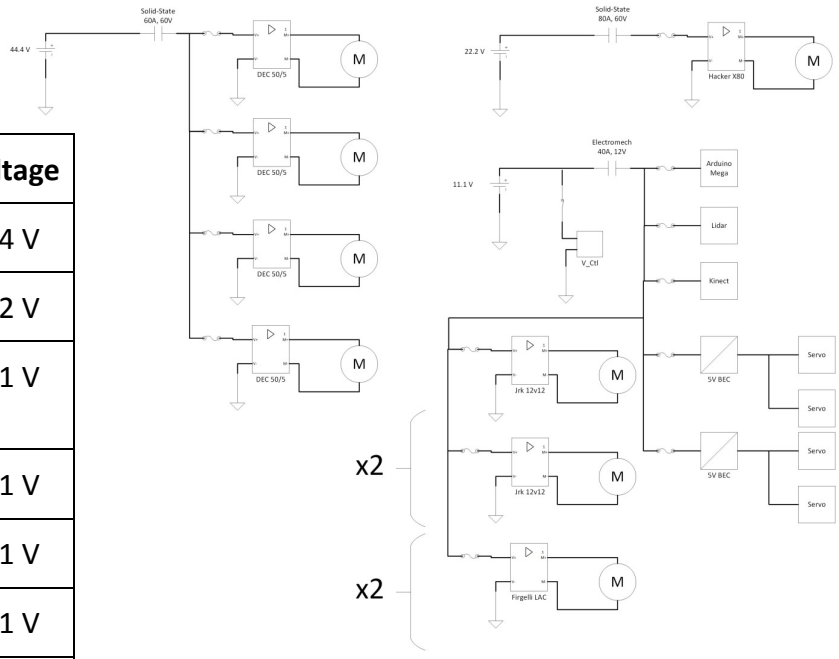


Figure 5 - Power System

Power Circuit

Based upon the required torque and an operational speed chosen to allow for two complete regolith collection cycles in a 10-minute match, the motors team began a search for a drive system with appropriate specifications.

Requirements:	Specifications to meet:
Torque	11 Nm to 14 Nm
Speed	18 rpm to 22 rpm

Based upon these specifications and more subtle analyses such as radial load capacity of the gearbox, we chose Maxon EC90 brushless DC motors equipped with Hall-sensor current feedback for speed control, mated to 74:1 reduction gearboxes.

In order to accommodate the drive motors, auger motor, and linear actuators, requiring 44V, 22V, and 12V, respectively, three power circuits will be used. The Zippy Flightmax batteries were chosen so that their charge would last a 20-minute session at full motor loads and with discharge ratings to allow for those full loads.

The E-Stop button was placed on the 11V power circuit. Pressing it opens the loop, signaling relays to activate on the high voltage circuits. Wire gauge varies throughout the Lunabot's circuitry, depending on amperage.

Control Circuit

Motor controllers were chosen for each of the 14 motors on board!

- 4 Maxon DEC 50/5 closed-loop speed controllers were the most affordable controllers that were compatible with our EC90 Brushless drive motors.
- Re-used Firgelli LACs work well for the 2 small door actuators.
- Pololu Jrk12v12 LACs can handle the 19 A max current in the other 3 actuators.
- 4 Servos have built-in controllers, and the Auger's Hacker X80 is reused

In order to have sufficient digital pins for all these motors, an Arduino Mega was ordered. Every PWM pin is accounted for, highlighting the benefit of careful systems interfacing. Arduino code was developed to map desired robot velocities to wheel velocities. Multiple steering schemes are supported by the omnidirectional locomotion design, and the optimum will be determined in testing.

Phase C: Final Design and Fabrication

Mechanical system

Excavation

The final design of the auger excavation system was analyzed in detail. Two different manufacturing options were studied: a composite-aluminum screw combining an aluminum drive shaft and carbon fiber blades, or a steel screw machined out of a single piece on a CNC machine. Both options were subjected to an extensive weight and mass flow rate analysis towards the goal of minimizing weight and maximizing mass flow rate. For the mass flow rate a simple relation between geometric parameter and operating velocities was found. The rate is

