Universität der Bundeswehr Hamburg
Fachbereich Maschinenbau
Laboratorium Fertigungstechnik
Prof. Dr.-Ing. Jens P. Wulfsberg

HELMUT-SCHMIDT
UNIVERSITÄT
Universität der Bundeswehr Hamburg

Diplomarbeit

„Development of a Cool Robot for the Antarctic“

Guido Gravenkötter
Wirtschaftsingenieurwesen 2000
e776220

Gunnar Hamann
Wirtschaftsingenieurwesen 2000
e776240

THAYER SCHOOL OF ENGINEERING
DARTMOUTH COLLEGE

Betreuer: Prof. Dr. Laura Ray

Ehrenwörtliche Erklärung

Hiermit versichere ich, dass ich diese Diplomarbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen hat.

Alle Ausführungen, die wörtlich oder sinngemäß übernommen worden sind,
sind als solche gekennzeichnet.

Hanover, N.H., USA, den 4. März 2004

Guido Gravenkötter

Gunnar Hamann

Table of Contents

List of Figures

Nomenclature

a	m/s^2	Acceleration
A	m^2	Area
a_t	m/s^2	Tangential acceleration
A_H	N	Horizontal force on bearing A
A_{total}	m^2	Emitting surface
A_V	N	Vertical force on bearing A
b	m	Beam width
B_H	N	Horizontal force on bearing B
C_0	kN	Static load rating
d	m	Diameter
D	m4Pa	Panel bending stiffness
E_c	MPa	Compression modulus of core
E_f	MPa	Modulus of elasticity of facing skin
F_G	N	Weight Force
F_M	N	Transverse force on one motor
f_s		Static performance figure
G_C	MPa	Core Shear modulus – in direction of applied load
G_L	MPa	Core shear modulus- Ribbon direction
G_W	MPa	Core shear modulus – Transverse direction
i_{Gear}		Gear transmission ratio
I	m^4	Geometrical moment of inertia
I_{Motor}	A	Motor current
k_b		Beam – Bending deflection coefficient
k_s		Beam - Shear deflection coefficient

K_T	Nm/A	Torque Constant
l	m	Beam span
L_{10}	10^6 h	Life time of a bearing
m	kg	Mass
M_b	Nm	Bending moment
P	N	Applied load
P_0	kN	Static load
P_{Motor}	W	Motor power
q	J/s	Heat flow
Q	J=Ws	Heat energy
R	N	Total Rolling Resistance
R_1	m	Inner diameter
R_2	m	Outer diameter
R_w	N	Rolling Resistance at one wheel
R_i	N	Internal Rolling Resistance
R_s	N	Snow Rolling Resistance
t	s	Time
T_1	Nm	Total Torque Demand
T_θ	Nm	Momentum due to mass moment of inertia
t_c	mm	Thickness of core
t_f	mm	Thickness of facing skin
U	V	Voltage
v	m/s	Velocity
W	m^3	Section modulus
δ	mm	Calculated deflection
ν,S		Safety factor
θ	kgm^2	Mass moment of Inertia
$\sigma_{b,max}$	N/mm^2	Stress because of bending
σ_c	N/mm^2	Core compressive stress
σ_f	N/mm^2	Calculated facing skin stress

σ_v	N/mm^2	Equivalent stress
τ_c	N/mm^2	Shear stress in core
τ_s	N/mm^2	Shearing stress
$\Delta\upsilon$	°C	Temperature difference inside/outside the box
ω	rad/s	Angular velocity

The meaning of other symbols is apparent from the description of the specific section.

Abbreviations

AC	Alternating Current
ADC	Analog-Digital Converter
ATV	All-terrain Vehicle
BLDC	Brushless Direct Current
CMU	Carnegie Mellon University
CRREL	U.S. Army Cold Regions Research and Engineering Laboratory
DAC	Digital-Analog Converter
DC	Direct Current
DGPS	Differential Global Positioning System
ECDF	Empirical distribution function
ECU	Electronic Control Unit
EOD	Explosive Ordinance Disposal
FEM	Finite Element Method
GPS	Global Positioning System
JPL	Jet Propulsion Laboratory
K-S	Kolmogorov-Smirnov Goodness-of-Fit Test
MBT	Main Battle Tank
MIT	Massachusetts Institute of Technology
NASA	National Aeronautics and Space Administration
NMEA	National Marine Electronics Association
PCM	Phase Change Materials
PTFE	Polytetraflourethylen
PWM	Pulse Width Modulation

RPM	Revolutions Per Minute
SCARA	Selectively Compliant Articulated Robot Arm
SPI	Serial Peripheral Interface
UAGV	Unmanned Antonomous Ground Vehicle
UGV	Unmanned Ground Vehicle
VIP	Vacuum Insulation Panel
WEB	Warm Electronics Box

Acknowledgements

This work was funded by the National Science Foundation

We would like to thank the following individuals for their help to complete this thesis:

Dr. Laura Ray –
For your counsel, support and wisdom throughout our stay here. Your open door and availability to help us has been incredible and has made our experience at Dartmouth one we will always cherish.

Dr. James Lever –
For your great support and help with your exceptional expertise on both the Antarctic region and robot design

Dr. Marc Lessard –
For imparting a minute amount your experience with project at the Antarctic to us

Dr. Horst Richter –
For building and supporting this exchange program and for your great support with the organization of this stay

The Thayer School Machine Shop –
Leonard, Mike, Pete, and Kevin – Thank you for your time, advice and humor

The Thayer Instrument Room –
Roger, Renee and Phil – Thank you for your efforts and support

The Thayer School community –
For making this stay an unforgettable great experience

Chapter 1 *Introduction*

Magnetometers record the way the Earth’s magnetosphere reacts when struck by the solar wind, a flow of particles from the sun. The collision causes auroras, electric currents and geomagnetic fields. It can also disrupt satellite communications, damage power lines, induce currents in underground pipelines and cause corrosion. If the data could be collected in a reliable way on the Antarctic plateau, it could be used to predict such problems.

Several Robots have been built to explore the Antarctic. They were all large, heavy and therefore driven by a combustion engine. But a robot, powered by a renewable source like solar cells and capable of traversing long distances autonomously doesn’t exist.

This thesis describes the conceptional work on a robot, which can be deployed around the South Pole station in order to record continuously data. The robot is designed to carry different instruments and sensors according to the mission. The new approach to power the robot with solar cells has a major influence on the mechanical design of the robot. All components have to be selected to aim a low total vehicle weight, which leads to a low power consumption. They have to work reliable in the harsh environment of the Antarctic. The effect of the cold temperature on the components will be tested in a cold chamber.

Waypoint navigation from one point to another has to be implemented without a magnetic compass, which is not working at the Antarctic. Thus, an algorithm has to be developed that is only based on a GPS signal and will provide the robot controller with the steering information.

With the market offering commercial solar panels at a high efficiency, the idea of an autonomous robot is now viable.

Chapter 2 Background

The following chapter describes first the overall project, emphasizing the main challenges. Secondly, the specifics of the Antarctic as the application area of the robot will be introduced. Finally state-of-the-art robot projects and solutions and their relevance to the project will be shown.

2.1 Project description

The project is to design a robot that can deploy various instruments autonomously during the summer months on the Antarctic plateau around the South Pole. The design will be scalable for deployment of small (up to five kg) instruments to large (greater than 25 kg) sets of instrumentation.

The project started in September 2003 and will last for eighteen months through December 2004. A single prototype robot will be constructed and evaluated for mobile robot-based instrument deployment.

This thesis covers the concept phase of the project including the evaluation of different design approaches, the selection of components and the testing of components and algorithms.

The robot will be scalable in the sense that the drive system, power system, chassis, and navigation algorithms can be scaled for small, low power versions of the robot to larger robots. One can envision deploying robot networks from the South Pole station, each with instruments secured, to desired geomagnetic grid locations on the surface of the plateau for long or short term observation, and retrieving or repositioning the network through Iridium-based communication links.

The deployment of a magnetometer to measure the Earth's magnetic field at the South Pole represents the main focus. However, with a stable robotic platform the potential exists to deploy different instruments for multiple mission requirements and reduce the need for manned stations. Possible other tasks for the robot would be:

- Long- and short-term automated monitoring functions in the Antarctic region as atmospheric sampling, surface weather monitoring, sampling for surface contaminants in soil, snow and ice, local population/ecology monitoring with remote cameras mounted on robots, and GPS-based glaciology study
- Enhanced capability for intelligent monitoring, e.g., autonomous redeployment of robot platforms based on results of ground-based observation of solar-terrestrial physics, glacial movement, or ecological monitoring
- Field support in the form of local area networks, data storage and transfer, and mobile power
- Deploying instrumentation in regions that are inaccessible or pose undue safety risks or ground site preparation for aircraft access.

Based on the scalable mobile robot design concept, a single robot will be constructed to carry instruments, its electronics and an Iridium modem. The robot design and navigation concepts will build on analogous systems design issues faced in, for example, the Mars explorer and Lunar Rover design with a low cost approach.

Though economics are one issue, freeing other resources, such as aircraft use, is valuable regardless of operational costs, due to the short summer season.

Harsh weather characteristics of polar environments along with substantial range, navigation, and variable terrain pose significant system design challenges in the development of unmanned vehicles for coordinated sensor deployment. In addition to challenging mobility specifications, instrumentation will be deployed for long periods of time in drifting snow and must have a stable operating environment with vibration and electromagnetic characteristics that provide a low noise floor. Sensors should be retrievable with high reliability to minimize both environmental impact and cost. The project will develop specifications for deployment, and will develop engineering concepts for meeting mobility, range, and navigation requirements in snow cover and weather characteristics of the Antarctic plateau. Economic tradeoffs between mobility and navigability of mobile robots will be established in order to predict the point at which performance capabilities of ground-based deployment exceed the cost of air-based deployment and retrieval. A model of the most promising robot design will be developed and tested in terrain characteristic of that at which the robot must operate, and robot components will be tested at temperatures at which the robot must operate.

To minimize environmental impact and non-renewable resource use, solar-electric designs are preferred, although economic and performance tradeoffs for non-renewable high energy density sources will be considered.

The project is sponsored by the Thayer School of Engineering at Dartmouth College, the Cold Regions Research and Engineering Laboratory (CRREL) of the US Army in Hanover, New Hampshire and a grant of the National Science Foundation.

Prof. Dr. Laura Ray of the Thayer School of Engineering and Dr. Jim Lever of CRREL advise the team of students involved in the project. The project group is divided into three areas. Guido Gravenkötter is responsible for the overall design of the robot. Gunnar Hamann takes over the navigation and control system and Ian J. Kahn, Xianghui Weng and Samuel B. Kasdon will design the power system.

The following aspects are the most challenging and critical requirements for this robot project that must be met by all three groups:

- The ability to traverse 500 km in about two weeks on dense snow and over small snow drifts (sastrugis)
- Sustain mobility in windy, drifting conditions while gathering data through the deployed instruments
- Minimize environmental impact
- Operate in summer temperatures of as low as –40° C
- Find low-cost and reliable solutions to make the robot a competitive solution for instrument deployment to existing air and man based approaches.

The main goal of the project is to show the feasibility of meeting the above mentioned constraints with an autonomous, scalable, reliable and low-cost robot [RAY03].

2.2 The Antarctic

The following description of the Antarctic focuses on the area at the Antarctic Plateau at the South Pole. It stresses the characteristics that are most important for the development of an UAGV that will be deployed in this region.

The Antarctic extends from latitude 60° S to the South Pole at 90° S. This includes the continent of the Antarctica, its off-lying islands and the encircling Southern Ocean.

Its massive ice sheet, extending over an area roughly the size of Australia and in places more than four kilometres thick, covers 98 percent of the continent and stores 80 percent of the earth’s fresh water. Most of the rock beneath the ice is depressed by the burden of the ice.

Antarctica’s origins have been traced to Gondwana where 140 million years ago it formed the centre-piece of that ancient southern super-continent, surrounded by present-day Australia, India, Africa and South America.

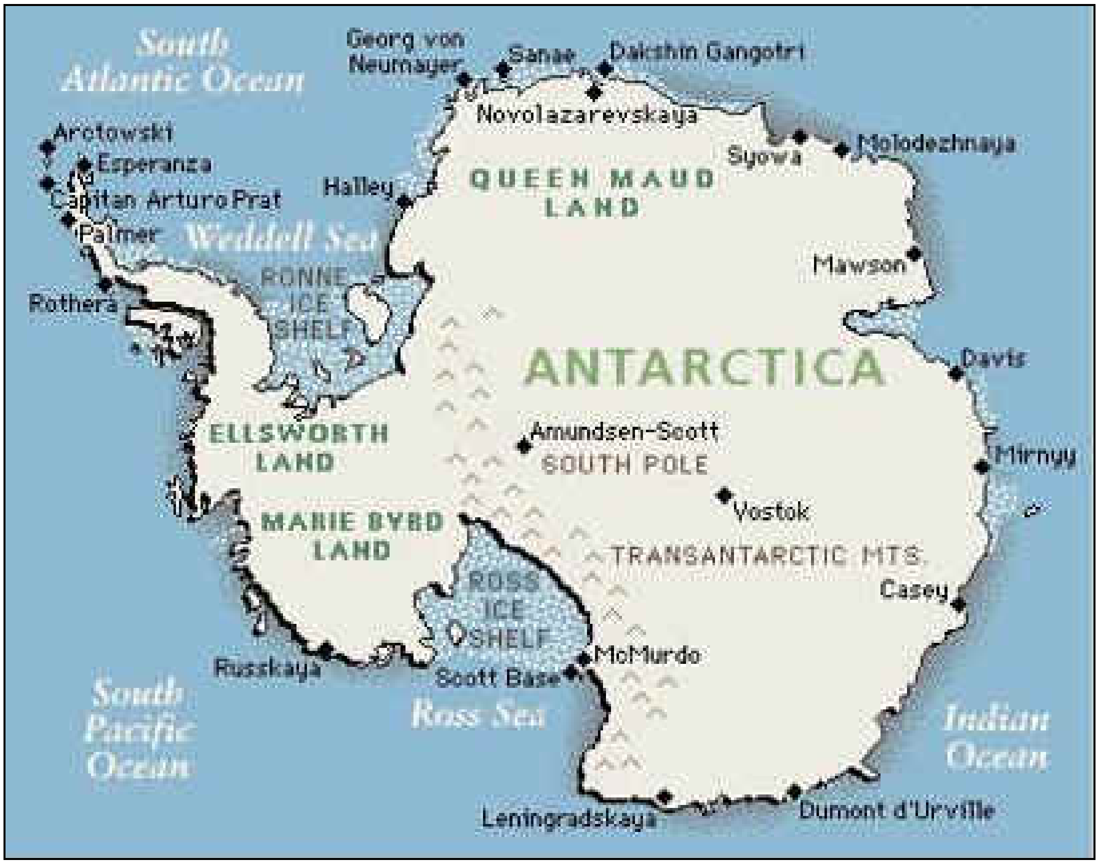


Figure 2.1: Map of the research stations on the Antarctica

The first historical records of the Antarctic come from the journals of Captain James Cook's second circumnavigation of the world. The expeditions to the Antarctic climaxed in the so-called 'heroic era' from the turn of the 20th century up through the First World War with the land-based explorations of men such as Robert F. Scott and Roald Amundsen. Subsequent expeditions initiated the tradition of joint national efforts which culminated in the International Geophysical Year (IGY) of 1957-58. During the IGY, research stations were established on the continent to investigate various branches of the biological and physical sciences. In 1961 the Antarctic Treaty System was ratified by twelve countries to protect the Antarctic and to open it for research. The effects of the increased international awareness and interest in the Antarctic are included in the Protocol of Environmental Protection of 1991 which prohibits the digging for mineral resources for another 50 years [HOR94, KIN69]. Even today it takes almost a week to get to the South Pole from the USA. Military transport planes fly from Christchurch in New Zealand to the American McMurdo station. Another plane covers the last 1500 km to the Amundsen-Scott station at the South Pole.

The continent is the highest, driest, coldest and windiest on earth. The lowest temperature ever measured on earth of -91° C was recorded at the abandoned Soviet Vostok station in 1997. At the South Pole the temperature varies from -20° C at the height of the summer in February to -70° C in mid-winter. The average temperature over the year is -49.5° C (**see figure 2.2**). During the summer months from November through February the sun angle climbs towards its peak on December 21 of about 23 degrees above the horizon. In the summer the sun is shining for 24 hours a day whereas in the winter it is completely dark.

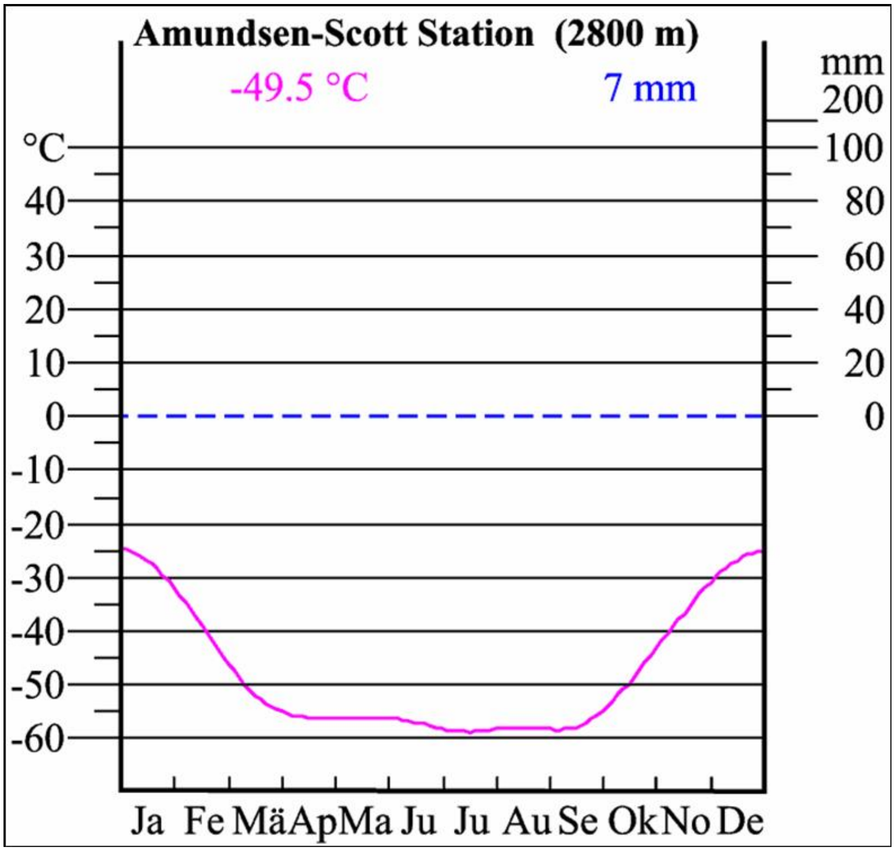


Figure 2.2: Temperature in degrees Celsius recorded at the South Pole

Winds in the Antarctic are mostly caused by heavy cold air located of the Antarctic plateau falling under gravity towards the sea. As the air falls it gathers speeds and reaches over 300 knots at the coast. At the South Pole the air is still moving slowly with top speeds of 30 knots. Also slow motions of the air raise the dry snow crystals from the ground and cause snow drifts. In combination with low clouds and diffuse light the snow drift increases the loss of contrast visibility and results in ‘white outs’ which sometimes last for a few days.

Despite all the frozen water the mean annual accumulation for the entire continent amounts to less than five centimetres of water equivalent. This is slightly more than the Sahara Desert and makes the Antarctic the driest continent [WEB01].

Due to these attributes there is almost no life outside the research stations. The biggest animal living on the continent is a small insect with a length of 12 mm. Only at the shore and islands around the Antarctica some animals such as penguins and whales live from the huge amounts of plankton in the sea.

2.3 State-of-the-Art Robot Projects

This section will introduce the reader to state-of-the-art robot programs particularly in regard of the cool robot project and its relevance. It will discuss mobile robots that are exposed to similar environments. The lessons learned from these projects were considered in building the COOL ROBOT. All described Robots are or have been deployed successfully.

2.3.1 NOMAD (CMU / NASA)

NOMAD is an autonomous mobile robot from Carnegie Mellon University (CMU), Pittsburgh, Pennsylvania and the National Aeronautics and Space Administration (NASA), and it was developed to traverse planetary analogous terrain. It is the first robot to “exhibit intelligence to explore an extreme polar environment and automatically classify indigenous rocks” [WEB03].

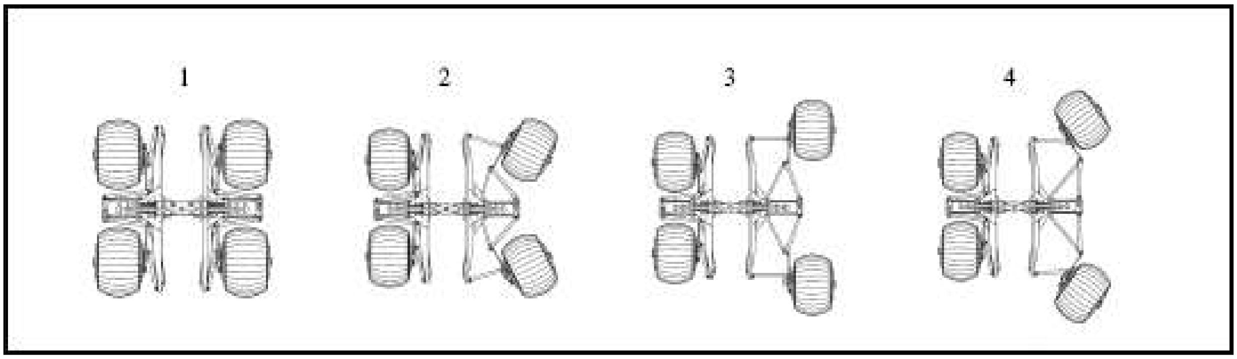


Figure 2.3: Transforming Chassis – Range of Motion for One Side [SHA98]

The 725 kg four wheeled robot (2.4 x 2.4 x 2.4 m) powered by a gasoline engine uses the Differential Global Positioning System (DGPS) to determine its location. One great feature of the mechanical design is the transforming chassis that can expand or compact by driving two pairs of four-bar linkages with two electric motors. Another feature is the in-wheel propulsion: all drive components are sealed within the wheel. This prevents any geometric or operational interference between the systems. No electromechanical components are needed for propulsion beyond those enclosed in the wheel. The advantages include stability from individual traction control, increased control flexibility and durability from sealed drive units.
The NOMAD carries a numerous set of sensors: three cameras (stereo, high resolution and panoramic), a laser rangefinder to detect obstacles in the robot's path and a spectrometer mounted on a manipulator arm (SCARA) in order to perform remote science.

In 1997 NOMAD executed the first mission in the Atacama Desert of southern Chile. It traversed 223 km and demonstrated successful teleoperation of a wheeled vehicle over a long distance [SHA98]. Rigid tires have been developed for the desert terrain, where elastomeric tires are inappropriate due to the heat.

In 1998 and 2000 CMU/NASA started an Antarctic Expedition using NOMAD. For this mission, NOMAD's task was to analyze meteorites, which are very well preserved in the Antarctica's frozen environment for thousands of years. It is known as the best place on earth to harvest meteorite specimens and thus it is of tremendous value for scientists. In contrast to the rigid tires for the desert, they used pneumatic tires, specialized for Antarctic terrain. Pneumatic tires with an aggressive rubber tread pattern and embedded metal studs increase the traction on snow and ice.
The mission was successful: NOMAD found and classified five indigenous meteorites. Two major lessons were learned from this project: A travel speed of 0.15 m/s (0.54 km/h) is way too slow for the mission requirements. Second, the high-resolution camera didn't work properly, because the bright sunlight reflecting off the ice closed the lens auto-iris.

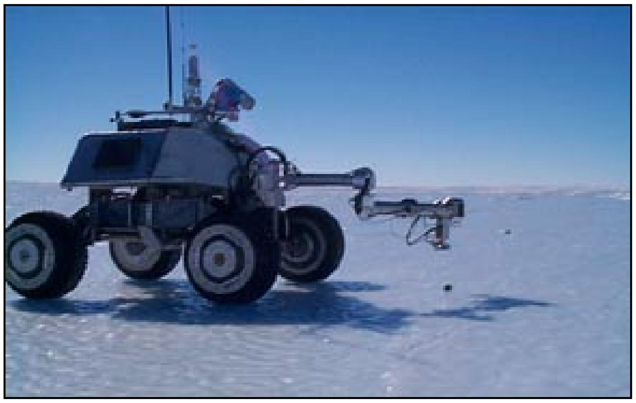


Figure 2.4: Nomad autonomously investigates a meteorite sample using its manipulator arm [SHA98]

2.3.2 Spirit and Opportunity (NASA)

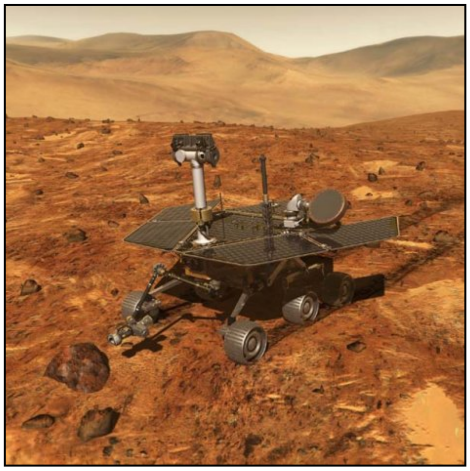


Figure 2.5: Mars Rover “Spirit” [WEB04]

Spirit and Opportunity are two similar Mars Exploration Rovers from NASA’s JPL (Jet Propulsion Laboratory), which have been built for the examination of rocks and soils that may reveal a history of past water activity. The Rover alone weights 174 kg and is 2.3 x 1.6 x 1.5 m (L x W x H) tall [WEB04]. It can go at a top speed of 0.05 m/s (0.18 km/h). The power source of the rover is a multi-panel solar array in combination with two rechargeable lithium-ion batteries. This enables the rover to generate 140 Watts of power for four hours per sol, when the panels are fully illuminated (Note: Sol is a Martian day. It is approx. 40 min longer than the 24-hours earth day) [WEB05]. However, the rover needs only about 100 W to drive at 0.18 km/h.

An important issue is the temperature control of the Warm Electronics Box (WEB). It contains all essentials such as the batteries, electronics and computer and should not exceed the temperature range of -40°C to +40°C. Several measures protect the body from the 113°C temperature swing during the Mars day: (1) gold-painted, insulated walls; (2) solid silica aerogel, which is one thousand times less dense than glass; (3) heaters with a thermostat and the (4) heat rejection system.

Located at the rover’s Pancam Mast Assembly is a Panoramic camera and a Miniature Thermal Emission Spectrometer. Furthermore there are, *inter alias*, the following instruments mounted at the rover arm: a Mössbauer Spectrometer, Alpha Particle X-Ray Spectrometer and a Rock Abrasion Tool to grind material from rocks.

The chassis of the Mars Exploration Rover consists of six wheels, each with its own individual motor; in addition to that the two front and the two rear wheels also have individual steering motors. This will allow the rover to make point turns.

The twin rovers were launched in June (Spirit) and July (Opportunity) 2003. They both landed on Mars in

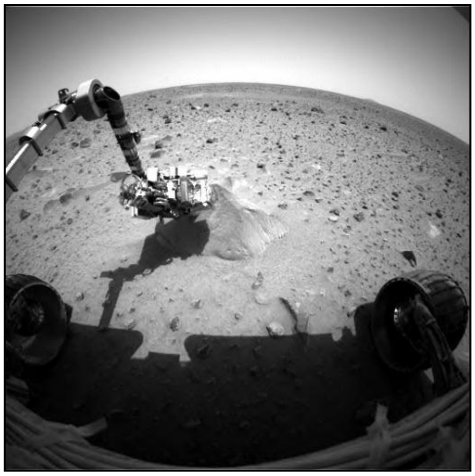


Figure 2.6: Spirit: Board Camera [WEB04]

January 2004 and the mission will last for 90-sol. They will address the question how past water activity on Mars has influenced the red planet's environment over time. While there is no liquid water on the surface of Mars today, the record of past water activity on Mars can be found in the rocks, minerals and geologic landforms, particularly in those that can only form in the presence of water.

2.3.3 Hyperion (CMU)

Hyperion is a rover designed and built for experiments in sun-synchronous exploration [SHA01]. The total mass of the vehicle including the 3.45 m² (nearly) vertical solar panel is 157 kg. The size of the robot is 2 x 2.4 x 3 m and it is able to travel at a speed of 0.3 m/s (1.08 km/h).

The chassis design was kept intentionally very simple: A brushless DC motor (1.5 Nm cont. torque, 150 W) combined with a harmonic drive for an 80:1 reduction ratio is driving a mountain bike wheel with a bicycle chain. But the key feature is a passively articulated steering joint for two free rotations. This gives the robot the capability of axle articulated steering (and skid steering), which gives the vehicle a moderate maneuverability maintaining a mechanical simplicity at the same time. "The innovation of the design implemented on Hyperion is to electronically control the velocity of the front wheels to maintain a desired angle of the front axle" [SHA01].

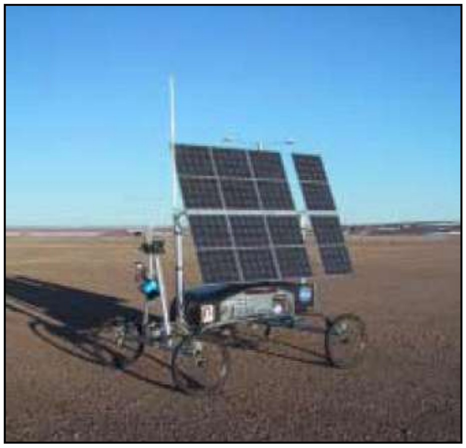


Figure 2.7: Hyperion with vertical solar panel [SHA01]

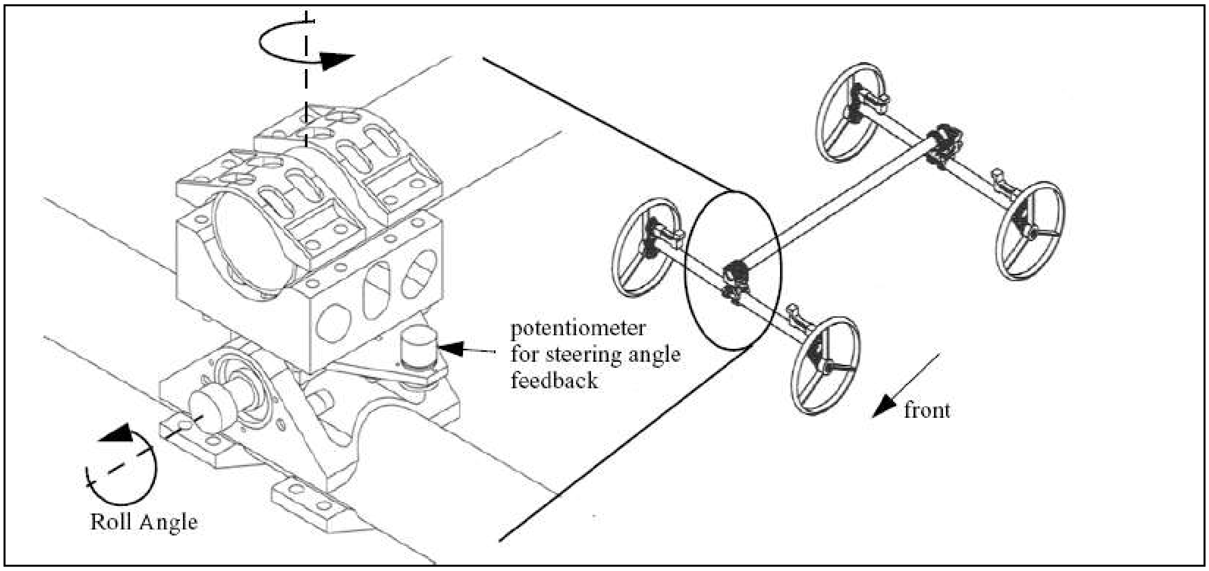


Figure 2.8:Detail of the front axle steer and roll pivot [SHA01]

By implementing several sensors into the steered front axle, a more robust autonomous navigation is achieved. The perception sensors include stereo cameras, a laser range finder and a panoramic camera.

In July 2001 the vehicle was tested in the Canadian Arctic where the sun angle is low. The mission investigated the capability of intelligently gathering solar power. With this ability the rover would be able to survive for months. The rover’s path was tracked by a Differential GPS (DGPS) and it turned out that the mean error magnitude of the actual path versus fit line is 6 cm with a standard deviation of 3.2 cm. One major lesson was learned that is important to the cool robot project: a potential failure was gear damage due to excessive drive motor torque. Another outcome of the project was the unstable driving in reverse.

However, the rover proved to be capable of handling the various disturbances of natural terrain.

2.3.4 Commercial Robots (iRobot)

iRobot is a major player in the market for autonomous vehicles and was founded by three MIT researchers of the Artificial Intelligence Lab. Two products will be introduced here: the ATRV (All-Terrain) and the Packbot.

The ATRV-Robot Line is based on a four-wheel drive platform, which turns through skid steering. The weight of the vehicle ranges from 39 kg (ATRV-Mini) to 118 kg (ATRV) depending on the model. The maximum payload is 100 kg for the ATRV (1.05 x 0.8 x 0.65 m) (L x W x H). Four Lead Acid Batteries provide 1,440 Whr which is enough to operate the robot for 4 to 6 hours depending on the terrain. Each wheel is driven by a high-torque 24VDC servo motor. With a speed of 7.2 km/h this robot is relatively fast.

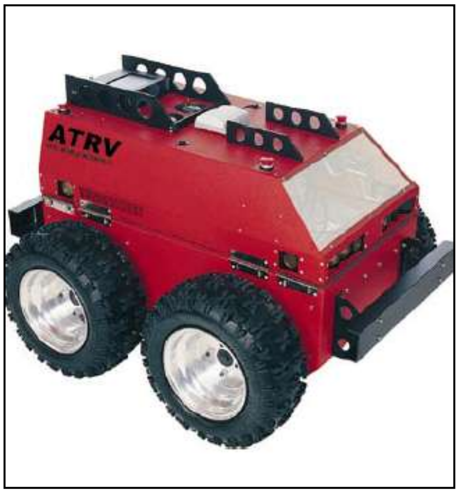


Figure 2.9: iRobot: ATRV [WEB06]

For obstacle avoidance the ATRV robot is equipped with 11 sonar sensors (6 front, 4 side, 2 rear). A selection of Navigation sensors is also available (SICK LMS Laser Scanner, Electronic Compass, (D)GPS etc.) [WEB06].

The company’s latest version (“CoWorker”) can be controlled remotely by any internet browser without installing additional hardware or software.

With its big knobby tires the ATRV is designed for mapping rough terrain or scouting out unknown territory.

iRobot’s military product “PackBot Scout” ($45,000) was designed to aid reconnaissance operations in Military Operations Urban Terrain (MOUT) warfare, terrorism and other 21st century battle environments. Weighing only 18 kg fully loaded, and 0.69 x 0.46 x 0.18 m (L x W x H) tall, PackBot can easily be carried by a Soldier in a backpack [WEB07]. The polymer tracks speed up the battery-driven robot to 14 km/h. Equipped with dual QuickFlip™ track articulations, the robot can perform a self-

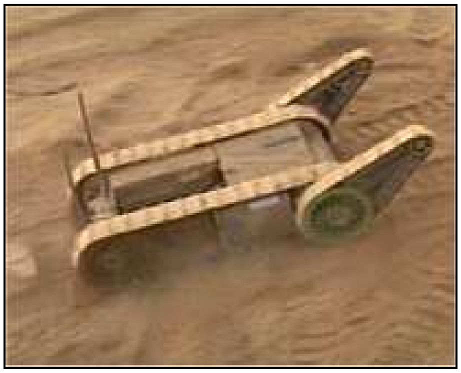


Figure 2.10:PackBot: Scout [WEB08]

righting maneuver within seconds [WEB08]. The aluminum chassis is designed to survive a 2-meter drop onto concrete (400 Gs) letting soldiers throw the robot into a building through a window for either basic tele-operation or full autonomy. Multiple sensors like GPS receiver, electronics compass, absolute orientation sensors and temperature sensors, help the robot to navigate through its environment.

PackBot Scout also offers five open payload bays for mission-specific upgrades like guns, grenades, fiber-optic spoolers, heads, sniper detection, chemical-agent testing and other payloads. This Unmanned Ground Vehicle (UGV) has been successfully in use in Afghanistan and Iraqi Freedom.

Besides the here introduced PackBot Scout, there are also the PackBot Explorer and the PackBot EOD (Explosive Ordinance Disposal) available.

Chapter 3 The Mechanical Design

The mechanical design is the main focus from Guido Gravenkötter in this thesis about the Robot for the Antarctic. It will examine the main robot layout and the circumstances which will influence the choice of the components. After the basic decisions have been made, the specific components for the robot will be chosen and tested for their ability to survive in the Antarctic climate. Besides the very conceptional scientific work the focus always remains also on a reasonable, economic selection in regards of designing the final robot within the limited budget.

This chapter provides the basic engineering work for building the final robot based on the detailed recommendations of components that have been theoretically verified and successfully tested.

3.1 Composing a Robot Model

This robot project starts from scratch. Thus, before starting to design the robot some general considerations have to be made in advance, particularly how to plan the basic layout of the robot. Based on these decisions, a robot model will be developed, which is the foundation for the mechanical design.

3.1.1 Wheel vs. Track

When it comes to the robot the first basic decision is without doubt the question, if the robot will be driving on *wheels or on tracks.* If the answer would be *"wheels"*, the next question has to be *"How many wheels?"* will be ideal for the given problem.

A wheel driven vehicle has a lot of advantages. Besides the fact that it is easier to design, the most important fact is a lower rolling resistance on firm snow compared to a track driven vehicle, leading to lower energy consumption. Hence, a wheel driven vehicle has in general a higher speed on firm terrain. Another advantage is easier maintenance and the change of a wheel, either in order to repair it or to replace it by a new type of wheel, e.g. according to the ground type.

The main advantage of a track driven vehicle is, due to low ground pressure and high traction forces, its great performance in slippery or easily deformed terrain. This of course is bought on the cost of a high energy requirement. The relatively low ground pressure enables the propulsion to drive even relatively high vehicle masses. (Note: that is the reason why main battle tanks (MBT) are usually track-driven. The heavy armor leads to high mass which needs to be distributed on a wide surface.)

Comparing the advantages and disadvantages of each option, a wheel-driven vehicle turns out to be more appropriate for the challenges of the Antarctic. The Antarctic terrain is widely flat and the snow consistency is different from what we know in our degrees of latitude. The snow is stiffer, leading to a lower sinkage, and there is no need for a very low ground pressure which would be the case with a track vehicle (e.g., compared to snowmobiles which are operating on powder snow). And, most of the terrain is quite smooth and difficult terrain such as crevasses and sastrugi can be avoided by good route planning.

A primary design specification is that the energy consumption has to be as low as possible, considering that the vehicle will be driven by solar-power and batteries. This consideration leads to a wheel-driven vehicle.

Pursuing the model of a wheel-driven robot, determination of the number of wheels (4-wheel configuration/6-wheel configuration) and making a reasonable trade-off between sinkage and traction are considered. Valuable information is provided by the Robotics Institute of Carnegie Mellon University. Their research with the NOMAD prototype for locomotion "LocSyn" has proved an "advantage to the four-wheel configuration primary because of the expected benefits in drawbar pull and slope climbing." [APO01] And what is even more important, is the outcome that the drive torque and power requirement is 25 % lower than the six wheel configuration.

There was only an insignificant difference in obstacle negotiation, where the six-wheel configuration proved to be slightly better. A second obvious advantage of the four-wheel configuration is from a technical standpoint the greater design simplicity and lower control complexity than the six-wheel configuration. The actual design of the tire - in terms of dimension and shape - will be discussed in detail at a later point in this chapter.

3.1.2 Steering Schemes

Various kinematics are available for steering a vehicle. Figure 3.1 gives an overview of some major steering schemes. For the Antarctic robot the best combination of mobility and simplicity is required. The requirements for mobility are defined by the mission task and the mission scenario. In case of the Antarctic, the mobility requirement is not very high because of the desert like terrain with a wide, open field for great distances, thus leading to large curve radii. It is unlikely to expect the necessity for the robot to perform a great number of short turns, or on-the-point turns.

The simpler the steering, the more desirable it is. Reducing the number of actuators will decrease the vehicle weight, which is very important in terms of power to drive the vehicle, and it will also increase the reliability of the robot, which will be exposed to the cold and harsh climate of the Antarctic. Thus, the simpler the overall robot design is, the more likely it will survive at the South Pole.

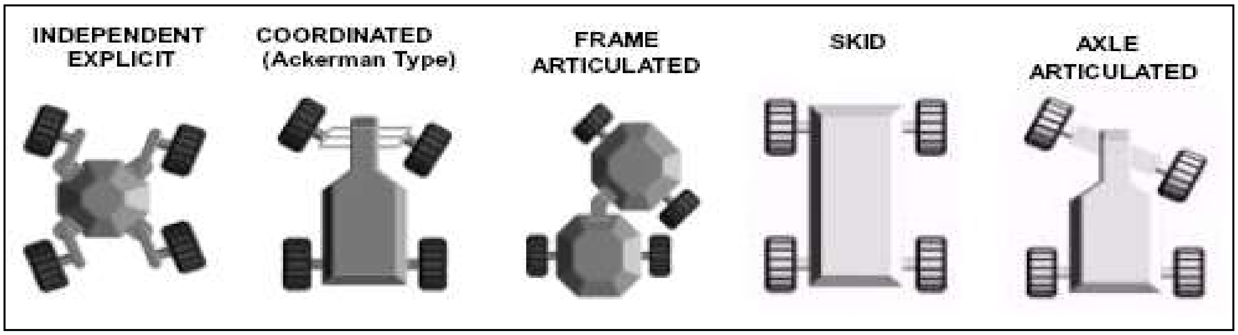


Figure 3.1: Steering Schemes for four-wheel vehicles [SHA01]

Independent steering explicitly articulates each of the wheels to the desired heading. This steering procedure is both, complex in actuation and complex in accuracy of coordination control. But the advantage is obvious: this steering method delivers the highest maneuverability in unprepared terrains. It also discloses the ability for crab steering, with all wheels turning by the same amount in the same direction, so the robot is even able to move sideways.

The most common type of steering on passenger cars is Ackerman steering which mechanically coordinates the angle of the front two wheels. In order to maintain all wheels in a pure rolling condition during a turn the wheels need to follow curved paths with different radii originating from a common center [LIT5].

Frame articulated steering is well known from large and heavy earth moving equipment. The heading of the vehicle changes by folding the hinged chassis units. The advantage is a greater maneuverability compared to coordinated steering (Ackerman) and a maximum thrust provided by the traction elements compared to skid steering. The mechanical complexity is very low.

“Skid steering can be compact, light, requires few parts, and exhibits agility from point turning to line driving using only the motions, components, and swept volume needed for straight driving.” By creating a differential thrust between the left and right sides of the vehicle, it changes its heading. The downside is the unpredictable power requirement caused by extensive skid steering which could be very high during the steering activity, dependent on the tire-soil interaction. Although there are no actuated joints for steering, the maneuverability is very high compared to all other steering mechanisms. (Note: This steering mechanism is used by track-driven tanks.)

The steering mechanism known from carriages is named axle articulated steering. In contrast to Ackerman steering, the single axle steering leads to the disadvantage that the wheels run in separate tracks when going around curves. Under difficult ground conditions this requires increased drive propulsion as each wheel is driving over fresh terrain. But a significant advantage is the low mechanical complexity and only low drive power that is needed for this steering maneuver.

Considering all advantages and disadvantages, skid steering promises to be the most appropriate steering method for the Antarctic region. Its very simple design and the low number of flexible parts make this method the best choice for the Antarctic robot, because each additional moveable part could lead to a possible failure of the system. The torque transmission is very easy, because the motion of the wheels is limited to rotation around only one axis. In the case of explicit steering, where wheels move around two axes, torque transmission is much more difficult.
Another important issue that has to be taken into account is the lateral forces, which are significantly higher in skid steering than in other steering methods. Therefore the wheels as well as the supporting structure have to be stronger in order to resist the higher forces.

3.1.3 General Conditions of the Antarctic Region

The Antarctic continent consists of various climatic regions with huge differences in wind, snow and temperature, depending on the location. For example, the winds near to the coast are stronger than at the South Pole Station. Those are important details which have to be considered by defining the actual circumstances of the mission scenario, because they will affect the layout of the UGV.

One of the major variables is the rolling resistance on the Antarctic snow. This number has to be identified first in order to calculate the necessary gearbox output torque to drive the vehicle. Based on the output torque, the proper motor-gearhead combination can be chosen. That makes the effect of the rolling resistance clear: a slightly difference in rolling resistance will have a major effect on all other components! The rolling resistance used in the following calculations is based on tests, which have been conducted by the scientists of CRREL (US Army Cold Regions Research and Engineering Laboratory). Nevertheless this value may vary from region to region and is not fixed. It rather has to be verified by further testing with the first prototype of the Cool Robot on the Antarctic plateau.

Wind and snow drifts were an undetermined concern in the beginning. The layout was tending to a streamline shape of the robot in order to minimize snow accumulation around the robot and also minimizing the wind resistance which has to be overcome. But by further research and experience of scientists who actually have been at the Antarctic, it turned out that the influence will not be as large as supposed. However the robot might need to cope with snow drifts when it is not many and it has to be able to free from the snow by itself to a certain extend. For example a occasional move of the robot for only a short distance will prevent it from getting stuck, while pausing at a particular spot.