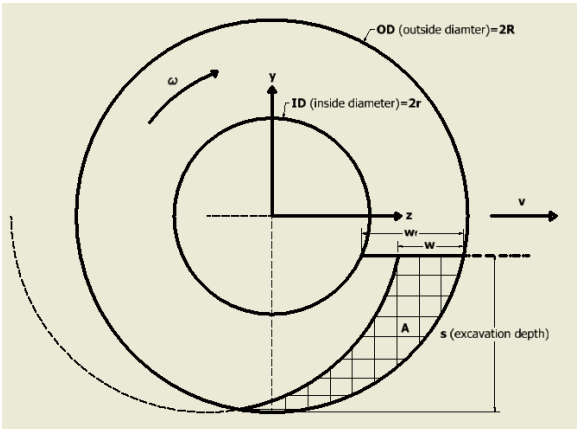


Figure 6 – Excavation Rate Calculations

limited by a choking condition where the mass inflow cannot be matched by the outflow. This condition is governed by the operating speeds and geometry of the screw:

$$2\frac{n}{w_f} < \left(\frac{\omega}{v}\right)$$

Here n is the number of turns in half the screw, w_f is the horizontal distance sand can travel inside the auger without reaching a choking condition, ω is the angular speed of the auger and v is the forward speed of the Lunabot. Several design iterations were considered for reasonable forward and angular velocities and a final design was selected with 5 turns and estimated mass flow rate of 1.72 kg/s. The drive train was reused with a maximum output shaft speed of 10 rev/s. Therefore the robot should not advance faster than 8.43 cm/s to avoid choking. Under these conditions the Lunabot should collect 50 kg in 30 seconds. The first manufacturing option was chosen because it provided a larger weight relief together with more affordable costs than the second option.

The bucket shape and the nozzle were designed based on parabolic trajectories of a distribution of regolith coming out of the auger with different angles and initial velocities. This shape was manufactured with fiberglass to optimize weight, cost, and customizability. The 4-bar dumping mechanism was optimized to sufficiently lift and tilt the bucket within the 8" stroke length of the actuator. Initially no configuration was found that would allow the desired range of motion without sacrificing all mechanical advantage in the actuator. Leverage was gained by placing the actuator's pin *inside* the bucket to increase the vertical component of the force.

Locomotion

The innovative wheel design consists of a deformable thermoplastic cellular structure surrounded by a 3D-printed outer tire. It takes advantage of the unique properties and flexibility that multi-material 3D printing technology affords by creating an outer tire that has rigid treads to improve traction, but at the same time is soft enough to deform along with the internal structure of the wheel. The precision of the 3D printer allowed us to closely match a complex cellular structure with the tire. The grousers designed based on suggestions in the literature (Paul W. Bartlett). Interfaces between the tire, wheel, hub, gearbox, and steering mechanism where carefully designed to avoid interferences and support the weight of the Lunabot.

The active suspension was refined to have one linear actuator per side, reducing the complexity from having two. The rocker-type system of linkages uses a parallel configuration to maintain the horizontal plane of the Lunabot. Simulations in Autodesk Inventor were used to calculate the required actuator force and to optimize the geometry for speed, force, and range. High precision Schaeffler bearings were used for all joints to minimize friction and play. High torque servomotors were chosen to actuate the steering mechanism because of their control simplicity. The servos are directly placed over the wheels so that little torque is required as the wheel drives forward. The drive gearbox is strong enough to support the radial load, isolating any weight from the motor. Geometric constraints allow 64º of rotation in mining mode and 180º in travel mode.

An analysis of the empirically derived equations for required driving and steering torques was attempted, but the models were complex and the team had little faith in the ability to choose appropriate coefficients. Instead, previous motors were observed to work with a nominal stall torque of 14 Nm. This became the baseline for comparison.

Manufacturing plan

The manufacturing plan took place on the second half of the academic year to meet the proposed manufacturing deadline of March 10th. McGill LunarEx Robotics organized parallel work streams for the different subassemblies of the Lunabot. The plan was to purchase the material, electronics, actuators, and other components of the robot during the month of December to start the manufacturing stage in January. The team put together a schedule to work in the machine tool laboratory to fabricate the mechanical components, as well as the composite parts. Simultaneously to the mechanical team, the electrical team set a schedule to connect, install and test individual components of the overall circuit. The CNC machined parts were outsourced and scheduled to start at the beginning of January to avoid any possible delays from the machine shop. The manufacturing deadline would allow the team to have a rolling chassis by the beginning of March, allowing the software team to take over the project and program the robot in order to start testing.

This plan was respected within its timeframe; however, the established deadlines where not always met; design, machining and components delays that were not accounted for in the original plan pushed our deadlines. The team took longer than planned working fixing design details of the Lunabot, which in turn delayed the project by a month. The parts that required conventional machining operation were completed on time with respect of the original timeframe. However, the composite manufacturing process took longer that expected together with the orders of CNC parts. On March 14th the team completed the mechanical system of the robot except for the auger mechanism that was completed on April 10th. The electric circuits were also completed in April due to delays of out-of-stock parts. Finally, minor changes were made until the final assembly on April 15th. The original planned timeline did not accounted for the months of December and April because of final exams; this helped to mitigate the pushed deadlines and was key to achieve our manufacturing objectives.

Software System

The Lunabot's high-level autonomous operation is summarized in Figure 9 in the Appendix; the implementation details of the major components follow.

Communications

The client uses multi-threading to operate both the GUI and the Socket concurrently. The GUI displays the data received from the Lunabot in a convenient, easy to read format using Java Swing, and detects key presses to send data to the server.

The client sends a byte array of length 6 to the server periodically. Each byte contains a code for a different command.

Navigation

The Navigation stack contains two major components:

1. AMCL, a probabilistic localization system for a robot moving in 2D
2. Move_base: Guides the robot to a destination, given global and local costmaps

Localization

Hector_mapping provides a map and robot pose, given LiDAR data, but the origin is defined as the point of initialization. A custom algorithm was implemented to detect the four corners on the map to give meaning to the coordinate system. The Hough Transform is applied by using a voting procedure to identify the most populated lines on the map: the four walls.

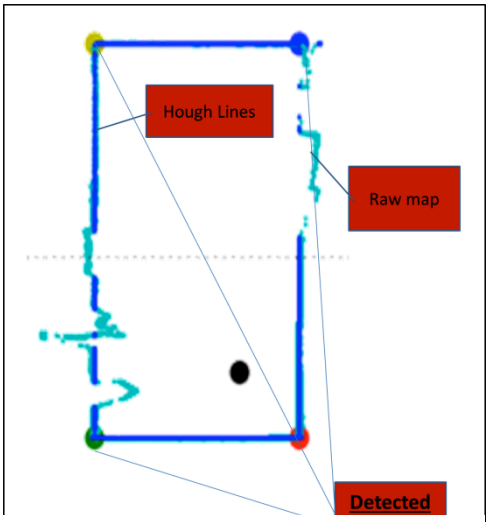


Figure 7 - Corner detection using the Hough Transform

Obstacle Avoidance

Two approaches were considered to perform obstacle avoidance:

- Geometrical Approach: using the depth image provided by the Kinect, it is possible to detect obstacles using vector geometry and filters.
- Machine Learning Approach: Using multiple labeled Kinect depth pictures of an environment containing rocks and craters, it is possible to use machine learning algorithms to 'teach' a program to recognize and locate obstacles in a picture.

After completing working prototypes for both approaches, the geometrical approach was chosen because (i) it is able to function in less predictable environments as it requires less calibration than the machine learning approach; and (ii) the lack of access to an environment that is similar enough to the competition environment to train the AI. The geometrical approach was implemented as follows:

- Get depth image as a 2d array.
- Transform each pixel into a vector pointing to the surface it displays using the pixel's depth value and its location in the image.
- Detect and correct for tilt and height using a selection of vectors
- Remap every vector to its (x,y) location on the floor and store its height at that point. This value can be either positive (for rocks and wall) or negative (for craters).
- Apply a Sobel filter to find the location of the obstacles.

Map integration & Control

The Map_builder node operates as follows:

- Subscribe to updates of the occupancy grid, the robot's pose from the Hector Mapping node
- Call the Kinect Node to obtain the locations of the obstacles
- Map the obstacles in the global coordinate system
- Compute the cost of each cell in the occupancy grid by taking an average of the N past values from the obstacle detection, where N is to be calibrated

The Command_node's operation involves implementing the control shown in Figure 9 in the Appendix.

Electrical System

Power Circuit

Upon the arrival of the linear actuators and servos, the necessary changes to the robot's frame and wheels were made to allow for the parts to be installed. Since the servos are powered by 5 VDC, the purchase of battery elimination circuits (dc/dc converters) was necessary to drop the battery's 11.1 V down to 5 V.

The battery housings were manufactured out of aluminum for thermal dissipation. Wires were prepared, cut, and attached to connectors for quick assembly, since they would be used with external power supplies long before the new batteries would be attached. The negative reference terminals of each battery bus were connected at a single point to ensure an equipotential reference at "common" or "ground" (especially important because the signals from the Arduino are referenced to common on the 12 V system, yet they are read by devices referenced by all bus voltages).

Control Circuit

Since the DEC 50/5 speed controllers were packaged as printed circuit boards, it was necessary to manufacture interface boards to allow for the proper connections to be made both between the controllers and outside devices, but across the controllers themselves. These boards were designed in accordance with the instructions given by the speed controller manufacturer (Maxon). Also, rudimentary DACs were built with low-pass filters to convert Arduino PWM output to analog levels for the DEC 50/5 speed control input. The control parameters of the Firgelli and Pololu controllers were adjusted via supplied software to allow for safe and smooth operation of the linear actuators under their operational conditions.