Tires provide a good performance on snowy and icy surfaces (e.g., the NOMAD robot). Experiments have shown that a *tire* pressure of 1.5 to 3 psi will provide the best traction on snowy surfaces. The *ground* pressure, determined by the vehicle weight and the wheel patch, should not exceed 4 to 6 psi.

The permanently cold temperature, even during the summer period, influences the characteristics of high-strength steels. Most aluminum and stainless steel alloys do not become brittle with decreasing temperature but have lower strengths. Many plastics become brittle at low temperatures (≤-20°C) and are susceptible to cracking by thermal stresses [LEV02]. Grease in gearheads, bearings and other components do not work properly, if they are not rated for very low temperatures. They get stiff and lose their ability to lubricate the contact area between mechanical parts. Usually this demands extra preparation, e.g. exchange of the grease with a special low temperature lubricant.

The cold has a highly significant influence on the capacity of batteries. A capacity drop of almost 50 % should be anticipated if the batteries cool down to the surrounding temperature (-40°C).

Snow fall and ice fog can influence navigation sensors, e.g., obstacle avoidance sensors.

A magnetic compass will not work close to the magnetic South Pole, because the magnetic field lines are almost vertical. Therefore, the indicated direction is not very accurate. This issue will be addressed in detail in Chapter 4.2.1 “Magnetic Compass”.

It becomes obvious that all assumptions have to be made very carefully, because they can not be verified immediately by conducting tests under Antarctic conditions. Therefore a lot of information relies on the experience of people who worked at the South Pole Station or worked on similar projects [LEV04], [BLA03], [LES03].

3.1.4 Analyzing the torque demand

As mentioned above, the torque demand is the initial value to start the mechanical design. It will determine the choice of the motor-gearhead configuration. When one of the basic assumptions changes, the definition of the layout might be an iterative process.

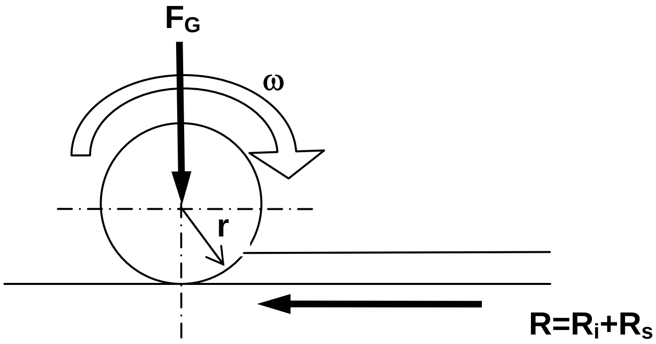


Figure 3.2: Rolling Resistance of a Rigid Wheel

A simplified scheme of a rolling rigid wheel is given in figure 3.2. Because it is a first approximation, it assumes that the tire shape remains round. The contact patch is only the *line* between the ground and the wheel and not a patch *area* (as it would be truly the case for a deformable cylindrical wheel). The total rolling resistance consists of two terms: the inner resistance, which describes the effort of a rigid wheel to roll over a hard surface, and the snow resistance, which considers the influence of a snow surface and the effort to compress the snow in order to be able to roll over it. (Note: Besides the Rolling Resistance R_i and the Soil Compaction resistance R_S, a detailed calculation the motion resistance would also include a term for the Bulldozing Resistance. This term can be expected to be small in Antarctic terrain, as sinkage is small.)

Based on testing conducted at CRREL, the following relationships evolved:

The internal resistance is

and the snow resistance is

$$\left.\begin{array}{l} \dfrac{R_i}{F_G} = .10 \\[2em] \dfrac{R_s}{F_G} = .10 \end{array}\right\} \quad \frac{R}{F_G} = .20 \qquad (3.1)$$

The estimated vehicle mass will be approximately 75 kg plus a payload mass of another max.15 kg. Therefore the total vehicle mass will be m =90 [kg].

The weight is

$$F_G = m\ g = 900\ [N] \qquad (3.2)$$

This leads to a total rolling resistance for the vehicle of

$$R\ = .2\ F_G = 180\ [N] \qquad (3.3)$$

And for one wheel

$$R_{\mathrm{W}} = 45\ [N]$$

Assuming that the UGV will drive 20'' tires (e.g. ATV) [1. Assumption], the diameter will be d=20''or r = 254 [mm].

The balance of moments will deliver a torque at one wheel of

$$\boxed{T_{\mathrm{W}}\ = R_{\mathrm{W}}\ r = 11.43\ [Nm] \qquad (3.4)}$$

This torque will be delivered from the motor through the gearhead to the wheel in order to overcome the motion resistance of a rolling wheel.

In addition to the steady torque is an extra term due to the mass moment of inertia, when the vehicle starts to accelerate. This term will be estimated here.

The vehicle should be able to travel 500 km in two Weeks. The average speed to fulfill this requirement is v=0.42 [m/s].

Assuming that the vehicle will accelerate to that speed in 2 seconds, the acceleration is:

$$v = a\ t \qquad (3.5) \qquad\qquad \imath := \frac{dv}{dt} = .21 \left[\frac{m}{s^2}\right]$$

The tangential acceleration

$$a_t\ = r\left[\frac{d\ \omega(t)}{dt}\right] = r\left[\frac{d^2\ \phi(t)}{dt}\right] \qquad (3.6)$$

determines the angular acceleration

$$\frac{d^2\,\phi(t)}{dt} = .83\left[\frac{\circ}{s^2}\right]$$

With the angular acceleration, the torque required to accelerate the wheel can be calculated as follows:

$$T_\Theta = \Theta\left[\frac{d^2\,\phi(t)}{dt}\right] \qquad (3.7)$$

and the mass moment of inertia for a thick pipe (R_1:inner diameter, R_2 outer diameter):

$$\Theta_{Wheel} = \frac{1}{2}\,m_{Wheel}\,(R_1^{\;2} + R_2^{\;2}) \qquad (3.8)$$

Assuming a wheel mass of m_W= 4.5 kg (1kg tube, 1kg rim, 2.5 kg tire) and a homogenous distribution of the mass from the rim (8’’) to the outer diameter (20’’) (therefore: R_1= 0.1 m, R_2=0.254 m) the mass moment of inertia is

$$T_\Theta = .14\ [Nm]$$

It turns out that the additional torque is of a very low magnitude and therefore will have not a significant effect on the total torque calculation. However, by considering the term with a safety factor of ν=2, it will balance some impreciseness in the assumptions (also the resitance may be substantially higher at rest due to static friction).

The total torque demand at one wheel is finally:

$$T_1 := T_W + \nu\ T_\Theta = 11.71\ [Nm] \qquad (3.9)$$

3.2 Selection of the Mechanical Components

In the previous chapter all design relevant information is identified. Based on this information the actual design work, the layout of the robot and the choice of components can begin.

For the propulsion of the cool robot one has to select the type of motor and the gearhead ratio. This goes along with the tire dimension (see above) and the resulting required torque. After choosing the tire dimension (based on availability), the rim size and design has to be defined as well as the tire width and tread pattern and other characteristics.

Besides the drive-train, the robots body e.g. a frame chassis has to be designed and the proper insulation needs to be determined to "keep the components warm".

The selection will be analyzed in detail and each component presented in a separate paragraph. One set of all components was been ordered and tested.

3.2.1 Propulsion: Motor and Gearhead

A motor is a machine that converts other forms of energy into mechanical energy and so imparts motion. The main purpose of a gear is the function of a torque converter; this goes hand in hand with revolution speed conversion. A third function of a gear is to control the direction of the rotation. Motor and gearhead influence each other, and therefore they have to be looked at combined. Because the gearhead ratio determines directly the necessary motor torque, which will be needed to drive the robot, the selection process will compare different motor-gearhead combinations.

The project analyzes the feasibility to drive a robot with non-fossil renewable resources. Therefore combustion engines will not be considered as possible drive-train solutions. Regarding electrical motors there is a great variety on the market.

The motor types are [MIL89]:

- d.c. commutator motors (e.g. Wound Field, Permanent-Magnet Brushless d.c. motors)
- Induction motors (e.g. Cage type three or single phase)
- Synchronous motors (brushless exciter, Permanent-Magnet)

- Reluctance motors (e.g. Synchronous/Switched/One-Phase reluctance)
- Stepper motors (e.g. single-stack, multiple-stack, PM, hybrid, Inductor)

The chosen motor type for the robot application is a “Brushless DC Servo Motor”, which is electronically commutated and exhibits the linear speed-torque characteristics of the conventional DC motor. It belongs to the class of square wave (trapezoidally excited) motors. Square wave motors are fed with three-phase waveforms shifted by 120° one from another, but these waveshapes are rectangular or trapezoidal. A servo motor drive is a drive with a speed or position feedback for precise control where the response time and the accuracy with which the motor follows the speed and position commands are extremely important [GIE02].

In conventional DC motors, the armature is the rotor, and the field magnets are placed in the stator. A brushless DC motor of this structure is difficult to make. The construction of modern brushless DC motors is very similar to the AC motor, known as the permanent magnet synchronous motor. The armature windings are part of the stator, and the rotor is composed of one or more magnets (2-pole/4-pole), so it is the other way round than the principle of a DC motor. Brushless DC motors are different from AC synchronous motors in that the former incorporates some means to detect the rotor position (or magnetic poles) to produce signals to control the electronic switches. The most common position / pole sensor is the Hall element, but some motor use optical sensors (Encoder) [KEN85].

A conventional DC motor has graphite brushes, which are in contact with copper commutators and the field magnets are on the stator. The rotors position is automatically detected by brushes. By having the field magnets (usually high energy neodymium magnets) on the *rotor*, with the brushless DC motor there is no need for brushes. The commutation is provided by electronic switching using transistors, and sensor elements, which detect the rotor’s position. The great advantage is that BLDC motors require no maintenance (no brushes = no wear) and are long-lasting in a compact design for high speed with a smooth and precise motion. The power density is extraordinary high. Figure 3.3 explains the interaction of Hall sensors and the commutation of a 4-pole motor (2x North, 2x South pole), which change the polarity (High/Off/Low) dependent on the position of the rotor provided by the Hall sensors.

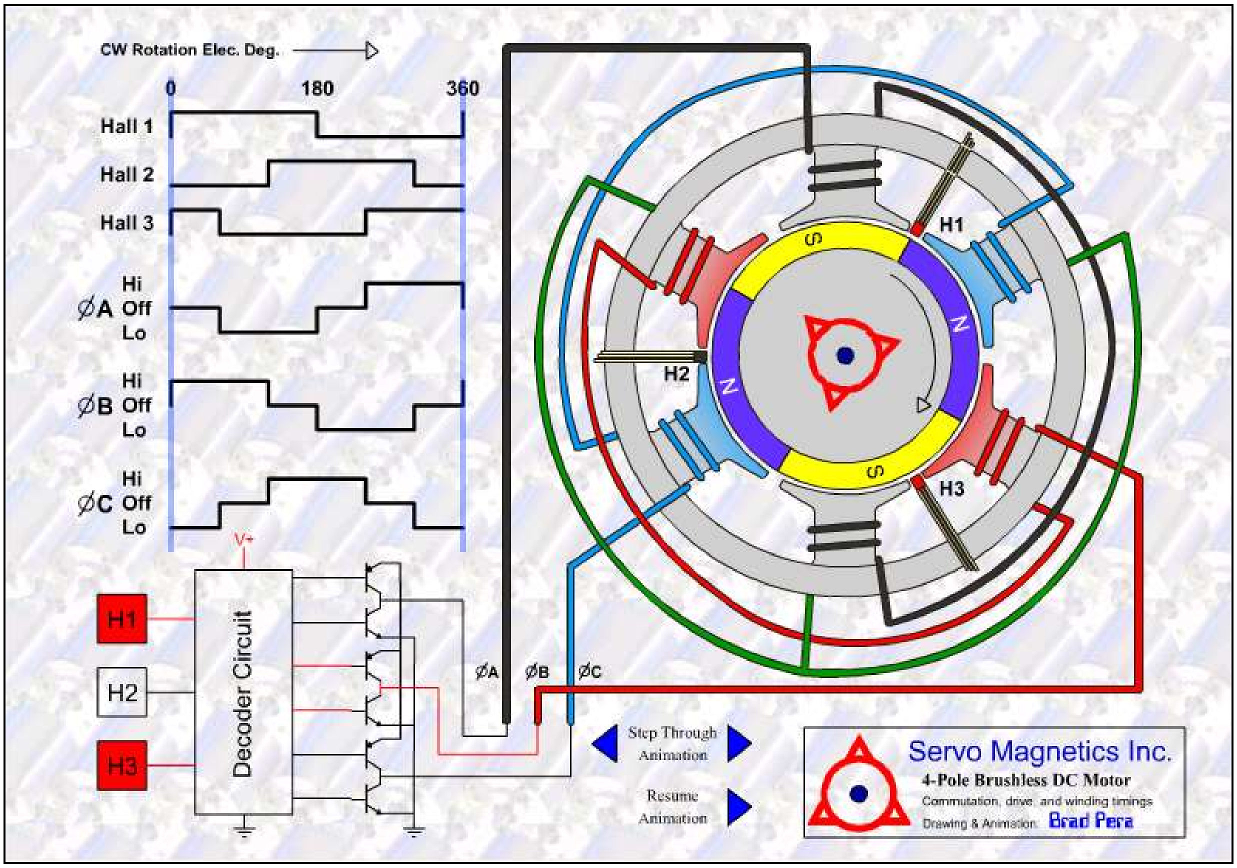


Figure 3.3: 4-Pole Brushless DC Motor

One result of the market research for BLDC motors is that the required torque for turning the wheel could not be provided with a reasonable direct drive in terms of size and power draw (and costs). Therefore it becomes obvious that a gearhead has to be mounted on the motor in order to increase the motor torque and reduce the motor speed (and still be able to work near the efficiency peak). The downside is that the gearhead has to be specially prepared for the cold temperature, because it contains a relatively huge amount of lubricant, which is usually not rated for very low temperatures (only to - 20° C). In addition, the bearings of both motor and gearhead need also be prepared with a special rated lubricant, otherwise the lubricant changes it viscosity and loses the ability to lubricate the moving parts.

The torque transmission could be done in many different ways:

- gear transmission by teeth
- crank drive
- cam mechanism
- belt drive

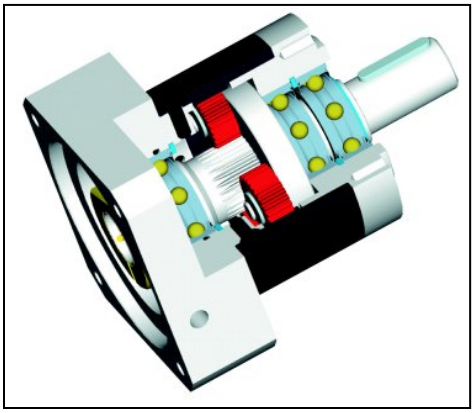


Figure 3.4: Planetary Gearhead [Neugart]

Looking for a compact, reliable design, the gearhead turns out to be the best option. All moving parts are enclosed in a sealed casing, protecting the parts from the unkind environment. Gearheads also offer the most flexible reduction ratio. For instance, the gearbox line of planetary gearheads used in this project starts with ratios from 3:1 ending with 512:1 (Figure 3.4). Planetary gearsets offer the following advantages: (1.) Remarkable for a planetary gear train is that the power is transmitted through two or more load paths rather than the single load path of a simple gear mesh. Because planetaries share load between several meshes, they are more compact than parallel shaft drives and offer significant envelope and weight savings. (2.) In addition to achieving minimum weight and envelope, the relatively smaller and stiffer components which result from the use of planetary gearing lead to reduced noise (55 dB) and vibration and increased efficiency (~90 % for 3 stages). (3.) The input and output shaft axes in planetary gearing are concentric. This can lead to space savings in some installations by allowing the driving and driven equipment to be in-line. (4.) The resultant radial forces on the input and output shafts of planetary gearboxes are zero since the arrangement cancels out all radial forces. In effect, the planetary gearbox transmits only torque. This simplifies bearing design and in some applications allows close coupling of the gearbox to the driven equipment [LYN83].

Besides the design advantages mentioned above, the gearhead used for this project is prepared for life-time lubrication, thus no maintenance is needed during use, which makes this selection both, reliable and user-friendly.

After defining the type of motor and gearhead, the market research leads to two major manufacturers of brushless DC motors (EAD motors and Maxon Motor) and two major gearhead manufacturers (Neugart and Maxon Motor). Different motor/gearhead combinations have been compared. The maximum continuous torque that the motor can provide and the torque constant play the most important role. The maximum continuous torque should be higher than the torque demand delivered to the gearhead. This value is usually limited by the heating up of the winding. A cold environment like the Antarctic will improve this value.

The torque constant, also referred to as "specific torque" represents the quotient from generated torque and applicable current. Therefore it could be considered as an efficiency of converting the electrical energy into mechanical energy. It turns out that 4-pole motors have a very good torque constant compared to 2-pole motors. The higher the torque constant, the lower is the power draw of the motor from the batteries or the solar panels. Due to the fact that the power draw is the most critical point in choosing the right combination, this value will mainly determine the selection of motor and gearhead. However, other parameters like stall torque, peak current, weight etc. have also to be taken into account.

Figure 3.5 compares different options of motor-gearhead combinations. The basis is the torque demand calculated above. An additional safety factor is not necessary, because the motor can deliver higher torque (up to the stall torque) for a short period of time if necessary, without any problems. To a certain extend it is a built-in safety factor.

To compare the options, they are all based on the same RPM at the wheel (n_{Wheel}=53 rev/min correspond to a max. vehicle velocity of 1.4 m/s), which corresponds for all options to a motor speed close to the efficiency peak (which is at this point not *exactly* known).

The calculated net torque is the torque that the motor has to provide constantly, in order to create the required motor torque at the wheel through the gearhead. It is calculated as follows:

$$T_{Motor}\,[mNm] = 1000\,\frac{T_1\,[Nm]}{i_{Gear}\,\eta_{Gear}} \qquad (3.10)$$

(T1= total motor torque, i_{Gear}=Gearhead ratio, η_{Gear}=Gearhead efficiency)

The necessary current is (K_T=Torque Constant):

$$I_{Motor}\,[A] = \frac{T_{Motor}\,[mNm]}{K_T\left[\frac{mNm}{A}\right]} \qquad (3.11)$$

This leads finally to the total power requirement by the motor of:

$$P_{Motor}\,[\mathrm{W}] = U\,I_{Motor}\,[V]\,[A]$$

Option	1	2	3	4	Unit
Safety-Factor	1	1	1	1	
M= [GearOut]	11.71	11.71	11.71	11.71	Nm
Gearhead	Neugart	GP 42C			
Part-#	PLE-40-100	203125			

Ratio (i_{Gear})	100	91	91	91	
RPM	5300	4823	4823	4823	
Max. continuous transmissible Torque	20	15	15	15	Nm
Efficiency (η_{Gear})	0.9	0.7	0.7	0.7	
Weight	550	460	460	460	g

Motor	EAD Motor	EC 45 150W	EC 45 150W	EC 45 200W	
Part-#	DA23 GBB-11	136200	136201	252464	
	4-pole			4-pole	
Voltage	48	36	48	48	
stall torque	1060	940	1040	3019	mNm
Max.Cont. TORQUE @5000rpm	380	175	176	266	mNm
Max.Cont.CURRENT @5000rpm	4.2	3.1	2.6	3	A
Efficiency	0.85	0.79	0.79	0.86	
NET TORQUE	130.11	183.83	183.83	183.83	mNm
Torque Constant	95.00	64.10	78.50	100.60	mNm/A
Current	1.37	2.87	2.34	1.83	A
Power	65.74	103.24	112.41	87.71	W
Weight	870	850	850	1000	g

Figure 3.5: Motor-Gearhead Options and the Resulting Power Requirement

In the table it is evident that option 1 requires 25 % less power than the best 4-pole motor-gearhead combination from Maxon (Option 4). In fact the torque constant is slightly better, but the proper gearhead adapted to the motor has such a low efficiency that it is in the end not better than the first option, in terms of power consumption. It was also considered to compare a motor at a lower voltage (36 V), but the low continuous torque and the low torque constant rule this solution out. Note: The power draw calculated at this point is based on a maximum robot speed of 1.4 m/s, but if the speed is lower than that speed (because the mission requires only an average speed of 0.42 m/s), the power draw will also be lower!

Therefore the selected components for the drive train will be the *EAD motors DA23-GBB-M300* brushless DC motor ($ 285.58 EA, including Encoder) and the *Neugart PLE40-100 –*OPX ($ 509 EA, including special lubrication and adapter to DA23-GBB BLDC motor) gearhead (Option 1), mounted to the motor with a special adapter provided by Neugart. The assembled components are shown in figure 3.6. A metal bracket fixes the two components on a wooden panel, so they can be tested (without a load attached). Both components are individually modified: the motor contains the encoder *and* the hall sensors (the reason for this is explained in the next paragraph; usually BLDC motors have only one sensor output) and the gearhead is prepared with a special lubricant, rated down to -50°C.

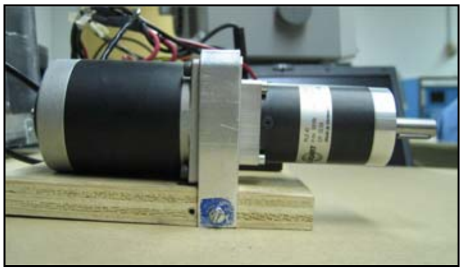


Figure 3.6: Gearhead Assembled to the BLDC-Motor

3.2.2 Motor Control: Brushless Servo Amplifier

Due to all the advantages mentioned above, brushless servo motors created a new phase in the development of control motors. As explained, the principle of brushless servo motors is that the function of the commutators of the dc brush motor is substituted by a pole sensor and a semiconductor power converter. Therefore a BLDC motor always requires a Controller, which gives the current commands depending on the angular position detector feedback.

The current waveform can be made close to a square wave by means of PWM (Pulse Width Modulation) because this inverter uses phase current feedback. With that, energy wasted in the motor, torque ripple, and transistor peak current can all be minimized. Figure 3.7 shows the block diagram of the driving system of a brushless servo motor. Three-phase sine (or trapezoidal) waves are generated by the signal produced by the angular sensor (Hall, Encoder). They are multiplied by the current command value (because the generated torque is proportional to the product of current and field flux), and compared with each of the feedback three-phase currents. If the motor current is larger than the command value, the inverter switches itself in the direction that limits the current. If the motor current is smaller than the command value, the inverter switches itself in the direction that increases the current. The speed signal is fed back to estimate the future angular signal, which reduces the time delay of the system [DOT90].

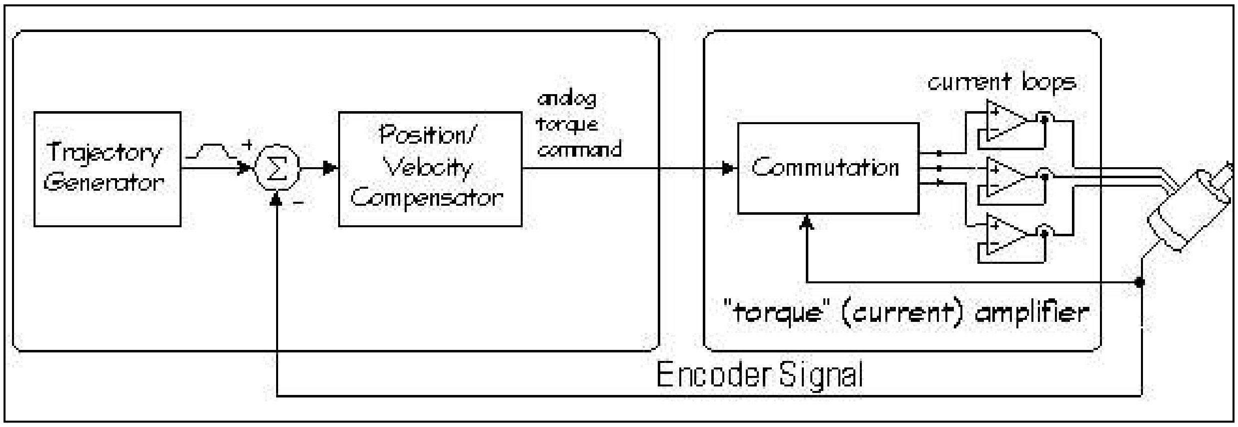


Figure 3.7: Block Diagram of the Driving System of a Servo Motor [worldservo.com]

Two control methods can be considered for the cool robot: the robot is controlled by a speed command or by a torque command (which is the motor current). Both methods have advantages and disadvantages. In a first scenario the speed command is without effect, if sufficient traction does not exist, i.e., if the wheel is spinning and therefore the robot is not moving. In a second scenario the controller gives a torque command to the motor, which aids its traction control, but that does not necessarily result in a constant speed, due to changes in the rolling resistance or driving up a slope. In general it is convenient to drive the motor by a speed control and follow the movement of the robot. If the vehicle loses traction, which is detected by a comparison with the GPS's speed monitor, the motor control switches from speed control to torque control. Also, in torque control, the motor can be driven at the maximum torque point and the robot can free itself from a stuck position (if traction permits).

While the motor is driven in velocity mode, an encoder signal is available to deliver the wheel speed signal for further processing by the "central" controller (which controls all four motors), e.g., to measure the shaft speed and compare the vehicle speed as determined by GPS to that determined by wheel speed.

An encoder is a feedback device which converts mechanical motion into electronic signals. Usually an encoder is a rotary device which outputs digital pulses which correspond to incremental angular motion. Example: A 1000 line encoder produces 1000 pulses every mechanical revolution. The encoder consists of a glass or metal wheel with alternating clear and opaque stripes which are detected by optical sensors to produce the digital outputs. A Hall sensor is a feedback device which is used in a brushless servo system to provide information for the amplifier to electronically commutate the motor. The device uses a magnetized wheel and hall-effect sensors to generate the commutation signals (Figure 3.8 and 3.9).

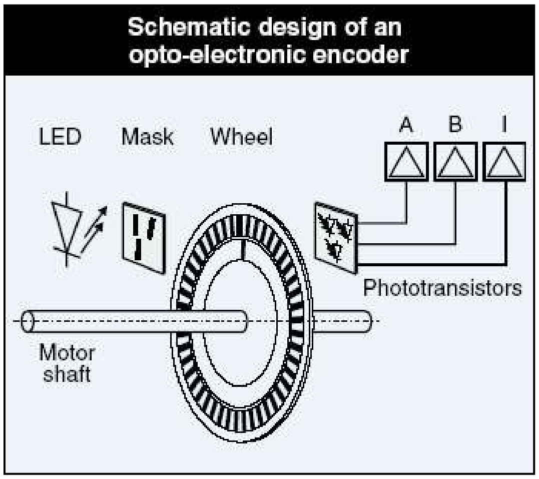


Figure 3.8: Design of an Encoder [Maxon Motor]

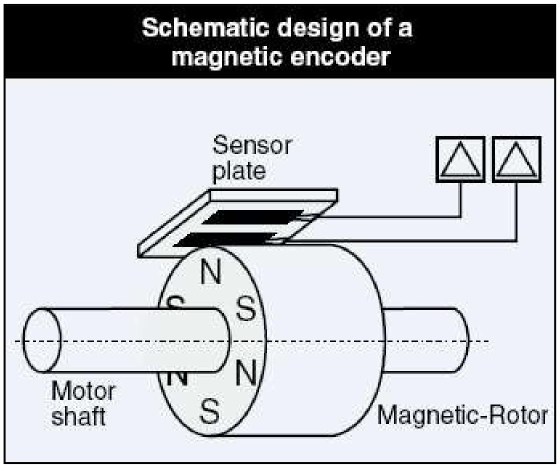


Figure 3.9: Design of an Encoder [Maxon Motor]

AMC (Advanced Motion Controls) offers servo amplifiers, which drive brushless, trapezoidally commutated DC motors. (Note: sinusoidal commutated servo drives are usually used for applications that require high precision velocity control and a perfectly smooth torque curve.)

They can operate in

- current,
- tachometer,
- open-loop and
- *Hall* or *Encoder* velocity

mode.

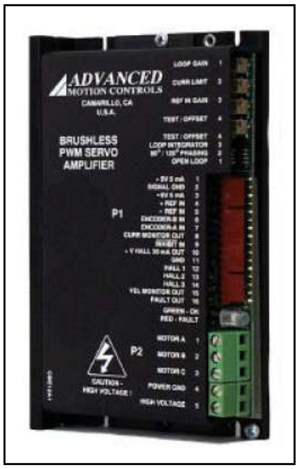


Figure 3.10: Motor Controller [AMC]

In current mode, the current command is used as a torque command, as both variables are proportional. A tachometer is an electromagnetic feedback transducer which produces an analog voltage signal proportional to rotational velocity. Open-loop is a system in which there is no feedback. Motor motion is expected to faithfully follow the input command, e.g. the higher the voltage, the higher the speed. Loop gain, offset, current limit and reference gain are adjusted by a potentiometer.

The component selected to control the EAD brushless DC motor is the *AMC BE15A8-H* servo amplifier (Figure 3.10). This controller offers the required control modes in 4-Q Operation ("four quadrant": accelerating and braking in clockwise and counterclockwise direction). The supply voltage ranges from 20-80 V, the peak current (max. 2 sec.) is +/-15A.
Including an educational discount of 50 %, this product is rated at $ 162.50 EA.

3.2.3 Robot Structure: Honeycomb

When it comes to the structure, there are two sorts one could think of: a frame structure, built of metal pipes, or a panel structure. The first one has the advantage that it could result in a very light structure, assuming that all planes are covered with modern carbon material sheets. The second option offers an unmatched strength and stiffness to weight ratio. Figure 3.11 shows the relative stiffness and weight of sandwich panels compared to solid panels. A honeycomb sandwich construction can compromise an unlimited variety of materials and panel configurations at temperatures from -55°C to 170°C. They offer excellent resistance to degradation due to moisture and humidity. The skin material and the core material can be different. The core cell shape is either hexagonal or rectangular. Figure 3.12 shows the construction of a sandwich panel compared to an I-beam. The facing skins of

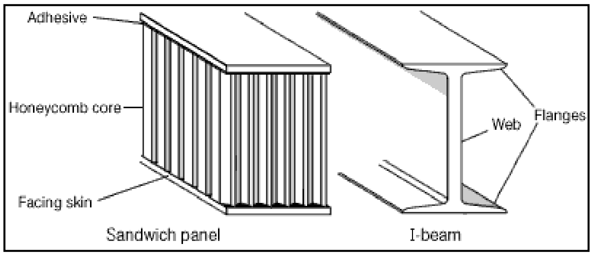
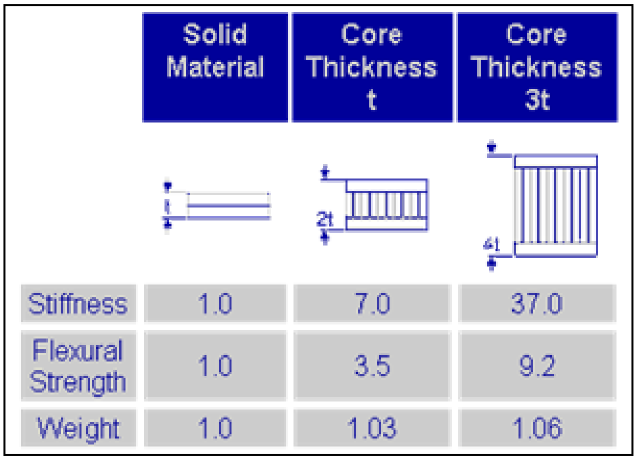


Figure 3.11: Construction of a Honeycomb Sandwich compared to an I-Beam [Hexcel]



	Solid Material	Core Thickness t	Core Thickness 3t
Stiffness	1.0	7.0	37.0
Flexural Strength	1.0	3.5	9.2
Weight	1.0	1.03	1.06

Figure 3.12: Comparison of Relative Stiffness and Weight with an increasing Thickness [Hexcel]

a sandwich can be compared to the flanges of an I-beam, as they carry the bending stresses to which the beam is subjected. The core resists the shear loads, increases the stiffness of the structure by holding the facing skins apart.

The core-to-skin adhesive rigidly joins the sandwich components and allows them to act as one unit with a high torsional and bending rigidity.

Considering all the advantages of a honeycomb structure (great strength/stiffness at a low weight, durability, vibration damping, insulation), these panels turn out to be most appropriate for the cool robot project.

The following part will deal with the choice of the honeycomb material (skin, core) and the stress analysis of the panel. A honeycomb structure is different from standard materials used for conventional stress analysis. Therefore, due to the complex structure of the core, a 100 % stress analysis can only be done with a FEM-analysis. However, the manufacturer *Hexcel Composites* offers guidance through this complex issue, enabling customers to do their own analysis.

(I.) Configuration Data

In a first step all the input data (material, constants) for the stress analysis is collected. Common aluminum panels offer all required properties for the robot project. The data is divided into the facing skins and the core material:

Facing Skins		Unit	**Core**		Unit
Material	**Aluminum 2024 T3**		Material	**Alu 3.4-1/4-5052**	
Thickness t_1 and t_2	0.518	mm	Thickness t_C	25.4	mm
Appendix II			Appendix I		
Yield Strength	**270**	MPa	E_C-Modulus	620	MPa
E-Modulus	70	GPa	Longitudinal Shear	1.6	MPa
Poissins Ratio	0.33		G_L Modulus	345	MPa
			Transverse Shear	**1.1**	MPa
			G_W Modulus	166	MPa
			Stabilized Compression	2.6	MPa

(all material constants are taken from Appendix I and II, Hexcel Composites)

(II.) Load Model

There is a table with load models for the stress analysis available from the manufacturer. Depending on the load model (Figure 3.13), the calculation for the maximum shear force and the maximum bending moment are different. Also the bending and shear deflection coefficient (k_b, k_s) are determined by the chosen load model. This application is similar to the beam type with both ends fixed and a central load (Appendix I, Exhibit 1).

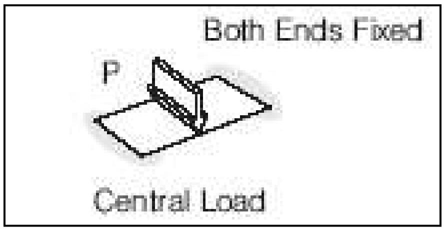


Figure 3.13: Load Model of a fixed panel

The load P in this case is estimated as follows:

The arrangement of the motors presents the load P. One motor distributes its force F of $F_M := \frac{1}{4} F_G = 225\ [N]$on a length of 0.05 m (fixture for the motor attachment). Because of the back to back installation of the opposite motors, the force doubles, leading to $P = 9000\ \frac{[N]}{[m]}$

Or for the total board (b=0.84 m) $P_T = 7560\ [N]$

The assumptions are very conservative, because in real the load is not distributed on the *entire* board, but only where the motors are fixed.

The Bending Stiffness *D* is where

$$D = \frac{1}{2} E_f t_f h^2 b \qquad (3.12) \qquad h = t_f + t_c$$

where E_f is the modulus of elasticity of the facing skin, t_f is the thickness of the facing skin and *b* is the beam width.

For the Shear Stiffness S

$$S = b\ h\ G_c \qquad (3.13)$$

where G_C is the Core Shear modulus in the direction of the applied load

And the results are presented in the following table:

width b	0.840	M		Deflection coefficient	
length l	0.560	M		bending k_b	0.0052
Load P	7560	N	center	shear k_s	0.25
height h	25.918	mm			
Panel Bending Stiffness D	10230.104	m^4Pa			
Panel Shear Stiffness S	3614005.92	m^2Pa			

(III.) Deflection

Two terms define the maximum deflection δ: bending and shear:

$$\delta = \frac{k_b\, P\, l^3}{D} + \frac{k_s\, P\, l}{S} \tag{3.14}$$

Where *P* is the applied load and *l* is the beam span.

BENDING	0.001	m
SHEAR	0.000292861	m
TOTAL	**0.969**	mm

Assuming the above described load model, the deflection of the honeycomb panel will be approximately one millimeter at the place where force P is applied.

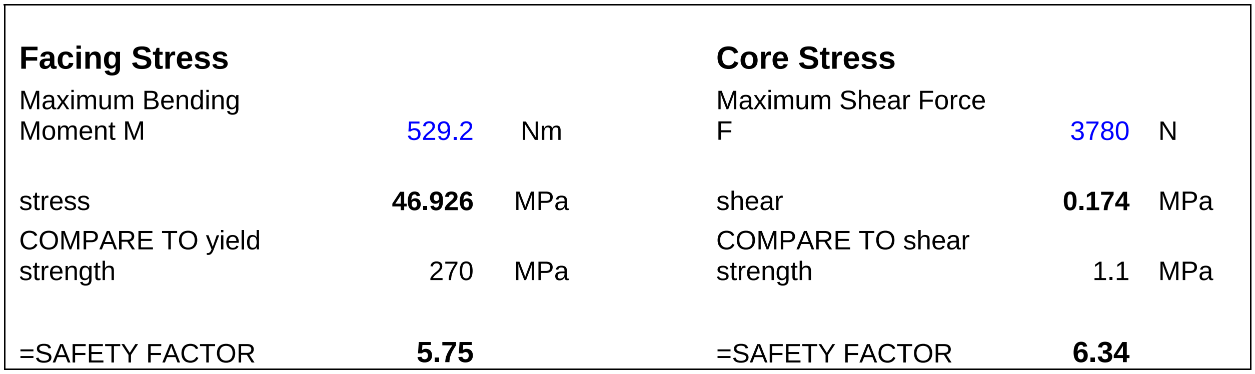
(IV.) Stress Analysis

Based on the maximum bending moment (M_B=P*l/8), the facing stress is compared to the yield strength of the specific material. The shear, which is influenced by the shear force (=P/2), is compared to the transverse shear. If the safety factor is considered to be too low, one has to modify the thickness dimension of the facing skins and/or the core, in order to get an adequate safety factor within an iterative process.

The stresses for the facing skins and the core are:

$$\sigma_{facing} = \frac{M_B}{h\,(t_1 + t_2)\,b} \qquad (3.15) \qquad\qquad \tau_{core} = \frac{F}{h\,b} \qquad (3.16)$$

For the cool robot the values are as follows:



Facing Stress			**Core Stress**		
Maximum Bending Moment M	529.2	Nm	Maximum Shear Force F	3780	N
stress	**46.926**	MPa	shear	**0.174**	MPa
COMPARE TO yield strength	270	MPa	COMPARE TO shear strength	1.1	MPa
=SAFETY FACTOR	**5.75**		=SAFETY FACTOR	**6.34**	

The results show that the selected one inch aluminum honeycomb panel resists the assumed load scheme. With the purpose of reducing the material costs, a commercial panel, which will fulfill all requirements, has been chosen. Both, the selected skin and core material, and the panel thickness are used in a large scale by customers and therefore very cost-effective. A standard 4'x 8' sqft panel (2.97 m²) *A510C* offered by TEKLAM costs ~$340.

3.2.4 Insulation: Vacuumpanel

One major effect of the cold temperature of the Antarctic is on the batteries. As the temperature drops, they lose their capacity dramatically (easily to 50 %). For that reason it is wise to keep the energy losses of the robot system inside the robot body. The losses are mainly in terms of energy losses by the motor and gearhead, which can be used to heat the interior of the chassis - if the body is properly insulated. Finally, *all* components will benefit from the warmer temperature. For that reason the idea of an in-wheel propulsion has been rejected.

One way to compare different insulation performance is by simply comparing the R-value of panels of the same thickness. The R-value is a measure of how well a material resists the flow of heat or cold through it. R-value is determined by a laboratory test (guarded hot box) in which an insulation material is sandwiched between a cool and a warm surface. The ability of the material to resist temperature changes results in an R-value for that material. The higher the R-value, the better is the resistance to the flow of heat.

Air leakage is often considered a secondary factor in keeping the heat in, but research shows that air leakage is a primary factor in heat loss. Therefore it is important to have a sealed envelope.

To understand how thermal insulation works, it helps to understand the three mechanisms of heat energy transfer: convection, conduction and radiation. *Convection* is the transfer of heat by physically moving the molecules from one place to another. It takes place when a fluid, such as gas, is heated and moved from one place to another. When warm air in a room rises and forces the cooler air down, convection is taking place. *Conduction* is the process by which heat transfer takes place in solid matter, such as the direct flow of heat through a material within a single or two separate bodies in direct contact. It is the molecule-to-molecule transfer of kinetic energy. For example, the outgoing shaft of the gearhead gets warm from the heat produced inside the gearbox. *Radiation* is the third way energy is transferred. This is evident in the way the sun warms the surface of the earth, which involves the transfer of heat through electromagnetic waves and absorption of that energy by a surface. The robot at the Antarctic absorbs radiant heat from the sun. Radiant heat transfer between objects operates independently of air currents and is controlled by the character of the surface and the temperature difference between warm objects emitting radiation and cooler objects absorbing radiation. In contrast to that, convection and conduction are functions of the roughness of surfaces, air movement and the temperature difference between the air and surface [BYN01].

Some interesting high-tech products are available on the market for insulation:

- Aerogels
- Vacuum Insulation Panels
- Gas-Filled Panels

- Phase-Change Materials (PCM) and
- Plastic Fiber Insulation.

The three most important insulation products will be presented: Aerogels, Vacuum Insulation Panels and Insulating Sheathings.

Aerogels are foam like and contain 99.9 per cent air. One inch of aerogel (silica gel) offers insulation equivalent to ten inches of fiberglass. Like a super sponge, an aerogel can be squeezed down to a tiny fraction of its original volume. When released, it springs back to its original shape. Aerogels are made of clusters of atoms linked to each other by chemical bonds that are unusually flexible. The linked clusters create an intricate web containing such high numbers of air-pockets that it becomes mostly air. So much so that one gram of aerogel may cover 1000 square meters of surface area.

Silica aerogel in a 90 % vacuum, which is simply and inexpensively produced, has a thermal resistance of R-20 per inch. By adding carbon, researches have improved the performance to R-32 per inch.

Vacuum insulation is formed by creating a metal foil or polymeric barrier envelope that surrounds an open cell core material, and then evacuating the enclosed gases (Figure 3.14). The envelope is then tightly sealed so that the vacuum can be maintained for years. The core material can be open-cell polystyrene or polyurethane foam, a silica powder or silica-based porous gels, or other insulating materials.

Based on the specific combination of barrier material, fillers, and vacuum level, R-values from 16 to 40 per inch can be obtained at room temperature. At lower temperatures, the R-value increases sometimes significantly.

S*heathings* are a very cost effective and light weight solution for insulation of buildings. It is a hardboard panel of a foam material and could be covered with a foil for a better radiant barrier.

The R-value for a one inch panel is 6.5. With the increase of the thickness it is almost linear: A 2’’ panel has a R-value of 12.8 (Atlas Roofing: Energy Shield Plus).

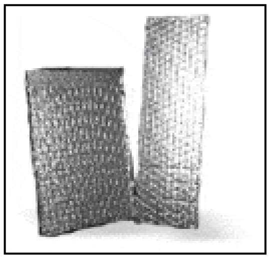


Figure 3.14: Vacuum Panel [EST]

For the cool robot project, a VIP from *Energy Storage Technologies Inc.* has been chosen. With a great performance of R-25 for a 1'' flat VacuPanel, at an internal pressure of less than 50 millibar (about 5 % of normal air pressure), these panels outperform other insulation products at this insulation thickness by far (Rigid Polyurethane R-7, Expanded Polystyrene R-4, Fiberglass R-3.3). The downside of these panels is that it is not possible to drill holes through the insulation because the vacuum cannot be damaged. And the shape of the insulation panel is also restricted from production on (which has only a minor effect on the simple shape of this cool robot).

As a second insulation panel the *Atlas Roofing "Energy Shield Plus"* has been chosen as a material to fill uncovered surfaces of the robot body (Figure 3.15). The material is simple to handle and process. Furthermore it is cheap and very light. It weighs only 0.86 kg/m² compared to the VIP, which is relatively heavy at 6.25 kg/m². Hence, weighing more than 7 times compared to the insulating sheathing at an insulation performance that is only 3.8x greater.

Figure 3.15: Hardboard Panel [Atlas Roofing]

3.2.5 Wheels

The decision to drive the vehicle with wheels instead of tracks is explained earlier in this chapter. But, nevertheless it is still a difficult decision because of a high leverage. Even if all components are thoroughly chosen, the entire project could fail with an inappropriate selection of wheels.

There are some parameters which still needed to be defined:

- Rigid or inflatable tire
- Tube or tubeless tire
- Tire and rim diameter
- Tire width
- Inflation pressure
- Material
- Tread pattern

Some commercial tires in the desirable size are available from vehicles like Go-Carts, lawn mowers, golf carts, ATV's etc., and there are some manufacturers of extraordinary tires like the balloon tires from Roolez Wheels (Figure 3.16).

On the custom-made side, different projects generated interesting tires: the rigid wheel of the moon rover, the mars rover wheel with a built-in suspension or the balloon tire from a Russian Antarctic vehicle, and not to forget the grinded wheel of the Antarctic bicycle.

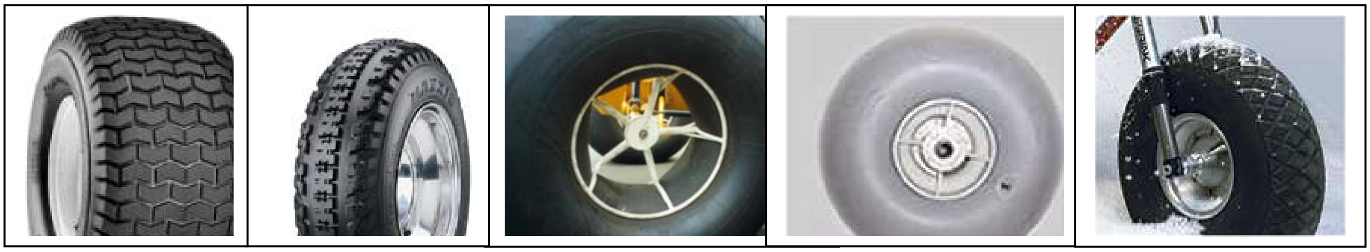


Figure 3.16: Different Tires (Left to Right): Golf Cart, ATV, Russian Snow Mobile, Roleez Balloon Tire, Antarctic Icy Bike

A balloon tire with a small, light weight rim seems to be promising. They are able to run on low inflation pressure, thus absorbing shock, and they promise good traction because of a very low ground pressure. But the attempt to create a custom made balloon tire with a Go-Cart rim and a tube has been discarded.

Commercial ATV tires are available in a great variety of sizes and tread patterns. But with a minimum of 5 kg for the tire only and another 5 kg for the rim, four of these tires will easily eat more than 50 % of the vehicle weight "budget".

A custom made rim is viable. It will optimize weight and design for the needs of the cool robot, because ATV rims at a size relevant for the cool robot are designed for at least three times the vehicle weight. A self-made aluminum rim will pay attention to the special requirements for the cool robot project. In combination with grinding down the "unnecessary" rubber from a commercial tire, the overall wheel weight could be reduced dramatically to the half.

The tires of the moon rover were also special: they were not made of rubber with the inner tube, but there was an elastic ring inside with a tight net of metal wires and the tread was made of titanium. A rigid tire can also be designed for the cool robot. One set of rigid tires could be taken to the Antarctic and tested with the cool robot for traction analysis. A metal tire could be designed for optimum stiffness and minimum rolling resistance and weight.

Although a tubeless tire suggests that it does not need an inner tube, it seems to be safer to still have the tube as a back-up. Unfortunately this measure will increase the weight by a maximum of 1 kg per wheel.

For the rim design, aluminum is a good choice. It has a low density and is good to process. It will not be brittle at the cold temperature of the Antarctic. ATV winter tires shouldn't have problems with the cold temperature either. The rubber should be rated down to -40°C. Unfortunately the tire manufacturers are usually not able to give reliable information on this. More reliable information could be provided through cold room testing at CRREL.

A wheel size of 20'' at a width of 7'' on an 8'' rim (20''/7''-8'') seems to be a reasonable sizing for the cool robot tire. Nonetheless this is an iterative process and it could turn out in the further process that a different rim size may turn out more appropriate. Changes to a smaller tire are more likely, than to a larger one. In terms of rolling resistance a slimmer tire is desirable if a low ground pressure can be maintained. On the other hand, a smaller tire will reduce weight and torque requirement, but it will also reduce ground clearance – which will affect maneuverability – and vehicle speed.

The wheel design is a project for itself. A second student has taken over this part for further research and design. Figure 3.17 shows a design of a rim, where all unnecessary material has been removed. Its hole pattern fits on a four-hole wheel-hub. Further computer aided stress analysis (ProMechanica) has to verify the design. The research may result in a grinding down the pattern of a *Carlisle Trail Wolf 19''x7''-8''* ATV tire on a self-made aluminum rim. Carlisle already agreed to donate the tires and support this research project.

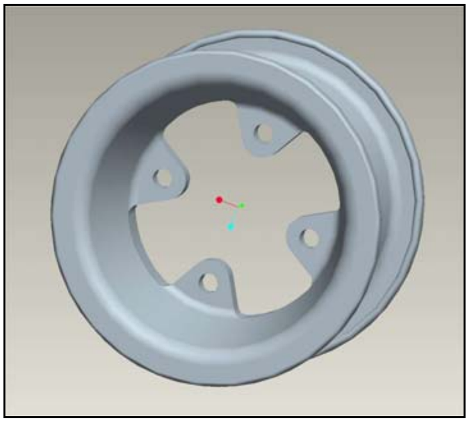


Figure 3.17: Hardboard Panel [A. Price]

3.2.6 Lubrication

In order to reduce friction and wear between the surfaces of moving parts, a lubricant is required. Especially in components with a high portion of grease (gearhead, bearings, O-rings etc), it is important that the grease is rated for low temperatures. Sometimes the manufacturer is able to provide a special grease (e.g. Neugart for the gearhead), but occasionally the manufacturer does not offer this service, so the grease needs to be changed afterwards.

This work might not have priority for the first prototype of the cool robot, but anyways it must be considered within the project. A vendor on this sector is SPI supplies. They offer lubricants for space applications and other extreme environments. *Braycote® Micronic® 1613* is a smooth, buttery, off-white grease based on a perfluoroether base oil. It is thermally stable, nonflammable, and chemically inert to commonly used fuels, solvents and oxidizers. The “Micronic” designation indicates that the grease has been extruded through a screen pack filter to remove any Teflon (PTFE) agglomerates larger than one µm. The temperature range of this braycote is from -73° C to 204° C. This product costs from \$100 for a 10 g syringe to \$280 for a 56 g syringe (Figure 3.18).

Figure 3.18: Special Cold Temperature Lubrication (Braycote) [SPI]

3.3 The Robot Layout

Having chosen all relevant components for the mechanical design in the chapter before, the thesis will now present the layout of the robot. All data (deviations, weight, power requirement) is collected to get an overview of the entire cool robot project. Therefore the set-up will also include components for navigation and power supply.

3.3.1 Meeting the weight limit

In order to be able to meet the upper weight limit of 75 kg for the robot vehicle only (+ 15 kg for the payload), the table below lists all components. With a total vehicle mass (without payload) of 71 kg, the table proves that the weight limit is still valid. However, the weight of the solar panels and the wheels is still subject to change.

Component	Body	Insulation	Motor	Encoder	Gearbox+Adapter	Shaft	Controller
Weight	10.955	4.038	3.480	0.400	2.200	2.481	1.120

Component	GPS	Antenna1	Modem	Antenna2	Magnetometer
Weight	0.025	0.090	0.610	0.084	0.100

Component	Batteries	Solar Panel	Recharger	RIM	Tube	TIRE
Weight	5.600	15.000	2.500	4.000	4.000	10.000

TOTAL	**65.28**	[kg]

(A full list, including the deviations and power draw is given in Exhibit 2.)

3.3.2 In-wheel vs. Out-wheel solution

The robot will be transported to McMurdo Station on the coast of Antarctica in a C-130 cargo plane and then from the McMurdo Station to the Amundsen-Scott South Pole Station with a smaller *"Twin Otter"* plane. The smaller Twin Otter plane dictates the overall dimensions of the robot, as this plane is the most likely way to transport the robot for the last part of the travel to the Antarctic. The robot has to fit through the cargo door of the Twin Otter, therefore the outer restriction is 1542 x 1270 x 1219.2 (W [mm] x H [mm] x D [mm]; 60’’x 50’’x 48’’) and limits at same time the overall robot dimensions.

Two principle solutions for robot configuration are considered: First, the solar panels are entirely around the robot. Therefore, also the wheels are covered by the solar panels (Figure 3.19). This solution maximizes the possible size of the solar panels, only limited by the cargo bay of the plane, and generates the highest possible power. The disadvantage of this solution is that an extra frame has to be attached to the vehicle, which holds the solar panels. This solution adds weight to the robot.

The second option provides a better vehicle driving stability, because in this solution the panels are directly mounted on the robot body. Therefore, no extra frame is needed for mounting the solar panels, which is an advantage in terms of weight. On the other hand the solar panels are partly covered by the wheels. So the generated power is less, because of the smaller panel size and the wheel shadow on the panels.

The trade-off between both solutions has to be made by the students working on the power system of the robot. An abstract of their work is presented in chapter 5.

For further work, this chapter follows the solution with the solar panels around the entire robot, because the ability of the robot to generate power is considered as the bottleneck. However, a closer look at the vehicle weight for the second solution could also result in favor of the out-wheel solution, which keeps the design very simple and provides easier access to the instruments in the inner robot body. Moreover, the solar cells will take a about 50 % of the budget, so the budget restriction might force to be in favor of this solution (although it is scientifically not the desired optimum).

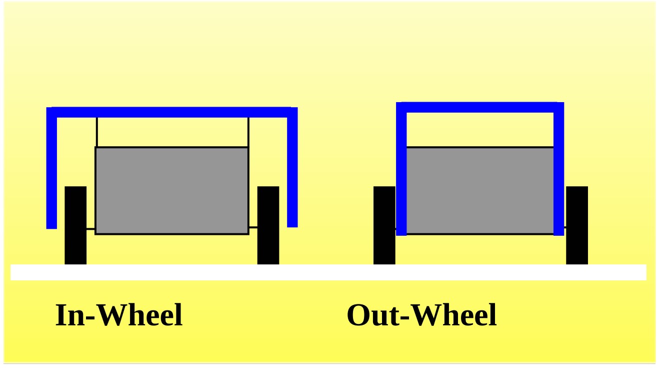


Figure 3.19: In-Wheel vs. Out-Wheel solution

3.3.3 First Layout

Figure 3.20 shows the TOP VIEW of the robot layout based on the in-wheel configuration. All important dimensions are measured in the view and based on actual sizes (all values in mm).

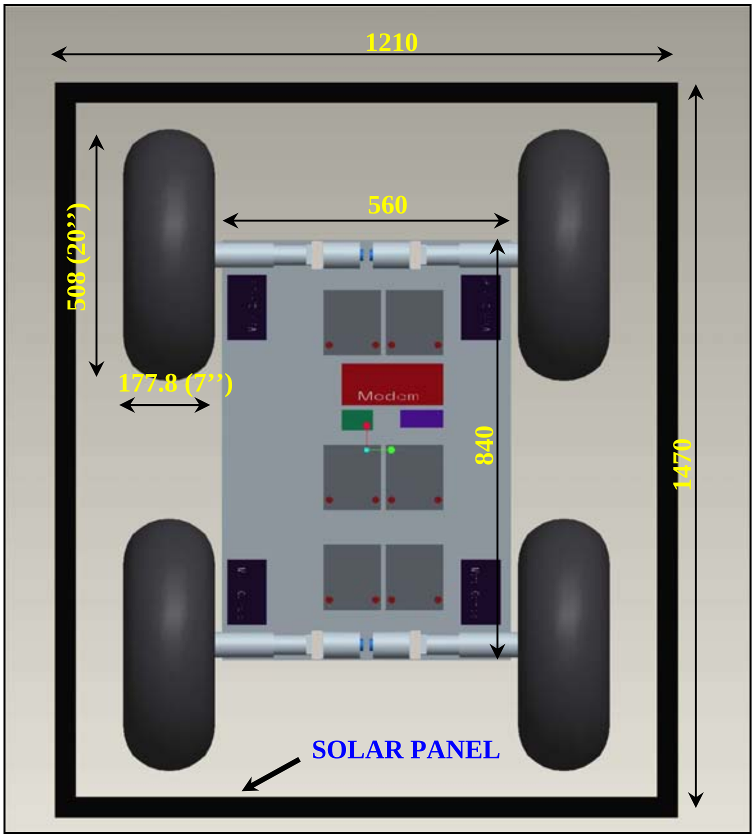


Figure 3.20: Robot Layout (Top View) [Pro Engineer]

The height of the vehicle is restricted to 914.4 mm. On the honeycomb base platform are the instruments mounted (magneto-meter, GPS, Antenna), the propulsion (a single motor incl. controller and gearhead configuration for each wheel) and the power system (batteries, recharger, maximum power point tracker). The surrounding line represents the solar panels. Their size is restricted by the cargo door of the plane minus 50 mm (2’’) for handling the robot through the door and maybe an attachment of handle bars for carrying and lifting the robot. With a frame architecture (not shown), the solar panels are mounted to the inner box. This robot model uses the suggested 20’’/7’’-8’’ tires. The entire box could be made of honeycomb panels, but if weight could be saved, it is recommended to use aluminum sheets only (instead of a full honeycomb structure), bonded with the insulation and an acrylic glass cover to protect the insulation. The assembling of the walls provides enough stiffness for the box.

3.3.4 Stress Calculation of significant parts

The following paragraph will deal with the stress calculation of some significant parts of the propulsion system. This will help to build the robot in the next phase, based on scientific calculations. After introducing an abstract model of the technical system, and a brief calculation of the bearing forces, the diameter of the shaft will be determined. In a second step, the strength of the shaft-to-collar connection is analyzed. Finally, a bearing for this application is chosen in detail, and the calculation proves the long life of the bearing under the assumed load.

(i) Mechanical System

The following picture (Figure 3.21) shows the simplification of the effective forces on one of the transmission shafts. It displays the shaft, going from the gearhead output to the wheel hub.

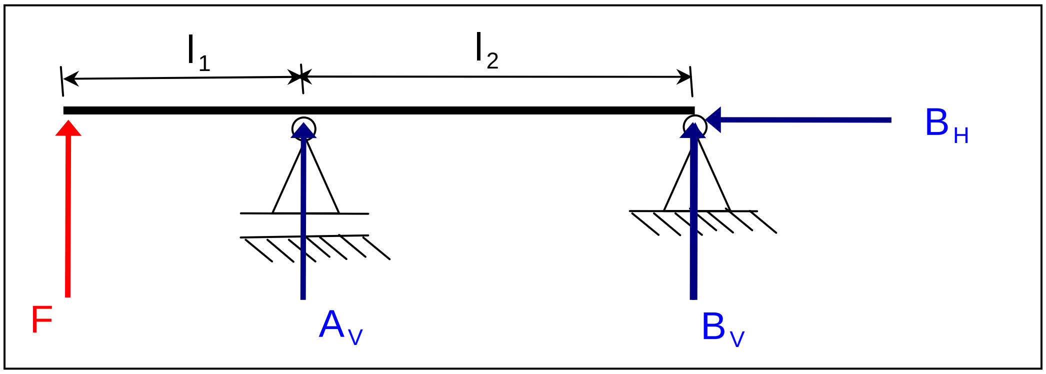


Figure 3.21: Mechanical System from the Gearhead to the Wheel Hub

The total vehicle mass will be 90 kg. Assuming that the robot will drive on 3 wheels instead of 4 for a short time because of the terrain, the Force **F=300 N**. This leads to a more conservative dimensioning of the shaft.

The balance of forces for each direction (vertical, horizontal) and the balance of momentum in Point A, leads to the following:

I. $\Sigma F_{iH} = 0 \rightarrow \mathbf{B_H = 0}$ (3.17)

II. $\Sigma F_{iV} = 0 \rightarrow F + A_V + B_V = 0 \Leftrightarrow \mathbf{A_V = -F - B_V}$ (3.18)

III. $\Sigma M_{i(A)} = 0 \rightarrow -F*l_1 + B_V*l_2 = 0 \Leftrightarrow \mathbf{B_V = F*l_1/l_2 < 200\ N}$ (3.19)

A maximum radial load of 200 N for the gearhead in radial direction is given by the manufacturer. This limits the force B_V and it becomes obvious that this limit could only be reached with a minimum 2:1 ratio of l_2:l_1(a higher ratio would be desirable). *Important:* The layout of the base panel indicates that the shaft length might not be longer than 100 mm on board. The resulting 50 mm length to the wheel hub might turn out not to be sufficient; therefore it strongly recommended placing the bearing *outside* the robot platform, supported by a tube. Without any turning action the horizontal force in bearing B is zero. But when the robot start to turn with skid steering, this force will increase. The maximum load in the axial direction is also 200 N for the gearhead. It is very difficult to predict the forces during skid steering.

Assuming l_2=0.15 m and maintaining the ratio of 2:1, it leads to l_1=0.30 m (which can only be reached with a tube-extension for the bearing) and, the bearing forces result in **A_V = -450 N** and **B_V = 150 N (<200 N)**.

(ii) Forces and Moments in the Shaft

Transverse Force distribution on the beam (shaft):

The following diagram (Figure 3.22) shows the shear force depending on the location in the beam.

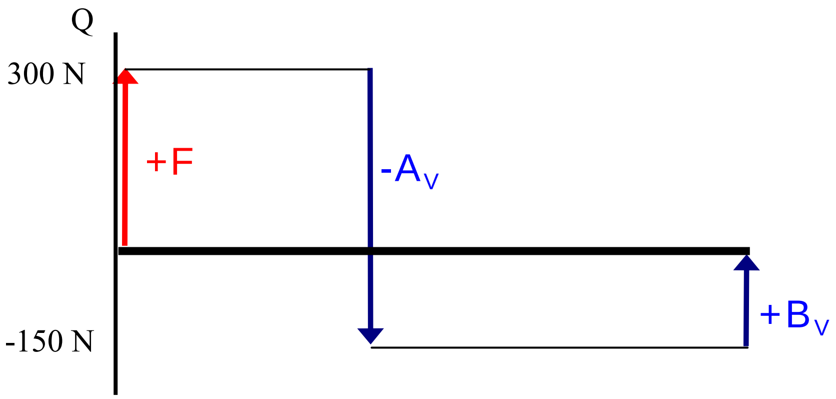


Figure 3.22: Shear Force Distribution in the Shaft

Bending Moment distribution in the beam (shaft):

The following diagram (Figure 3.23) shows the bending moment depending on the location in the beam. At the ends, the moment is zero. The maximum bending moment is located at bearing A with 45 Nm.

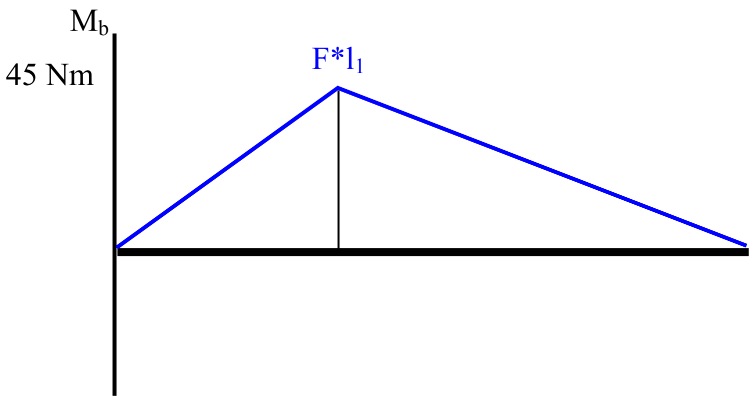


Figure 3.23: Bending Moment Distribution in the Shaft

(iii)Dimensioning of the Shaft

Stress because of bending:

To find a reasonable value for the shaft diameter, an iterative process is necessary. The start value for the diameter was 0.01 m (and M_b=45 Nm), because that is the dimension of the outgoing shaft from the gearhead. A solid shaft diameter of **d = 0.015** m is the result of the iterative process with an Excel-sheet including a sufficient safety factor (see below).

with

$$\sigma_{b,\,max} = \frac{M_b\, y_{max}}{I_{circle}} = \frac{M_b}{W_b} \qquad (3.20) \qquad\qquad W_{b,\,circle} = \frac{1}{32}\pi\, d^3 \qquad (3.21)$$

$$\Rightarrow \qquad \sigma_{b,\,max} = 135.8 \left[\frac{N}{mm^2}\right]$$

Shearing Stress because of bending:

The basic equation for stress solves the shearing stress in the beam (with Q=300 N)

$$\tau_s := \frac{Q}{A} = 1.70 \left[\frac{N}{mm^2}\right] \qquad (3.22)$$

Stress because of torsion:

The torque from the motor (T=11.71 Nm) twists the shaft.

with

$$\tau_{t,\,max} = \frac{1}{2}\frac{T\,d}{I_t} = \frac{T}{W_t} \qquad (3.23) \qquad W_{t,\,circle} = \frac{1}{16}\pi\, d^3 \qquad (3.24)$$

$$\Rightarrow \quad \tau_{t,\,max} = 17.67 \left[\frac{N}{mm^2}\right]$$

Equivalent stress:

The combination of all calculated terms above to the *equivalent stress* (GEH=Gestaltänderungsenergiehypothese or von Mises) allows to merge different stress terms into one value. The formula for ductile materials and fatigue load is as follows:

$$\boxed{\sigma_V = \sqrt{\sigma_{B,x}^2 + \sigma_{B,y}^2 - \sigma_{B,x}\,\sigma_{B,y} + 3\,\{\tau_T^2 + \tau_{B,x}^2 + \tau_{B,y}^2\}} \qquad (3.25)}$$

$$\Rightarrow \quad \sigma_V = 139.25 \left[\frac{N}{mm^2}\right]$$

Calculation of safety factors:

Material constants for steel E335 (St60-2):	tensile strength S_{ut}=R_m=590...770 N/mm²
(all values from DIN-EN 10 025)	elastic limit S_y=$R_{p0.2}$=335 N/mm²
	strength properties under alternating flexural stress σ_{bW}=290 N/mm²

1. Safety against breakage (static): $S_B = \frac{S_{ut}}{\sigma_V} = 4.24$ (should be 2 to 3)

2. Safety against plastic deformation (static): $S_D .= \frac{S_y}{\sigma_V} = 2.41$ (should be 1.2 to 1.5)

3. Safety against breakage due to fatigue (dynamic): $S_F = \frac{\sigma_{bW}}{\sigma_V} = 2.44$ (should be 1.5 to 3)

All safety factors are within the limits. A shaft (steel E 335) with a diameter of 0.015 m will withstand all stresses. Further calculation has proved that even a torque of 52 Nm, which is more than four times greater than the usual torque, will prevent permanent damage to the shaft.

(iv)Shaft-to-Collar Connection

Through a shaft-to-collar connection at the output-shaft of the gearhead, the torque is transmitted into the shaft which holds the wheel hub. The following section will analyze if the standard DIN 6885 keyway could transfer the torque from the gearhead to the driving-shaft.

According to DIN 6885, the following values are valid (Figure 3.24):

d=10 mm
h=3 mm
b=3 mm
t_1=11.4 mm
t_2=1.4 mm
t_3=8.2 mm
t_4=1.8 mm

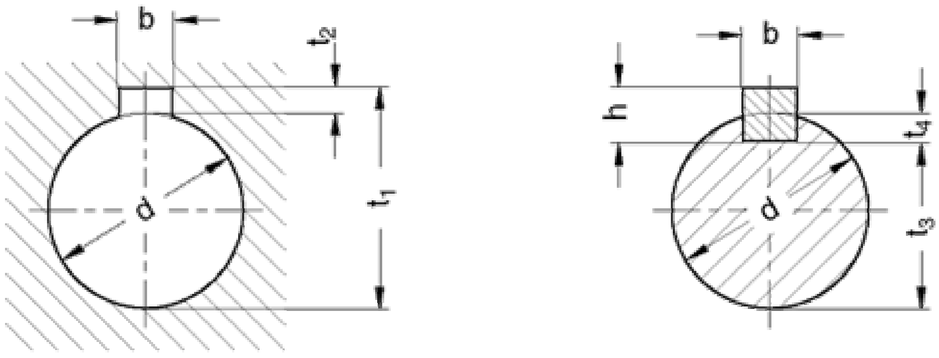


Figure 3.24: Dimensions of Shaft / Hub [DIN 6885]

Continuous transmissible torque (with a safety factor of 3):

$$T = \frac{1}{2}\, l\, d_m\, h_t\, i\, \rho\, p_{zul} \qquad (3.26)$$

$$\Rightarrow \qquad T = 18.09\,[Nm] \quad (SF=3)$$

with: $d_m=d=0.01$ m

$l=18$mm (manufacturer)

$h_t=t_4=1.8$mm

$i=1$ (number of keys)

$\rho=1$ (reduction factor)

$p_{zul}=S_y/SF=111.67$ N/mm²

With only a safety factor of only 1.5, the transmissible torque doubles to 36.18 Nm, which is almost the maximum continuous torque of that motor (0.38 Nm) can provide. As a result of this analyzes, it turns out that there is no problem to expect with the available keyway.

(v) Bearing Selection

As it is shown in 3.3.4 (i) the mechanical system needs two bearings two support the turning shaft. A ball bearing is a machine element which transfers a load between counterrotating surfaces at very low friction. A common configuration consists of one thrust bearing (here: bearing B) and one floating bearing (A). The thrust bearing is inside the gearhead and is able to take forces from two directions (both up to 200 N). The second bearing has one component for the radial force.

Grooved ball bearings and cylinder roller bearings are typically used as a floating bearing, because they allow movement in one direction and restrict the second direction (see Figure 3.25). In this case bearing A has to stand a radial force of 450 N as calculated before (shear force in the shaft in point A).

With FAG Kugelfischer, Germany, being the third biggest ball bearing manufacturer in Europe, this company provides the technical data which is essential for the bearing selection. A calculation on durability could only be done with data for a specific bearing. Apart from FAG

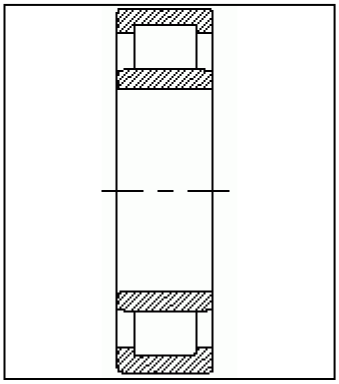


Figure 3.25: Cylinder Roller Bearing [FAG]

and scientific literature, many other manufacturers also provide the formulas to calculate the lifetime of a bearing which will lead to similar results for the same bearing size.

The chosen cylinder roller bearing is: FAG NU 204E Typ2 (with cage)

inner diameter:	d=20 mm
outer diameter:	D=47 mm
width:	B=14 mm
dynamic load rating:	C_{dyn}= 27.50 kN

(a) durability (dynamic)

If the revolution speed is higher than n>10 min^{-1} a dynamic load analysis has to be conducted. The bearing can stand the nominal life-time L_{10} (in 10^6 RPM):

$$\boxed{L_{10} = \left(\frac{C}{P}\right)^p \qquad (3.27)}$$

with the dynamic equivalent load P [kN] = $X_0F_r + Y_0F_a$ (X_0=1, Y_0=0 for cylinder roller bearings) and p=10/3 (for roller bearings):

⇒ L≈900,000 Mio. RPM

Assuming an average speed of 1.4 m/s (n=53 min^{-1}), the life-time of the chosen bearing will last for 32,308 years (if there is no influence on lubrication, temperature and cleanliness).

(b) static performance figure

This figure indicates if the roller bearing resists the plastic deformation at the touching spot of the rolling parts. The higher this number is, the better for the running smoothness.

$$f_s = \frac{C_0}{P_0} \qquad (3.28)$$

with C_o= 24.5 [kN] (static load rating)

⇒ f_s= 54.4

This cylinder roller bearing will easily stand the static load, because a value between 1..1.5 is already sufficient.

Exhibit 3 shows the technical drawing of the selected cylinder roller bearing.

3.4 Experimental Evaluation of Drive System

One set of drive components was ordered, and a test box made out of the selected honeycomb material was built. The main purpose was to create a box in which one set of the propulsion components could be tested under "real" conditions at the US Army CRREL facility. This test would provide new information on component operation under the extreme cold of the Antarctic, only protected by the robot body and its insulation.

This "mini-robot-body" type box also visualization of a robot body made out of the selected material and suggests how to mount things on the honeycomb panel and work with it.

The box consists of three layers:

- honeycomb aluminum panel
- vacuum panel insulation
- Lexan glass protection

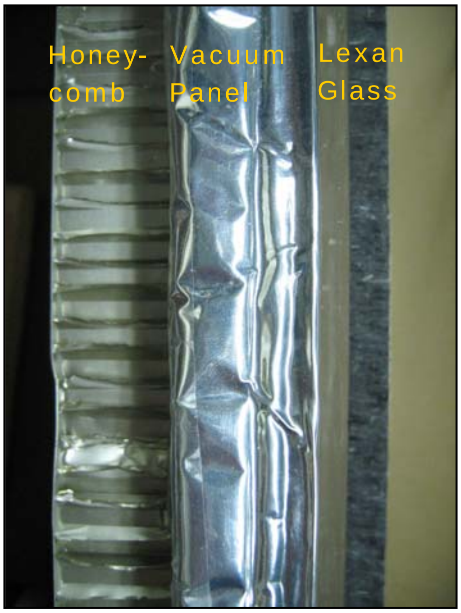


Figure 3.26: Insulation Layers of the Test Box

The honeycomb is the inner layer which provides the structural strength of the box. All walls are glued together with a silicone paste. They hold the walls together and eliminate air flow. The insulation panels are glued on the *outside* of the walls. Therefore, components can be mounted on the aluminum panel and bores can be drilled without destroying the vacuum inside the panel. For insulation purposes it doesn't matter, if the insulation is glued outside or inside the box. The vacuum panels are protected from the surrounding terrain with a fiber glass type cover. The

General Electric LEXAN (1/4'') sheets are more flexible than plexi glass and therefore do not crack so easily. These sheets prevent the panels from being destroyed by sharp edges and other things that would harm the insulation. The stiffness of the box is enforced with aluminum angles holding the fiber glass together. Figure 3.26 points up the three layers of insulation.

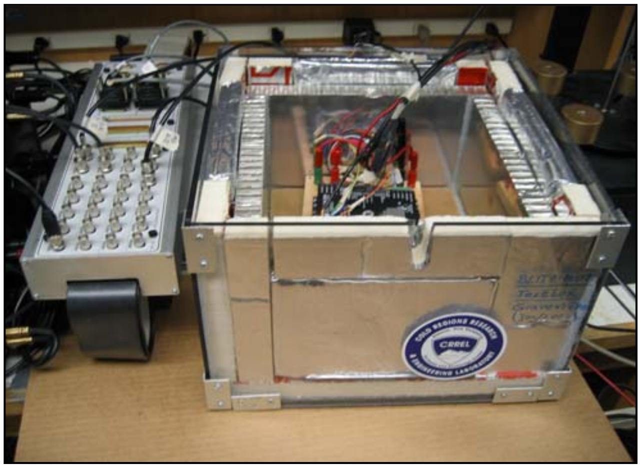


Figure 3.27: dSpace and Test Box with Compon. inside

The *outer* dimensions of the test box are: **365 x 365 x 265** (L [mm] x W [mm] x H [mm]). The *inner* dimensions are: **250 x 250 x 145**. This is large enough to enclose one motor-gearhead-controller configuration inside the box, and still fit the box into the test chamber. Figure 3.27 shows the component-setup inside the box with the wires going to the DC power supply (not visible on the right) and to the dSpace experimental environment (left from the box). dSPACE provides complete solutions for electronic control unit (ECU) software development.

On the input side, the dSpace device receives the current and encoder signal and sends it to a connected laptop. On the output side, it sends the programmed reference voltage to the motor controller. The laptop connected to the dSpace board is running the block diagram-based modeling software MATLAB/Simulink®, which is designed to use and validate control functions. Moreover MATLAB/Simulink® is used for recording all data for additional interpretation.

3.5 Testing the drivetrain

Some information on the motor can only be gathered by doing actual testing on the components. This section deals with two major test series: One in the laboratory at Thayer School to quantify the motor efficiency, and a second one at US Army CRREL (Cold Regions Research and Engineering Laboratory) in order to test the components in a realistic Antarctic-like environment.

3.5.1 Laboratory: Motor Efficiency

One important value for the overall system design is the motor efficiency. This value has a direct influence on the layout of the power system and the mission's demand on energy. Sometimes the vendor is able to state the motor efficiency at a particular RPM, sometimes the vendor can provide the information only partly by giving performance curves and parameters.

EAD motors do not provide any motor efficiency curves. But with a no load test and some further calculation, the efficiency curve depending on the torque can be determined. It can be calculated from the *speed-torque* curve and the *current-torque* curve.

(I.) The **speed-torque diagram** is given by EAD motors in the specification sheet of the motor as a graph. The formula determined from this graph, is:

$$\boxed{n\,[RPM] = no\_load\_speed\ [RPM] - 3777\ T\,[Nm] \qquad (3.29)}$$

$$\Rightarrow\ n\,[RPM] = 5250 - 3777\ T\,[Nm]$$

where *T* is the motor torque in Nm and *n* is the speed in RPM.

(II.) The **current-torque diagram** has to be determined with the help of the no-load speed, which has to be measured, and the torque constant of the motor, which is provided by the manufacturer in the winding data. The no-load speed will be the intercept point with the y-axis (T=0; current axis in the current-torque diagram), and the torque constant represents the slope of the current-torque curve (how current increases with an increase of torque).

Sending a full reference voltage to the motor controller will make the motor turn at the maximum speed (with the gearbox mounted, but without any load attached). The current monitor of the controller displays the required current as an output signal to the graphic interface of the laptop. The current value has been recorded 3600 times and the average current was calculated to 0.272 A at 48 V. Hence it is leading to a no-load speed of 5,249 RPM (full motor speed). The advertised no load speed of the motor is 5000 RPM.

Considering an efficiency loss of 90 % because of the gearhead, the motor-only current must be less: 0.9 x 0.272 A = **0.245 A**.

The torque constant for the selected motor is 0.095 Nm/A (y-axis: torque, x-axis: current). Therefore the gradient in the current-torque diagram is **1/0.095 A/Nm** (y-axis: current, x-axis: torque). Thus, leading to the following current-torque curve:

$$I\,[A] = no\_load\_current \; + \frac{T\,[Nm]}{K_T} \qquad (3.30)$$

$$\Rightarrow \quad I\,[A] = .245\;[A] + 10.53\left[\frac{A}{Nm}\right] T\,[Nm]$$

Figure 3.28 shows both curves in one diagram: the speed-torque and the current-torque curve

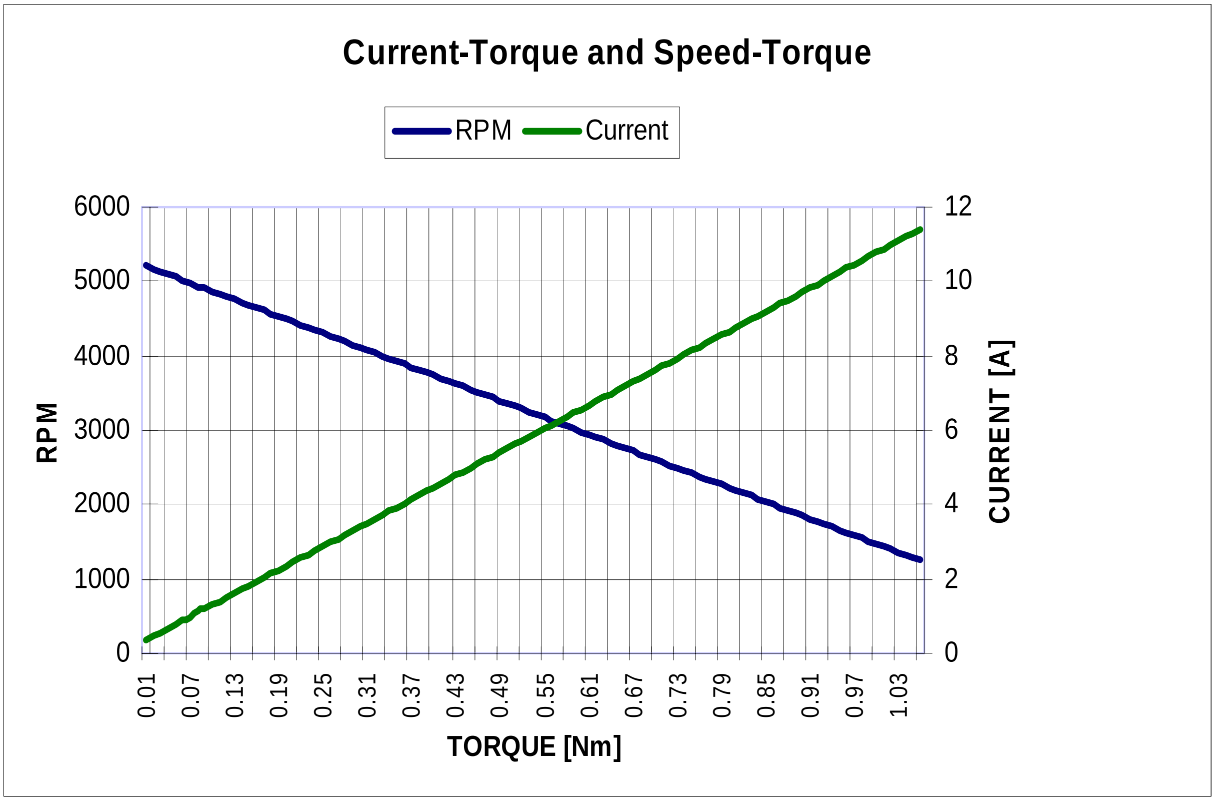


Figure 3.28: Current-Torque and Speed-Torque Diagram for Brushless DC-Motor

(III.) With the help of the calculations done before, the **motor efficiency** can now be quantified. The efficiency is calculated by comparison of the electrical power that goes *into* the motor, provided by the DC power supply, and the power, which is *delivered by* the motor in terms of mechanical power at the motor shaft.

$$\eta_{Motor} = \frac{P_{M\_out}}{P_{M\_in}} \qquad (3.31) \qquad \begin{cases} P_{M\_out} := T\,[Nm]\,\omega\left[\frac{rad}{\sec}\right] = \frac{1}{30}\,T\,[Nm]\,\pi\,n\left[\frac{1}{min}\right] & (3.32) \\ P_{M\_in} = I\,U\,[V]\,[A] & (3.33) \end{cases}$$

Where I is the motor current and U is the voltage.

Now, the efficiency of the motor can be calculated for *every torque value*. The calculation steps are as follows:

(i) select a torque value for which you want to know the efficiency

(ii) determine the motor speed (RPM) for a particular torque, by using the speed-torque curve

(iii) calculate the necessary motor current for this torque with the help of the current-torque curve

(iv) current and voltage (U_{Motor}=48V) determine P_{M_in}

(v) torque and RPM solve P_{M_out}

(vi) (iii) and (iv) result in the motor efficiency for this torque

By generating many torque values as an input and letting the efficiency value calculated by a spread sheet, the resulting efficiency curve is shown in the following figure (3.29):

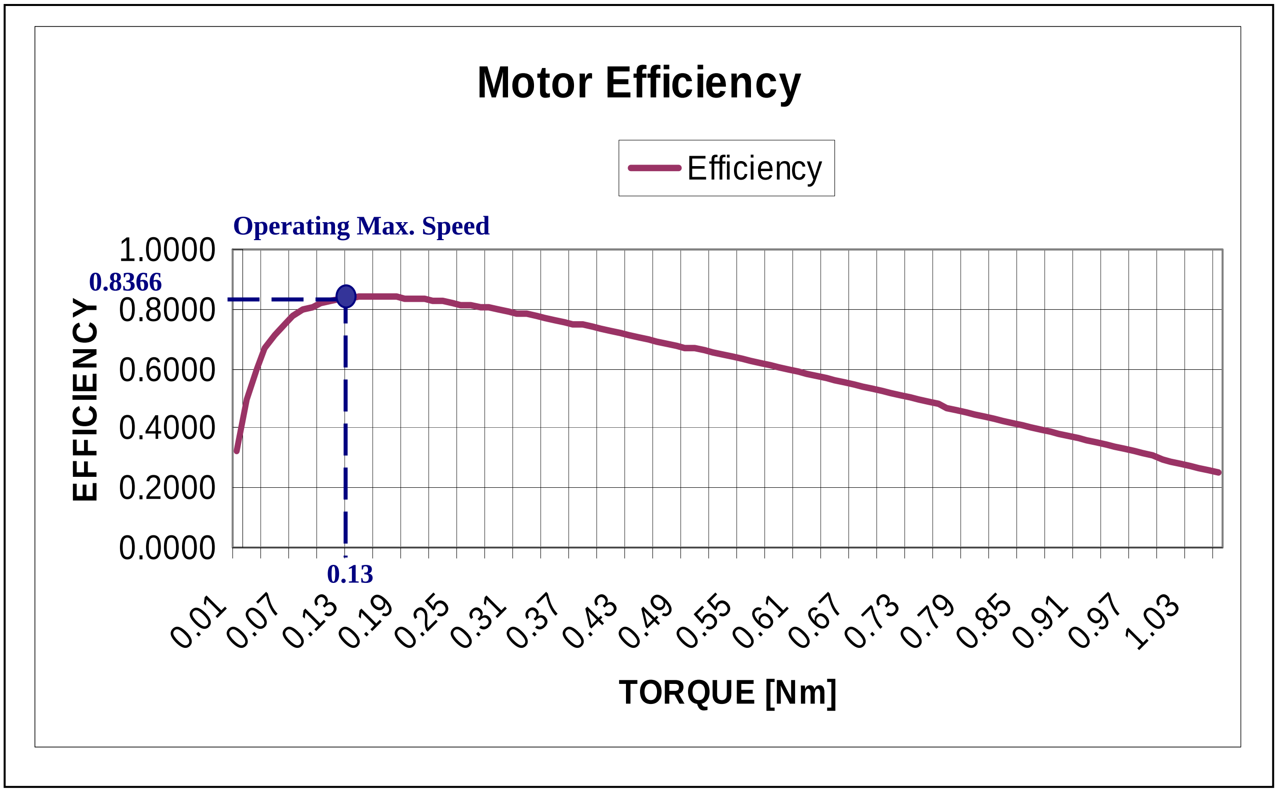


Figure 3.29: Motor Efficiency Curve

In the graph it is visible that the top motor efficiency is ~84 % at a torque of 0.16 Nm, which is a very high efficiency for a motor. In order to be able to deliver the required 11.71 Nm at the output shaft, the motor has to provide 0.13 Nm (because of the 90 % efficiency in the gearhead). At this torque, the motor efficiency is 83.66 %, which is still close to the maximum efficiency. A motor speed of 4,759 RPM will be reached with 48 V and a current of 1.61 A. This speed is equal to an outgoing shaft speed of 47.6 RPM, which leads to a vehicle velocity of 1.27 m/s (4.6 km/h). The recommended motor speed into the gearhead is 4500 RPM.

(Exhibit 4 provides a full list of torque-efficiency values. It also includes the calculated current and speed.)

3.5.2 CRREL: Cold Chamber and Test Setup

As it is important to know if the setup would work under Antarctic conditions, facing the problem that some components are not rated for -40° C (motor to -20° C, controller officially only to 0° C), the only solution is to conduct a test run in the cold chamber available at CRREL research facilities. Those results will give an idea of how reliable the configuration will work in the Antarctic climate.

The cold chamber *Blue M Electric Company (Model #1004-8-BX)* is a cascade refrigeration system with a low stage compressor and a high stage compressor. Its temperature ranges from -73° C to +240° C with an accuracy of 0.3° C. The internal dimensions of the cold chamber itself are 1219 x 1219 x 1219 mm (48''x 48''x 48''). Figure 3.30 shows the setup of the test. The test box is inside the cold chamber (with a small window in the door), and the wires are led through a whole on the right wall to the dSpace and the DC power supply. The laptop records all data for final analysis and control of the parameters (motor speed, current, reference signal). The temperature is measured with a thermometer, which has two separate temperature sensors. One channel is used for measuring the room temperature; another one is inside the box, for measuring the temperature change due to the heat emission of the drivetrain components. The following sections describe a series of experiments performed to validate drivetrain system components. A table of results is in Appendix I, Exhibit 5.

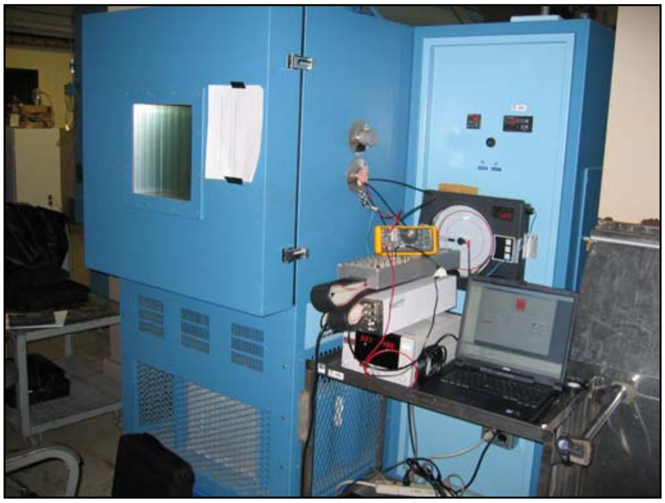


Figure 3.30: Cold Chamber Test Setup at CRREL

3.5.3 CRREL: 1. Cooling Down Procedure

Description : in the first test the temperature in the cold chamber is cooled down from room temperature to -40° C over a period of 120 minutes

Box : open

Motor : running under no load until cold temperature is reached

Goal : analyzing the interrelationship of chamber temperature, box temperature and current provided by the power supply

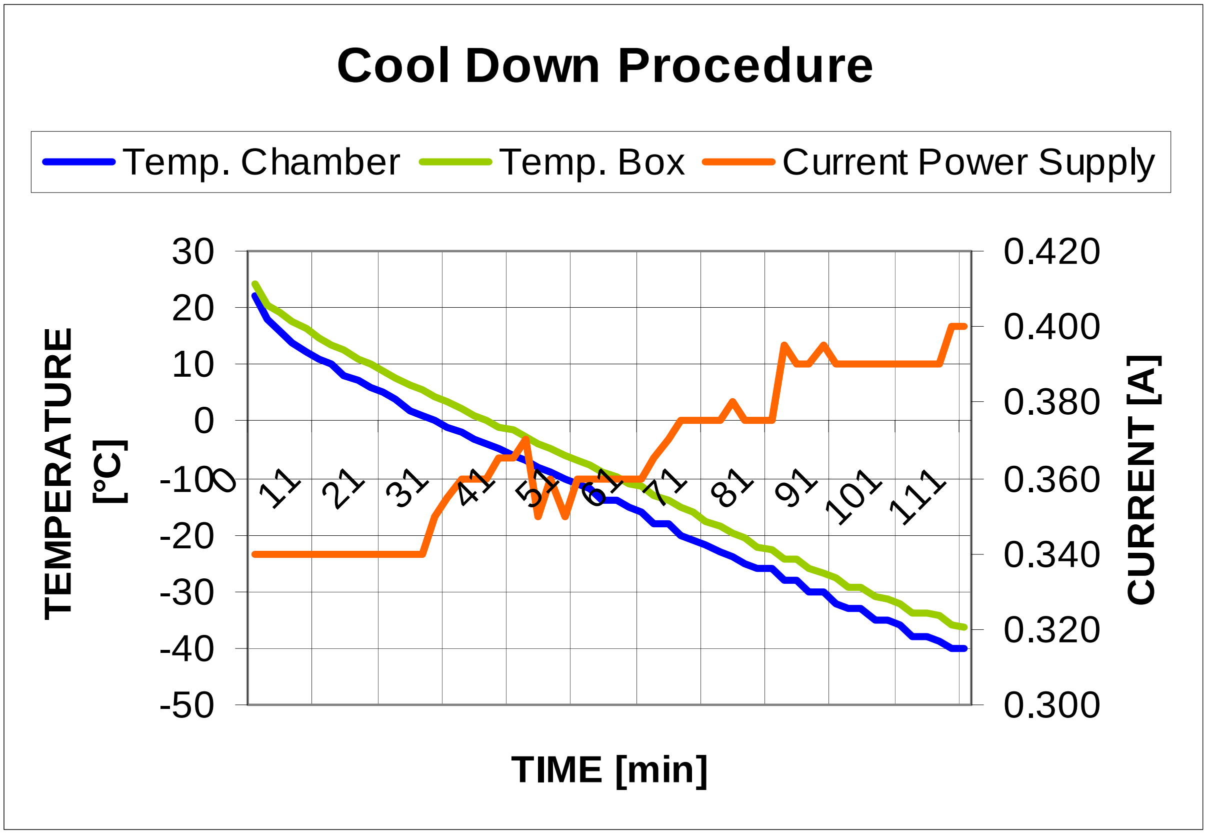


Figure 3.31: Cold Chamber Test Setup at CRREL

Result: As expected the box temperature is always a little bit warmer than the chamber temperature. This is because; the cold air has to cool down not only the air inside the box, but also the material of the box. Therefore the temperature change is delayed. But both curves are linear, which indicates a continuous cool down process of the cold chamber and the temperature difference is constant.