The Arduino was fixed at the center of our electronics board to allow for convenient access to all other controllers. The Kinect and LiDAR were positioned in the areas arranged for them in the frame design. The Arduino code and ROS code were verified to work together properly. The current and voltage sensors were placed on the 3 buses right after the power relays.

Financial Budget

At the critical design review, the projected budget was verified and deemed to be within affordable, thanks to generous sponsor contributions. The total came to $14,420. As of April 2013, the total spending is 13,370.71 – under budget. The details are included in the appendix.

Phase D: System Assembly, Integration and Test, Launch

Mechanical System

Final mass budget

	Excavation	Dumping	Frame	Electronics	Actuators	Wheels	Locomotion	Total
Estimated Mass [kg]	12	7	10	7	15	8	7	**66**
Actual Mass [kg]	8.11	8.66	8.68	7.13	21.59	8.07	11.21	**73.44**

The estimated mass was consistently underestimated by approximately 10%. Fortunately a safety margin was allowed before reaching 80 kg.

Excavation

The auger and bucket were assembled according to plan. The 4-bar dumping mechanism originally designed had to be modified to avoid a singularity position that caused problems when the bucket was in the low position.

Locomotion

The manufacturing of the wheels followed a 3D printing sampling process: testing different material ratios to ensure a mix that gave the desired mechanical properties. When the requirements were met, all four tires were manufactured. The original configuration of the cellular structure of the wheels proved to be too stiff. So, certain cells were strategically removed to obtain the desired deformation under typical loads.

The assembly of the active suspension linkages with the bearings had a lot of play in the connections. It was found that the main reason was the tolerances on the connectors and spacers. High-precision parts were purchased to solve the problem.

Software system

Communications

Correct operation of the telecommunication system was tested by running the client, connecting it to the server machine through a TCP/IP connection, and sending commands to the server. Three main functions of the telecommunications client need to be tested:

1. Sending a start signal and observing the robot activate autonomous mode
2. Sending an override signal and observing the robot activate teleoperation mode.
3. Monitoring feedback data sent from the robot to the client, and comparing against the onboard laptop and observed robot behavior.

Navigation

Testing was done in Stage, a robot simulator for ROS, in four phases:

1. In an empty arena: getting the robot to go to a goal position without obstacles
2. In an arena with obstacles. Put simulated obstacles in the arena and test the obstacle avoidance function of navigation (still doing...).
3. Testing navigation on the physical robot, with a hard-coded map of its environment, to observe the command velocity outputs of navigation
4. Testing navigation on the physical robot in a realistic environment

Localization

Initially, the localization routine was designed to be run just once, after the robot had performed a 180◦ rotation at the start of its run for Hector_mapping to accumulate a complete map of the arena. In the integration stage, however, we realized that it would be desirable to be able to call the localization node at will during the run to refine the estimate of the corners' positions, and benefit from hector mapping's more complete map that accumulates with more scans. As such, the node was made into a ROS Service that can be called from any other node in the system.

To test the corner detection node, the LiDAR was rotated by 180◦ in a room with dimensions similar to the arena's. During the rotation, Hector_Mapping was running and the state of the ROS system during this rotation was captured using a testing tool called ROS bag. The bag was visualized to ensure correct data acquisition, and the corner detector was called with the last map from the ROS bag.

Obstacle Avoidance

Modifications

The obstacle avoidance algorithm was initially designed to output the obstacles (if any) found at a set rate throughout the run. This was subsequently modified (by making the node into a ROS Service) to allow for the Map_builder node to query the Kinect when it was necessary to find the surrounding obstacles. The advantage of this method is (i) to minimize computational load by only running the obstacle avoidance algorithm when needed; and (ii) only calling the obstacle avoidance service when the robot is in the obstacle area, thus avoiding obstacles being erroneously placed in an area of the arena where there are no obstacles.

Testing

The obstacle avoidance node was tested in 3 phases:

1. On a stationary surface visualizing the output as objects of varying sizes were put in its field of view.

2. The node was tested on a moving platform, where it was fully integrated with the hector_mapping and Map_builder nodes to create complete occupancy grids.

3. The SLAM systems will have to be tested on the robot in an environment similar to the Lunarena. This will be done both locally in Montreal and in Florida when we get the chance to test our systems in the Lunarena before the competition.

Electrical System

Power Circuit

In order for the drive motors to run properly with the speed controllers, wire harnesses were fabricated to mate the Molex connectors of the motors with the screw terminals on the motor controller housings.
The linear actuators were connected to the linear actuator controllers and the servos were connected to the BECs for power and the Arduino for the PWM signal.
The batteries were installed with the complete power distribution system in order to evaluate the performance of the robot with the final power supply.