The motor current increases with dropping temperatures, because the lubricant gets stiffer and the inner resistance increases. In the test it doesn’t seem to be linear, but the trend is clearly visible. A reason for a non-linear relationship might lie in the different distribution of the temperature gradient around the motor and the gearhead. Thus, it takes time for the cold air to affect the motor performance.

3.5.4 CRREL: 2. Motor Start

Description : the motor will run and pause 30 times at specified intervals with chamber temperature constantly at -40° C

Box : open

Motor : running for 10 seconds under no load – pause for 30 seconds, repeat 30x

Goal : Simulate a motor start under Antarctic temperature conditions (-40° C); this situation is similar to when the robot is on low batteries but needs to move occasionally in order to avoid being covered by snow

Result: The table (Figure 3.32) shows the average motor current draw during the 10 second starting phase of the motor. The temperature inside the box was almost -40° C. The no-load current to start the motor is twice as much as under room temperature conditions (but drops back to normal relatively fast). There is no significant change in the current between the attempts; hence, the start of the motor is of no permanent effect for the following starts. This is because 10 seconds are not enough to evolve any significant influence on the drivetrain components.

Trial #	Temp – Box [°C]	Current "I" [A]
1	-37.7	0.61
2	-37.6	0.60
3	-37.7	0.62
4	-37.7	0.61
5	-37.7	0.61
6	-37.7	0.62
.	.	.
.	.	.
.	.	.
26	-37.7	0.60
27	-37.7	0.60
28	-37.7	0.60
29	-37.7	0.57
30	-37.7	0.57

Figure 3.32: Start Test Series

3.5.5 CRREL: 3. Long-Run Analyzes

Description : the motor will conduct a long-run at constant -40° C

Box : closed

Motor : running for 2 hours without load at a speed of 5,127 RPM

Goal : Simulate a robot traverse over a longer distance and gather information about the reliability of the components and both the temperature change inside the box and its effect on the motor current; this test is the basis for further *R-value* calculation (see below 3.4.7)

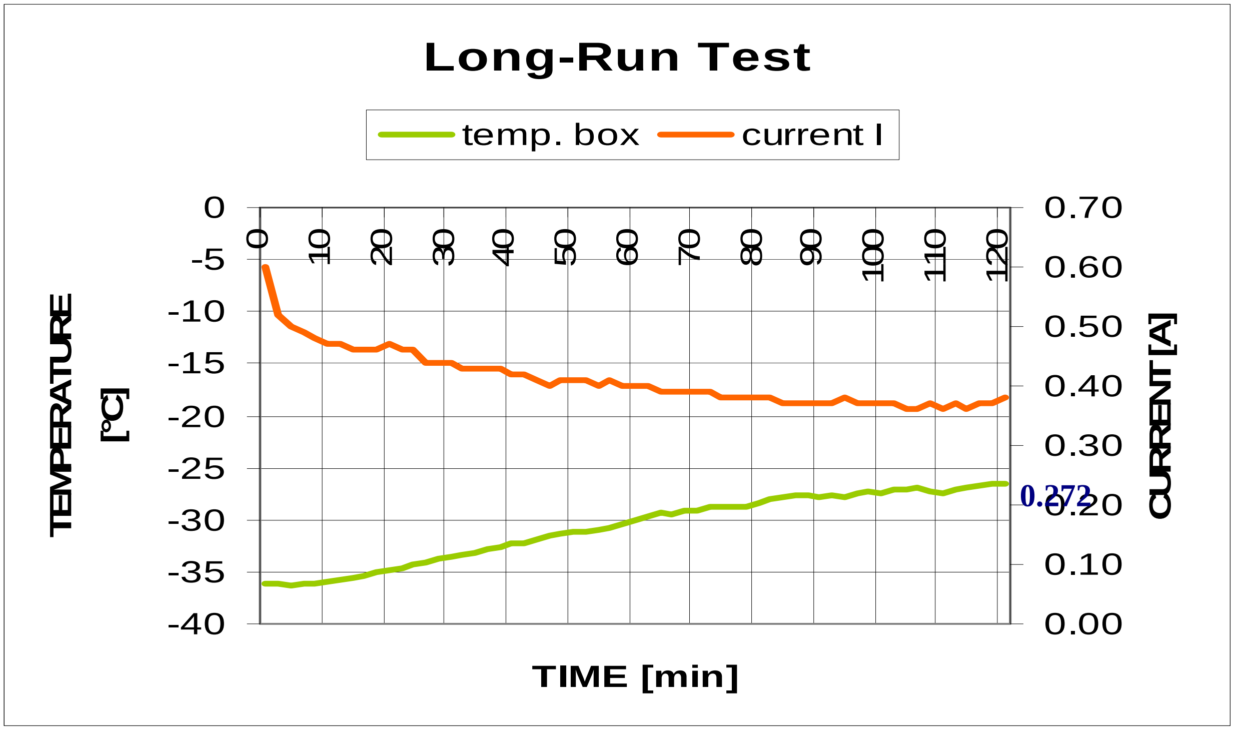


Figure 3.33: Long-Run Test

Result: The temperature inside the box increased within the two hours from -36° C to -26.5° C because of the heat emitted by the components. The R-value of the robot test box is calculated later. While the temperature rises, at the same time the current drops 37 % because the lubricant warms up and the viscosity reduces. Especially within the first 10 minutes the current fell for more than 20 % (Figure 3.33)! The no-load current of the motor (without gearbox) is at 0.272 A at room temperature.

3.5.6 CRREL: 4. Motor Start At Extreme Cold (-50° C)

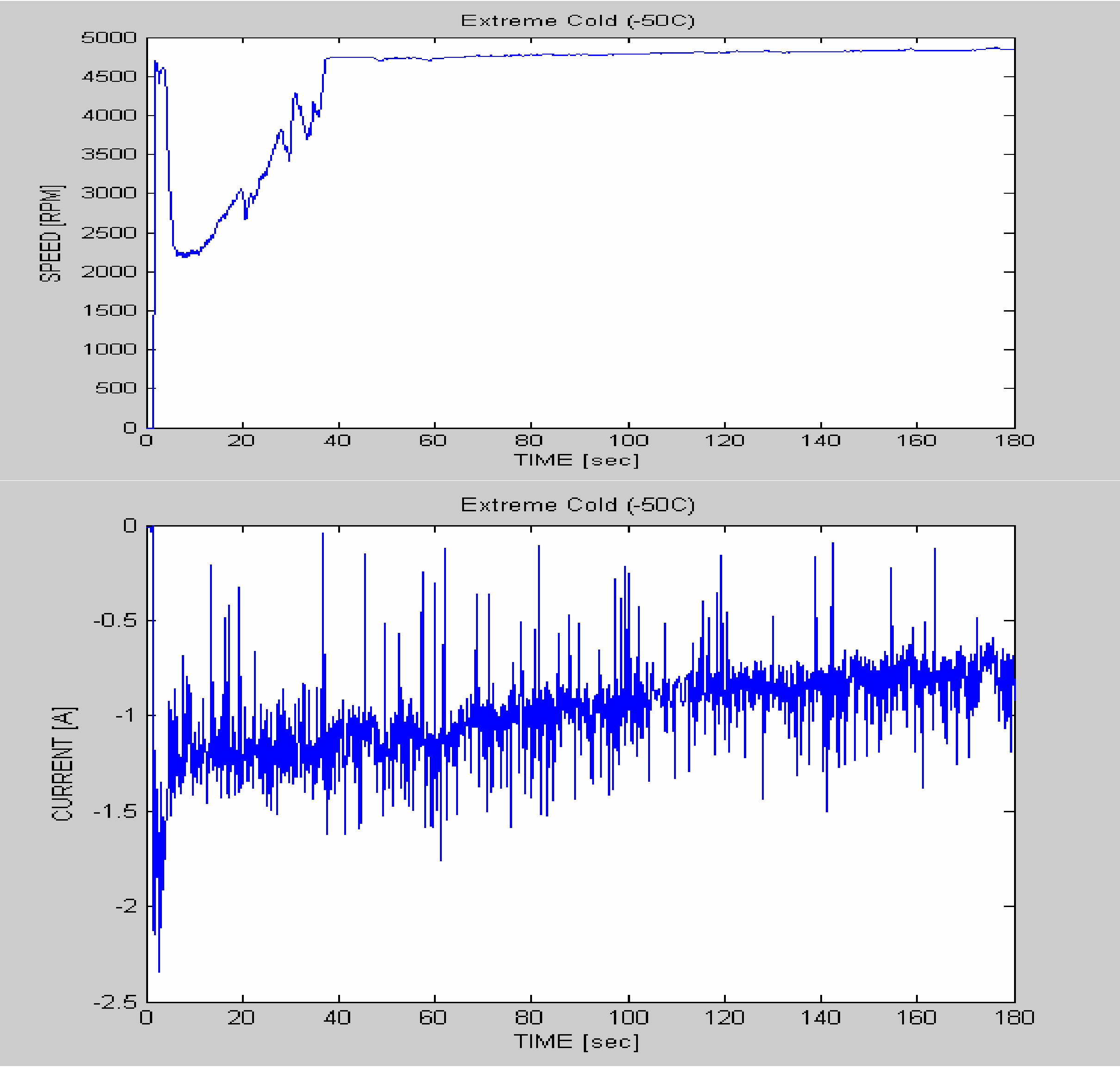
Description : the chamber is cooled down to -50° C; the motor has to perform a start test

Box : open

Motor : start test only under no load

Goal : Simulates a situation where the robot is exposed to temperatures less than -40° C (due to a weather change or any other reason the temperature may drop from the average temperature to an extreme temperature)

Figure 3.34: Extreme Cold Motor Start



Result: The above diagrams are inserted from MATLAB. They illustrate the motor behavior for the first 3 minutes after starting the motor. The cold chamber was cooled down to -50° C, and the temperature inside box was almost the same (-47.5° C). One thermocouple was placed directly on the motor and the display indicates a temperature of -39.6 on the motor casing.

The speed of the motor drops dramatically from 5000 to 2000, but increased rapidly back to 5000 within the first 40 seconds.

The peak current during this extreme starting procedure was about 2.5 A, but the current stabilizes after 10 seconds back to ~1.3 A, and decreases further with time to a more moderate value around 1 A after 3 minutes. The current curve shows spikes, which can be suppressed by using a capacitor. This electrical component will also reduce the peak current draw upon starting the motor by discharging the capacitor.

This final test resulted in an important conclusion: the test confirms that *the propulsion configuration is capable of operating in temperatures colder* than the assumed average temperature of -40° C, but even -50° C. A period of driving time will warm up all components again, and it will take time to cool them down to the surrounding temperature. Only the gearhead was rated to -50° C, but the other components all the same (like gearhead and motor controller) worked perfectly, too. This is a very encouraging outcome of the test and demonstrates a confirmation of a proper component selection.

Of course this is a worst case scenario, because no internal heat was generated and motor and controller are at ambient temperature -50° C. Actually some heat is dissipated in the box and the temperature during the Antarctic summer is usually not -50° C.

3.5.7 R-Value of the Test Box

The R-value has been introduced in 3.2.4 (insulation), therefore this section deals only with the calculation of the R-value for the test box, based on the test results of the long-run test (3.4.5).

$$\left.\begin{array}{l}\textit{Input data: } \text{temperature outside the box for } t_0\text{=0: } \upsilon_0 = \text{-40° C} \\ \qquad\qquad\quad \text{temperature inside the box for } t_1\text{=120: } \upsilon_1 = \text{-26.5° C}\end{array}\right\} \quad \Delta\upsilon\text{=13.5° C; } \Delta t\text{=120 min}$$

V_{Box}=0.00906 m³

heat energy: $Q = (1 - \eta_{Motor})\, W_{el}$ (3.34)

$\Rightarrow Q = 23614\ [J]$

Note: The electrical energy delivered to the motor is based on U=48V and the motor current according to the long-run test. W_{el} = U x I x t

R-Value: The following formula calculates the R-value, in particular regards to the units

$$\boxed{R = \frac{\Delta\upsilon \left[\frac{h\ °F}{BTU}\right] A_{total}\ [sq - ft\,]}{q} \qquad (3.35)}$$

(i) with

$$q := \frac{Q}{\Delta t} = 3.28 \left[\frac{J}{s}\right] \qquad (3.36) \qquad \Rightarrow \quad \frac{\Delta\upsilon}{q} = 4.12 \left[\frac{K\ s}{J}\right]$$

(ii) 1 K ≅ 9/5 ° F
(iii) 1 sec = 1/3600 h

$$\Rightarrow \quad \frac{\Delta\upsilon}{q} = 7.42 \left[\frac{°F\ s}{J}\right] \quad \Rightarrow \quad \frac{\Delta\upsilon}{q} = .00206 \left[\frac{°F\ h}{J}\right]$$

(iv)1 J ≅ $9.4781\text{x}10^{-4}$ BTU

$$\Rightarrow \quad \frac{\Delta\upsilon}{q} = 2.17 \left[\frac{h°F}{BTU}\right]$$

(v) A_{total} = emitting area (w/o the bottom panel) = *4x* 365x265mm² + *1x* 365x365mm²

= 666,125 mm² = 7.17 sq-ft

(vi) Finally leading to an R-value of:

$$R := \frac{\Delta\,\upsilon\,A_{total}}{q} = 15.6\left[\frac{h\ {}^{\circ}F}{BTU}\right][sq - ft]$$

Interpretation: An **R-value of R-15.6** is a reasonable value for this test run, because not the entire emitting box surface is covered with vacuum panel R-25. Only less than 50 %, namely the center of the box walls, is covered with the high-tech vacuum insulation. The rest is covered with a simple R-6.5 insulation sheet.

Furthermore, the formula above for the R-value is valid for the steady state, meaning constant temperatures on both sides of the box. If the steady-state is not reached yet, and assuming that the box is not perfectly sealed (e.g., hole for cable in box wall; cover does not seal perfectly) thus leading to leakage, the R-value could be in real even better!

However, it seems that a minimum R-value of R-15.6 is guaranteed by the combined insulation of the test box walls. Not to forget that the one inch honeycomb and ¼’’ Lexan glass add further R-value to the insulation performance.

Assuming above all that the steady state was not reached within the two hours long-run test, it is more likely that the R-value is higher, close to R-20. Depending on the material, the steady state can be reached from within a few hours (plexi glass) to a couple of days (foam).

Another fact is likely to contribute to a better insulation performance: the surrounding solar panels will warm up in the Antarctic sun and emit heat to the robot box, hence reducing the temperature at the outer box wall and also reducing the air flow from the inside of the box to the outside.

3.6 Vision of the Next Generation Robot

This first autonomous cool robot prototype is part of a *feasibility study* for the Antarctic region. It is meant to demonstrate that a robot of this size can traverse the Antarctic autonomously without any fossil energy source – only driven by the sun power. This cool robot should confirm the overall concept underlying this project.

However, based on the experience and tests with this prototype, the successor could exhibit a number of mechanical improvements. The design could be optimized to the task, thus, leading to the following recommendations, which are considered for a second generation robot:

(i) A high potential lies in the optimization of the robot wheels: for the first prototype inflatable tires are recommended, but further testing could proof that rigid tires perform at least as good as inflatable tires. A rigid tire could reduce the total vehicle weight significantly. The selection of the diameter would also be more flexible.

(ii) Depending on a weight trade-off and the temperature inside the main body, the best insulation material has to be balanced: Vacuum Panel vs. Standard Insulation Sheet.

(iii) The weight of the robot box could be optimized a by new design and by a new stress analysis based on the actual (measured) rolling resistance. This will probably lead to new dimensions for the honeycomb robot box

(iv) All lubricants – including the bearings – should be changed with a high performance braycote with a low temperature rating.

(v) Surfaces which are directly exposed to the snow could be painted with a special low adhesive lacquer, so the snow will not stick to the surface.

3.7 Summary of the Mechanical Design

This chapter deals with the design of the robot body and the selection of components based on a stated robot model. The components were tested and the results are encouraging for a successful deployment of this cool robot.

The model analyses different locomotion methods, comparing a track-driven vehicle with a wheel-driven vehicle. As it turns out that a wheel-driven vehicle would be capable to deal with the Antarctic terrain at lower energy consumption, one has to identify the proper steering method. Among a number of different steering schemes, skid steering seems to be the ideal steering mechanism for the Antarctic mission, because of its simple design and the low curve radii declared in the mission description. Mentioning the special conditions at the polar plateau, which will strongly affect the robot design, the demanding torque is calculated. This value determines the most important component selection: motor and gearbox.

All components are chosen very deliberately. Besides the propulsion and its motor controller, also the structural material needs to be selected. A honeycomb panel is able to deliver the structural stiffness and strength for a robot body. This robot chassis accommodates all important components and instruments at a very light weight. The idea to keep the batteries inside the main body as warm as possible, leads to a special insulation type: Vacuum Panels. They exhibit an extraordinary insulation performance, but they are relatively heavy. One of the most difficult choices is without any doubt the proper wheel. Different tires were examined and finally a 20’’ ATV tire has been chosen. However – further research has to be done on this issue and may lead to a different (rigid) tire.

The restricting weight limit of 75 kg for the vehicle without payload (90 kg w/ payload), predetermined in the vehicle requirements, has the major influence on the overall robot layout and has to be taken into account in every decision. An in-wheel solution, with solar panels surrounding the whole vehicle, seems to win the energy-gain vs. weight-increase trade-off (Figure 3.35). Therefore, the further robot layout presumes this vehicle design. The overall design is completed by some stress analysis and dimensioning.

In order to gather further information, a test box was built. The test box is a smaller scale of the future robot body and is used for testing the components at -40° C and -50° C in the cold chamber of the CRREL research facility. The tests were successful and confirmed the right selection of the components.
A second test, conducted in the laboratory, identifies the motor efficiency. This value has a strong influence on the overall power-system layout of the robot. As the required power determines the size of the solar panels, this area is only restricted by the total robot weight limit and the overall dimensions of the robot, which has to fit into the cargo plane.

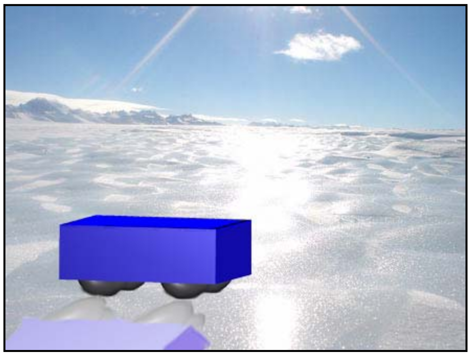


Figure 3.35: Robot at the Antarctic [Simulation]

As a conclusion, it turns out that the mechanical design of the cool robot is feasible within the weight and the budget limit. With state-of-the-art solar cells and batteries, the power system will be able to provide enough energy required by the propulsion. This comprises the entire robot model.

Chapter 4 *The Navigation and Control System*

The navigation and control system is Gunnar Hamann's part of this thesis about the Robot for the Antarctic. It first introduces the main challenges and presents the component selection. Secondly, it shows the analysis of the selected components for the navigation and the control system. Subsequently, the overall navigation program and its programmed functions are presented. Finally, a summary of the project results is given.

4.1 Introduction in the Navigation needs for this project

Navigation has its origin as a part of nautical science. Today it generally stands for the determination of the position and bearing of ships, planes, spacecrafts and ground vehicles.

The navigation process can be divided in two major parts. One being the positioning and guidance from waypoint to waypoint. The second being the obstacle avoidance occurring during the overall travel from waypoint to waypoint. The positioning and the obstacle avoidance imply very different requirements to the components needed.

At the Antarctic Plateau there are no major obstacles of concern for a robot of this size. The well known crevasses that are dangerous to any ground vehicle or person on the ground are all outside the Plateau. The only possible obstacles are sastrugis, which are a type of old, windblown snow waves. They can grow to bigger sizes but are mainly almost not visible. (**see figure 4.1**)

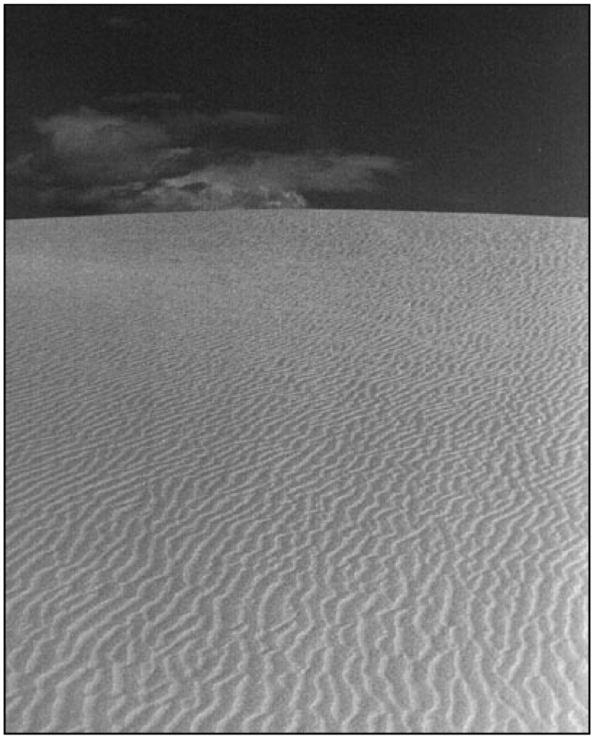


Figure 4.1: Picture of sastrugis at the Antarctic

Therefore this chapter will focus on waypoint navigation of the robot. Furthermore the results gathered from experiments at the Antarctic with existing robots regarding the performance of the components used for the obstacle avoidance will be discussed.

All the components of the navigation and control system including the positioning and obstacle avoidance have to meet the requirements that result from the robot's task and the application area. This ensures the reliable and safe functionality of the overall system and the interaction with the components of the mechanical design group and the power system group.

The most general and restraining requirements of the project arise from the harsh environment of the application area. The deployment period will be the summer at the South Pole. In the summer the temperatures rise above -40° C in November and stay at around -20° C until they drop below -40° C again in February. To be conservative with the requirements, all components have to be rated to -40° C or have to be tested for this cold temperature. Humidity and condensed water are not a big concern, because first the Antarctic is the driest continent (**see chapter 2.2**) and second the temperatures inside and outside the robot will always stay below 0° C. Although the Antarctic is known to be the windiest continent on the earth the wind at the South Pole is moderate and does not exceed 20-30 miles per hour.

Another challenge especially for vision based instruments for the obstacle avoidance is the bright sunlight in connection with the reflection of the snow. In combination with the snowdrift the contrast vision is very poor but still the robot has to be able to travel to its destination.

Due to the long distances that will be travelled and the wide open surface of the Antarctic the demand for a very high accuracy of the navigation system is not crucial. However the accuracy needs to be high enough to ensure the ability to travel from waypoint to waypoint. This helps to achieve the overall goal of building a low-cost robot.

The second group of constraints and challenges originate from the mission of the robot. In order to reach its arrival point for the instrument deployment with a distance of about 500 km in a reasonable time of two weeks the robot has to be able to travel with an average speed of about 1.5 km/hr and can sustain a maximum speed of approximately 4.4 km/hr (1.2 m/s). This is about three times as fast as all the existing robots travel and requires simple algorithms to reduce the calculation time.

In the project description, a network of the robots is envisioned that can be regrouped in real time to react on unpredictable changes of the mission settings. Therefore the robot needs the ability to receive new waypoints and to send its current position. The ability to send its current position back to the base station is also a safety feature for rescue purposes if the robot gets stuck and can not free itself.

Finally the robot itself presents an instrument to gather information. During its travel it can store settings of its attributes regarding the control system or the power system that will lead to a better understanding of the properties of the Antarctic region and help improve the next generation of the robot. Many characteristics of the Antarctic such as the rolling resistance, the power density of the sunlight, the sinking of the wheels on the snow, etc., are only known on frequently used paths between the stations or at the immediate surrounding area of the stations at the Antarctic.

4.2 Analysis of mobile robot positioning components

To select the best components for the navigation system of this robot, first reasonable common and existing components will be analyzed with regard to the requirements described in chapter 4.1. With the selected components a matching, programmable controller will be selected.

The main components of navigation systems today are the magnetic compasses, the gyroscope, the Global Positioning System (GPS) and cameras. In addition sensors to measure speed, acceleration and inclination are deployed. The usability is strongly depending on the application area and budget restrictions.

4.2.1 Magnetic Compass

The robot's heading is a very significant navigation parameter. For this reason, sensors which provide a measure of absolute heading are very important in solving the navigation needs of autonomous platforms. The magnetic compass is such a sensor.

The magnetic compass points to a location close to the north magnetic pole which is located about 600 miles from the geological North Pole. This variation is the first of two compass errors. The value of the variation depends on where on earth the compass is. The variation is the angular difference between true north and magnetic north. The value can increase to a point where magnetic compasses are useless.

The second compass error is the deviation. The deviation is the influence of the immediate environment upon the compass. The deviation is influenced mainly by iron bearing metal and other magnets caused for example by flowing electricity. The deviation is not constant and different in every vehicle. It varies with changing instruments deployed on the platform [BOR96].

The error of the magnetic compass increases the closer it gets to the poles. As the application area is located directly at the South Pole another fact has to be considered. First the position of the geographic South Pole changes about 10 meters per year along the 60 West meridian because of the moving ice. Second and more importantly the magnetic south pole travels 10 to 15 km north-westerly each year. Electric currents and the rolling motion of the liquid iron core of the Earth dictate the position of the magnetic poles. The pole wanders daily in a roughly elliptical path around this average position, and may be as far as 80 km away from this position when the Earth's magnetic field is disturbed [WEB02].

Taking these conditions into account the use of a magnetic compass as part of the navigation system for this application can not be recommended because the important heading information is imprecise with a continuously changing error.

4.2.2 Gyro compass

The gyro compass is essentially a north seeking gyroscope. It consists of a spinning wheel, which is held by two gimbals. The spinning wheel always points in one direction which is aligned with a point to true north. The gyro is a mechanical device with an inherent error caused by friction, motion of the platform and power malfunctions. The error is the difference between the gyro reading and the true bearing. A gyro compass in combination with accelerometers is used for inertial navigation. Measurements are integrated once or twice to yield position. Inertial navigation systems have the advantage that they do not need external references. But inertial sensor data drifts with time because of the need to integrate data to yield position. Any small constant error increases without bound after integration. This makes inertial sensors mostly unsuitable for accurate positioning over extended periods of time [BOR96].

Gyros that work at low temperatures and with a good performance are very expensive and range in the area of $ 10,000 each which would be almost 50% of the budget for the robot project.

4.2.3 Global Positioning System

The GPS is the latest technology for outdoor navigation. The system provides accurate, continuous, worldwide, three-dimensional position and velocity information to users with appropriate receiving equipment. It was developed as a joint services program by the Department of Defense in the 1960s. The system consists of 24 satellites which transmit encoded radio frequency signals. A worldwide ground control network monitors the health and status of the satellites. GPS can provide service to an unlimited number of users since the user receivers operate passively [KAP96].

The ground-based receivers can compute their position by measuring the travel time of the satellites’ radio frequency signals, which include information about the satellites’ momentary location. Four satellites are required to determine user latitude, longitude, height and receiver clock offset from internal system time. If either system time or altitude is accurately known less than four satellites are required [BOR96].

Until the late 90s the US government applied small errors in timing and satellite position to prevent a hostile nation from using GPS in support of precision weapons delivery. This intentional degradation was called selective availability and decreased the accuracy for other than military receivers to around 100 meters. However in the late 90s the selective availability was turned off and the accuracy of the GPS dramatically improved to less than 10 meters. Furthermore the accuracy of the GPS receiver can be improved with a practice known as differential GPS (DGPS). This concept is based on the premise that a second GPS receiver in fairly close proximity of about 10 km to the first will experience basically the same error effects when viewing the same reference satellites. The second receiver must be fixed at a precisely known location. Its calculated solution can be compared to the known position to generate a composite error vector representative of prevailing conditions in the immediate locale. This information can be passed to the first receiver to null out the unwanted effects to effectively reduce the position error for a commercial system [BOR96].

Tests with a GPS receiver with real time differential correction capability mounted on a moving platform in the Antarctica were conducted by the Robotics Institute of the Carnegie Mellon University in 1998. The results show an accuracy better than 10 cm. However the tests area was at Patriot Hills which is about 1500 km north-west of the South Pole [FOE99].

Due to its worldwide availability the GPS is the main navigation component used by scientists at the South Pole. On the traverse for research or between stations they rely on handheld GPS receivers. Due to the great distances between the stations and harsh conditions almost no broadcast stations for DGPS are available [LEV04].

The GPS meets all requirements for this robot. Off the shelf receivers are rated to an operating temperature of -40° C and are low on energy consumption and retail for as low as $ 100. To be conservative and to ensure that the GPS itself is reliable and precise enough, the accuracy of the selected GPS receiver needs to be tested without the differential correction. Furthermore the nature of the signal needs to be evaluated, e.g. whether the signals randomly “jump” around a mean with a certain standard deviation or “walk” around the mean in a path.

4.2.4 Camera Systems

Camera systems are vision based sensors that can be used for both positioning and obstacle avoidance. Most camera systems deployed on robots for autonomous navigation consist of two cameras for stereo vision. These cameras represent the “eyes” of the robot and give it the ability to visualize its environment. To process the information of the three dimensional pictures from the cameras an extensive processor with sophisticated algorithms is needed. For the navigation with cameras the technique of landmark navigation or map-based navigation can be applied. With landmark navigation the robot compares its picture with known natural or artificial landmarks to determine its position. To use map-based navigation the robot compares its picture of the environment with a stored map until it finds a matching picture to position itself on the map [BOR96]. Both techniques require a detailed knowledge of the application area and distinct landmarks to determine the position. They work fine in lunar-like or desert environments. Tests conducted by the Robotics Institute of the Carnegie Mellon University with their Ratler Rover on an undulating plateau in 1996 have shown that only about 2 % of the image is needed for reliably detecting features on the order of 20 cm high [SIM96].

However the Antarctic at the South Pole does not provide distinct natural or artificial landmarks and makes the use of a camera system for positioning impossible.

Tests with the Nomad robot of the Carnegie Mellon University at Patriot Hills in 1998 have shown that the contrast visibility in the Antarctic is too low to use the images from the cameras for obstacle avoidance. Deployed algorithms to evaluate the images were not able to distinguish obstacles from blowing snow or flat surface [MOR00].

The results of these tests help to conclude that a vision-based system consisting of cameras is not applicable for positioning an autonomous robot in the Antarctic. It requires not only high calculation power and sophisticated algorithms but also the components needed are rather expensive. Most important is its failure at similar application areas.

4.2.5 Iridium modem

The requirements for this project demand the ability of the robot to communicate with its base station. Both sending and receiving positions or commands is required to either rescue the robot when it is stuck in place or reprogram its waypoints remotely.

For short distances, the wireless network could be deployed to communicate with the robot. But this solution would significantly limit the operating radius of the robot. In order to fulfil the constraint of travelling one way distances of 500 km the robot needs a solution that is independent of the distance to the base station.

The Iridium Satellite System is the only provider of global, mobile satellite data solutions with complete coverage of the Earth including the Polar regions. Through a constellation of 66 low-earth orbiting satellites, Iridium delivers essential communications services to and from remote areas where terrestrial communications are not available. An iridium modem can call any regular phone. This reduces the need to one modem carried on the robot. First the service was only for military use but the commercial service was launched in 2001. The provided bandwidth of the modem is capable of processing a regular phone call and certainly fast enough to send short waypoint information.

Available Iridium modems are rated to operating temperatures of at least -40° C and cost around $ 1,400.

4.2.6 Additional sensors for the navigation system

To safely navigate the robot through unknown terrain additional sensors are employed on existing robots. First, additional sensors to improve the positioning of the robot will be described. Second, sensors to preserve the robot from tipping over or to drive into obstacles will be discussed.

To gather more information about the movement of the robot, the feedback from the wheel speed sensors on each motor can be implemented in the overall program. It is possible to use this information if necessary improve the position beyond that obtained using GPS. A filter that uses the GPS information and the information from the wheel speed sensors could help sensing the error of the GPS itself. Otherwise the extra information can be used to detect certain states, e.g., the robot is not moving but the wheels are spinning, or to set the vehicle velocity vector to zero when the wheels are stopped.

To decrease the error in the velocity determination evoked by slipping wheels, a touchless sensor can be implemented in the system. Such sensors are available and work with high frequency radar technology and a high accuracy. These devices work fine on streets etc. but fail on snow or in hard rain because of the reflections. Furthermore they are high in energy consumption and retail for as much as $ 1,600 [PRZ04].

Another feedback loop can be taken from the current sensors at the motor controllers. The current draw of the motor is proportional to the applied torque to the wheel. If the torque increases, then the current draw increases. This information can be used for the power management or for the task to use the robot as a test instrument itself. By monitoring and storing this information, the existing rolling resistance on a given terrain can be evaluated.

Almost all existing autonomous robots deploy two inclinometers to prevent the robot from tipping over. These inclinometers are connected to the overall controller and run on an interrupt that stops the robot as soon as a defined angle of inclination is reached. This event starts an evasion algorithm to find another way for the robot to proceed to its waypoint [SIM96, MOR00].

For obstacle avoidance, conducted tests have shown that the only reliable sensor to detect major obstacles in a white environment is an active laser sensor. The laser was unaffected by terrain type working as well in Antarctica as in tests made with the Nomad in Pittsburgh. Even the specular surface of the blue ice fields had no effects on the return signal, and overcast conditions also had no effect on the laser range finder. But the laser encountered problems during periods of blowing snow. The laser could reflect off snow flakes. If it reflected back a short distance would be measured whereas if it reflected away no return signal would be received. In mild levels of blowing snow, filtering was able to remove these effects but in heavy storms the laser could not be used [MOR00].

4.3 Description of selected components

To conclude the results of the analysis in chapter 4.2 the selected components will be described. This includes additional components to satisfy the need for communications and the presentation of the selected suitable controller to navigate the robot. To justify the selected controller and its suitability for the system, both the required interfaces of the components and the provided interfaces of the controller will be presented.

4.3.1 Specification of the selected navigation components and its interfaces

After evaluating the test results of the navigation components implemented in existing robots and extensive research on further existing problem solutions the navigation system consists of the following components:

- 1 * Non-differential GPS receiver (differential update if necessary)
- 1 * Iridium modem
- 4 * Wheel speed encoders
- 4 * Current draw sensors
- 2 * Inclinometers for roll and pitch angle
- 1 * Laser range finder

The GPS receiver is the main navigation component. It can provide information about the position, altitude, speed, true bearing and time. Because of its vital function a second identical GPS receiver could be deployed for redundancy. The non-differential GPS receiver has to be tested and validated to determine whether it is accurate enough for the robot's mission to travel from waypoint to waypoint. If not, the feasibility to achieve the goal, a DGPS receiver has to be evaluated. Furthermore, the information gained from the wheel speed and current draw sensors can be used to improve the position accuracy by implementing a filter to merge the information of both sensors. This solution demands a far higher computation power and more sophisticated algorithms. Otherwise the sensors can be used to detect certain states as described in section 4.2.6 or to react on a higher current draw according to the available power.

Being the only worldwide accessible mobile communication system the Iridium modem is part of the navigation system to meet the communication requirement.

A standard component of state-of-the-art autonomous robot navigation systems are two inclinometers to prevent the robot from tipping over. One inclinometer monitors the pitch angle and a second inclinometer the roll angle. These sensors must interrupt the controller to stop the robot if either the maximum pitch or roll angle is reached.

For the obstacle avoidance cited tests in section 4.2 have shown that most of the deployed solutions on existing robots fail in the Antarctic environment. The only fairly reliable component appears to be the laser range finder. Although test results show that this active laser fails in heavy snow storms they point out that it works reliably in mild weather conditions. Because of the low wind speed of around 20 to 30 mph at the South Pole, heavy snow storms as they occur at the coastal area of Patriot Hills (test area for the tests conducted by the Carnegie Mellon University) are very rare on the plateau [LEV04]. In order to prove this assumption the functionality of the laser range finder needs to be validated in the first tests of the robot at the South Pole.

At this point of the project the subsequently described components for the navigation system have been selected and tested. These components enable the further testing of the overall navigation program on an autonomous platform. All missing components at this stage are either needed for communications or for the obstacle avoidance which is not essential to the primary waypoint navigation system.

The selected GPS receiver is the M12+ Oncore from Motorola, Inc. There are several similar GPS receivers on the market from Trimble, Garmin, Ublox and Motorola. The Motorola receiver stands out of this group of receivers for two reasons. First, the receiver comes with extensive software and an evaluation board that allows separate testing of the receiver with various power supplies. Second is the positive experience with the receiver by another research group at the Thayer School of Engineering under the supervision of Marc Lessard, Assistant Professor of Engineering, for a different project in the Antarctic.

The M12+ Oncore receiver is a twelve channel GPS receiver. It has the ability to track up to twelve satellites at the same time. This has no direct influence on the accuracy of the receiver, however it increases the redundancy. One satellite may be in a bad status or blocked for different reasons. With more than four satellites tracked, the receiver selects the best satellites possible for every measurement and is not immediately affected if one satellite is blocked. This ensures the visibility of at least four satellites in the rare event of one being blocked at the South Pole. To improve the signal quality the receiver has an implemented filter [WAR04] .

The receiver was purchased from Synergy Systems, LLC, which distributes Motorola’s GPS receiver. The proper antenna and the receiver are rated to an operating temperature of -40° C and weigh less around 100 grams together. The operating voltage is three volts, the current draw is 60 mA and it uses a serial port as an interface to the controller (**see figure 4.2**).

Research on iridium modems shows that the NAL Research Corporation has the greatest experience with iridium modems at cold temperatures. Scientists in the Arctic, Antarctic and at the Thayer School of Engineering have experienced no problems with operating temperatures of as low as -70° C although the modems are only rated to -30° C [NGO04]. The modem is available with or without an internal GPS receiver. For more flexibility and more convenient maintenance the modem and the GPS receiver will be separate. The best suited modem is the model A3LA-IG. It weighs about 640 grams and operates at a voltage range from four to five volts and a current draw of 120 mA. The interface with the controller to send and receive commands is a serial port (**see figure 4.2**). The modem has not yet been purchased because NAL Research Corp. offered to donate one to support this educational research project as soon as one testing modem is available from their own laboratories.

The wheel speed sensor and current monitor are integrated in each motor controller package. The current monitor of each motor sends an analog signal to the main controller. The wheel speed sensor emits square waves to the main controller, which can be sensed and decoded with encoder channels.

The inclinometers and the laser range finder have not yet been selected, because they are not vital for the first evaluation of the navigation system. Both sensors will send analog signals to the main controller, which need to be permanently monitored to stop the robot in case that the signals exceed a certain limit.

4.3.2 Presentation of the controller and control applications

The main controller's selection is not only determined by the interfaces of the navigation components but also by future demands. To control the navigation components selected as described in chapter 4.3.1 the controller needs a minimum of two serial ports for the GPS receiver and the iridium modem. Moreover are up to four wheel speed encoders needed and an analog to digital converter with a minimum of seven channels. Four channels are used by the current monitors from each motor, two channels for the inclinometers and one for the laser range finder.

In addition, the controller needs a digital to analog converter with four channels to control each motor controller separately. As specified in chapter 3.1.2 the robot uses skid steering. This steering method requires the ability to control each motor separately. The turns of the robot are performed by sending different velocity commands to each motor controller. The motor controllers require an analog input signal in the range of -10 to 10 volts to control wheel speed between ± 50 RPM.
To perform the algorithm calculation and to control the robot in real time the controller needs to be fast and flexible in its programming possibilities. For further modifications and extensions, extra interfaces need to be available.

Due to positive experiences gathered from a project supervised by Prof. Laura Ray and confirmed with the records from a project of Prof. Marc Lessard the selection was focused on the products of Z-World, Inc. This company has a reputation for low-cost solutions of flexible off the shelf controllers. The product that meets the requirements for this project the best is the RCM 3100 Jackrabbit. It has a powerful 29.4 MHz processor with very low emission of electro magnetic impulses. It comes with a 512 KB flash Random Access Memory (RAM), 54 digital Input/Output (I/O) connection headers that share 6 programmable serial ports and two quadrature encoders for the wheel speed sensors. It operates at 3.3 volts and is rated to -40° C. The controller is programmed with both Dynamic C which is an extended version of the C program language and directly with assembly language. All programs are executed on the chip itself. For first testing the controller is mounted on an evaluation board.
For the analog input of the sensors and the analog output to the motor controllers extra converters are needed that use a serial port as the interface. Figure 4.2 shows the overall system diagram.

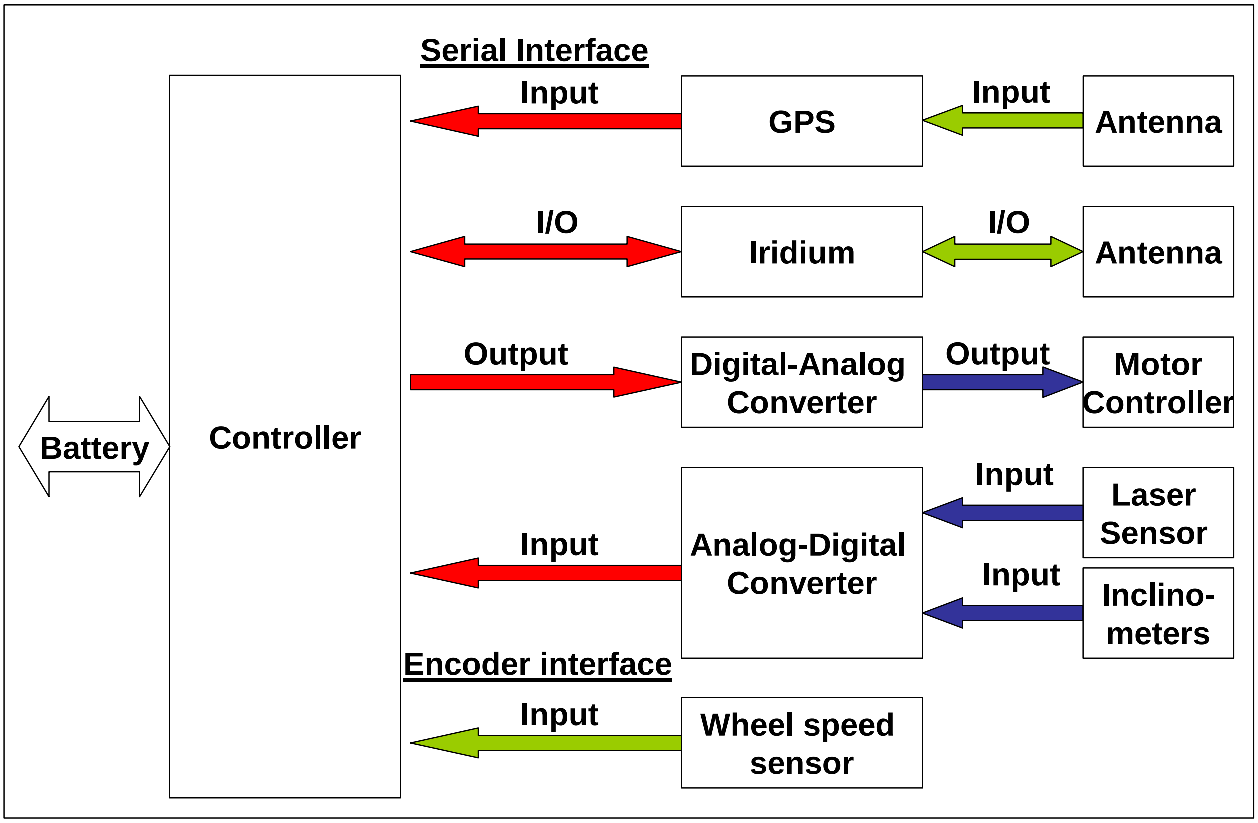


Figure 4.2: System diagram of the navigation system [HAM04]

The system diagram shows the connection of the controller with all navigation components. The orientation of the arrows displays the direction of the information flow they represent. The red arrows are serial port interfaces, the blue arrows analog signals and the green arrows digital signals. The interface with the battery/ power system is not yet determined.

The Analog-Digital Converter (ADC) and Digital-Analog Converter (DAC) have been selected from Maxim Integrated Products which is the leading manufacturer of integrated circuits. The ADC is the MAX 186. It is an eight channel converter with a resolution of 12 bits and a serial port interface. The input can be either 0 to 5 volts or -5 to 5 volts for each channel. The MAX 186 is suitable for prototypes because it is a through-whole solution, which is convenient to test on prototyping boards before being implemented in the final circuit board.

The DAC is the MAX 536 which is the best fit with the requirements. It is a four channel converter with a resolution of 12 bits and a serial port interface. It is able to provide an output range of -5 to 10 volts for each channel separately. As the MAX 186 it is a through-whole solution which helps the implementation in the navigation system in this early stage of the project.

A table summarizing the voltage needs and power draws is presented in figure xx in section 4.5.4.

4.4 Results of tests with the GPS receiver

To develop and verify the navigation algorithm for the overall program, first the components and their characteristics need to be tested and analyzed. Only checked components that fulfil the requirements for the project can be selected for further tests of the interaction of the whole system.
The GPS receiver is the main and most important component for the navigation system. The performed tests determine the accuracy of the position signal to determine if further filters or sensors are necessary. Furthermore the quality of the information about the true bearing is evaluated to draw conclusions for the overall navigation program.

4.4.1 Description of tests with the GPS receiver

Two main different forms of tests are performed with the GPS receiver. Each conducted test layout has different intentions.
The first test is made for a better understanding and analysis of the GPS position signal. The GPS receiver is fixed on one position and the GPS signal is taken for twenty minutes with steps of one signal per second. The results are used to help answer first the question of the relationship between sample signals, i.e., whether the signals randomly “jump” around a mean with a certain standard deviation or “walk” around the mean in a path. The residual distribution is determined using a Kolmogorov-Smirnov Goodness-of-Fit test.
The second test is performed on a moving platform. The platform is a car that moves with about the same speed as the robot on a straight line. The samples are taken every second. The goal is to estimate the path error. Also, the bearing information of the GPS receiver is plotted to understand its accuracy and usefulness for the navigation program. This second test is repeated several times under different circumstances but on the same straight line to verify the conclusions.
The results of both tests are processed and visualized with the mathematic program Maple. The source code of all tests is presented in appendix II.1. Subsequently the main results are presented and the conclusions for each test are provided.

4.4.2 Results of steady state test with the GPS receiver

The first test setup was performed on November 26 in 2003 on a sunny day with no clouds at -5° C at the Thayer School of Engineering on 43° 42,2564’ N and 072° 17,6683’ W. Twelve satellites were visible and, on average, five satellites tracked. The test started at 1.30 p.m. local time and ended twenty minutes later. The GPS receiver was connected to a laptop and 1274 samples were collected and stored through a logging program for evaluation. The data were prepared for evaluation with Excel. During the test only the numbers behind the comma of the longitude and latitude changed and were stored in the file “031126 GPS Lon Lat.txt”.
First the means of 2564 for the latitude and 6683 for the longitude are calculated (**see exhibit 1**):

$$dx := [2564.603611, 6683.142072]$$

Second the standard deviation in meters is determined. This calculation needs the transformation of the value of the samples from degrees to meters.

The Earth is segmented in 360 degrees. To determine any point on Earth the exact longitude and latitude is necessary. The longitudes are vertical and the latitudes horizontal (**see figure 4.3**). The longitudes value from 0° at Greenwich in Great Britain to 180° East or West. The latitudes value from 0° at the equator to 90° North or South at the poles. For the latitude the distance between two degrees is always close to one nautical mile or 1.852 km. The length of one degree for the longitude which represents East and West depends on the degree of the latitude because all longitudes meet at the poles. At the equator one minute of the longitude is approximately one nautical mile or 1.852 km. At the poles however the distance is 0 km. On all degrees in between the distance decreases from 1.852 km to 0 km.

To determine the standard deviation first the standard deviation in degrees is calculated. Subsequently it is transformed to meters using a scale factor of 1.852 km for the latitude and 1.359 km for the longitude. 1.359 km is the distance between two minutes at the test position of 43° N [WEB09].

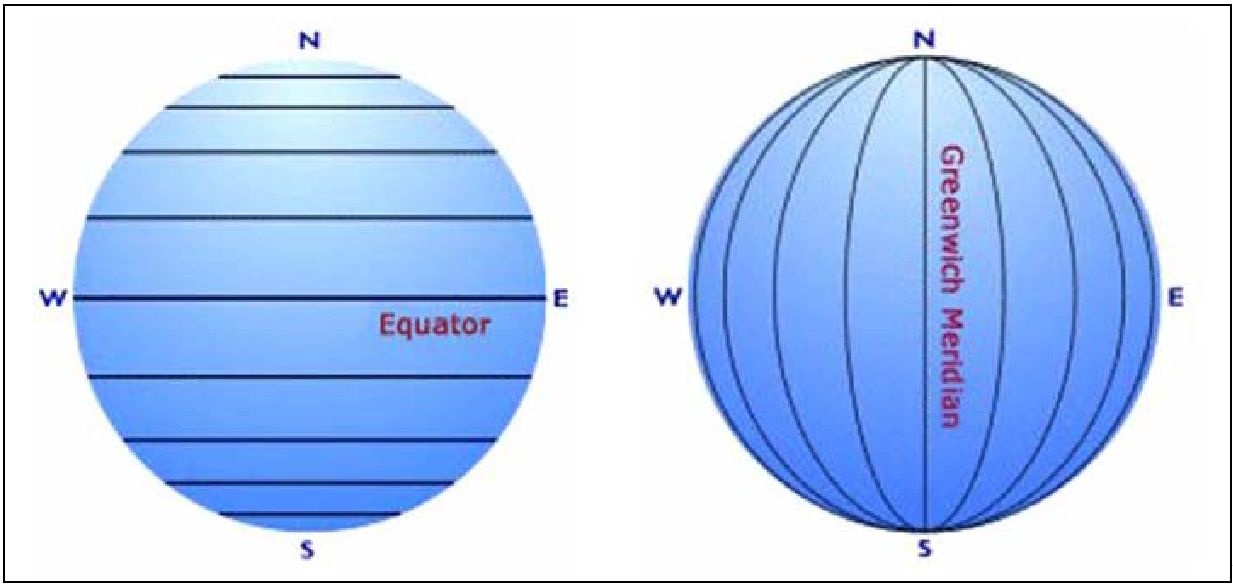


Figure 4.3: Drawing on Latitude and Longitude on Earth [HAM04]

This results in a standard deviation s_Lat_meter for the latitude of 2.096 meters and for the longitude of 1.657 meters (**compare exhibit 1**):

$$sx\_Lat\_meter := 2.095895532$$

$$sx\_Lon\_meter := 1.657165845$$

To understand the nature of the signals, both streams for the longitude and the latitude are plotted over time. To visualize the average of the samples each mean respectively is included in the plot. Both streams are similar in their general behavior. The longitude stream is shown as an example in figure 4.4.

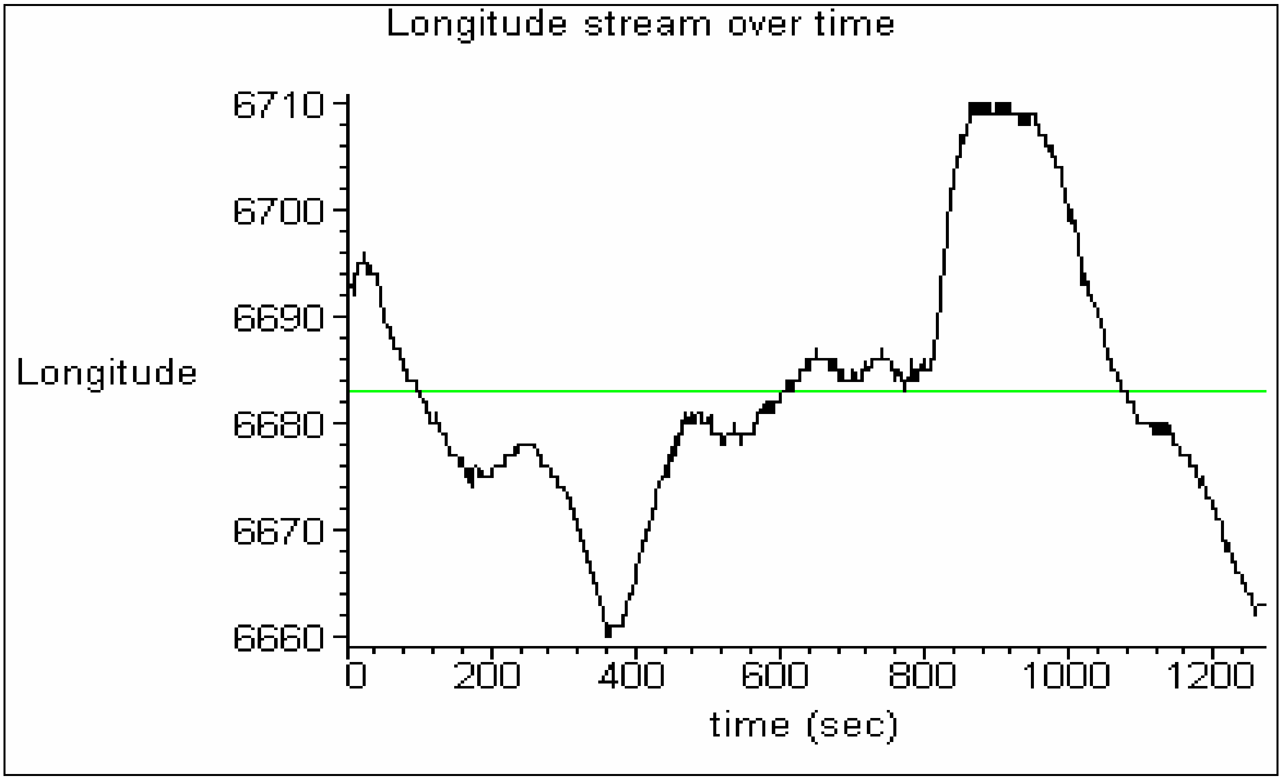


Figure 4.4: Longitude stream over time in black and mean in green [HAM04]

The signals are not randomly distributed over time with big distances from sample to sample. Instead, the signals appear to follow an oscillating "path" around the mean value of 6683. The overall maximum distance between any two points is (**compare exhibit 1**):

$d\_max\_meter\_Lat := 3.777411243$

$d\_max\_meter\_Lon := 3.649992415$

This is in agreement with the standard deviation of about 2 meters for each stream. However the maximum distance between only to consecutive samples is (**compare exhibit 1**):

$d\_max12\_meter\_Lat := .3704000000$

$d\_max12\_meter\_Lon := .1349000000$

This value is even better for the longitude presumably because of the lower distance between two minutes.

Most of the samples on the oscillating path (**see figure 4.4**) are located close to the mean. This gives rise to the assumption that they come from a population with a specific distribution, most likely a normal distribution.

The Kolmogorov-Smirnov Goodness-of-Fit Test (K-S) represents a common test to verify if a sample fits to a continuous distribution with a selected significance level. It is mostly used to test a sample on a normal distribution. The attractive feature of this test is that the distribution of the K-S test statistic itself does not depend on the underlying cumulative distribution function being tested. It is an exact test in comparison to the chi-square goodness-of-fit test which depends on an adequate sample size for the approximations to be valid.

It uses a hypothetical distribution function $F_0(x)$ and compares it with the empirical distribution function $F(x)$ of the sample $x_1,x_2,..,x_N$. The sample is sorted and divided in classes *NKA*. Each class has the same width calculated by dividing the range of the sample by the number of classes *NKA.* All samples are assigned to the according class. The classes are accumulated to make the sample comparable to a distribution function. After the relative fraction *rel_SnA* of each class in relation to *N* has been calculated the empirical distribution function $F(x)$ is compared to the hypothetical distribution function $F_0(x)$ with the same mean and standard deviation as the sample [ROT02].

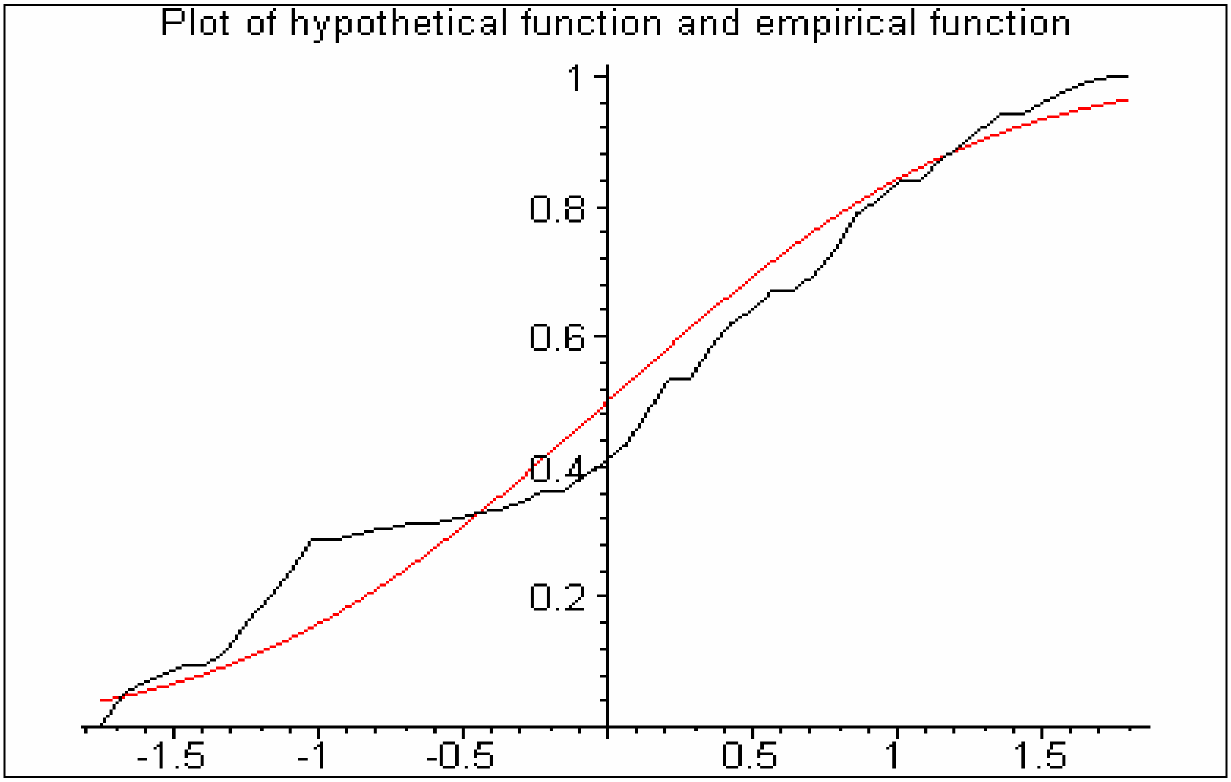


Figure 4.5: Plot of $F(x)$ in black and $F_0(x)$ in red [HAM04]

The hypothesis:

H_0: $F(x) = F_0(x)$ for all x

is tested against the alternative

H_1: $F(x) \neq F_0(x)$ for at least one value of x

using the test statistic

$TS = \sqrt{N} * D_N$ with $D_N = max| F_0(x) - F(x)|$

D_N is the maximum vertical difference between hypothetical and empirical distribution. The hypothesis is rejected if the test statistic is greater than the critical value $\lambda_{N;\ 1-\alpha}$ obtained from a table. For a significance level $\alpha = 0.01$ and a sample number $N > 100$ the critical value $\lambda_{N;\ 1-\alpha}$ is 1.62 [ROT02]. The resulting *TS* of the test is *5.35* (**see exhibit 1**):

$TS = 5.35 > \lambda_{N;\ 0.99} = 1.62$

This result rejects the hypothesis H_1 and confirms that the empirical distribution function $F(x)$ fits with a hypothetical normal distribution $F_0(x)$ for a significance level of $\alpha = 0.01$. Figure 4.5 shows the assumed and actual distribution for the test data.

The result of the K-S test helps to better understand the behaviour of the GPS signal. It confirms that the residuals of the GPS signals come from a standard distribution which has the biggest probability around its mean.

4.4.3 Results of moving tests with the GPS receiver

A second set of tests with the GPS receiver is necessary to analyze its performance under application conditions. The maximum speed of the robot is about three mph and it will mainly travel on straight lines. Because the robot uses skid steering to minimize the energy, the turns to correct its position have a rather big turning radius. For the simulation of the robot's behaviour on both the straight line and turns, a slow moving car functions as the test platform. The test is repeatedly performed on the same straight line under changing conditions. The number of tracked satellites varies from four to seven, the outside test temperature from -20° C to -5° C and the coverage of the sky from no clouds to snow fall. The test is conducted on different times throughout the day on the grounds of Dartmouth College. An explicit description of all test conditions for every test and the respective test results can be found in exhibit 2-7. However one test will be discussed in detail. The other tests do not have the exact same results but confirm the conclusions drawn from any test for the navigation system and program. On each straight line test a regression analysis is performed, which provides a least square fit to the straight line. The least square fit method minimizes the square value of the residuals. The residuals represent the vertical distance between the samples and the least square fit function. For the straight line, a first degree function is calculated for the regression analysis. The least square fit function represents the track, and the samples show the noise in the GPS signal. The value of the residuals is plotted but more importantly the offset of the track. These results help to decide whether an extra filter is necessary to enhance the accuracy and performance of the whole navigation system.

The sample test was conducted on January 9th 2004 at 11.00 a.m. on a sunny day with no clouds and an outside temperature of -20° C. The GPS receiver had 12 satellites visible and tracked 7 of them for its position determination. The GPS signal samples were collected every second for 277 seconds from the beginning to the end of the straight line.

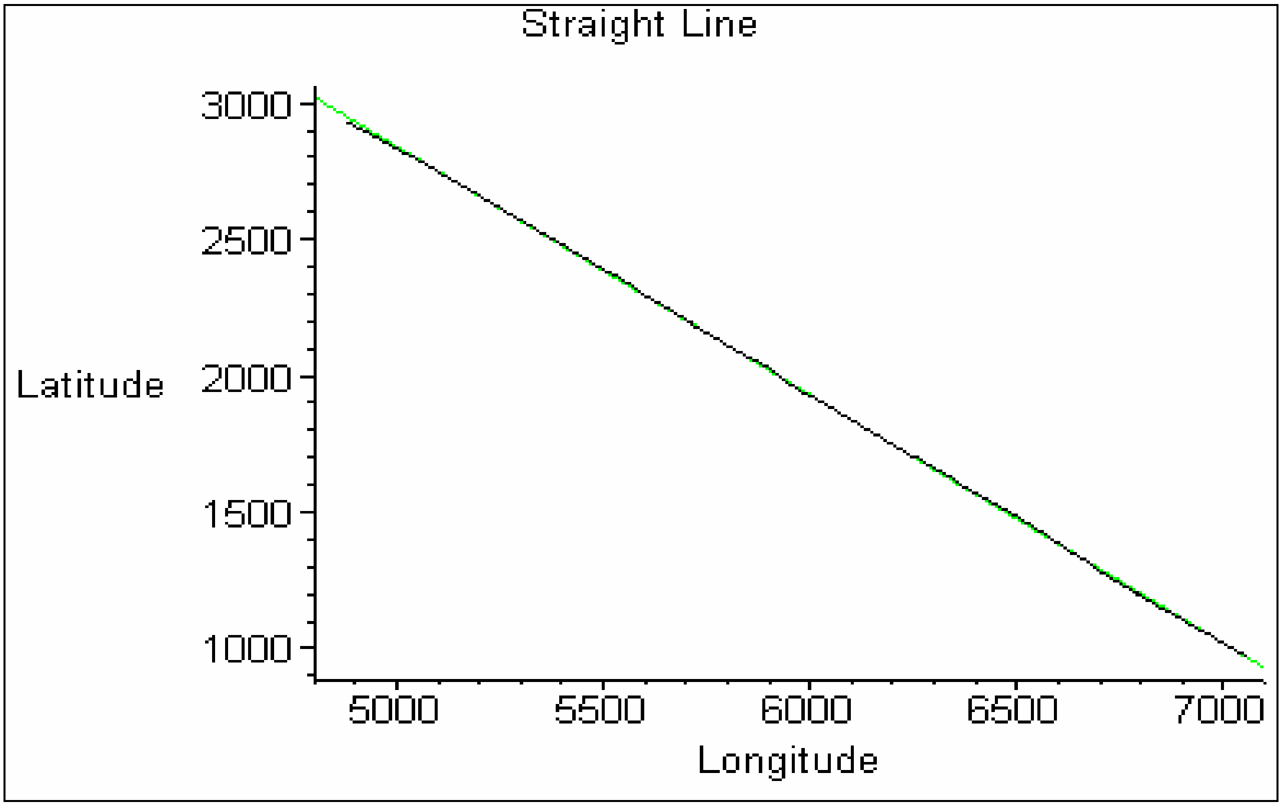


Figure 4.6: Plot of the samples in black and the least squares fit function in green [HAM04]

Figure 4.6 shows the plot of both the GPS signals and the least square fit function *gerfit* in the area of the samples. The units on x and y axis represent the four digits of the longitude and latitude position after the comma highlighted in red below. The rest of the position information does not change:

Latitude:	43° 42,4877
Longitude:	72° 17,2934

Due to the large sample size the offset is almost not visible in figure 4.6 . To visualize the offset, first the residuals from each sample point to the *gerfit* have to be calculated and transferred in the scale of meters. The value in meters depends on both the noise in the latitude and longitude signal. To be conservative in the transforming process, the distance between two minutes of 1.852 km is used. However, the actual values of the offsets are smaller, which increases the performance of the GPS receiver. The residuals are transformed in the value of the offset with basic geometric operations (**see exhibit 2**). The offset represents the orthogonal distance of a sample point of the *gerfit* function which represents the track.

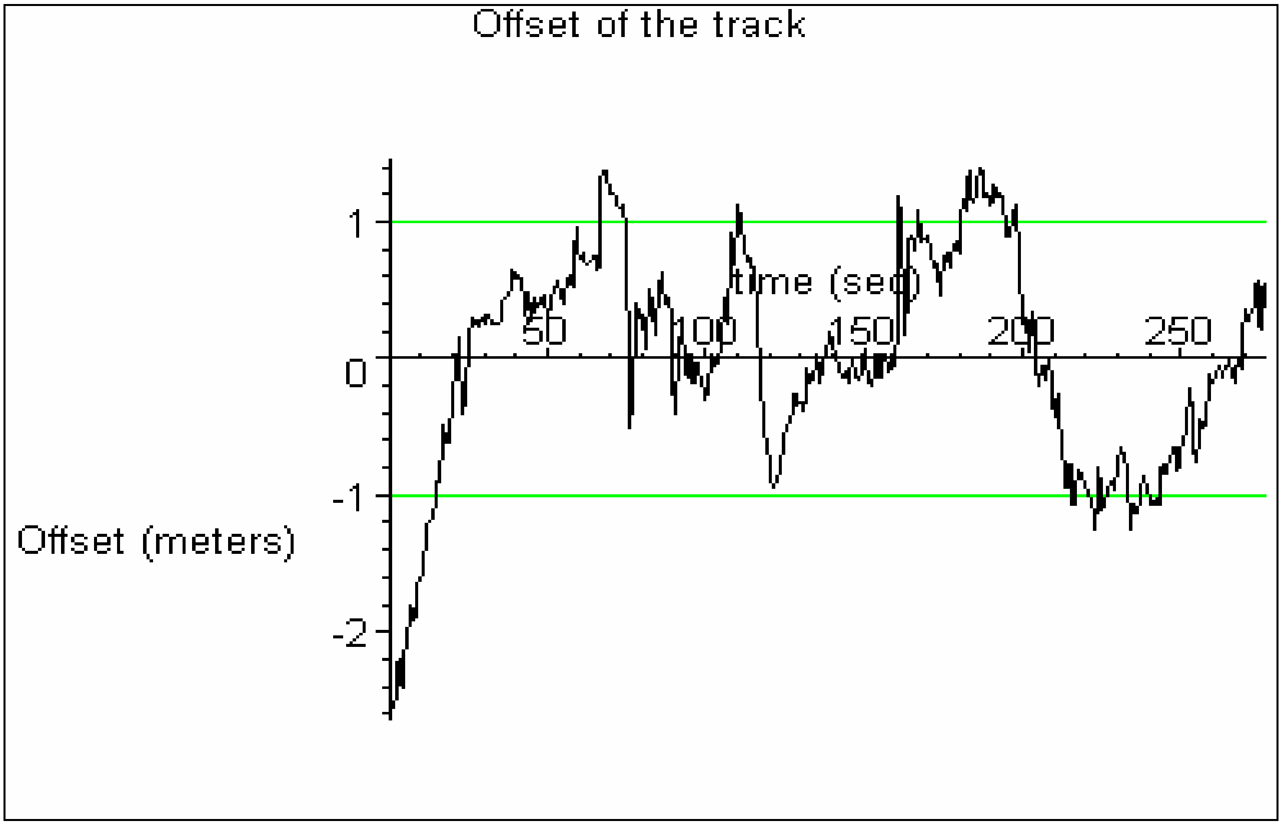


Figure 4.7: Visualization of the offset in meters as a function of sample [HAM04]

Figure 4.7 presents the offset of the track in meters for the 277 seconds sample. Taking into account the added errors through an imperfect straight line driven by the test vehicle and the conservative scale factor of 1.852 km for one arc-minute the GPS signal is almost always within an one meter distance off the track. This result can be improved with a GPS receiver using differential correction input. However to be fair and to ensure that the navigation system still works reliably under worst conditions without any correctional signals the test has been conducted with the GPS signal only.

Other important information for the navigation system that is provided by the GPS receiver is the current bearing. The analysis of the quality of the bearing information represents the second part of the results obtained from the straight line test. To decide whether to correct its course autonomously depends primarily on the bearing information. Due to the great distances between waypoints and to ensure proper decisions, the bearing information needs to be accurate.

For the bearing analysis, first the mean and the standard deviation in degrees of the sample are calculated (**compare exhibit 3**):

$$dx := 32.79386282$$
$$sx := 3.820131443$$

Second, the logged data is plotted in true degrees to visualize the course of the samples and to find outliers (**see figure 4.8**).

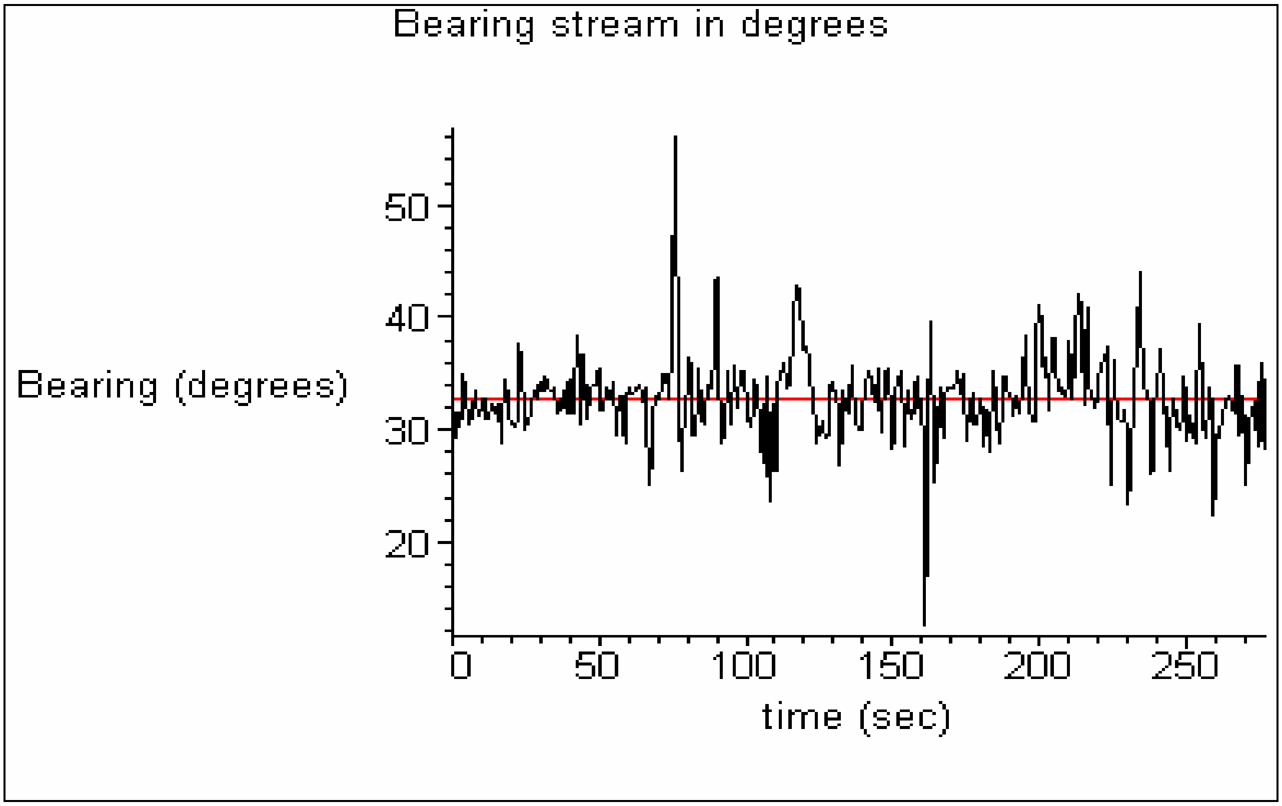


Figure 4.8: Plot of the bearing stream in black and the mean in red [HAM04]

The GPS receiver determines the bearing from point to point. This explains the great range in which the sample bearings lie. The maximum range between two outliers is almost 34° which is about the value of the actual course. This result clearly shows that the instantaneous bearing information is hardly of use for the robot to autonomously decide whether to correct is course.

However the possibility exists to take a number of consecutive samples and use their mean for the decision making process. This approach helps to reduce the impact of outliers on the bearing information.

In the following analysis fifteen samples are tested to find out whether the information gained is precise enough to base the correction decisions on it. To draw sound conclusions the analysis computes the mean based on the past 15 samples for every second of the time line. This means that for a sample of fifteen bearing signals the first mean is determined after fifteen seconds and every second after that, i.e., a "moving average". The result of the sample mean is compared to the overall mean *dx* of the sample and the difference plotted in figure 4.9.

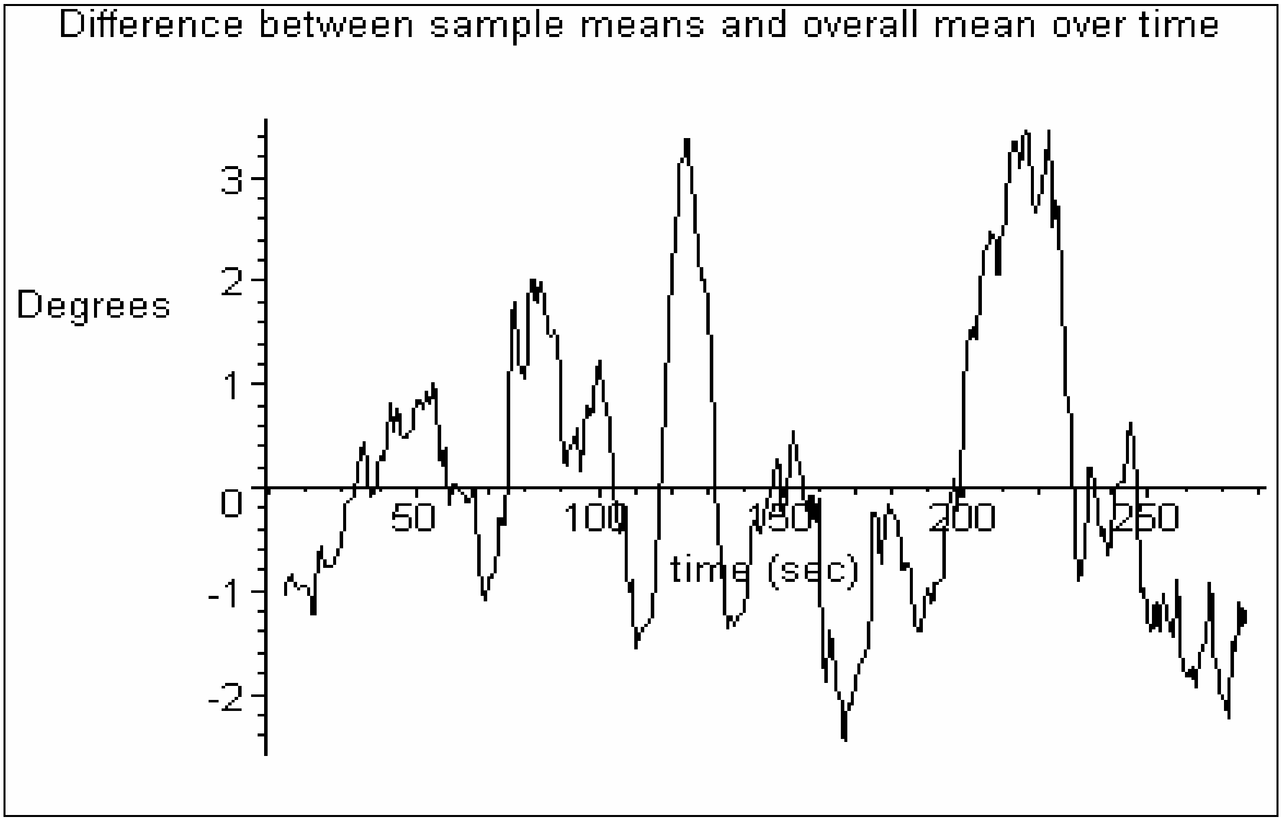


Figure 4.9: Time history of the mean based on past 15 samples [HAM04]

Figure 4.9 shows the differences between the sample means and the overall mean *dx* over time. Using fifteen samples the maximum difference to the overall mean *dx* is still 3.47°.

This result confirms that the bearing information from the GPS receiver is not feasible for the correction determination process. To increase the number of samples beyond fifteen does not immediately improve the result. It helps reduce the impact of outliers but the main problem comes from the wandering residuals of the GPS signal (**see chapter 4.4.2**). If the signals move in one direction for a longer period of time the bearing information is biased for this time and so is the average of the sample.

These test results are confirmed by further tests (**compare exhibit 5,7**) and show the need for another approach to help the robot's decision making on course corrections. This approach is described in section 4.6 and 4.7 in detail.

4.4.4 Discussion if an extra filter as the "Kalman Filter" is necessary

With the results of the stationary and the moving test of the GPS receiver the need for further sensors and filters in the navigation system can be discussed.

To improve the accuracy of a mobile navigation system different filters can be deployed. The most common filter is the "Kalman Filter". It was introduced in 1960 by Dr. R. E. Kalman. The Kalman Filter is a recursive algorithm that provides optimum estimates of the user position and velocity based on noise statistics, measurements, and system dynamics. The filter contains a dynamical model of the GPS receiver platform motion and outputs a set of user receiver position and velocity state estimates as well as associated error variances [KAP96, WEL03, STE86].

The benefit in the improvement of the accuracy by deploying a Kalman Filter in the navigation system has several side effects. A dynamical model of the robot needs to be developed and implemented in the navigation program. Due to the skid steering solution the dynamic model of the robot can be simplified but still needs to be included. To apply the filter, the dynamic model and state covariance matrices must be integrated. This requires extra calculation power of the controller and the need matrix algebra. As the controller's programming language is based on C it is not designed for matrix algebra and requires extensive extra libraries even for small matrices.

With regard to the accurate positioning information of the GPS receiver presented in chapter 4.4.2 and the overall goal to keep the navigation system simple and reliable the deployment of a filter e.g. the Kalman Filter is not feasible.

The accuracy of the GPS receiver itself is high enough to help the robot navigate autonomously in its application area. As described before the Antarctic at the South Pole does not have major obstacles which makes the need for a higher accuracy than a standard deviation of two meters less important.

However the further information provided by the GPS receiver such as bearing and speed are not as accurate. The robot needs to calculate the bearing itself. The speed information can be used as in extra input to determine whether the wheels are slipping and the robot is not moving to stop wasting energy and if necessary decide to send a rescue signal.

4.5 Explanation of the controller and converters

The controller and the DAC and ADC are the control system of the robot. The controller uses the information from the GPS receiver to navigate the robot autonomously from waypoint to waypoint. The DAC is used to drive the four motors and the ADC is used for the feedback loop of the control system. A description of the control system's layout is followed by the explanation of the implementation of the DAC and ADC as third party products in the control system.

4.5.1 Detailed description of the control system of the robot

At this point of the project the layout of the control system is not determined. One can envision three possible layouts for the control of the robot. They differ in the complexity of especially the algorithms needed for the programmable Jackrabbit and its immediate implementation in the control loop for the motors. All three solutions are presented and explained beginning with the most simple layout to the most complex solution and a mixed solution of the first two layouts for the control system.

Figure 4.10 presents the layout of the control system for the velocity control approach. This solution keeps the Jackrabbit controller out of the control loop.

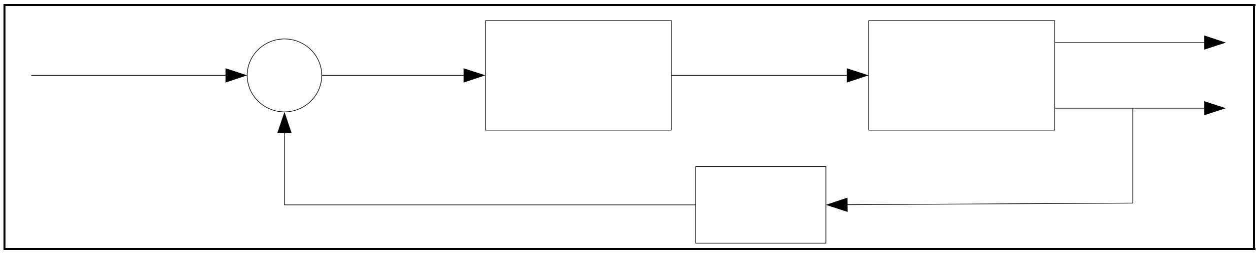


Figure 4.10: Layout for velocity control [HAM04]

The Jackrabbit sends a desired velocity command as a voltage through the DAC to the motor controller of each motor. The motor controller has a proportional integral controller inside the controller. It runs either in velocity or torque control mode. The mode is set by a jumper.

In the velocity control mode it sends a voltage command to the motor which results in an actual velocity. The feedback loop contains a gain with a velocity constant K_v. The motor controller adjusts and maintains the velocity command of the Jackrabbit controller.

In the torque control mode the motor controller sets the current to the motor, based on a desired torque command from the Jackrabbit. This current results in an actual velocity which depends on the applied torque. In the torque control mode the feedback loop consists of a torque constant K_t. Until a new command is given the motor controller adjusts and maintains the desired torque through a torque command to the motor.

In either mode the actual current draw can be logged for later evaluations of the rolling resistance of the application area.

This layout represents the most simple approach for the control system. There are no complex algorithms needed for the Jackrabbit which only sends a command for the desired torque or velocity. However, restrictions for reasonable application of this layout in either the velocity or torque control mode exist.

With the Jackrabbit out of the loop and the motor controller in velocity control mode the Jackrabbit has no immediate feedback on the actual velocity. Slipping wheels can only be sensed by comparing the desired velocity command with the actual velocity information gathered from the GPS receiver. However this information from the GPS receiver appears to be affected by a rather great error as stated in chapter 4.4.3. This could result in spinning wheels that are not detected for a extended period of time and evoke loss of energy. It is possible that slip can be detected by comparing the four wheel speeds to each other.

If the motor controller is set on torque control mode, it transforms the desired torque command in a current command to the motor. The motor controller adjust the torque through the feedback loop.

This solution helps to control slipping wheels and to maximize traction. If the wheel is slipping, the current draw drops and the motor controller changes the torque command. However, the relation between torque command and wheel speed must be determined through the motor model and the resulting speed is strongly depending on the environmental conditions. In steady state with the robot travelling at a steady speed, the relation between the applied torque and the resulting velocity can be measured. If the rolling resistance changes for different reasons the controller will maintain the torque and the robot's velocity will change. Furthermore problems appear if the robot accelerates. The Jackrabbit sends a command for a desired velocity and the motor controller applies the relating torque to the motors. But if the torque is not sufficient to exceed the rolling resistance the robot does not move and wastes energy.

The described restrictions call for a second approach for the layout of the robot's control system which includes the Jackrabbit in the control loop (**see figure 4.11**). Here, the Jackrabbit implements control logic to maximize traction, based on wheel velocities and slip estimates.

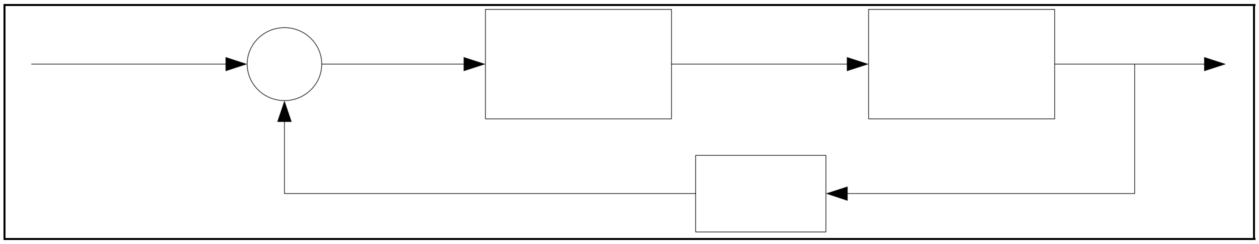


Figure 4.11: Layout for torque control [HAM04]

The desired torque is processed through the Jackrabbit to a voltage command to the motor controller and motor which leads to an actual current.

In the second control layout the motor controller runs in open loop mode or a velocity control mode. The Jackrabbit controller has a proportional integral controller to read the feedback through its ADC. The feedback is transferred through a gain with a torque constant K_t which is directly related to the current draw. This layout requires complex control algorithms implemented on the Jackrabbit to realize the ability to adjust and maintain a desired torque for each of the four motors. This solution helps the robot's control system to detect spinning wheels and gives the Jackrabbit the opportunity to react to different torques on each wheel. Being a skid steered robot different torques lead to a turning moment and change the robot's direction. In this layout, the Jackrabbit can adjust the torque of each wheel immediately if the torque of one wheel changes significantly.

However this solution requires extensive testing on and knowledge of the application area about the relation between torque and resulting velocity.

Because of these complications the third solution provides a mixture of the layouts presented above (**see figure 4.12**).

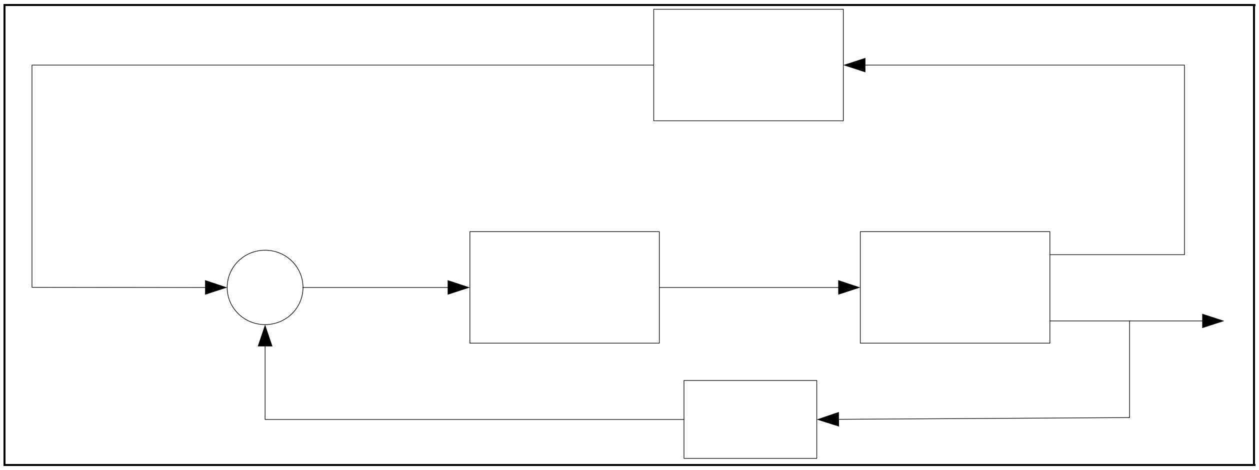


Figure 4.12: Layout for mixed control [HAM04]

The idea of the layout for mixed control is that the system works in a velocity mode like the first layout most of the time. The Jackrabbit controller sends a desired velocity as a voltage command through its DAC to the motor controller which operates in closed loop and velocity control. The motor controller drives the motor and reads the feedback of the actual velocity through a gain with the velocity constant K_v. At the same time the Jackrabbit controller monitors the current draw through its ADC. In case of slipping wheels the Jackrabbit senses the drop of the current and adjusts the desired velocity command accordingly through logic. This change is based on the information presented in figure 4.13.