Control Circuit

All the motor controllers were implemented with their appropriate control signals coming from the Arduino. The Arduino code was modified as needed to account for changes in driving conditions experienced when testing under mock competition conditions. The Kinect and LiDAR were tested when exposed to dust on the operating robot. Power monitoring data collection was tested and the final energy consumed calculations were checked with estimated to ensure that they were valid.

Phase E: Operations and Sustainment

Risk

Despite every effort to build a reliable system, there is always some risk of failure. A graph plotting each of the possible failure modes at competition against their probabilities of occurrence and consequences can be found in Table 3 of the appendix.

Software

Communications

The procedure to run the Lunabot's communications platform is as follows:

1. Run the `roscore`, Python Server and Java Client
2. The server waits for a connection automatically; pressing SPACE in the client establishes a connection
3. The client detects keyboard input and changes the data in the byte array
4. The client sends the byte array to the Server periodically
5. The server receives the data from Client, parses it and publishes the information to corresponding ROS topics
6. The server sends the key robot status indicators back to the client periodically
7. The client displays the status of the robot accordingly

Localization

To run the SLAM system, the following nodes are to be run in this order: roscore, Hokuyo_node, Hector_mapping, and Corner_detector

Obstacle Avoidance

The calibration in Florida will involve finding the right thresholds for the filters in order to ensure that all obstacles are detected but that random terrain features like the robot's tracks aren't wrongly classified as obstacles.

Phase F: Closeout

The completion of the Fourth Annual NASA Lunabotics Mining Competition on May 24, 2013 will mark the end of the largest task ever undertaken by the members of McGill LunarEx Robotics. Over the course of a year, students from across the Faculty of Engineering came together to learn about how their expertise can integrate with others. Guided by the systems engineering process, Mechanical, Electrical, and Software components were developed and integrated into a functional lunar rover.

May 24th is more than just an ending, however. It marks the beginning of the 2014 McGill LunarEx Robotics team. With over 20 members attending the competition, the new cycle will begin with a strengthened base of experience that can be leveraged to teach new students about robotics, and transfer skills to younger peers.

Meeting with students from around the world will be incredibly inspiring and the team will return to Montreal with a wealth of new ideas for improvement next year. The first step on the agenda is to renovate the workshop, optimizing the space for design teams and clubs at McGill.

Conclusion

The Systems Engineering process guided the development of a Lunabot from an idea to a competitive robot. System reviews after each phase of development kept the team on track, assured interfacing compatibility, and drew attention to areas that design flaws. The timeline is currently on track, with operations scheduled to begin on May 20, 2013. Testing is scheduled for the month of May, where system requirements will be verified.

The quest to gain experience drove innovation for McGill LunarEx Robotics. Beyond developing novel robotic systems such as an auger or an autonomous controller, at the end of the day it’s the team that makes or breaks the system. We have learned how to work with engineers of all disciplines and how the dynamics of diversity can be an incubator for brand new ideas.

References

Hokuyo. (2012, 11 27). *UTM-30LX/LN Specification .* Retrieved from http://www.hokuyo-aut.jp/02sensor/07scanner/utm_30lx.html

Nildeep Patel, R. S. (2010). The ExoMars rover locomotion subsystem. *Science Direct* .

OpenKinect. (n.d.). *OpenKinect - Main page*. Retrieved 04 19, 2013, from http://openkinect.org/wiki/Main_Page

Paul W. Bartlett, D. W. (n.d.). Traps, Design of the Scarab Rover for Mobility & Drilling in the Lunar Cold. *Carnegie Mellon University* .

ROS. (n.d.). *navigation - ROS Wiki*. Retrieved 04 20, 2013, from http://www.ros.org/wiki/navigation

Appendix

Table 3 - Lunabot risk assessment

Probability \ *Consequence*	1	2	3	4
1	Wheel cellular structure breaks	Bolts come loose	Wheel oversteers due to software glitch and interference breaks something	Batteries run out
		Regolith sticks to lidar		Communication is lost with Client
		Dust messes up vision systems		Arduino wires fall out
				Actuation rips out cables
2	Kinect fails to recognize obstacles	Lidar loses arena when robot tilts	Wheel material breaks	Wheels don't have traction
			Servos break	Dust gets in a motor, causing it to fail
			Auger gets stuck on rock	Overcurrent fries electronics
			Suspension can't lift the full weight	
3	Auger blades snap/abrade			Robot topples due to bad center of gravity
4	Regolith leaks out of seams	Out of sync motors/actuators		
	We don't collect the right amount of regolith (too much or too little)			