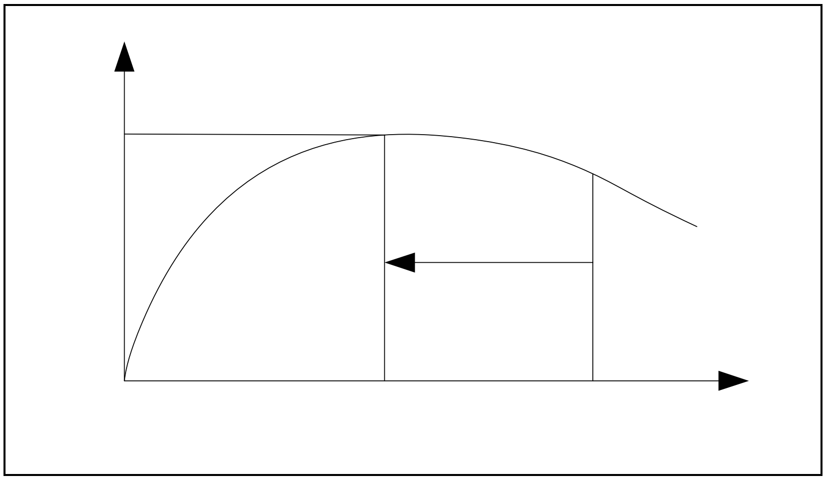


Figure 4.13: Relation between torque and slip [RAY04]

Figure 4.13 shows the relation between torque F_t and slip. The torque increases until the wheel starts to slip. At this turning point the slip increases but the torque decreases. The logic of the Jackrabbit needs to take advantage of this effect. With the wheel spinning in point 1 it needs to send a decreased command for the desired velocity. The wheel eventually stops spinning in point 2 and the maximum torque can be applied. The exact shape of the curve is not known in any case but the effect is the same.

Ft

The third layout solution presents a mixture of the first and the second solution and is less complex than the second solution. However in order to keep the system simple and failure-free tests should start with the first solution. Only if major problems regarding undetected slip of single wheels occur the third layout is the next step up. The third solution demanding complex algorithms is the last choice for the control system. More complex algorithms present a greater risk for system failure.

To realize the control loops above the controller needs DAC and ADC for information exchange with the motor controller and motor. The Jackrabbit's six programmable serial ports are the main interfaces of the system (**see figure 4.14**).

Connection	Pin Name	Connection	Components	I/O
planned		required		
Serial Port A	TXA, RXA		Programming port	
Serial Port B	TXB, RXB	RS 232	Iridium Modem	
Serial Port C	TXC, RXC	RS 232	GPS	
Serial Port D	PC0, PC1	2 wire serial interface	Four Channel D/A	Motor Controller 1
				Motor Controller 2
				Motor Controller 3
				Motor Controller 4
Serial Port E	PG0, PG1	2 wire serial interface	Eight Channel A/D	Current Monitor 1
				Current Monitor 2
				Current Monitor 3
				Current Monitor 4
				Inclinometer Pitch
				Inclinometer Roll
				Laser Range Finder
				free
Serial Port F	PG6, PG7	free		

Figure 4.14: Spreadsheet of controller connections [HAM04]

The serial port A is the programming port. It is also available for further devices. Serial port B and serial port C have a RS 232 interface.

The RS 232 interface standard was set by the Electronic Industries Association for data communications equipment. It specifies signal voltages, signal timing, signal function, a protocol for information exchange and mechanical connectors for reliable communication. The Jackrabbit controller provides two serial ports with this standard that is required by the iridium modem and the GPS receiver as long as the GPS receiver is mounted on the evaluation board.

The DAC is connected to the serial port D and the ADC to the serial port E through a Serial Peripheral Interface (SPI) connection. SPI was created by Motorola and uses a shift register to send and receive data. The four output channels of the DAC drive the four motors. Four of the eight input channels of the ADC are used for the current monitor of the motor. Two channels are reserved for the inclinometer input and one for the laser range finder. The eighth channels is free at this time. The serial port F is not in use.

Although it is planned to keep the deployed instruments independent of the robot’s control and navigation system, at this time there are two serial ports not in use. These serial ports can be used e.g., for an external data storage flash card or to read from an instrument quality magnetometer. The necessity for an external data card depends on the final size of the overall program and if the leftover space from the 512 kb internal storage is not sufficient to store all collected information.

However an external serial flash card from Z-World with 16 MB exists and can be connected to either one of the serial ports.

4.5.2 Presentation of the program to run the DAC MAX536

The MAX536 combines four 12-bit, voltage-output DACs and four precision output amplifiers in a 16-pin package. The DAC is operated at -5 V V_{SS} and +10 V V_{DD} supplies. It uses a SPI connection (**see chapter 4.5.1**) for the information exchange with the controller and is connected to serial port D on the Jackrabbit controller (**see figure 4.15**).

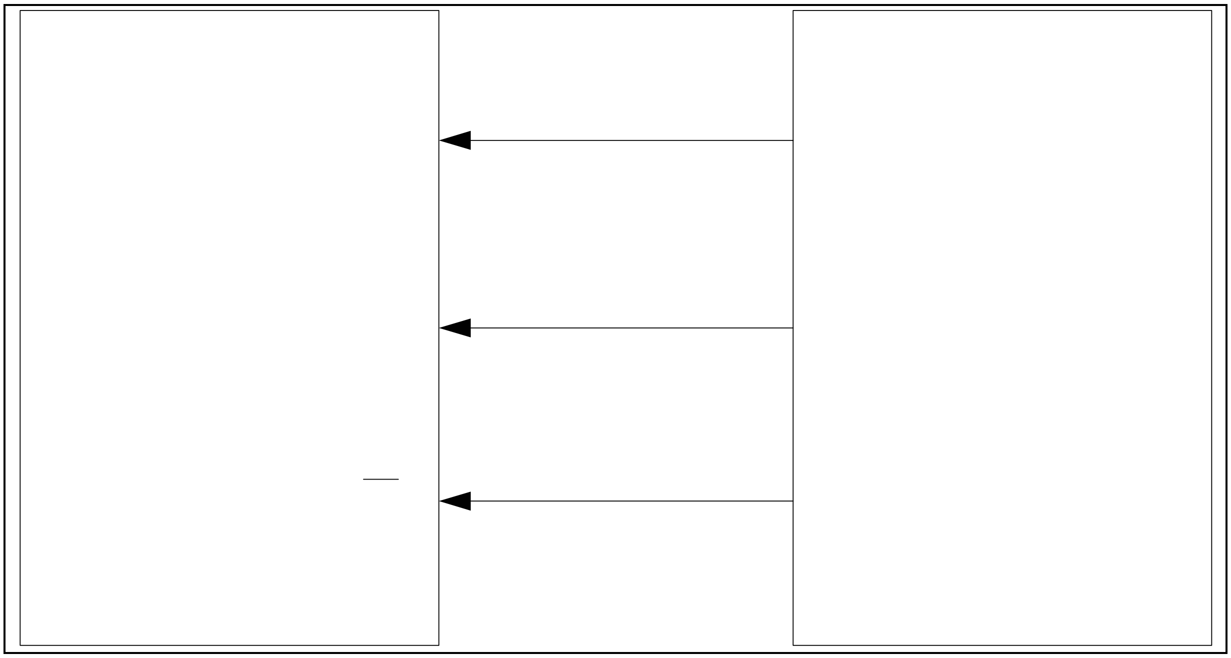


Figure 4.15: SPI connection for DAC and Jackrabbit controller [HAM04]

The serial data input *SDI* is connected to the master serial data output *MOSI* of the Jackrabbit controller. The *SCK* input of the DAC is connected to the respective *SCK* of serial port D and the chip select line *CS* to an *I/O* line of the controller. The serial data output *SDO* of the DAC is not necessary for writing purposes and not connected. A 16 bit serial word is used to load data into each register. The first four bits are control bits and the last 12 bits set the output value. The four control bits select the output channel. In the *SPI_DAC_MAX 536.c* program the output channel selection is set for an immediate update of the output value (**see exhibit 8**):

```
(15)    #define DACA 0x3000          // Definition of the output channels
        #define DACB 0x7000
        #define DACC 0xB000
        #define DACD 0xF000
```

The following 12 bits set the output value of the selected channel in relation to the reference voltage. At the current setting the reference voltages *REFAB* and *REFCD* are tied to the positive power supply V_{DD} of +10 V. The analog and digital ground pins *AGND* and *DGND* are tied with the power supply and the Jackrabbit controller for a common ground which is essential for the functionality of the DAC.

The MAX536 DAC requires the most significant bit first of the serial word. Dynamic C sends the least significant bit of an integer first. This requires the function *SwapBytes* before sending the 16 bit serial word. To activate the DAC the chip select line *CS* needs to go low to send the data. The register and the output value are updated as soon as the *CS* line goes high (**compare exhibit 8**):

```
(26 i.) i=820; // actual value in proportion to the RefVoltage (i/4096)*10 V REF
        j = SwapBytes(i|DACB);      // update DACB immediately with value i
        BitWrPortI ( PBDR, &PBDRShadow, 0, 2 );        // CS = 0
        SPIWrite( &j, 2 );
        BitWrPortI ( PBDR, &PBDRShadow, 1, 2 );        // CS = 1
```

The program uses the function *SPIWrite* that is included in the library *SPI.lib*. The library is a part of the software provided with the Jackrabbit controller.

Each output channel is connected to the dedicated motor controller's input reference voltage to drive the four motors independently. The provided program *SPI_DAC_MAX 536.c* will be integrated in the overall program of the robot's control and navigation system.

4.5.3 Explanation of the program to run the ADC MAX186

The MAX186 is a 12-bit data acquisition system with an 8-channel multiplexer. The ADC has a SPI compatible serial interface and operates at 0 to +5 V V_{DD} that is provided by the development board of the Jackrabbit controller. It has a built in reference voltage *VREF* of 4.096 V. All connections between the Jackrabbit controller and the ADC are specified in the *SPI_ADC_MAX 186.c* file (**see exhibit 9**). The ADC is currently connected to the serial port B of the Jackrabbit controller.

The MAX186 requires three 8-bit transfers to perform a conversion. One 8-bit transfer to configure the ADC and two more 8-bit transfers to clock out the 12-bit conversion result. The 8-bit control transfer starts the ADC and selects the channel. Currently the ADC is set to unipolar mode which makes it possible to convert an analog input signal from 0 V to *VREF*. Besides the ADC is set to single-ended conversions to refer the input signal voltage to analog ground *AGND* and it uses the internal clock All settings are defined and combined in one char for each channel *READ_CH0..7* using exclusive ors for the first byte and leaving the last two bytes for the reading from the ADC open (**compare exhibit 9 lines 40 to 47**):

```
(40)   const char READ_CH0[]
       ={MAX186_START|MAX186_UNIPOL|MAX186_CH0|MAX186_SINGLE|MAX186_MOD
       E_ICLK, 0, 0 };
```

The main program sets up the chip select port, defines the *ScaleFactor* and performs the *ReadAD* function. The *ReadAD* function commands the ADC to take readings from the selected channel. It takes the average of the specified number of readings and converts the value to volts using the predefined scale factor. It requires the address of the command bytes *READ_CH0..7* and an integer number of readings to average as an input. The function returns the voltage using the globally defined *ScaleFactor*:

```
(72)   float ReadAD ( char *Command, int Samples )
```

The function takes the readings by enabling the ADC through the chip select line *CS*. The actual writing and reading is performed by the *SPIWrRd* function that is included in the *SPI.lib* file. The *ReadAD* accumulates the results of the readings and finally multiplies the accumulated result with the *ScaleFactor* and divides it by the number of readings *Samples*. The result is the average voltage of the readings and is returned to the main program:

```
(78)   i = 0L;                                              // init accumulator
       for ( Count = 1; Count<=Samples; Count++ )
       //      for ( ;; )
```

```
{
BitWrPortI ( SPI_MASTER_CS_PORT, &SPI_MASTER_CS_SHADOW, 0,
SPI_MASTER_CS_BIT );                                    // enable /CS
SPIWrRd ( Command, &data, 3 );
BitWrPortI ( SPI_MASTER_CS_PORT, &SPI_MASTER_CS_SHADOW, 1,
SPI_MASTER_CS_BIT );                                     // disable /CS
j = data[1]*256 + data[2];
i += (j>>3);                                            // update accumulator
for (j=0; j<100; j++);                                  // cheap time delay
}
Voltage = (float)i * ScaleFactor / (float)Samples;
}
```

The first four channels 0..3 are reserved for the current monitor of each motor controller. Channels 4,5 are reserved for the inclinometers and channel 6 for the laser range finder. The eighth channel is free at this stage of the project. The provided program *SPI_ADC_MAX 186.c* will be integrated in the overall program of the robot's control and navigation system.

4.5.4 Presentation of the test configuration of the control components

At this stage of the project the Jackrabbit controller and its development board are mounted on a board with the ADC and DAC. Due to the separate testing the GPS receiver and its evaluation board are separate and not permanently connected to the Jackrabbit controller (**see figure 4.16**).

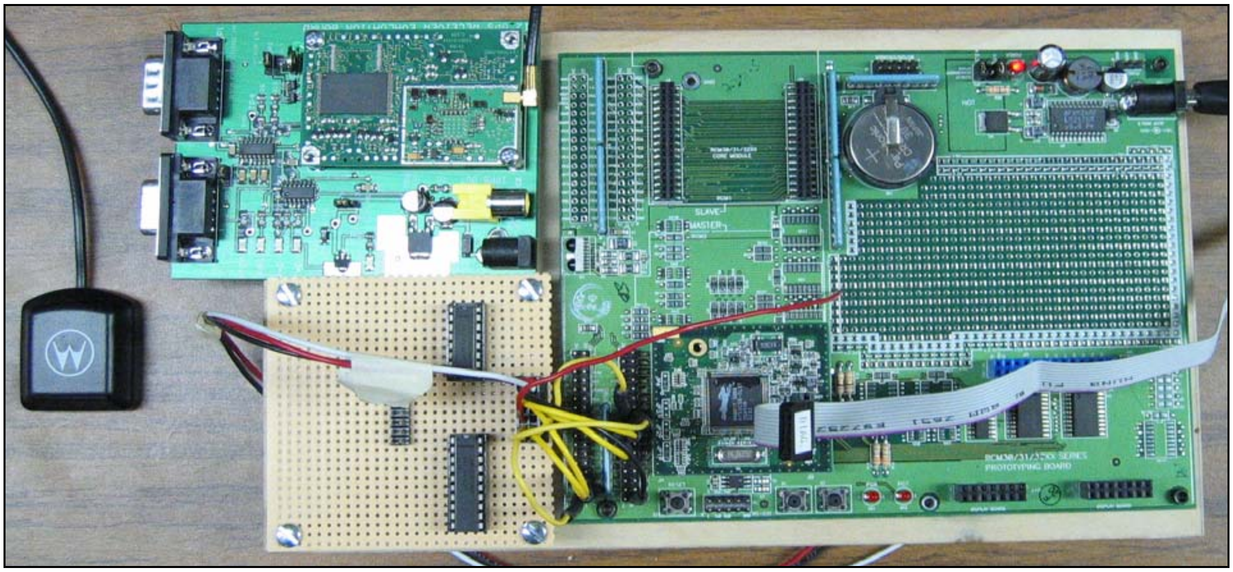


Figure 4.16: Picture of all components [HAM04]

Figure 4.16 shows the Jackrabbit controller and its development board on the right. The GPS receiver is on the top left with the GPS antenna next to it. The 20-pin ADC is the lower chip on the board on the bottom left. The DAC is mounted on the same board above the ADC. The 14-pin-connectors on the left and on the right of the ADC and DAC are for the connections to the Jackrabbit controller and for the input or output values of the converters.

The grey ribbon cable connected to the Jackrabbit controller is the programming cable that goes to the serial port of a computer or laptop. The black connector on the top right is the power supply for the Jackrabbit controller.

Red coloured wires are for positive supply voltages, white coloured wires for negative supply voltages, black coloured wires for ground and yellow coloured wires for signals.

The pin assignment for the pins next to the converters is as described in figure 4.17:

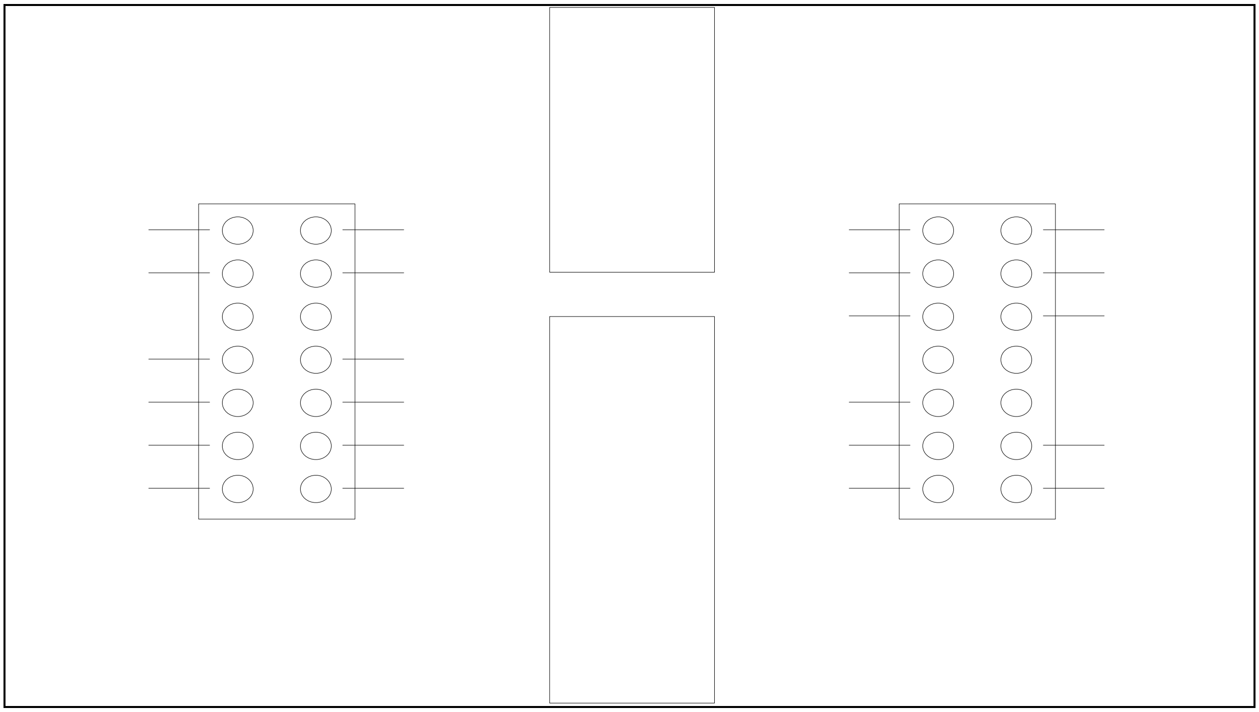
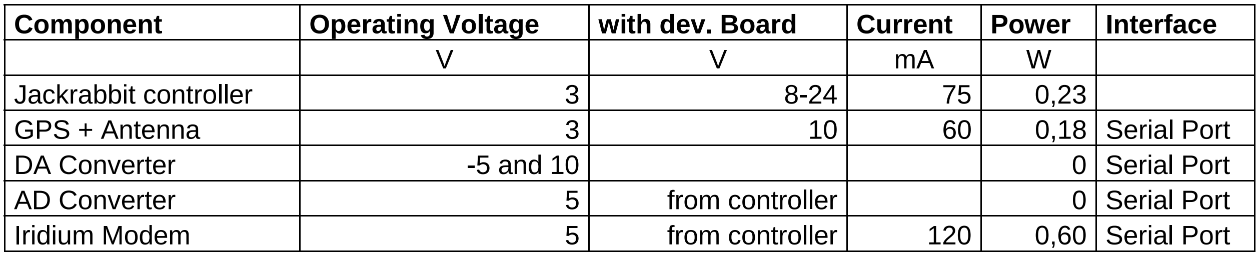


Figure 4.17: Pin assignment on test board [HAM04]

The pins are wired to the respective pins on the ADC and DAC. The ground pin needs to be connected to the ground of the Jackrabbit controller and the power supply to obtain a common ground for the whole system. The DAC positive power supply V_{DD} requires +10 V and the negative power supply V_{SS} requires -5 V. The positive power supply V_{DD} for the ADC is connected to the +5 V output of the Jackrabbit controller. The serial port input and output pins and the serial port clock pins are currently connected to the serial port D for the DAC and to serial port B for the ADC. The *CS* line for the DAC is *PB 2* and for the ADC *PB 7* on the Jackrabbit controller.

The four output pins of the DAC and the eight input pins of the ADC are ready to be connected to the respective devices for further testing.

To mount the components on a moving robot the following direct current (DC) voltages need to be supplied:



Component	Operating Voltage	with dev. Board	Current	Power	Interface
	V	V	mA	W	
Jackrabbit controller	3	8-24	75	0,23	
GPS + Antenna	3	10	60	0,18	Serial Port
DA Converter	-5 and 10			0	Serial Port
AD Converter	5	from controller		0	Serial Port
Iridium Modem	5	from controller	120	0,60	Serial Port

Figure 4.18: Specification of necessary DC power supplies [HAM04]

The ADC and the iridium modem can be powered by the development board of the Jackrabbit controller which provides +3.3 V and +5 V at xx A. The GPS receiver and the Jackrabbit controller with their development boards both run with +10 V as does the DAC. This results in only three DC voltages -5 V, +5 V and +10 V needed to run the whole navigation and control system on a moving platform for testing purposes or later on the robot.

4.6 Presentation of the overall navigation program

The robot's layout of the overall navigation program is based on the test results of the GPS receiver from section 4.4. It takes into account that the robot has no immediate information for its heading. This asks for the robot to correct its heading on the move. The robot's basic strategy is to travel from waypoint to waypoint. One can argue that the robot needs to travel on a sun-synchronous way being a solar powered vehicle. But as described in the abstract of the power group in chapter 5 and in the description of the mechanical design in chapter 3 the robot has solar panels on each side, the top and possibly the bottom. The sides have almost the same size. As shown in chapter 5 the reflection of the sun from the snow adds extra energy to the power system although the solar panels are not facing the sun directly. All the energy from the reflected light adds up to almost twice the energy that can be absorbed by the solar panel facing the sun. Furthermore the sunlight is available 24 hours a day for the whole summer. Tests for "Solar Navigational Planning for Robotic Explorers" have been conducted by the NASA and the Carnegie Mellon University with the Nomad and the Hyperion robot. These tests for sun-synchronous and sun-seeking paths on different application areas show that on the Antarctic field are no shadows as it is flat [WHI01,WET02]. These results confirm that a robot with no primary solar panel side is not depending on solar navigational planning and is able to go straight on its track all day long.

However the selection of the waypoints needs planning to ensure a safe trip of the robot. As the robot is only equipped with close range detection for obstacle avoidance the waypoints need to guide the robot around known dangerous areas. But operating in an application area with a radius of 500 km around the South Pole there are no major obstacles of danger for the robot [LEV04].

The robot travels from waypoint to waypoint by selecting the following waypoint as soon as it gets within a defined range of the current waypoint. The overall program is set up for the robot travelling a circle with the last waypoint having the same position as the starting point. However this can be flexible but the idea is that the robot leaves its base for the summer and returns by the end of the summer to the same station at the South Pole.

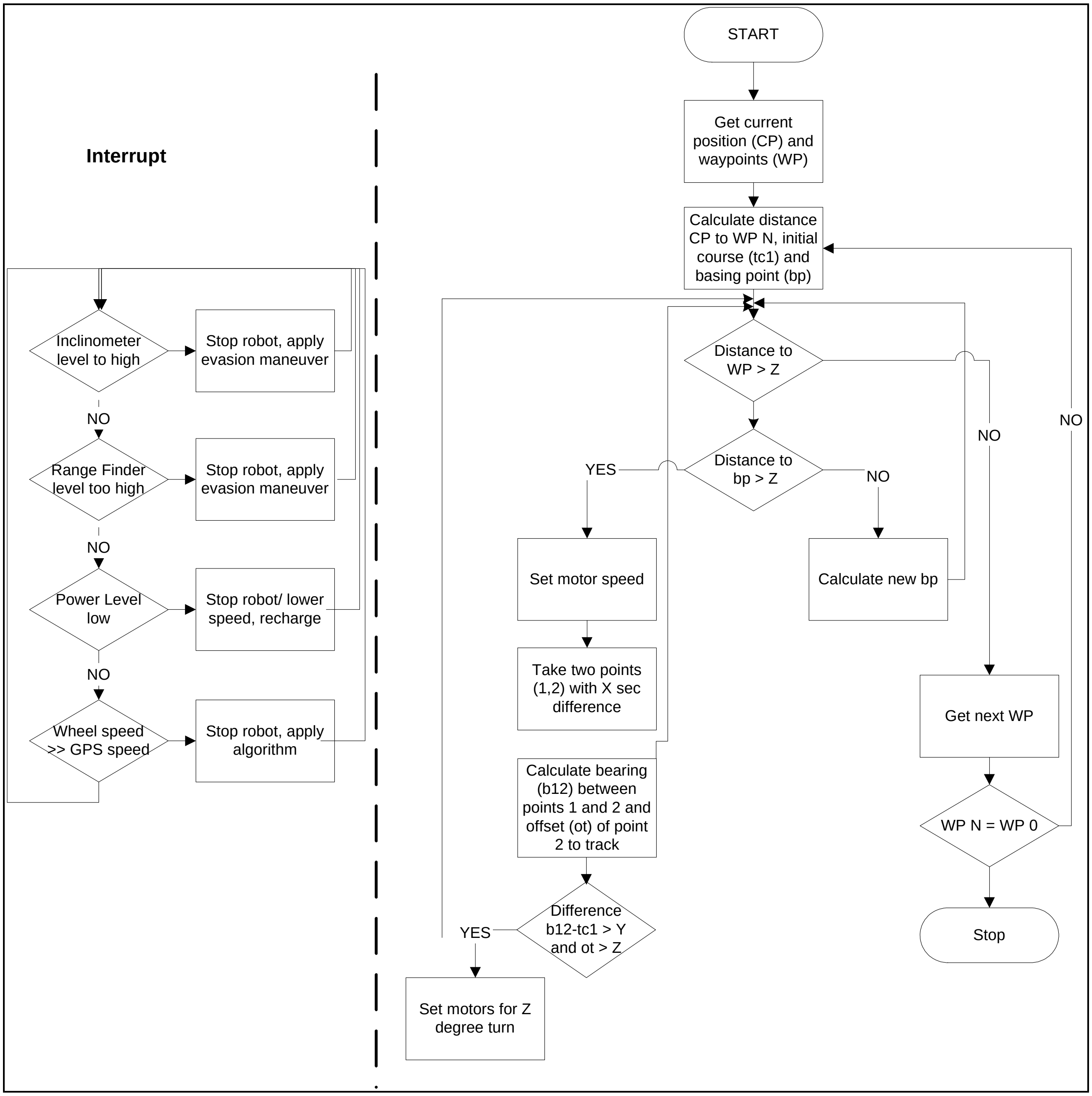

START
Get current position (CP) and waypoints (WP)
Calculate distance CP to WP N, initial course (tc1) and basing point (bp)
Interrupt
Inclinometer level to high
Stop robot, apply evasion maneuver
NO
Range Finder level too high
Stop robot, apply evasion maneuver
NO
Power Level low
Stop robot/ lower speed, recharge
NO
Wheel speed >> GPS speed
Stop robot, apply algorithm
Distance to WP > Z
NO
NO
Distance to bp > Z
YES
NO
Set motor speed
Calculate new bp
Take two points (1,2) with X sec difference
Get next WP
Calculate bearing (b12) between points 1 and 2 and offset (ot) of point 2 to track
WP N = WP 0
Difference b12-tc1 > Y and ot > Z
YES
Stop
Set motors for Z degree turn

Figure 4.19: Flowchart of overall navigation program [HAM04]

Figure 4.19 presents the flowchart of the overall program. It can be divided in two major parts. One being the navigation algorithm and one the interrupts that monitor crucial input signals permanently to prevent the robot from damage.

The navigation program starts with the collection of data. The robot receives its current position (CP) information through the GPS receiver and the waypoints (WP) either from its memory or through the iridium modem. Subsequently it calculates the distance between the CP and the first WP, followed by the determination of the initial course (tc1). Taking into consideration that the distance between two waypoints can be up to 500 km the correction on the move would be impossible if the robot uses the waypoints only. Offsets from the track that require a course correction would be hardly measurable in a triangle with the axes being 500 km long. In order to enable the ability to correct on the move the robot needs to work with base points (bp) that lie on the track. These base points are calculated in distances from each other of e.g. 0.25 km. The robot can either calculate and store all base points immediately or, better, only store the next base point. After checking, if the distance to the next WP and to the next bp is bigger than Z it sets the motor speed.

The results of the tests with the GPS receiver in chapter 4.4 show that the position information is accurate but the bearing information is not precise and has extensive outliers. This precludes the obvious approach to include the bearing information of the GPS receiver into the decision process. The robot needs to take two sample positions 1 and 2 within X seconds and calculate the bearing (b12) between these two points using the same algorithm as used for the calculation of the initial course tc1. Also, the robot calculates the offset of the track (ot) at the position of point 2. Only if both the difference between b12 and tc1 and the offset ot are two big the robot turns. If only the course b12 is off and the robot is still close the track, no correction is necessary. On the other hand if the offset ot is too big but the current course b12 is close or the same as the initial course tc1 the robot travels on a parallel line to the initial track which causes no problems on the overall goal to reach the waypoint. If both the b12 and ot are off the limit the robot needs to set the motors for a Z degree turn depending on the value. These settings need to be evaluated empirically first on a similar ground an later on the South Pole area itself. If only one or no information is off, the robot checks the distance to the base point and the waypoint again and continues its path. The distance to the waypoint is checked too to ensure that the robot does not accidently pass a waypoint on its way to the following base point.

As soon as the robot reaches the base point it calculates or takes a new base point and checks the distance to the waypoint. This process continues until the robot reaches the waypoint. In this case, it either takes the following stored waypoint or receives the next waypoint through the iridium modem. If the following waypoint is not the same as the starting point it will continue travelling until it reaches the last waypoint and the program stops. On its way the robot can be stopped through time delays at any time. The algorithm does not have to be performed every tenth of a second with the robot travelling at around 3 mph. A good start is to check the position and the necessity to correct the robot every thirty meters or at 3 mph every thirty seconds.

At any travel time, the interrupts permanently monitor different sensors. In case that one of the input signals exceeds a security level, the interrupt forces the overall program to stop and perform the respective algorithm. If an inclinometer exceeds the security level and the robot is about to tip over the robot stops and applies a predefined evasion maneuver. If the range finder level is too high the robot is too close to some obstacle and it performs another evasion maneuver. Another critical level is the power level. If for different reasons the power level drops the robot either has to lower its speed or even stop to recharge its batteries. The last interrupt necessary to prevent the robot from damage is the control of the wheel speed in comparison to the GPS speed. If both information diverge significantly, the wheels are slipping and the robot is stuck wasting energy while driving its motors. At this point the robot needs to first perform a rescue algorithm and if not successful send its current position via iridium modem for a rescue pickup.

At this point of the project after having analyzed all tests with the components the design of the overall navigation program is finalized. The functions needed to perform the navigation program are presented in the following chapter.

4.7 Description of functions for the overall navigation program

In order to successfully perform the overall navigation program presented in chapter 4.6 the following functions are necessary. The functions will be implemented in the program and have been tested individually for their functionality in the *GPS.c* file. They are arranged as they occur in the overall program:

- Function to receive data from GPS receiver through a serial port
- Function to calculate the distance between two points
- Function to calculate the course between two points
- Function to determine and store base point information
- Function to set the motor speed with the DAC is presented in chapter 4.5.2
- Function to read from the motor controller’s current monitors with the ADC is presented in chapter 4.5.3

- Function for variable time delays
- Algorithm to calculate the offset from track with available functions

Some functions come with the Jackrabbit controller as samples. But the samples either needed to be revised or new additional functions for this particular problem needed to be developed and programmed. The functions are presented with a description of their purpose first. Subsequently the most important of the source code is cited and the whole source code presented in appendix II.2.

4.7.1 Function to receive data from GPS receiver through a serial port

The function *gps_get_position* to receive data from the GPS receiver through a serial port is mostly adopted from the sample program included in the software with the Jackrabbit controller. The function is implemented in the library *GPS.lib.* The sample function is able to read different NMEA 0183 protocol strings.

The National Marine Electronics Association (NMEA) is a non-profit association of manufacturers, distributors, dealers, educational institutions, and others interested in peripheral marine electronics occupations. The NMEA 0183 standard defines an electrical interface and data protocol for communications between marine instrumentation. The NMEA 0183 is a voluntary industry standard that was first released in 1983 and has been updated with the latest version from August 2001. NMEA 0183 devices are designated as either *talkers* or *listeners* (with some devices being both), employing an asynchronous serial interface with the following parameters:

Baud rate: 4800
Number of data bits: 8 (bit 7 is 0)
Stop bits: 1 (or more)
Parity: none
Handshake: none

All data is transmitted in the form of *sentences*. Only printable ASCII characters are allowed, plus carriage return (CR) and line feed (LF). Each sentence starts with a *$* sign and ends with <CR><LF>. There are three basic kinds of sentences: *talker sentences*, *proprietary sentences* and *query sentences*. **Talker Sentences.** The general format for a talker sentence is:

$ttsss,d1,d2,....<CR><LF>

The first two letters following the *$* are the *talker identifier*. The next three characters (sss) are the *sentence identifier*, followed by a number of *data fields* separated by commas, followed by an optional *checksum*, and terminated by carriage return/line feed. The data fields are uniquely defined for each sentence type. An example talker sentence is (taken from the steady state test on November 26 in 2003):

*$GPRMC 185026.00 A 4342.2563 N 07217.6692 W 0.3 153.4 261103 *25*

where *GP* specifies the talker as being a GPS, the *RMC* specifies the "Recommended Minimum Navigation Information" message follows. The *185026.00* is the time, *A* the status. A *V* signifies a GPS receiver warning, that the signal is not accurate. *4342.2563* is the latitude *N* or *S* and is followed by the longitude *07217.6692* on either *W* or *E*. Subsequently the speed over ground in knots *0.3* is sent and the track made good in true degrees *153.4*. The last information is the date *261103* and the checksum in this case **25*. The RMC sentence can hold additional information of the magnetic variation in degrees *W* or *E*.

A sentence may contain up to 80 characters plus *$* and CR/LF. If the data for a field is not available, the field is omitted, but the delimiting commas are still sent, with no space between them. The checksum field consists of a * and two hex digits representing the exclusive OR of all characters between, but not including, the *$* and * [BET00].

The NMEA sentence with the most information for the navigation system is the quoted RMC sentence. Besides the sample function *gps_get_position* is able to read the Global Positioning System Fix Data (GGA) and the Geographic Position – Latitude/Longitude (GLL). All tests have been performed with the RMC sentence. However after having confirmed that the bearing information is not precise, the GGA sentence is the better choice. It contains the position information in longitude and latitude. Furthermore it sends a GPS quality indicator (0 – fix not available, 1 – GPS fix, 2 – Differential GPS fix), the number of satellites in view (00-12), further information about the height of the receiver and the differential reference station ID. These extra information might be interesting in regard of using the robot as instrument itself but for the navigation program only the longitude and latitude information is of vital importance.

For the presentation of the function the part for the RMC sentence is used. All other sentences are read in an analog way. The function requires a GPSPosition structure to fill **newpos* and a string containing a line of GPS data in NMEA-0183 format **sentence* and returns either a *-1* if the sentence is unknown or a *0* if the sentence is written in the structure (**compare exhibit 10**):

```
(47)    int gps_get_position(GPSPosition *newpos, char *sentence);
```

The function first determines the sentence through the information of the first letters of the sentence:

```
(55)    else if(strncmp(sentence, "$GPRMC", 6) == 0)
```

The GPS receiver needs to be set up to only send one of the NMEA sentences through the Motorola software WinOncore. Subsequently the function starts storing the parts of the sentence in the following structure for the GPS information:

```
(21)    typedef struct {
                int lat_degrees;
                int lon_degrees;
                float lat_minutes;
                float lon_minutes;
                char lat_direction;
                char lon_direction;
                 char sog;                    //speed over ground
                 char tog;             //track over ground
                } GPSPosition;
```

The function and the structure have been extended to additionally read and store the speed over ground information *sog* and the track over ground *tog*:

```
(72)    if(i == 6)                      //speed over ground, knots
             {
             newpos->sog = *sentence;
             }
(73)    if(i == 7)                      //track over ground, degrees true
             {
             newpos->tog = *sentence;
             }
```

The function uses the *gps_parse_coordinate* function to split the longitude and latitude information in degrees and minutes. This function is part of the *GPS.lib* library.

4.7.2 Function to calculate the distance between two points

The second function that occurs in the overall program is the *gps_ground_distance* function. This function and the following functions are based on spheroid geometry to describe the Earth's shape.

Two general approaches can be used to calculate distances, bearings, base points, etc. One approach uses a table of the lengths in meters of one minute the latitude and longitude at the respective degree of the latitude. As stated in the description of the test results of the GPS receiver the length for one minute for the latitude is almost always constant anywhere on Earth. However the length for one minute of the longitude varies from 0 meters at the poles to 1855 meters at the equator. This causes difficulties with the accuracy of the algorithms. The geometry used is more simple but over long distances the change in the length for one minute especially of the longitude is significant. Taking the application area of the robot at the South Pole and assuming an operating radius of 500 km covers almost the whole area of the latitude from 90° to 85° South. For this area the length of one minute of the longitude varies from 0 meters at 90° South to 162,24 meters at 85° South. This problems displays the difficulty in precision of this approach and calls for a more accurate and general approach [WEB09].

The second approach uses a spheroid model of the earth. This simplification is entirely adequate for most purposes to model the earth. However, in reality, the earth's mean sea-level surface is better approximated by a different geometric shape, an oblate spheroid. The surface of an oblate spheroid is created by rotating an ellipse about its polar axis. Compared to a sphere, an oblate spheroid is flattened at the poles. The earth's flattening is quite small, about 1 part in 300, and navigation errors induced by assuming the earth is spherical, for the most part, do not exceed this. As the robot operates in the immediate range of the South Pole the error can be minimized by taking the earth's radius at the poles of 6356 km in comparison to the radius at the equator of 6378 km [WIL02]. This approach is used by the *gps_ground_distance* function that is included in the sample library from Z-World and for all following functions. Further explanations about the origin of the formulas is available in Ed Williams publication "Navigation on the spheroidal earth".

The *gps_ground_distance* function requires two GPS points *a* and *b*, a defined earth radius *GPS_EARTH_RADIUS* and returns the distance between the points in km (**compare exhibit 11**):

```
#define GPS_EARTH_RADIUS 6356  // in km
float gps_ground_distance(GPSPosition *a, GPSPosition *b);
```

The function merges the degree and minute information of both points and transforms the minute information in a decimal number. To separate *N* and *S*, *E* and *W*, *N* and *E* are defined as positive whereas *S* and *W* are negative. Finally the longitude and latitude are changed from degree measure to radian measure:

```
lat_a = a->lat_degrees + a->lat_minutes/60;
if(a->lat_direction == 'S')
        lat_a = -lat_a;
lat_a = lat_a * PI / 180;
lon_a = a->lon_degrees + a->lon_minutes/60;
if(a->lon_direction == 'W')
        lon_a = -lon_a;
lon_a = lon_a * PI / 180;
```

After having transformed both points to the required input characteristic of the formula the calculation of the ground distance is performed and the distance returned in km:

```
angle = sqrt(cos(lat_a)*cos(lat_b)*pow(sin((lon_a - lon_b)/2), 2) +
        pow(sin((lat_a - lat_b)/2), 2));
angle = 2*asin(angle);

return angle * GPS_EARTH_RADIUS;
```

The calculated ground distance is necessary for the following calculation of the bearing.

4.7.3 Function to calculate the course between two points

The function to calculate the course between two points was written and tested as part of this project for the robot's navigation system. The function *gps_bearing* is implemented in the *GPS.lib* library and requires the same two points *a* and *b* as before plus the additional input of the calculated distance *dist* in km. It returns the necessary course to get from point *a* to point *b* in true degrees (**compare exhibit 12**):

```
(2)     float gps_bearing(GPSPosition *c, GPSPosition *d, float dist);
```

The *gps_bearing* function performs the same transformation of the longitude and latitude information as shown in chapter 4.7.2 for the *gps_ground_distance* function. The function takes into consideration that the immediate course from the South Pole or the North Pole as a starting point is either 360° or 180°:

```
(30)    if (cos(lat_c) < 0.0001)            // Small number
                if (lat_c > 0)
                bearing = 180;              // Starting from N pole
                else bearing = 360;   // Starting from S pole
```

If the starting point is not one of the poles or very close to it the function applies the formula for the bearing calculation between two points depending on whether the starting point is "above" or "below" the ending point [WEB10]:

```
(34)    dist = dist / GPS_EARTH_RADIUS;             // Convert distance to radian

         if (sin(lon_d - lon_c) < 0)          // Calculation of bearing
                bearing = acos((sin(lat_d)-sin(lat_c)*cos(dist))/(sin(dist)*cos(lat_c)));
                else
                bearing = 2*PI-acos((sin(lat_d)-sin(lat_c)*cos(dist))/(sin(dist)*cos(lat_c)));

(35)            return bearing * (180 / PI);
```

Finally the *gps_bearing* function returns the calculated course in true degrees to the main program.

4.7.4 Function to determine and store base point information

The function to determine and store base point information was written and tested as part of this project for the robot's navigation system. The function *gps_basing_point* is implemented in the *GPS.lib* library and requires the same starting point *a* and a predefined base point structure *bp* plus the additional input of the calculated true bearing *tc1* between *a* and *b.* It has no return value but fills the structure of the base point. It requires the definition of the distance from the starting point *a* to the base point *bp* in km (**compare exhibit 13**):

```
(25)    #define dbp 0.25                        // distance to next base point in km
(2)     nodebug gps_basing_point(GPSPosition *c, GPSPosition *bp, float tc1)
```

The *gps_basing_point* function first transforms the starting point into radian measure analog to the *gps_ground_distance* function. Moreover the transferred course input *tc1* and the defined distance to the base point *dbp* are transformed to radian measure:

```
(26)    tc1 = tc1 * (PI / 180);
(27)    dist = dbp / GPS_EARTH_RADIUS;
```

With all input parameters transformed to radian measure the formula for the calculation of the base point's longitude and latitude is performed [WEB10]:

```
(28)    lat_d = asin(sin(lat_c) * cos(dist) + cos(lat_c) * sin(dist) * cos(tc1));
        if (cos(lat_d) == 0)
        lon_d = lon_c;  // endpoint a pole
                 else
                 lon_d = (lon_c - asin(sin(tc1) * sin(dist) / cos(lat_d)) + PI);
                dummy = (int) ((lon_d) / (2 * PI));
                dummy2 = (float) dummy * (2 * PI);
                lon_d = lon_d - dummy2;
                lon_d = lon_d - PI;
```

Due to the reason that Dynamic C does not support the modulo operation for float variables as parameters as needed for the formula the variables *dummy* and *dummy2* are introduced to perform the modulo operation in singular steps.

Both the values of the latitude and the longitude of the base point *bp* need to be first transformed back to degree measure and filled in the corresponding structure *bp.* This process is analog for both the latitude and longitude is cited here for the latitude only (**see exhibit 13**):

```
(29)    lat_d = lat_d * 180 / PI;      // assign lat value to base_point stucture
         if(lat_d < 0)
                 bp->lat_direction = 'S';
                 else bp->lat_direction = 'N';
        dummy = (int) lat_d;
        bp->lat_degrees = dummy;
        lat_d = (lat_d - (float)dummy) * 60;
        bp->lat_minutes = lat_d;
```

The process takes into consideration the definition of *N* and *E* as positive and completes the *gps_basing_point* function.

4.7.5 Function for variable time delays

The function to determine and store base point information was written and tested as part of this project for the robot's navigation system in cooperation with the technical support of Z-World. The function *fnMsDelay* is stored in the *TimeDelay.c* and requires an integer number of milliseconds as an input. It has no return value (**compare exhibit 14**):

```
(11)    void fnMsDelay ( unsigned int iDelay )
```

The *fnMsDelay* function takes the delay value *iDelay* and adds it to the *MS_Timer* to get the *StopTime*. It waits until the difference *MS_TIMER – StopTime* turns positive and delays the program for this period of time:

```
(13)    StopTime = MS_TIMER + (unsigned long)iDelay;
                while ( (MS_TIMER - StopTime) < 0L );
```

This function can be implemented in the overall program at any time to delay the controller's speed if necessary.

4.7.6 Algorithm to calculate the offset from track with available functions

The last important part to write the overall program is presented with the algorithm to calculate the offset off the track with the available functions. It is presented and implemented in the *GPS.c* file. In the overall navigation program presented in chapter 4.6 the robot bases his decision whether to correct its bearing on two pieces of information, one being the bearing *b12* between the sample points *1* and *2,* the other being the offset of the track *ot* at the time of the decision respectively the second sample point. This algorithm is based on basic geometric calculations (**see figure 4.20**).

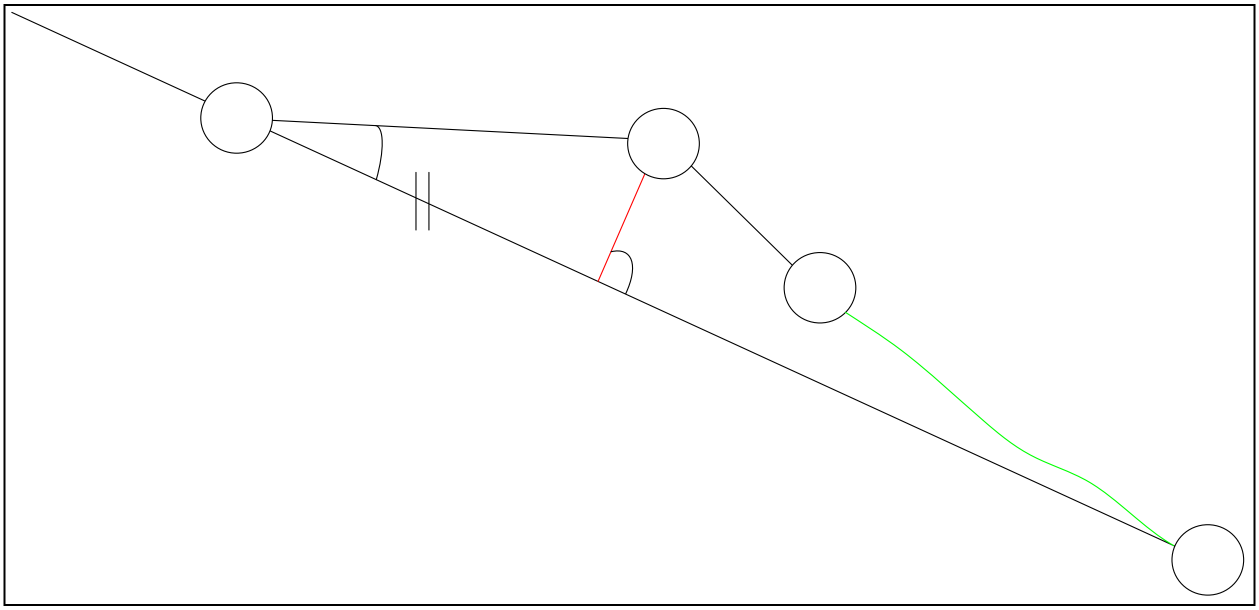


Figure 4.20: Drawing of parameters necessary to calculate the offset [HAM04]

It first determines the bearing to the base point *b2bp* with the *gps_ground_distance* and the *gps_bearing* functions (**see exhibit 15**):

```
(95)    distance2 = gps_ground_distance(&end_p, &way_p); // calculate ground dist
        printf("Distance from End_Point to Way_P: %f km\n", distance2);
        b2bp = gps_bearing(&end_p, &way_p, distance2);  // calculate bearing
        printf("Bearing in true degrees to Way_P: %f degrees\n", b2bp);
```

Subsequently it takes the positive difference between *b2bp* and the initial course *tc1* depending on which is bigger to determine the angle *α*. Knowing that the offset is orthogonal to the initial course it can be calculated with the sin-function using the angle *α* and the distance from point *2* to *bp*:

```
(99)    if (b2bp > tc1)
                {
                alpha = b2bp-tc1;
                }
                else if (b2bp < tc1)
                        {
                        alpha = tc1-b2bp;
                        }
        alpha = alpha * PI / 180;
        offset = sin(alpha) * distance2;
        printf("Offset from initial track: %f km\n", offset);
```

Dynamic C requires that the input value of *α* for the sin-function is of radian measure.

4.8 Summary of the results and further steps

At this point in the project all components for the navigation and control system have been selected (**see section 4.2 and 4.3**). Extensive tests have been performed with especially the GPS-receiver to understand the distribution of the signals and to analyse the accuracy of all output parameters from the GPS-Receiver (**see section 4.4**). The position information turns out to be precise enough for this project even without the differential GPS. However, the bearing and speed information both imply great errors and demand another approach for the overall navigation system.

At this stage the implementation of the main Jackrabbit controller in the control loop is not set. Three different and possible solution have been presented and their advantages or disadvantages discussed (**see section 4.5**). Functions for the DAC to run the motor and the ADC for feedback information have been programmed and tested with the Jackrabbit controller. Both converters although being third party products performed well with the controller and can be implemented in the overall system.

The concept for the overall navigation program has been developed and presented (**see section 4.6**). Taking the test the results into account, the decision making process does not rely on the direct bearing information from the GPS receiver. Instead the controller uses the more accurate position information and calculates the real bearing with two points that are taken in time more than one second apart from each other. Besides it checks the robot's offset off the track for the decision making process.

All necessary functions for the overall navigation program have been written and successfully tested on the Jackrabbit controller with the navigation components (**see section 4.7**). The accuracy of the applied spheroid model of the Earth's shape is good enough to determine distances between points of as little as 10 meters and the respective bearing between the two points. This allows the overall navigation program to be independent of the bearing information from the GPS receiver and still keeps the program and algorithms simple. This solution and the accuracy of the position information reduce the need for the implementation of additional filters in the navigation system.

At this point in the project and as a result of this thesis the feasibility to develop and run a low cost navigation and control system for the Antarctic Plateau has been confirmed. The components are assembled and connected and ready to be tested on a moving platform. The programmed functions need to be implemented in the overall navigation program according to the presented flowchart and the chosen control loop solution. Later the iridium modem can be tested for communication purposes.

Chapter 5 *Abstract of the Power System*

The power system for the robot must be a self sustained, renewable power supply. To satisfy this requirement, solar panels in combination with batteries will be used. The combination of solar panels and batteries is dependant on the size of the robot, and the mounting surface for solar cells.

In order to better understand how much power that can be expected from the robot, some 10 and 20 Watt panels were taken into an open snow covered plain to do some preliminary testing. The azimuth angle of the panel was varied, and the voltage and current was recoded over several different resistances. This data was then used to calculate the maximum power point for each of the different panel configurations. These maximum power points were used to indicate the percent of nominal power that can be expected from each side of the robot.

It was first determined through experimentation that because some power was gained by reflection from the snow, the panels were most effective when they were situated vertically up and down. The percentage of the nominal power that could be expected from each side of the robot assuming it were a square shape with a panel laying flat across the top was calculated. Figure xx shows a drawing of the "cubed" robot, along with the percent of nominal power that can be expected from each side.

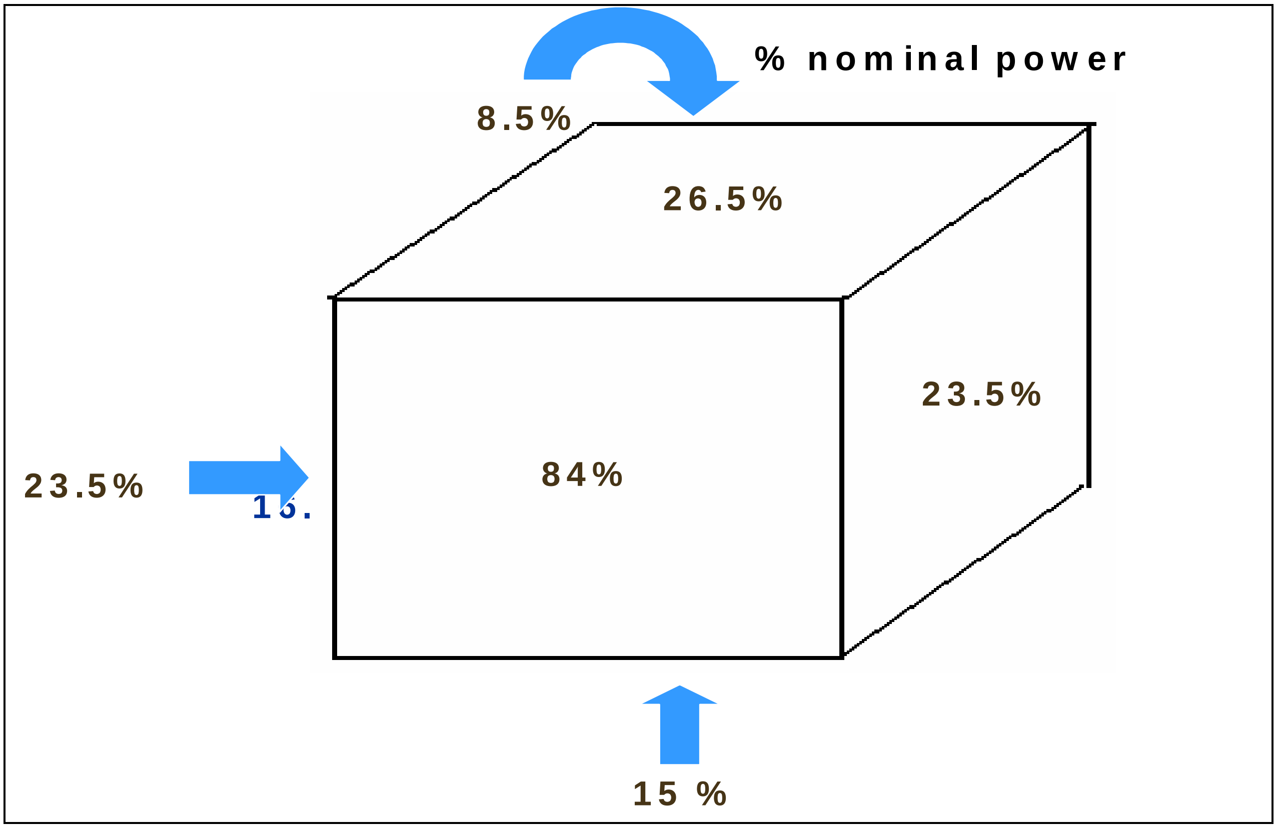


Figure 5.1: Nominal power that can be expected from each side of the robot

Once it was known how much of the nominal power could be harnessed from each side of the robot, it was possible to calculate how much power could be expected from each side of the robot. This was done using the following equation:

Power = (% nominal power)*(Watt/m^2)*(Area of panel)

It should be noted that the panels used in testing were rated at 12% efficiency, and the panels to be used on the robot will be at least 16% efficient. Thus in the calculations there was an additional scaling factor of 1.33.

The total power to be expected from the robot is then simply the summation of the power accumulated by all sides. Figure xx displays the "cubed" robot, along with how much power is expected from each side, and the total expected power. This data shows that it will be possible to power the robots four motors directly with the solar panels.

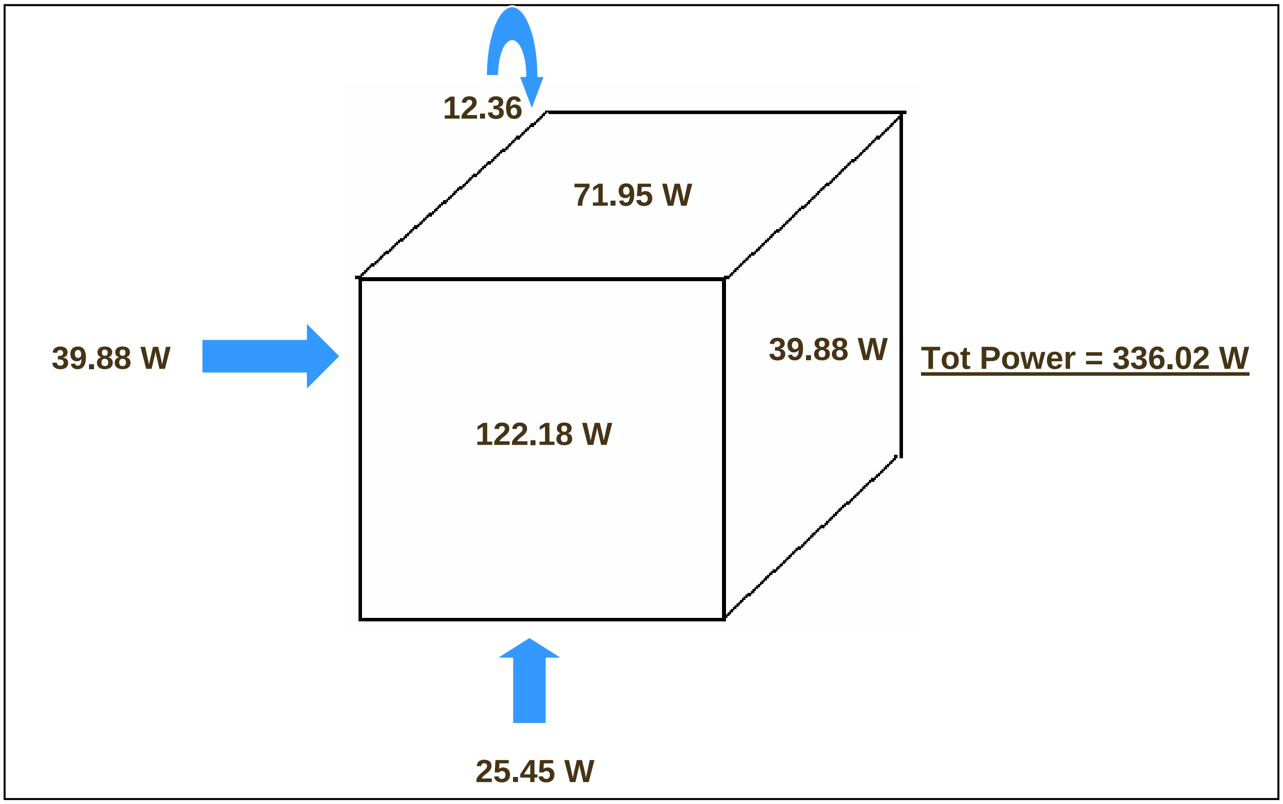


Figure 5.2: Absolute power that can be expected from each side of the robot

Another experimental test involved testing the voltage and current on panels at varying heights above the snow. This data was collected to determine whether or not a significant amount of reflected power would be lost or gained if the height of the chassis were to change. Results from this experiment showed no significant change in power output with varying height above the snow. The percent of nominal power output for azimuth angles between the surface and edge of the panels was also measured. This was to better understand the amount of power that can be expected when the sun was shining between sides of the robot (i.e. a corner). This data was then added to the previous data recorded to better track the robots power output over the course of a standard Antarctic day.

The robot will be equipped with maximum power point trackers for each panel on the robot. These trackers will regulate the panel's current and voltage over different loads, to ensure each panel is producing the maximum possible power. Experimental results show that the maximum power points all lie at roughly the same voltage, with varying current. This is to be expected as the voltage produced by the cells is only temperature dependant, whereas the current produced by the panels is incident dependant.

Because the output voltage of the maximum power point trackers will be between 12 and 24 volts, each one will then be connected to a DC-DC boost converter. These converters will regulate the output voltage of the maximum power point trackers up to the 48 volts necessary to run the four motors.

The robot will also utilize three UBI-5150 Lithium Ion rechargeable batteries for backup and emergency power. These batteries will be used to occasionally move the robot in times of cloud cover, to ensure the motors do not freeze. They will also be able to provide the navigation unit with power in case of solar panel failure.

Chapter 6 Summary of Project Results

These diploma theses confirm the feasibility to deploy a cool robot in the Antarctic over the summer months from November to February.

Chapter 3 describes the development of a mechanical design for the robot that meets all requirements from the mission and the application area. Lightweight and stiff materials such as honeycomb structures are available that help to keep the robot's weight down. The components for the propulsion system are tested at low temperatures and perform at a high overall efficiency. It is possible to build a box for all components with a high insulation to keep the heat inside. With regard to the power need of the overall system the design solution with the solar cells at the outside is favoured.

Subsequently the development of the navigation and control system is presented in chapter 4. Again a low-cost and low-weight solution of suitable components is shown that match the requirements of the project. Extensive tests of the GPS receiver show that the accuracy is high enough to make the overall navigation program work without further positioning components. The layout of the control system helps to keep the need for computational power low. All needed functions for the waypoint navigation of the robot are programmed and successfully tested with the main controller.

Finally the abstract of the power system in chapter 5 confirms the possibility to collect enough energy from the sun with state-of-the-art solar panels at the outside of the robot. However extra high-density Lithium Ion batteries are part of the power system to ensure enough energy if needed for the vital electronics of the robot. Taking the results of the theses a prototype can built and used as the platform to test the navigation and control system.

References

[APO01] Dimitrios S. Apostolopoulos : Analytical Configuration of Wheeled Robotic Locomotion
Carnegie Mellon University, Pittsburgh, PA 15213, USA, 2001

[BET00] Klaus Betke: The NMEA 0183 Protocol

May 2000, revised August 2001

[BLA03] George L. Blaisdell: Private Communication
George.L.Blaisdell@erdc.usace.army.mil
Research Civil Engineer, CRREL, Hanover, New Hampshire, USA, 2003

[BOR96] J. Borenstein, H.R. Everett, L. Feng, D. Wehe: Mobile Robot Positioning – Sensors and Techniques, University of Michigan, Advanced Technologies Lab, 1996

[BYN01] Richard T. Bynum: Insulation Handbook
McGraw Hill, New York, NY 10020, 2001

[DOR98] Richard C. Dorf et al.: Modern Control Systems
Addison Wesley Longman, Inc., Menlo Park, CA 94025, 1998

[DOT90] Y. Dote: Servo Motor and Motion Control Using Digital Signal Processors
Prentice Hall, Englewood Cliffs, New Jersey 07632, 1990

[FOE99] Alex Foessel et al.: Radar sensor for an autonomous Antarctic explorer
The Robotic´s Institute, Carnegie Mellon University Pittsburgh, PA 15213, 1999

[GIE02] Jacek F. Gieras et al.: Permanent Magnet Motor Technology
Marcel Dekker, Inc., New York, NY 10016, 2002

[GRA96] J.O. Gray et al.: Advanced Robotics & Intelligent Machines
Institution of Electrical Engineers, United Kingdom, 1996

[HOR94] John J. Horton: The Antarctic
Clio Press, Oxford, England, 1994

[KAP96] Elliott D. Kaplan: Understanding GPS: principles and applications
Artech House, Inc., 1996

[KEN85] T. Kenjo et al.: Permanent Magnet –Motor and Brushless DC Motors
Oxford University Press, New York, 1985

[KIN69] H.G.R. King, The Antarctic
Blandford Press, London, England, 1969

[LES03] Marc R. Lessard Private Communication marc.r.lessard@dartmouth.edu
Ph.D., Assistant Professor of Engineering, Thayer School of Engineering, Dartmouth College, Hanover, New Hampshire, 2003

[LEV02] James H. Lever: Cold-Regions Issues for Unmanned Ground Vehicles
Ph.D., Engineer, CRREL, Hanover, New Hampshire, USA, 2002

[LEV04] James H. Lever: Private Communication James.H.Lever@erdc.usace.army.mil
Ph.D., Engineer, CRREL, Hanover, New Hampshire, USA, 2004

[LYN83] Peter Lynwander: Gear Drive Systems – Design and Application
Marcel Dekker, Inc., New York, NY 10016, 2002

[MAX04] Maxim Integrated Products: Specifications for MAX 536 and MAX186
http://www.maxim-ic.com, CA, USA, 2004

[MIL89] T.J.E. Miller: Brushless Permanent-Magnet and Reluctance Motor Drives
Oxford University Press, NY, USA, 1989

[MOR00] Stewart Moorehead et. al.: Autonomous navigation field results of a planetary analog robot in Antarctica
The Robotic´s Institute, Carnegie Mellon University Pittsburgh, PA 15213, 2000

[NGO04] Ngoc Hoang: Private Communications nth@nalresearch.com
Application Engineer, NAL Research Corporation, USA, 2004

[PRZ04] Mike Przybyla: Private Communications priz@gmheng.com
Marketing Manager, GMH Engineering, USA, 2004

[RAY03] Laura Ray: Project Description
Associate Professor of Engineering, Thayer School of Engineering, Dartmouth College, Hanover, New Hampshire, 2003

[RAY04] Laura Ray: Private Communications Laura.E.Ray@Dartmouth.EDU
Associate Professor of Engineering, Thayer School of Engineering, Dartmouth College, Hanover, New Hampshire, 2004

[ROT02] Hendrik Rothe: Measurement and test engineering lecture notes,
Helmut-Schmidt-University/University of the Federal Armed Forces Hamburg, 2002

[ROT96] Karlheinz Roth: Konstruieren mit Konstruktionskatalogen, Band 3, 2. Aufl.
Springer Verlag Berlin Heidelberg, 1996

[SHA98] Ben Shamah, William Whittaker et al.: Field Validation of Nomad's Robotic Locomotion
Carnegie Mellon University, Pittsburgh, PA 15213, USA, 1998

[SHA01] Ben Shamah, William Whittaker et al.: Steering and Control of a Passively Articulated Robot
Carnegie Mellon University, Pittsburgh, PA 15213, USA, 2001

[SIM96] R. Simmons et al.: Experience with Rover Navigation for Lunar-Like Terrains, The Robotic´s Institute, Carnegie Mellon University Pittsburgh, PA 15213, 1996

[STA95] Wolfram Stadler: Analytical Robotics and Mechatronics
McGraw Hill, New York, NY 10020, 1995

[STE86] Robert F. Stengel: Stochastic optimal control, Department of Mechanical and Aerospace Engineering, Princeton University, New Jersey, 1986

[WAR04] Randy Warner: Private Communication randy@synergy-gps.com
Senior Applications Engineer, Synergy Systems, LLC, USA, 2004

[WEL03] Greg Welch, Gary Bishop: An introduction to the Kalman Filter
Department of Computer Science, University of North Carolina, 2003

[WET02] David Wettergreen, et.al. : First Experiment in Sun-Synchronous Exploration
Carnegie Mellon University, Pittsburgh, PA, USA, 2002

[WHI01] William Whittaker, Kimberly Shillcut: Solar Navigational Planning for Robotic Explorers
Carnegie Mellon University, Pittsburgh, PA, USA, 2001

[WIL02] Ed Williams: Navigation on the spheroidal earth
USA, 2002

Internet:

[WEB01] Quark Expeditions;
http://www.quarkexpeditions.com/antarctica/enviro.shtml

[WEB02] British Antarctic Survey;
http://www.antarctica.ac.uk/About_Antarctica/FAQs/faq_05.html

[WEB03] NOMAD – The Robot;
http://www.frc.ri.cmu.edu/projects/meteorobot2000/nomad/index.html

[WEB04] Mars Exploration Rovers – Mission Overview
http://www.marsinstitute.info/

[WEB05] Spacecraft: Surface Operations: Rover
http://www.marsrovers.jpl.nasa.gov/mission/spacecraft_rover_energy.html

[WEB06] iRobot: The All-Terrain Robot Line
http://www.irobot.com/rwi/p01.asp

[WEB07] PackBot Specifications
http://www.packbot.com/products/scout/applications.asp

[WEB08] PackBot Scout: Benefits
http://www.packbot.com/products/scout/benefits.asp

[WEB09] The Earth according to WGS 84
http://home.online.no/~sigurdhu/Grid_1deg.htm

[WEB10] Ed Williams:Aviation Formulary V1.41 http://williams.best.vwh.net/avform.htm

Appendix I

1. Presentation of Tables

Exhibit 1: Beam Coefficient [Hexcel Composites]

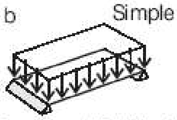
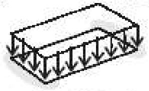
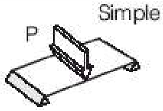
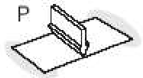
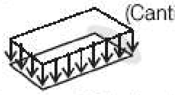
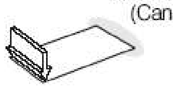
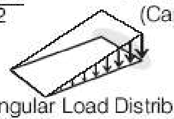
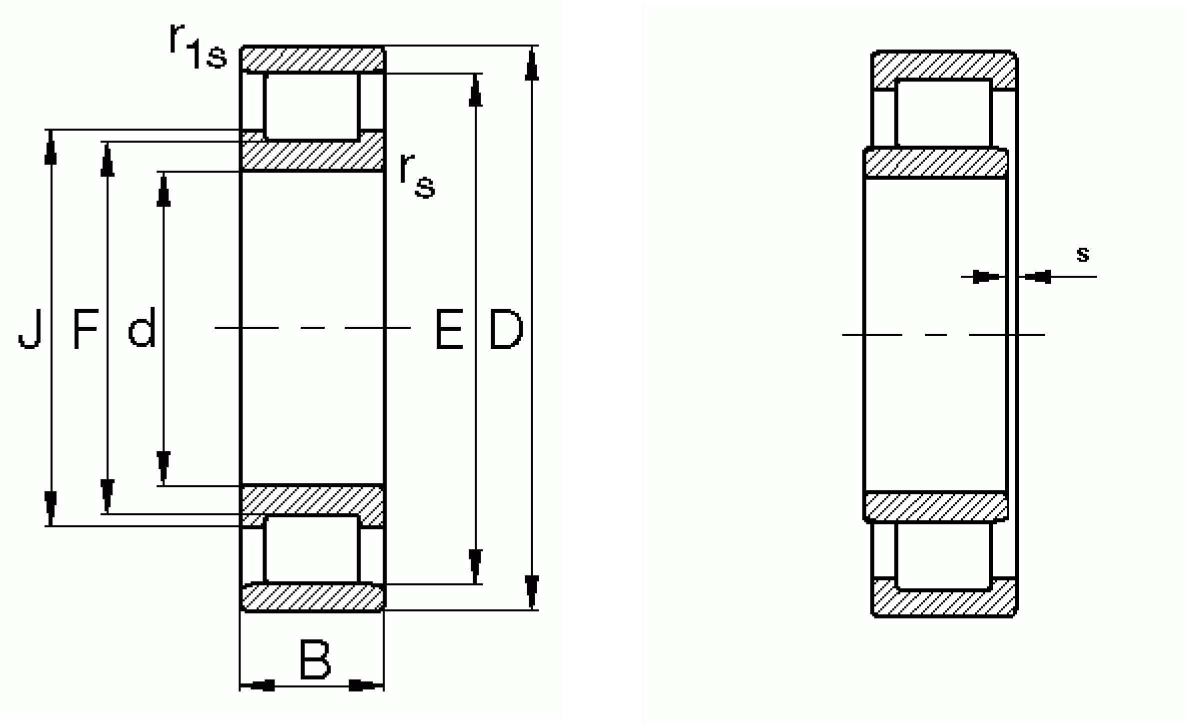
BEAM TYPE	MAXIMUM SHEAR FORCE F	MAXIMUM BENDING MOMENT M	BENDING DEFLECTION COEFFICIENT k_b	SHEAR DEFLECTION COEFFICIENT k_s
$P = ql/b$ Simple Support  Uniform Load Distribution	$\frac{P}{2}$	$\frac{Pl}{8}$	$\frac{5}{384}$	$\frac{1}{8}$
$P = ql/b$ Both Ends Fixed  Uniform Load Distribution	$\frac{P}{2}$	$\frac{Pl}{12}$	$\frac{1}{384}$	$\frac{1}{8}$
Simple Support P  Central Load	$\frac{P}{2}$	$\frac{Pl}{4}$	$\frac{1}{48}$	$\frac{1}{4}$
Both Ends Fixed P  Central Load	$\frac{P}{2}$	$\frac{Pl}{8}$	$\frac{1}{192}$	$\frac{1}{4}$
$P = ql/b$ One End Fixed (Cantilever)  Uniform Load Distribution	P	$\frac{Pl}{2}$	$\frac{1}{8}$	$\frac{1}{2}$
One End Fixed (Cantilever) P  Load One End	P	Pl	$\frac{1}{3}$	1
$P = \frac{ql/b}{2}$ One End Fixed (Cantilever)  Triangular Load Distribution	P	$\frac{Pl}{3}$	$\frac{1}{15}$	$\frac{1}{3}$

Exhibit 2: Meeting the weight limit; Voltage and Current requirement of the Components

WEIGHT / ENERGY

Component	Deviations [LxBxH] [mm]			Volume [cm³]	Weight [kg]	Voltage [V]	Current [A]
	4.68	kg/m²		2.34	m² *(SURFACE)*		
Body	840	560	500	235200.00	10.955		
Insulation	840	560	500		4.038	Vacupanel	1"Styro R6.5
density	1.725	kg/m² (2")				6.25	0.863
Motor	76.2	57.15	57.15	995.51	3.480	48	1.750
Encoder	8.9	42	42	62.80	0.400	5	0.070
Gearbox	117.6	60	60	1693.44	2.200		
	7800	kg/m³					
	Diameter						
Shaft	[mm]	15	450	318.09	2.481		
Controller	129.3	75.8	25.1	984.01	1.120	48	0.900
Amount #	4						
GPS	60	40	10	24.00	0.025	3	0.062
Antenna1	13	38	34	16.80	0.090	3	0.020
Modem	196.4	82.6	39	632.68	0.610	5	0.120
Antenna2	14	55	55	42.35	0.084		
Magnetometer	82.6	35.1	32.3	93.65	0.100	5	0.025
Batteries	130.2	111	43	1864.33	4.200	15.2	7.500
# of Batt.	3						
Solar Panel	1456.92	731.88	36.48	38898.28	15.000	35	4.500
Recharger	63.5	31.8	127	256.45	2.500	24	12.000
RIM	8"		EA	1.00	4.000	TOTAL	
Tube			EA	1.00	4.000	TOTAL	
TIRE	20"/7"		EA	2.50	10.000	TOTAL	
TOTAL				**281089.24**	**65.28**		

Exhibit 3: Cylinder Bearing



FAG N204E.TVP2

d=	20.00	mm
D=	47.00	mm
B=	14.00	mm
$r_{s\ min}$=	1.00	mm
J≈	29.70	mm
E=	41.50	mm
F=	26.50	mm
$r_{1s\ min}$=	0.60	mm
s=	0.80	mm

mass ≈	0.11 [kg]		
Dynamic load rating C_{dyn} =	27.50 [kN]	Static load rating C_0 $_{stat}$ =	24.50 [kN]
Marginal speed =	16000 [min^{-1}]		
Basic speed =	15000 [min^{-1}		

Exhibit 4: Motor Efficiency

NO LOAD TEST

#	gain	U [V]	I [A]	w [rad/sec]	n
1	0.025	48			
2	0.05	48	0.136	21.5	205.31
3	0.075	48	0.186	77.8	742.94
4	0.1	48	0.196	130.1	1242.36
5	0.125	48	0.215	177.8	1697.86
6	0.15	48	0.227	224.3	2141.91
7	0.175	48	0.237	270.1	2579.27
8	0.2	48	0.244	315.6	3013.76
9	0.225	48	0.254	360.8	3445.39
10	0.25	48	0.249	411.2	3926.67
11	0.275	48	0.275	461.7	4408.91
12	0.3	48	**0.272**	550	**5252.11**
13	0.325	48	**0.291**	549.7	**5249.25**
14	0.35	48	**0.272**	549.7	**5249.25**

(NO-LOAD CURRENT based on 3600 values)

speed - torque diagramm RPM = 5250 - 3777xT [Nm] *MOTOR ONLY*

n0= 5250

Tstall= 1.390

current - torque diagramm I = 0.245 + 1/0.095 x T [Nm] *MOTOR ONLY*

I0= 0.11

Kt= 10.5

Efficiency: Etha = (T[Nm] x 2PIxn /60 [rad/sec]) / (U x I)

T	RPM	I		Efficiency	Output T Gear	$n_{[out]}$
[Nm]		[A]			[Nm]	[RPM]
0				motor only	90% loss	./.100
0.01	5212.23	0.350263		0.3247	0.90	52.12
0.02	5174.46	0.455526		0.4956	1.80	51.74
0.03	5136.69	0.560789		0.5995	2.70	51.37
0.04	5098.92	0.666053		0.6681	3.60	50.99
0.05	5061.15	0.771316		0.7158	4.50	50.61
0.06	5023.38	0.876579		0.7501	5.40	50.23
0.07	4985.61	0.981842		0.7755	6.30	49.86
0.084	4932.732	1.129211		0.800533055	7.56	49.32732
0.09	4910.07	1.192368		0.8086	8.10	49.10
0.1	4872.3	1.297632		0.8192	9.00	48.72
0.11	4834.53	1.402895		0.8270	9.90	48.35

Appendix I

0.12	4796.76	1.508158		0.8327	10.80	47.97	v[m/s]
0.13	**4759**	**1.6134**		**0.8366**	**11.70**	**47.59**	1.27
0.14	4721.22	1.718684		0.8390	12.60	47.21	
0.15	4683.45	1.823947		0.8403	13.50	46.83	
0.16	4645.68	1.929211		0.8406	14.40	46.46	
0.17	4607.91	2.034474		0.8400	15.30	46.08	
0.18	4570.14	2.139737		0.8387	16.20	45.70	
0.19	4532.37	2.245		0.8369	17.10	45.32	
0.2	4494.6	2.350263		0.8344	18.00	44.95	
0.21	4456.83	2.455526		0.8315	18.90	44.57	
0.22	4419.06	2.560789		0.8283	19.80	44.19	
0.23	4381.29	2.666053		0.8246	20.70	43.81	
0.24	4343.52	2.771316		0.8206	21.60	43.44	
0.25	4305.75	2.876579		0.8164	22.50	43.06	
0.26	4267.98	2.981842		0.8119	23.40	42.68	
0.27	4230.21	3.087105		0.8072	24.30	42.30	
0.28	4192.44	3.192368		0.8022	25.20	41.92	
0.29	4154.67	3.297632		0.7971	26.10	41.55	
0.3	4116.9	3.402895		0.7918	27.00	41.17	
0.31	4079.13	3.508158		0.7864	27.90	40.79	
0.32	4041.36	3.613421		0.7808	28.80	40.41	
0.33	4003.59	3.718684		0.7751	29.70	40.04	
0.34	3965.82	3.823947		0.7693	30.60	39.66	
0.35	3928.05	3.929211		0.7634	31.50	39.28	
0.36	3890.28	4.034474		0.7573	32.40	38.90	
0.37	3852.51	4.139737		0.7512	33.30	38.53	
0.38	3814.74	4.245		0.7450	34.20	38.15	max cont.
0.39	3776.97	4.350263		0.7387	35.10	37.77	
0.4	3739.2	4.455526		0.7324	36.00	37.39	
0.41	3701.43	4.560789		0.7259	36.90	37.01	
0.42	3663.66	4.666053		0.7195	37.80	36.64	
0.43	3625.89	4.771316		0.7129	38.70	36.26	
0.44	3588.12	4.876579		0.7063	39.60	35.88	
0.45	3550.35	4.981842		0.6997	40.50	35.50	
0.46	3512.58	5.087105		0.6929	41.40	35.13	
0.47	3474.81	5.192368		0.6862	42.30	34.75	
0.48	3437.04	5.297632		0.6794	43.20	34.37	
0.49	3399.27	5.402895		0.6726	44.10	33.99	
0.5	3361.5	5.508158		0.6657	45.00	33.62	
0.51	3323.73	5.613421		0.6588	45.90	33.24	
0.52	3285.96	5.718684		0.6519	46.80	32.86	
0.53	3248.19	5.823947		0.6449	47.70	32.48	
0.54	3210.42	5.929211		0.6379	48.60	32.10	
0.55	3172.65	6.034474		0.6309	49.50	31.73	
0.56	3134.88	6.139737		0.6238	50.40	31.35	
0.57	3097.11	6.245		0.6167	51.30	30.97	
0.58	3059.34	6.350263		0.6096	52.20	30.59	
0.59	3021.57	6.455526		0.6025	53.10	30.22	
0.6	2983.8	6.560789		0.5953	54.00	29.84	

0.61	2946.03	6.666053		0.5881	54.90	29.46
0.62	2908.26	6.771316		0.5810	55.80	29.08
0.63	2870.49	6.876579		0.5737	56.70	28.70
0.64	2832.72	6.981842		0.5665	57.60	28.33
0.65	2794.95	7.087105		0.5592	58.50	27.95
0.66	2757.18	7.192368		0.5520	59.40	27.57
0.67	2719.41	7.29763		0.5447	60.30	27.19
0.68	2681.64	7.40289		0.5374	61.20	26.82
0.69	2643.87	7.50815		0.5301	62.10	26.44
0.7	2606.1	7.61342		0.5228	63.00	26.06
0.71	2568.33	7.71868		0.5154	63.90	25.68
0.72	2530.56	7.82394		0.5081	64.80	25.31
0.73	2492.79	7.92921		0.5007	65.70	24.93
0.74	2455.02	8.03447		0.4933	66.60	24.55
0.75	2417.25	8.13973		0.4859	67.50	24.17
0.76	2379.48	8.245		0.4785	68.40	23.79
0.77	2341.71	8.35026		0.4711	69.30	23.42
0.78	2303.94	8.45552		0.4637	70.20	23.04
0.79	2266.17	8.56078		0.4562	71.10	22.66
0.8	2228.4	8.66605		0.4488	72.00	22.28
0.81	2190.63	8.77131		0.4413	72.90	21.91
0.82	2152.86	8.87657		0.4339	73.80	21.53
0.83	2115.09	8.98184		0.4264	74.70	21.15
0.84	2077.32	9.08710		0.4189	75.60	20.77
0.85	2039.55	9.19236		0.4114	76.50	20.40
0.86	2001.78	9.29763		0.4040	77.40	20.02
0.87	1964.01	9.40289		0.3965	78.30	19.64
0.88	1926.24	9.50815		0.3889	79.20	19.26
0.89	1888.47	9.61342		0.3814	80.10	18.88
0.9	1850.7	9.71868		0.3739	81.00	18.51
0.91	1812.93	9.82394		0.3664	81.90	18.13
0.92	1775.16	9.92921		0.3588	82.80	17.75
0.93	1737.39	10.0344		0.3513	83.70	17.37
0.94	1699.62	10.1397		0.3437	84.60	17.00
0.95	1661.85	10.245		0.3362	85.50	16.62
0.96	1624.08	10.3502		0.3286	86.40	16.24
0.97	1586.31	10.4555		0.3211	87.30	15.86
0.98	1548.54	10.5607		0.3135	88.20	15.49
0.99	1510.77	10.6660		0.3059	89.10	15.11
1	1473	10.7713		0.2983	90.00	14.73
1.01	1435.23	10.8765		0.2908	90.90	14.35
1.02	1397.46	10.9818		0.2832	91.80	13.97
1.03	1359.69	11.0871		0.2756	92.70	13.60
1.04	1321.92	11.1923		0.2680	93.60	13.22
1.05	1284.15	11.2976		0.2604	94.50	12.84
1.059	1250.157	11.3923		0.2535	95.31	12.50
			MAX.:	0.8406		

Exhibit 5: CRREL Test Results

TEST #	**1**
Description:	Cool down from room temp. to minus 40C
box:	open
motor:	running

time	temp - chamber	temp - box	current "I"
[min]	[°C]	[°C]	[A]
0	22	24.0	0.340
3	18	20.4	0.340
5	16	19.1	0.340
7	14	17.5	0.340
9	12	16.2	0.340
11	11	14.8	0.340
13	10	13.5	0.340
15	8	12.4	0.340
17	7	11.1	0.340
19	6	9.9	0.340
21	5	8.9	0.340
23	4	7.6	0.340
25	2	6.5	0.340
27	1	5.5	0.340
29	0	4.3	0.350
31	-1	3.4	0.355
33	-2	2.3	0.360
35	-3	1.1	0.360
37	-4	0.2	0.360
39	-5	-0.9	0.365
41	-6	-1.7	0.365
43	-7	-2.8	0.370
45	-8	-3.9	0.350
47	-9	-4.7	0.360
49	-10	-5.9	0.350
51	-11	-6.8	0.360
53	-12	-7.7	0.360
55	-14	-9.1	0.360
57	-14	-9.7	0.360
59	-15	-11.1	0.360
61	-16	-11.6	0.360
63	-18	-13.1	0.365
65	-18	-14.0	0.370
67	-20	-15.3	0.375
69	-21	-16.1	0.375
71	-22	-17.7	0.375
73	-23	-18.5	0.375
75	-24	-19.9	0.380
77	-25	-20.4	0.375

79	-26	-22.1	0.375
81	-26	-22.5	0.375
83	-28	-24.2	0.395
85	-28	-24.5	0.390
87	-30	-26.0	0.390
89	-30	-26.7	0.395
91	-32	-27.7	0.390
93	-33	-29.1	0.390
95	-33	-29.2	0.390
97	-35	-30.8	0.390
99	-35	-31.4	0.390
101	-36	-32.0	0.390
103	-38	-33.7	0.390
105	-38	-34.0	0.390
107	-39	-34.3	0.390
109	-40	-35.9	0.400
111	-40	-36.3	0.400

Motor Stop - Cool Down for 1hr 35min

TEST #	**2**
Description:	Motor start
box:	open
motor:	Run for 10 sec. - Pause for 30 sec.