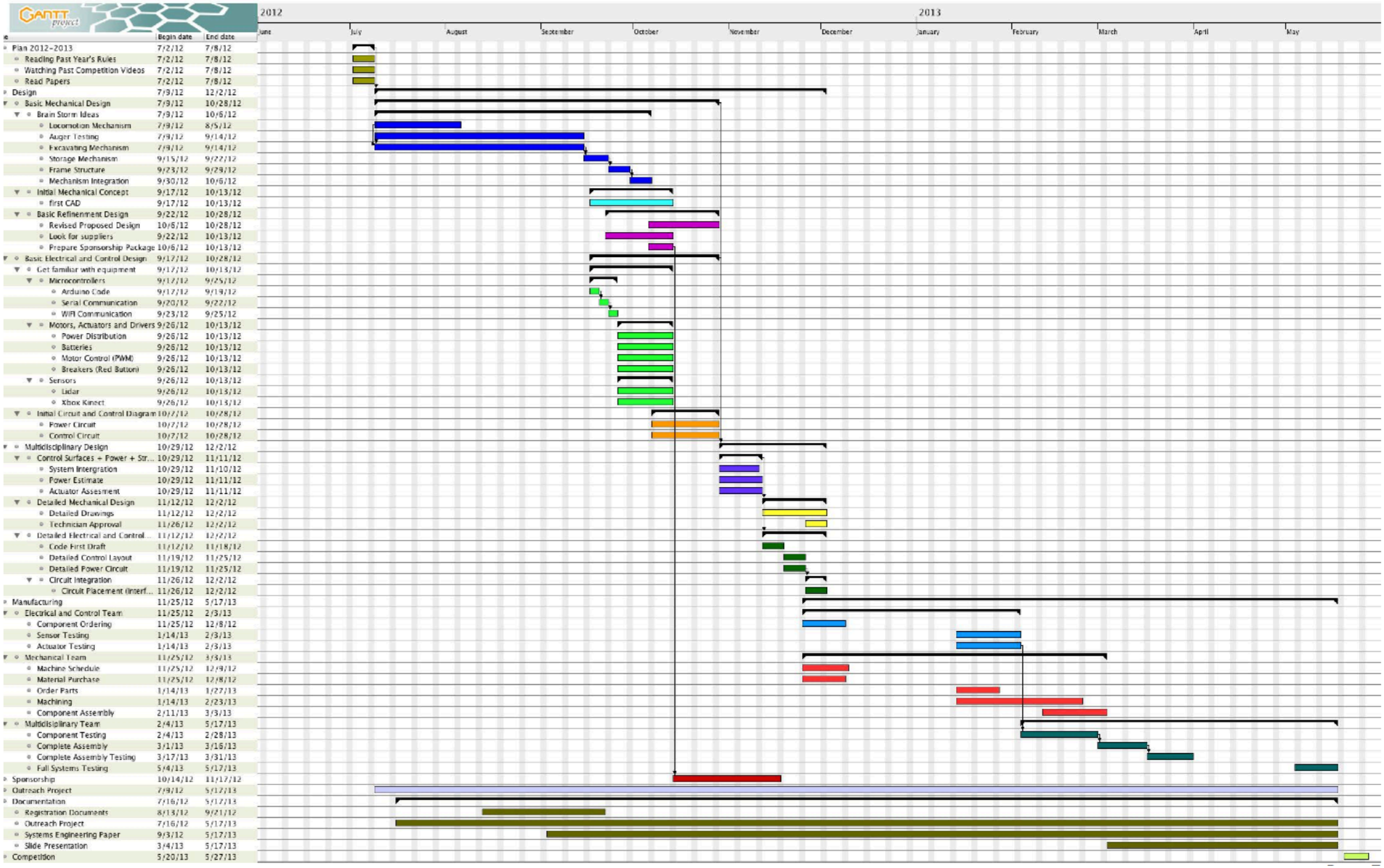


Figure 8 - Preliminary Gantt chart of the full project

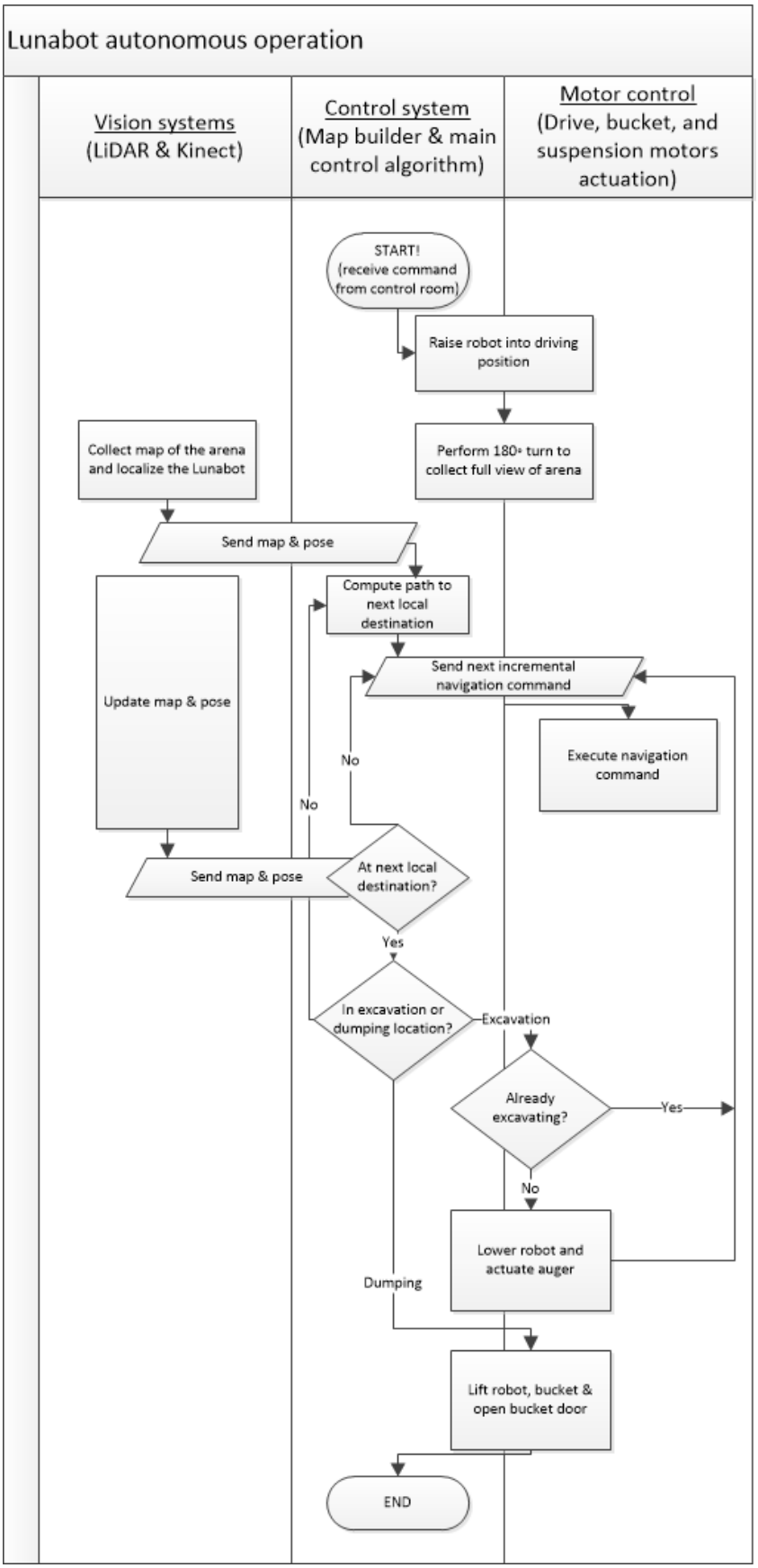


Figure 9 - High-level Lunabot software operation

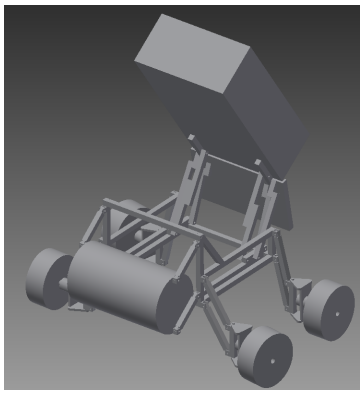


Figure 12 - CAD 1

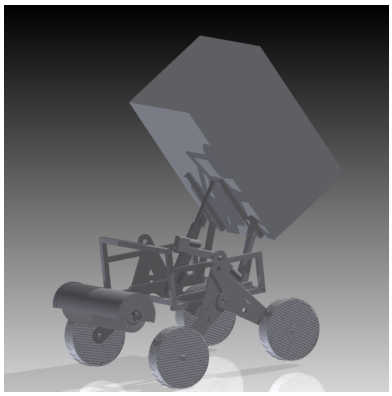


Figure 11 - CAD 2

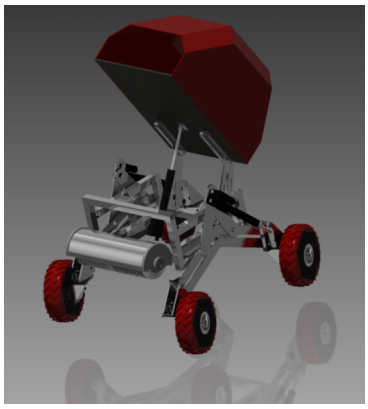


Figure 10 - CAD 3

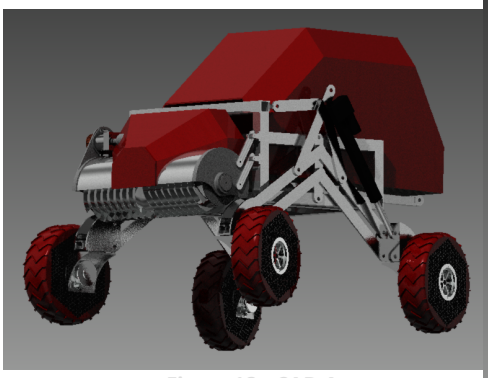


Figure 13 - CAD 4

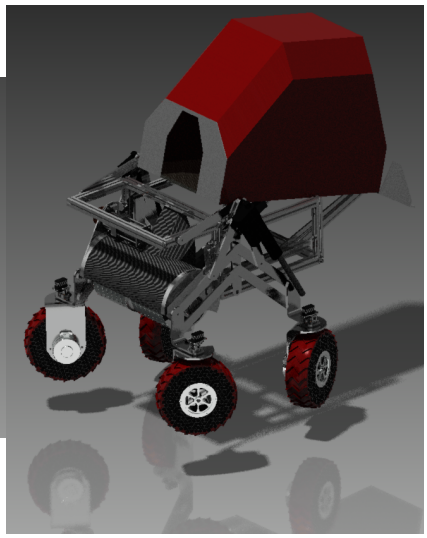


Figure 14 - CAD 5

Figure 17 - CAD 6

Figure 16 - CAD 7

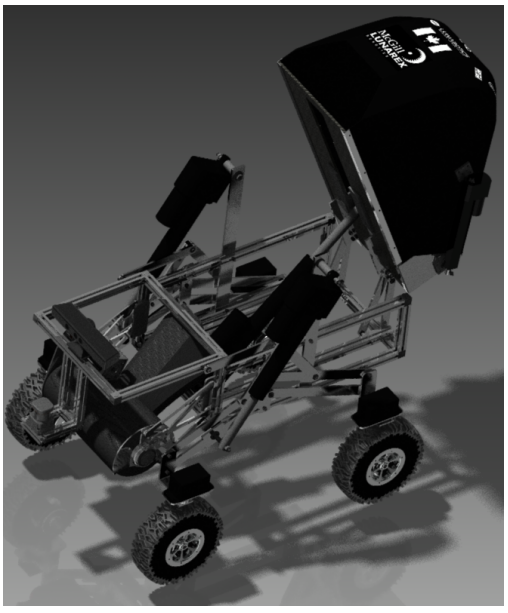


Figure 15 - CAD 8

Financial Budget

Category	Projected Cost	Actual Cost
Testing Facility	$500	$352.91
Stock Metal	$1000	$750.00
Servos and Linear Actuators	$3,500	$3,607.89
Motors/Gearboxes/Motor Controllers	$3,320	$3,403.63
Composites	$800	$589.46
Miscellaneous Supplies (paint, tools, bolts)	$1500	$1,033.83
CNC Machining	$1,600.00	$1,600.00
Batteries	$700	$544.55
Wires, Sensors, Fuses, Diodes, Sockets	$500	$606.25
Shipping/Service Charges/Service Fees	$500	$454.30
Motor Controllers + Microprocessor	$500	$427.89
TOTAL	**$14,420**	**$13,370.71**

Acknowledgements

McGill LunarEx Robotics would like to acknowledge the support we have received throughout the year. Without the sage guidance of our advisors and the generous contributions of our sponsors, we never could have come this far.

Mentors & advisors: Prof. Radziszewski, Dr. Martins, Mathieu Beauchesne, Sam Minter, Andreas Hofmann, Tony Micozzi, McGill Racing Team, McGill Baja Racing, Taryn Tomlinson, Michel Wander, Sebastien Gemme, Anthony Marineau, Prof. Sprecher, Prof. Pasini, Francois Leblanc, Prof. Barfoot, Prof. Angeles, Prof. Kovesces,

Sponsors: Schaeffler, Stratysys, EUS, Quebec, Raytheon, SSMU, Metso, LIPHE, MESC, Maxon Motor, Electromate, MDA, Willow Garage, MIAE, S&C Electric, ServoCity, Joachim Fricke Management Service