Attempt # 	**temp - box** [°C]	**current "I"** [A]
1	-37.7	0.61
2	-37.6	0.60
3	-37.7	0.62
4	-37.7	0.61
5	-37.7	0.61
6	-37.7	0.62
7	-37.6	0.60
8	-37.6	0.58
9	-37.6	0.58
10	-37.6	0.58
11	-37.6	0.56
12	-37.6	0.58
13	-37.5	0.58
14	-37.5	0.59
15	-37.5	0.60
16	-37.5	0.60
17	-37.4	0.60
18	-37.6	0.60
19	-37.6	0.61
20	-37.6	0.61
21	-37.6	0.61
22	-37.6	0.61
23	-37.6	0.61
24	-37.7	0.59
25	-37.7	0.59
26	-37.7	0.60
27	-37.7	0.60
28	-37.7	0.60
29	-37.7	0.57
30	-37.7	0.57

TEST # **3**

Description: long-time run / chamber temp = const. -40C

box: closed

getting an idea of R-value

motor: running for 2 hours

U=48V

		power supply		
time	**temp - box**	**current "I"**	Power	Energy
[min]	[°C]	[A]	[W]	[Ws]
0	-36.2	0.60	28.8	3456.00
2	-36.2	0.52	24.96	2995.20
4	-36.3	0.50	24	2880.00
6	-36.2	0.49	23.52	2822.40
8	-36.2	0.48	23.04	2764.80
10	-35.9	0.47	22.56	2707.20
12	-35.8	0.47	22.56	2707.20
14	-35.5	0.46	22.08	2649.60
16	-35.3	0.46	22.08	2649.60
18	-35.1	0.46	22.08	2649.60
20	-34.8	0.47	22.56	2707.20
22	-34.6	0.46	22.08	2649.60
24	-34.3	0.46	22.08	2649.60
26	-34.1	0.44	21.12	2534.40
28	-33.8	0.44	21.12	2534.40
30	-33.6	0.44	21.12	2534.40
32	-33.4	0.43	20.64	2476.80
34	-33.1	0.43	20.64	2476.80
36	-32.9	0.43	20.64	2476.80
38	-32.6	0.43	20.64	2476.80
40	-32.3	0.42	20.16	2419.20
42	-32.2	0.42	20.16	2419.20
44	-31.9	0.41	19.68	2361.60
46	-31.6	0.40	19.2	2304.00
48	-31.4	0.41	19.68	2361.60
50	-31.2	0.41	19.68	2361.60
52	-31.1	0.41	19.68	2361.60
54	-30.9	0.40	19.2	2304.00
56	-30.7	0.41	19.68	2361.60
58	-30.4	0.40	19.2	2304.00
60	-30	0.40	19.2	2304.00
62	-29.7	0.40	19.2	2304.00
64	-29.4	0.39	18.72	2246.40
66	-29.5	0.39	18.72	2246.40
68	-29.2	0.39	18.72	2246.40
70	-29.1	0.39	18.72	2246.40
72	-28.8	0.39	18.72	2246.40
74	-28.7	0.38	18.24	2188.80
76	-28.7	0.38	18.24	2188.80

78	-28.8	0.38	18.24	2188.80
80	-28.4	0.38	18.24	2188.80
82	-28.1	0.38	18.24	2188.80
84	-27.8	0.37	17.76	2131.20
86	-27.7	0.37	17.76	2131.20
88	-27.6	0.37	17.76	2131.20
90	-27.9	0.37	17.76	2131.20
92	-27.7	0.37	17.76	2131.20
94	-27.9	0.38	18.24	2188.80
96	-27.5	0.37	17.76	2131.20
98	-27.2	0.37	17.76	2131.20
100	-27.4	0.37	17.76	2131.20
102	-27.1	0.37	17.76	2131.20
104	-27.1	0.36	17.28	2073.60
106	-26.9	0.36	17.28	2073.60
108	-27.2	0.37	17.76	2131.20
110	-27.4	0.36	17.28	2073.60
112	-27.1	0.37	17.76	2131.20
114	-26.9	0.36	17.28	2073.60
116	-26.8	0.37	17.76	2131.20
118	-26.6	0.37	17.76	2131.20
120	-26.5	0.38	18.24	2188.80
				144518.40

2. Contact Information

EAD Motors Inc. *(BLDC Motor)*
1 Progress Dr.
Dover, NH 03820

www.eadmotors.com

Applications Engineer: Michelle Cullen
Phone: 603.742.3330
Fax: 603.742.9080
mcullen@eadmotors.com

Customer Service: Kelly Pennell
Phone: 603.742.3330 X663
kpennell@eadmotors.com

Neugart USA *(Gearhead)*
3047 Industrial Boulevard
Bethel Park, PA 15102-2537

www.neugartusa.com

Vice President of Sales: Timothy L. Francis
Phone: 412.835.4154
Fax: 412.853.4194
Cell:412.498.3057
tfrancis@neugartusa.com

ADVANCED MOTION CONTROLS (AMC) *(Motor Controller)*
3805 Calle Tecate
Camarillo, CA 93012

www.a-m-c.com

Technical Support: Solomon Cheng
Phone: 805.389.1935 X308
scheng@a-m-c.com

Customer Support: Christy Ferry
Phone: 805.389.1935
Fax: 805.389.1165
cferry@a-m-c.com

Senior Customer Support: Karen Summa
Phone: 805.389.1935
Fax: 805.389.1165
ksumma@a-m-c.com

Cold Regions Research and Engineering Laboratory (CRREL) *(Cold Chamber)*
72 Lyme Road
Hanover, NH 03755

James.H.Lever@erdc.usace.army.mil (Ph.D. Engineer)
Christopher.R.Williams@erdc.usace.army.mil (Cold Chamber)
Jason.C.Weale@erdc.usace.army.mil (Batteries)

Energy Storage Technologies, Inc. (EST Global) *(Vacuumpanel)*
7610 McEwen Road
Dayton, OH 45459

www.vacupanel.com

Customer Support : Lyndy Lancaster
Phone : 937. 312.0114
Fax : 937.312.1277
LLancaster@estglobal.com

SPI Supplies *(Lubricate/Braycote)*
PO BOX 656
West Chester, PA 19381-0656

www.2spi.com

Customer Support: Charles A. Garber, Ph. D.
Phone: 610.436.5400
Fax: 610.436.5755
cgarber@2spi.com
Manager: M. Kim Murray
mmurray@2spi.com

TEKLAM *(Honeycomb Panel)*
1121 Olympic Drive
Corona, CA 92881

www.teklam.com

Sales: Bob Simmons, JR.
Phone: 909.278.4563
Fax: 909.278.0389
bobjr@teklam.com

Hecel Composites *(Honeycomb Panel)*
www.hexcelcomposites.com

Appendix II

1. Presentation of Maple scripts with complete GPS test results

Exhibit 1: Test results for steady state test on 11/26/2003with the GPS receiver

```
restart;with(stats):with(describe):with(linalg):with(fit):with(plo
ts):
```

```
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

```
A:=importdata("C:/GPS_test.txt",2):

X:=[A][1],[A][2]:N1:=count(X[1]);N2:=count(X[2]);dx:=[seq(mean(X[i
]),i=1..2)];
```

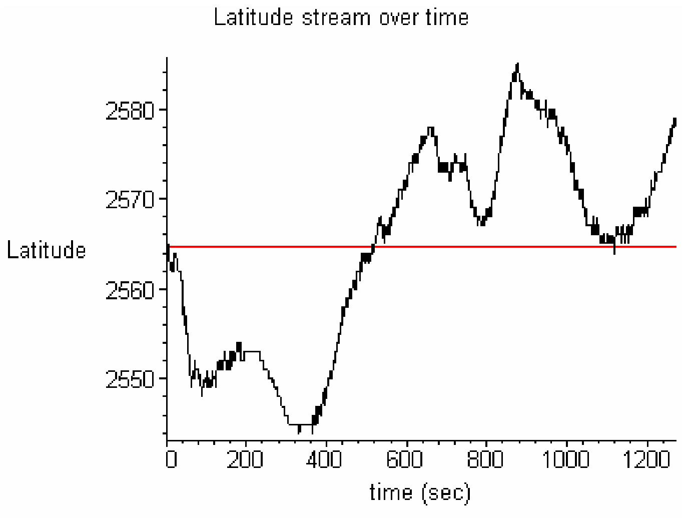
$$N1 := 1274$$

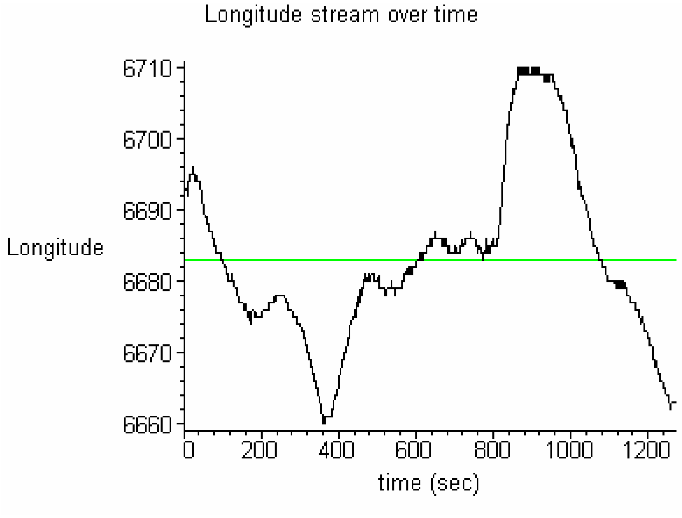
$$N2 := 1274$$

$$dx := [2564.603611, 6683.142072]$$

```
p:=plot([seq([k,X[1][k]],k=1..N1)],
color=black):p3:=plot(dx[1],k=0..N1, color=red):
 display(p,p3,title="Latitude stream over time",labels=["time
(sec)",Latitude]);p:=plot([seq([k,X[2][k]],k=1..N2)],
color=black):p3:=plot(dx[2],k=0..N2, color=green):
 display(p,p3,title="Longitude stream over time",labels=["time
(sec)",Longitude]);
sx:=[seq(standarddeviation[1](X[i]),i=1..2)];vx:=[seq(variance[1](
X[i]),i=1..2)];
sx_Lat_meter:=(sx[1]/10)*1.852;vx_Lat_meter:=(vx[1]/10)*1.852;
d_max:=[seq(max(seq((X[i][j]-
dx[i]),j=1..N1)),i=1..2)];d_max_meter_Lat:=(d_max[1]/10)*1.852;
d_max_meter_Lon:=(d_max[2]/10)*1.359;


d_max12:=[seq(max(seq((X[i][j]-X[i][j-
1]),j=2..N1)),i=1..2)];d_max12_meter_Lat:=(d_max12[1]/10)*1.852;
d_max12_meter_Lon:=(d_max12[2]/10)*1.349;
sx_Lon_meter:=(sx[2]/10)*1.359;vx_Lon_meter:=(vx[2]/10)*1.359;
```

Appendix II





$$sx := [\,11.31693052,\ 12.19400916\,]$$

$$vx := [\,128.0729163,\ 148.6938593\,]$$

$$sx\_Lat\_meter := 2.095895532$$

$$vx\_Lat\_meter := 23.71910410$$

$$d\_max := [\,20.396389,\ 26.857928\,]$$

$$d\_max\_meter\_Lat := 3.777411243$$

$$d\_max\_meter\_Lon := 3.649992415$$

$$d\_max12 := [\,2.,\ 1.\,]$$

$$d\_max12\_meter\_Lat := .3704000000$$

$$d\_max12\_meter\_Lon := .1349000000$$

$$sx\_Lon\_meter := 1.657165845$$

$$vx\_Lon\_meter := 20.20749548$$

```
# KS-Test for the Latitude
YA:=sort(X[1]):
YB:=sort(X[2]):
NKA:=50;
SnA:=vector(NKA,0);
rel_SnA:=vector(NKA,0);
```

NKA := 50

SnA := [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

rel_SnA := [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

```
NKA := 50;
```

NKA := 50

```
for i from 1 to N1 do
iclass:=round((YA[i]-YA[1])*NKA/(YA[N1]-YA[1])-0.55)+1; #Class
number between 1..NKA
if iclass < 1 then iclass := 1 fi; #Class 1 is the lowest
SnA[iclass]:=SnA[iclass]+1; #Increase number of each class
od:

print(SnA);
```

[4, 60, 24, 16, 19, 0, 29, 58, 43, 52, 60, 0, 11, 10, 6, 9, 0, 8, 10, 8, 13, 21, 0, 30, 27, 34,
62, 65, 0, 62, 49, 26, 39, 0, 31, 49, 70, 25, 37, 0, 44, 27, 34, 29, 0, 24, 21, 16, 10, 2]

```
rel_SnA[1]:=evalf(SnA[1]/N1);
```

$rel\_SnA_1 := .003139717425$

```
for iclass from 2 to NKA do
SnA[iclass]:=SnA[iclass-1]+SnA[iclass]: #Sum up classes
rel_SnA[iclass]:=evalf(SnA[iclass]/N1);
od:
print(SnA);
print(rel_SnA);
```

```
[4, 64, 88, 104, 123, 123, 152, 210, 253, 305, 365, 365, 376, 386, 392, 401, 401, 409,
    419, 427, 440, 461, 461, 491, 518, 552, 614, 679, 679, 741, 790, 816, 855, 855, 886,
    935, 1005, 1030, 1067, 1067, 1111, 1138, 1172, 1201, 1201, 1225, 1246, 1262, 1272,
    1274]
```

```
[.003139717425, .05023547881, .06907378336, .08163265306, .09654631083,
    .09654631083, .1193092622, .1648351648, .1985871272, .2394034537,
    .2864992151, .2864992151, .2951334380, .3029827316, .3076923077, .3147566719,
    .3147566719, .3210361068, .3288854003, .3351648352, .3453689168, .3618524333,
    .3618524333, .3854003140, .4065934066, .4332810047, .4819466248, .5329670330,
    .5329670330, .5816326531, .6200941915, .6405023548, .6711145997, .6711145997,
    .6954474097, .7339089482, .7888540031, .8084772370, .8375196232, .8375196232,
    .8720565149, .8932496075, .9199372057, .9427001570, .9427001570, .9615384615,
    .9780219780, .9905808477, .9984301413, 1.]
```

```
dmaxA:=0;
```

$$dmaxA := 0$$

```
PhileftA[1]:=statevalf[cdf,normald[dx[1],sx[1]]](YA[1]);# Y-
coordinate of distribution at YA[1]
```

$$PhileftA_1 := .03433379239$$

```
ClasswidthA:=(YA[N1]-YA[1])/NKA;# Equidistant class width
```

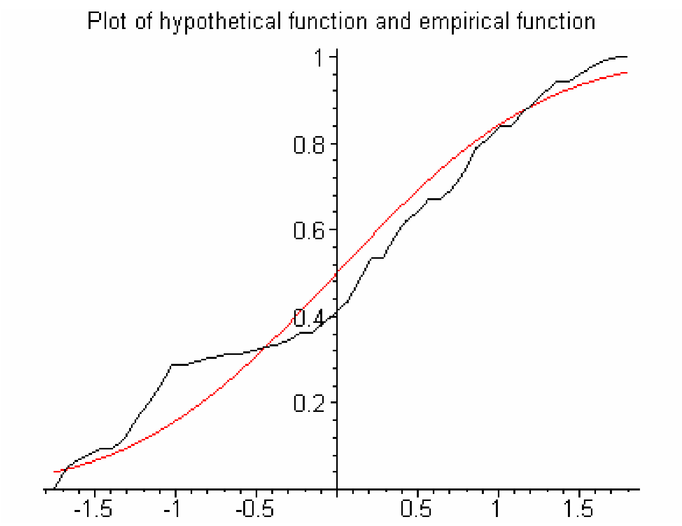
$$ClasswidthA := .8200000000$$

```
for iclass from 1 to NKA do
#define right side
xclassA:= iclass*ClasswidthA+YA[1];
ui[iclass]:=evalf((xclassA-dx[1])/sx[1]);
#Y-coordinate of distribution at xclassA
PhirightA[iclass]:=statevalf[cdf,normald[dx[1],sx[1]]](xclassA);
#Calculate vertical differences of the rel. cumulative frequency
of distribution at left side
di_lA:=abs(rel_SnA[iclass]-PhileftA[iclass]);
#Calculate vertical differences of the rel. cumulative frequency
of distribution at right side
di_rA:=abs(rel_SnA[iclass]-PhirightA[iclass]);
#Y-value of distribution function on the right becomes Y-value of
the left
PhileftA[iclass+1]:=PhirightA[iclass];
#Save biggest distance in dmax1
if (di_lA > dmaxA) then dmaxA:=di_lA; fi;
if (di_rA > dmaxA) then dmaxA:=di_rA; fi;
```

```
od:

printf("dmaxA: %f und TS=sqrt(N1)dmaxA: %f \n",dmaxA,
evalf(sqrt(N1)*dmaxA));
```

```
dmaxA: .149965 und TS=sqrt(N1)dmaxA: 5.352713
```

```
p1:=plot([seq([ui[k],rel_SnA[k]],k=1..NKA)], color=black):


p2:=plot([seq([ui[k],statevalf[cdf,normald](ui[k])],k=1..NKA)],col
or=red):

 display(p1,p2,title="Plot of hypothetical function and empirical
function");
```



```
TS:=sqrt(N1)*dmaxA;
```

$$TS := 1.049753328\,\sqrt{26}$$

```
evalf(%);
```

$$5.352712704$$

```
alpha=0.01;
```

$$\alpha = .01$$

Exhibit 2: Test results for moving test on 01/09/2004 with the GPS receiver Lat/Lon

```
restart;with(stats):with(describe):with(linalg):with(fit):with(plots):
```

```
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

```
A:=importdata("C:/GPSTrackTest01.txt",2):

Y:=[A][1],[A][2]:N1:=count(Y[1]);N2:=count(Y[2]);
```

$$N1 := 277$$

$$N2 := 277$$

```
z:=leastsquare[[x,y],y=b1*x+b0,{b1,b0}]([Y[1],Y[2]]);

gerfit:=unapply(rhs(z),x);b0:=7393.417540;
b1:=-.9105519629;o_leg:=-b0/b1;alpha:=arctan(o_leg,b0);alpha_degrees:=alpha*180/3.1415;

res:=[seq(Y[2][i]-gerfit(Y[1][i]),i=1..N1)]:res_meter:=[seq(res[i]*0.1852,i=1..N1)]:
dist_path:=[seq(res_meter[i]*sin(alpha),i=1..N1)]:
```

$$z := y = -.9105519629\,x + 7393.417540$$

$$gerfit := x \to -.9105519629\,x + 7393.417540$$

$$b0 := 7393.417540$$

$$b1 := -.9105519629$$

$$o\_leg := 8119.709628$$

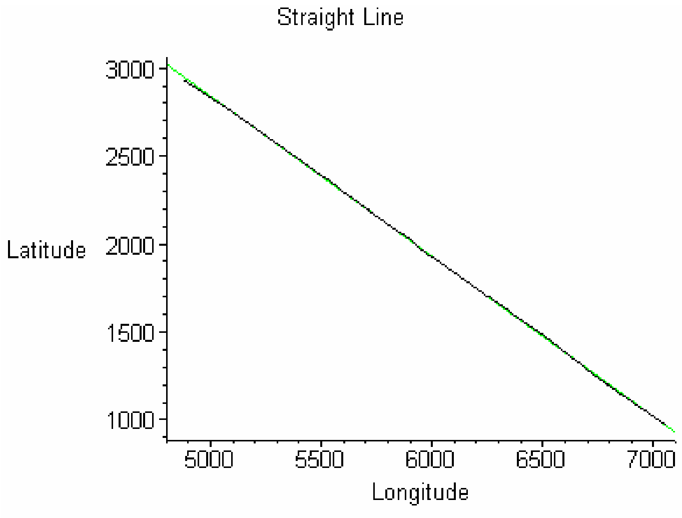
$$\alpha := .8321819047$$

$$alpha\_degrees := 47.68191719$$

```
b0 := 7393.417540;
```

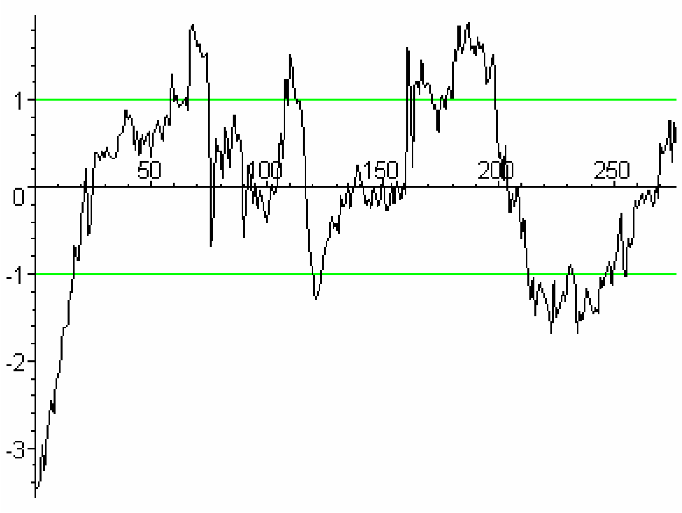
$$b0 := 7393.417540$$

```
p1:=plot([seq([Y[1][k],Y[2][k]],k=1..N1)],
color=black):p2:=plot(gerfit(i), i=4800..7100, color=green):
display(p1,p2,title="Straight Line",labels=[Longitude,Latitude]);
```



```
p5:=plot([1,-1],x=0..N1, color=green):
```

```
p3:=plot([seq([k,res_meter[k]],k=1..N1)],
color=black):display(p3,p5);
```



```
p4:=plot([seq([k,dist_path[k]],k=1..N1)],
color=black):display(p4,p5,title="Offset of the
track",labels=["time (sec)","Offset (meters)"]);
```

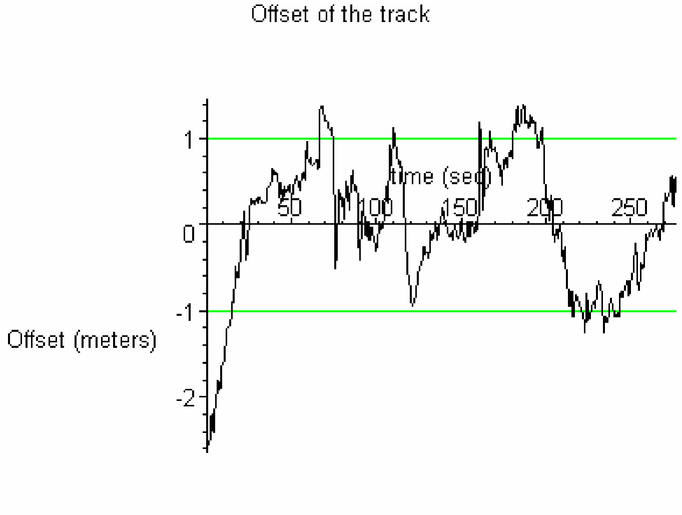

Offset of the track
Offset (meters)
time (sec)
1
0
-1
-2
50
100
150
200
250

Exhibit 3: Test results for moving test on 01/09/2004 with the GPS receiver Bearing

```
restart;with(stats):with(describe):with(linalg):with(fit):with(plots):
```

```
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

```
X:=[importdata("C:/GPSTrackTest10.txt",1)]:

N1:=count(X);dx:=mean(X);
```

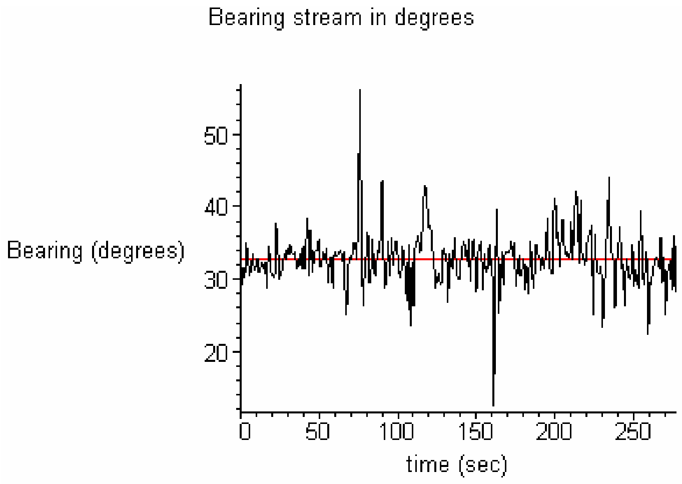
$$N1 := 277$$

$$dx := 32.79386282$$

```
sx:=standarddeviation[1](X);vx:=variance[1](X);
p:=plot([seq([k,X[k]],k=1..N1)], color=black):p3:=plot(dx,k=0..N1,
color=red):
display(p,p3,title="Bearing stream in degrees",labels=["time
(sec)","Bearing (degrees)"]);
```

$$sx := 3.820131443$$

$$vx := 14.59340424$$



```
for i from 14 to N1 do
Y:=[X[i],X[i-1],X[i-2],X[i-3],X[i-4],X[i-5],X[i-6],X[i-7],X[i-8],X[i-9],X[i-10],X[i-11],X[i-12],X[i-13]];Z[i]:=mean(Y);Z[i]:=Z[i]-dx;od:
```

```
d_max:=[max(seq(abs(Z[i]),i=14..N1))];
```

```
d_max := [3.47042289]
```

```
p1:=plot([seq([k,Z[k]],k=14..N1)],
color=black):display(p1,title="Difference between sample means and
overall mean over time",labels=["time (sec)","Degrees"]);
```

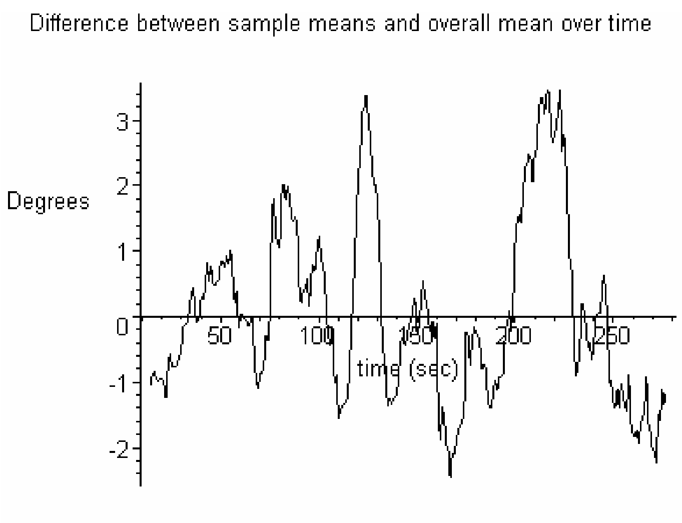


Exhibit 4: Test results for moving test on 01/28/2004 with the GPS receiver Lat/Lon

```
restart;with(stats):with(describe):with(linalg):with(fit):with(plo
ts):
```

```
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

```
A:=importdata("C:/GPSTrackTest05.txt",2):

Y:=[A][1],[A][2]:N1:=count(Y[1]);N2:=count(Y[2]);
```

$$N1 := 193$$

$$N2 := 193$$

```
z:=leastsquare[[x,y],y=b1*x+b0,{b1,b0}]([Y[1],Y[2]]);

gerfit:=unapply(rhs(z),x);b0:=7366.889027;
b1:=-.9065447880;o_leg:=-
b0/b1;alpha:=arctan(o_leg,b0);alpha_degrees:=alpha*180/3.1415;

res:=[seq(Y[2][i]-
gerfit(Y[1][i]),i=1..N1)]:res_meter:=[seq(res[i]*0.1852,i=1..N1)]:
dist_path:=[seq(res_meter[i]*sin(alpha),i=1..N1)]:
```

$$z := y = -.9065447880\, x + 7366.889027$$

$$gerfit := x \rightarrow -.9065447880\, x + 7366.889027$$

$$b0 := 7366.889027$$

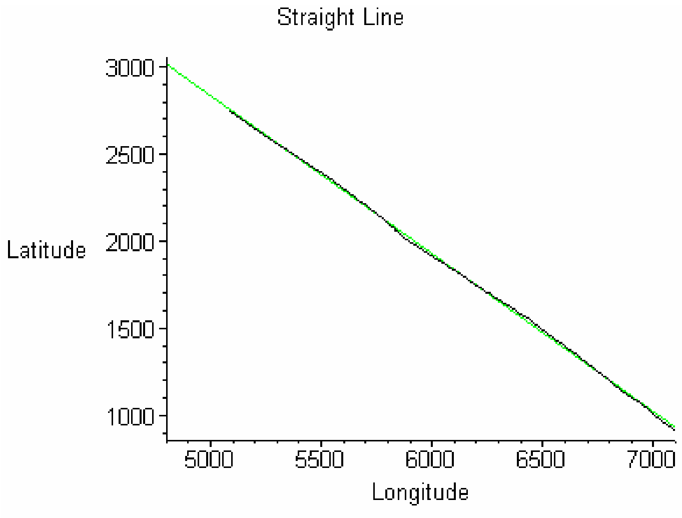
$$b1 := \text{-}.9065447880$$

$$o\_leg := 8126.337633$$

$$\alpha := .8343770647$$

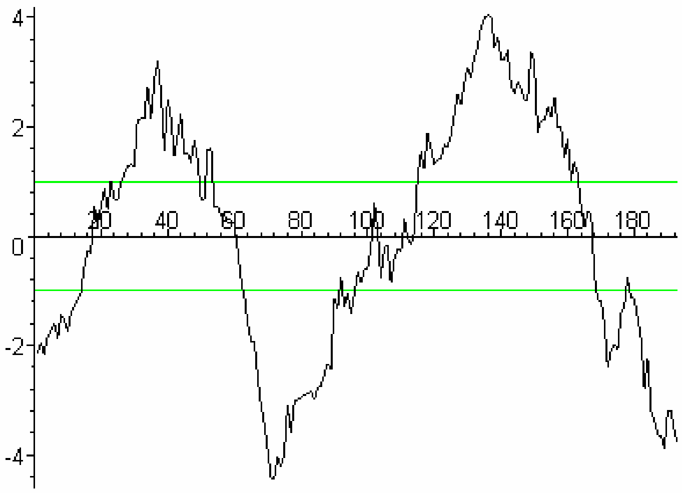
$$alpha\_degrees := 47.80769430$$

```
p1:=plot([seq([Y[1][k],Y[2][k]],k=1..N1)],
color=black):p2:=plot(gerfit(i), i=4800..7100, color=green):
display(p1,p2,title="Straight Line",labels=[Longitude,Latitude]);
```



```
p5:=plot([1,-1],x=0..N1, color=green):
```

```
p3:=plot([seq([k,res_meter[k]],k=1..N1)],
color=black):display(p3,p5);
```



```
p4:=plot([seq([k,dist_path[k]],k=1..N1)],
color=black):display(p4,p5,title="Offset of the
track",labels=["time (sec)","Offset (meters)"]);
```

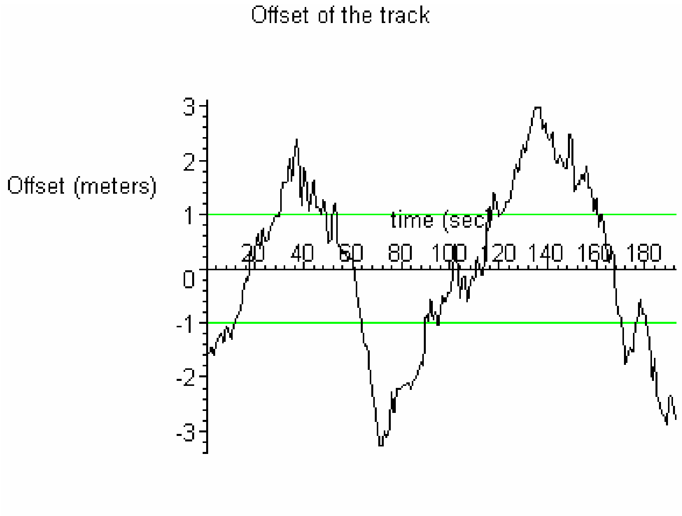

Offset of the track
Offset (meters)
time (sec)
3
2
1
0
-1
-2
-3
20
40
60
80
100
120
140
160
180

Exhibit 5: Test results for moving test on 01/28/2004 with the GPS receiver Bearing

```
restart;with(stats):with(describe):with(linalg):with(fit):with(plo
ts):
```

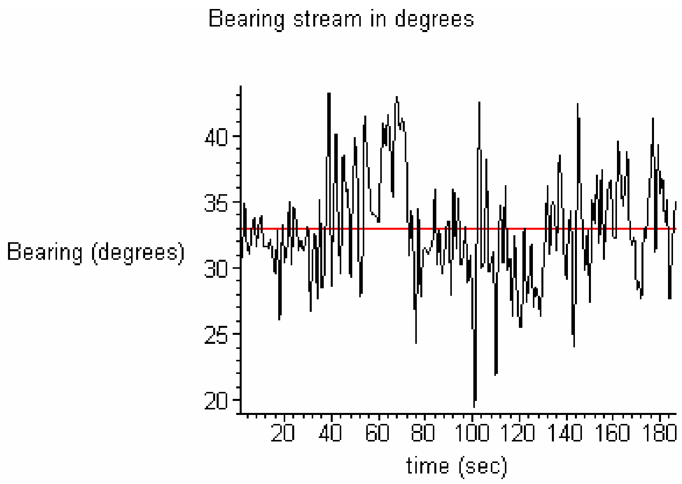
```
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

```
X:=[importdata("C:/GPSTrackTest11.txt",1)]:

N1:=count(X);dx:=mean(X);
```

```
N1 := 187
dx := 32.92459893
```

```
sx:=standarddeviation[1](X);vx:=variance[1](X);
p:=plot([seq([k,X[k]],k=1..N1)], color=black):p3:=plot(dx,k=1..N1,
color=red):
display(p,p3,title="Bearing stream in degrees",labels=["time
(sec)","Bearing (degrees)"]);
```

```
sx := 4.197020733
vx := 17.61498303
```



```
for i from 14 to N1 do
```

```
Y:=[X[i],X[i-1],X[i-2],X[i-3],X[i-4],X[i-5],X[i-6],X[i-7],X[i-
8],X[i-9],X[i-10],X[i-11],X[i-12],X[i-
13]];Z[i]:=mean(Y);Z[i]:=Z[i]-dx;od:
d_max:=[max(seq(abs(Z[i]),i=14..N1))];
```

$$d\_max := [\,6.13254393\,]$$

```
 p1:=plot([seq([k,Z[k]],k=14..N1)],
color=black):display(p1,title="Difference between sample means and
overall mean over time",labels=["time (sec)","Degrees"]);
```

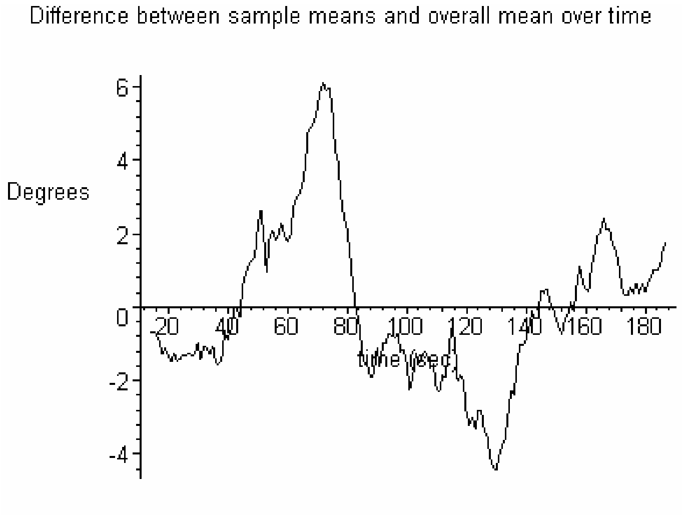


Exhibit 6: Test results for moving test on 01/29/2004 with the GPS receiver Lat/Lon

```
restart;with(stats):with(describe):with(linalg):with(fit):with(plo
ts):
```

```
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

```
A:=importdata("C:/GPSTrackTest15.txt",2):

Y:=[A][1],[A][2]:N1:=count(Y[1]);N2:=count(Y[2]);
```

$$N1 := 227$$

$$N2 := 227$$

```
z:=leastsquare[[x,y],y=b1*x+b0,{b1,b0}]([Y[1],Y[2]]);

gerfit:=unapply(rhs(z),x);b0:=7702.370208;
b1:=-.9582335610;o_leg:=-
b0/b1;alpha:=arctan(o_leg,b0);alpha_degrees:=alpha*180/3.1415;

res:=[seq(Y[2][i]-
gerfit(Y[1][i]),i=1..N1)]:res_meter:=[seq(res[i]*0.1852,i=1..N1)]:
dist_path:=[seq(res_meter[i]*sin(alpha),i=1..N1)]:
```

$$z := y = -.9582335610\ x + 7702.370208$$

$$gerfit := x \to -.9582335610\ x + 7702.370208$$

$$b0 := 7702.370208$$

$$b1 := \text{-}.9582335610$$

$$o\_leg := 8038.092717$$

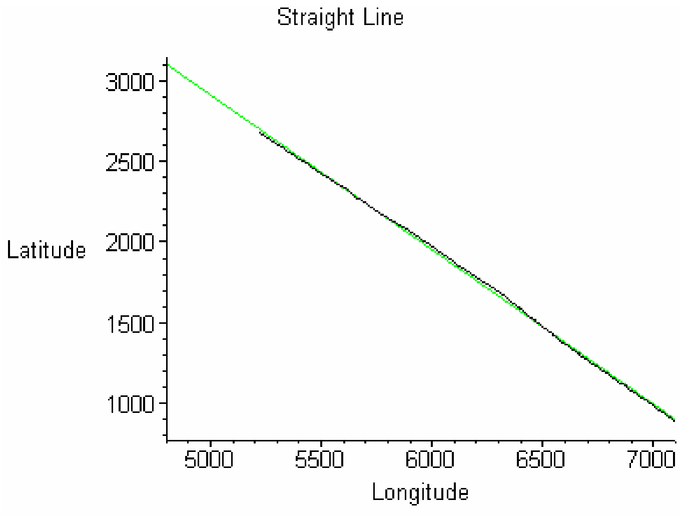
$$\alpha := .8067235600$$

$$alpha\_degrees := 46.22321846$$

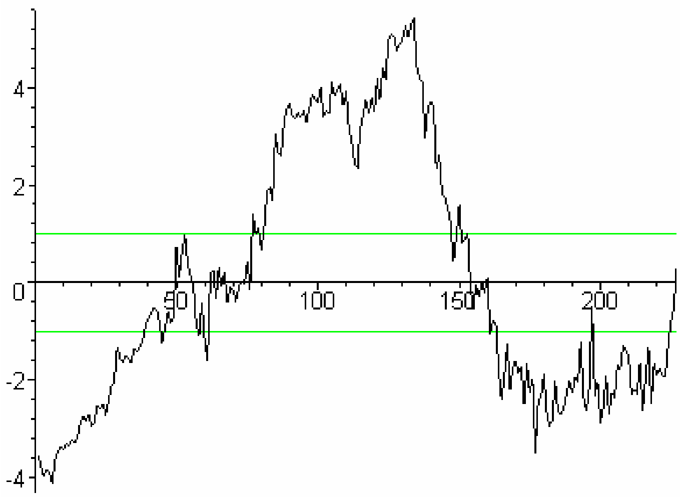
```
b0 := 7702.370208;
```

$$b0 := 7702.370208$$

```
p1:=plot([seq([Y[1][k],Y[2][k]],k=1..N1)],
color=black):p2:=plot(gerfit(i), i=4800..7100, color=green):
display(p1,p2,title="Straight Line",labels=[Longitude,Latitude]);
```



```
p5:=plot([1,-1],x=0..N1, color=green):
p3:=plot([seq([k,res_meter[k]],k=1..N1)],
color=black):display(p3,p5);
```



```
p4:=plot([seq([k,dist_path[k]],k=1..N1)],
color=black):display(p4,p5,title="Offset of the
track",labels=["time (sec)","Offset (meters)"]);
```

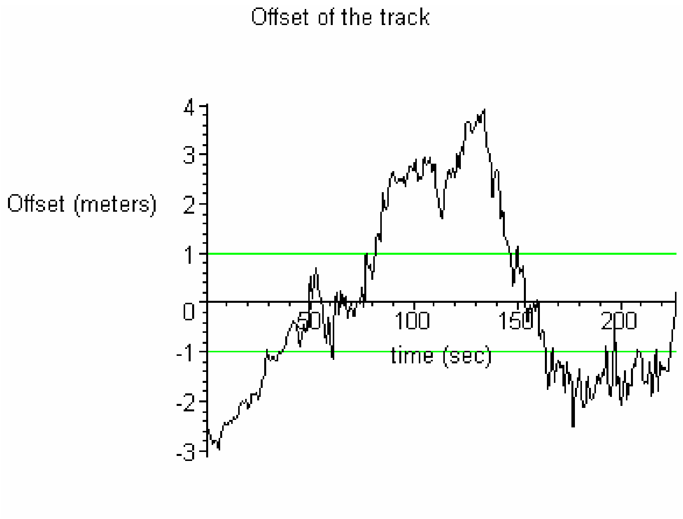

Offset of the track
Offset (meters)
4
3
2
1
0
-1
-2
-3
50
100
150
200
time (sec)

Exhibit 7: Test results for moving test on 01/292004 with the GPS receiver Bearing

```
restart;with(stats):with(describe):with(linalg):with(fit):with(plots):
```

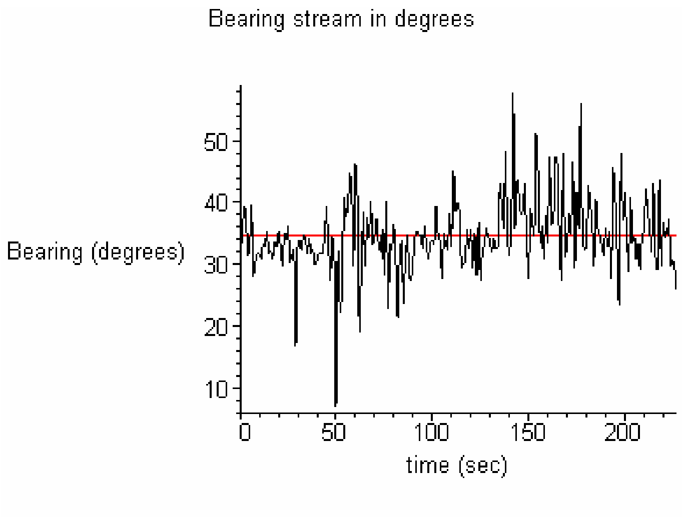
```
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

```
X:=[importdata("C:/GPSTrackTest16.txt",1)]:

N1:=count(X);dx:=mean(X);
```

```
N1 := 227

dx := 34.52819383
```

```
sx:=standarddeviation[1](X);vx:=variance[1](X);
p:=plot([seq([k,X[k]],k=1..N1)], color=black):p3:=plot(dx,k=0..N1, color=red):
display(p,p3,title="Bearing stream in degrees",labels=["time (sec)","Bearing (degrees)"]);
```

```
sx := 5.987058150

vx := 35.84486529
```

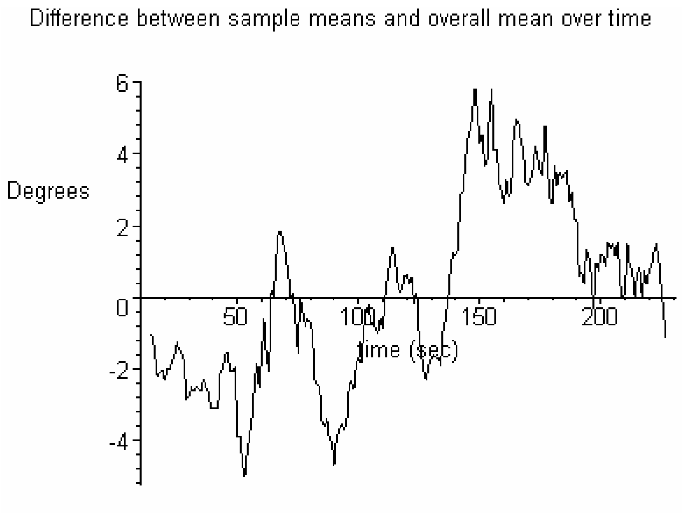


```
for i from 14 to N1 do
Y:=[X[i],X[i-1],X[i-2],X[i-3],X[i-4],X[i-5],X[i-6],X[i-7],X[i-8],X[i-9],X[i-10],X[i-11],X[i-12],X[i-13]];Z[i]:=mean(Y);Z[i]:=Z[i]-dx;od:
```

```
d_max:=[max(seq(abs(Z[i]),i=14..N1))];
```

```
d_max := [ 5.82894903 ]
```

```
p1:=plot([seq([k,Z[k]],k=14..N1)],
color=black):display(p1,title="Difference between sample means and
overall mean over time",labels=["time (sec)","Degrees"]);
```



2. Display of source code of presented functions in Dynamic C

Exhibit 8: Source code of *SPI_DAC_MAX 536.c* file

```
(1)  Gunnar Hamann 2004

(2)  This program uses serial port D to program a MAX536 quad DAC.
(3)  PB2 is the Chip Select.

(4)  The function SwapBytes is used to swap the order of the two
(5)  byte output value. This is necessary because the Rabbit and
(6)  Dynamic C are Little Endian - the LS byte is sent first. The
(7)  MAX 536 requires that the MS byte be transmitted first.
(8)  This function could be made more efficient; however, I decided
(9)  to make it easier to understand rather than have a more complex
(10)         formula.

(11)         */

(12)         #define SPI_SER_D   // Selection of the serial port
(13)         #define SPI_CLK_DIVISOR 10 // Set clockspeed
(14)         #use spi.lib

(15)         #define DACA 0x3000 // Definition of the output channels
(16)         #define DACB 0x7000
(17)         #define DACC 0xB000
(18)         #define DACD 0xF000

(19)         int SwapBytes ( int value );

(20)         main()
(21)         {
(22)         int i, j;

(23)         WrPortI ( PBDDR, &PBDDRShadow, 0xFF );       // PB = all output

(24)         SPIinit();
(25)         while (1)
(26)         {
                  i.  i=820;  // actual value in proportion to the RefVoltage (i/4096)*10 V REF

                          1.  j = SwapBytes(i|DACB);       // update DACB immediately with value
                              i
                          2.  BitWrPortI ( PBDR, &PBDRShadow, 0, 2 );          // CS = 0
                 ii.  SPIWrite( &j, 2 );
                          1.  BitWrPortI ( PBDR, &PBDRShadow, 1, 2 );          // CS = 1

                iii.  for ( i=0; i<5; i++ );            // delay
}
}

// reverse the byte order for sending to the MAX536
int SwapBytes ( int value )
{         int i0;
i0 = (value<<8) & 0xFF00;  // put low byte into high byte
i0 |= (value>>8) & 0x00FF; // put high byte into low byte
return i0;
}
```

Exhibit 9: Source code of SPI_ADC_MAX 186.c file

```
(1) /*
(2) SPI_A2D_MAX186.c
(3) Copyright Z-World Inc. Feb 2004
(4) Requested and first written by Gunnar Hamann 2004

(5) This program will set up and read a MAX186 12 bit A/D convertor which uses an SPI. It
    uses serial port B for testing in its synchronous mode and uses PB7 for the chip select.

(6) Connections between the RCM3000 prototype board and the MAX186:

(7) MAX186          RCM Jackrabbit
        1               input voltage to be measured <= 4volts
(8) 2..8            open
        9               ground
        10              +5
        11              4.7uf capacitor to ground
        12              open
        13              ground
        14              ground
        15              PC5
        16              open
        17              PC4
        18              PB7
        19              PB0
        20              +5

(9) */

(10)        #class auto

(11)        float ReadAD ( char *Command, int Samples );
(12)        int SwapBytes ( int i );

(13)        float ScaleFactor;

(14)        // SPI library definitions
(15)        #define SPI_SER_B
(16)        #define SPI_CLK_DIVISOR                 10
(17)        #define SPI_CLOCK_MODE 1
(18)        #define SPI_MASTER_CS_PORT              PBDR
(19)        #define SPI_MASTER_CS_SHADOW            PBDRShadow
(20)        #define SPI_MASTER_CS_BIT               7
(21)        #use SPI.LIB

(22)        // MAX 186 command byte definitions
(23)        #define MAX186_START        0x80
```

```
#define MAX186_CH0          0x00
#define MAX186_CH1          0x40
#define MAX186_CH2          0x10
#define MAX186_CH3          0x50
#define MAX186_CH4          0x20
#define MAX186_CH5          0x60
#define MAX186_CH6          0x30
#define MAX186_CH7          0x70

#define MAX186_UNIPOL 0x08
#define MAX186_BIPOL 0

#define MAX186_SINGLE 0x04
#define MAX186_DIFF 0

#define MAX186_MODE_PDFULL 0
#define MAX186_MODE_PDFAST 0x01
#define MAX186_MODE_ICLK 0x02
#define MAX186_MODE_ECLK 0x03


const char READ_CH0[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH0|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0 };

const char READ_CH1[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH1|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0 };

const char READ_CH2[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH2|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0 };

const char READ_CH3[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH3|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0 };

const char READ_CH4[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH4|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0 };

const char READ_CH5[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH5|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0 };
const char READ_CH6[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH6|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0 };

const char READ_CH7[] ={MAX186_START|MAX186_UNIPOL|MAX186_CH7|MAX186_SINGLE|MAX186_MODE_ICLK, 0, 0};

void main ()
{
int Value;
float volts;

// set up chip select port
BitWrPortI ( PBDR, &PBDRShadow, 1, 2 );          // PB2 = 1
BitWrPortI ( PBDDR, &PBDDRShadow, 1, 2 );        // PB2 = output

ScaleFactor = 4.096/4096.0; // 4.096 volts/4096 counts
SPIinit();

while (1)
{
      volts = ReadAD ( READ_CH7, 100 );          // average 100 readings
      printf ( "Value = %5.3f \r", volts );
}
}


/* ReadAD - this function will command the A/D to take a reading
on the selected channel. It will take the average of the
specified number of readings and convert the value to volts using the
predefined scale factor.

Input Parameter 1: address of command bytes - must be 3 bytes
Input parameter 2: (int) number of readings to average
Input Global: (float) ScaleFactor
Return value: (float) volts

In order to scale the returned value to a real voltage the result needs to
be multiplied by the scale factor.

*/

float ReadAD ( char *Command, int Samples )
{
int Count, j;
char data[3];
unsigned long i;
```

```
        float Voltage;

        i = 0L;                                                     // init accumulator
        for ( Count = 1; Count<=Samples; Count++ )
        //      for ( ;; )
        {
        BitWrPortI ( SPI_MASTER_CS_PORT, &SPI_MASTER_CS_SHADOW, 0,
    SPI_MASTER_CS_BIT );                                      // enable /CS
        SPIWrRd ( Command, &data, 3 );
        BitWrPortI ( SPI_MASTER_CS_PORT, &SPI_MASTER_CS_SHADOW, 1,
    SPI_MASTER_CS_BIT );                                      // disable /CS
        j = data[1]*256 + data[2];
        i += (j>>3);                                                // update accumulator
        for (j=0; j<100; j++);                                          // cheap time
    delay
        }

        Voltage = (float)i * ScaleFactor / (float)Samples;
        }
```

Exhibit 10: Source code of *gps_get_position* function from *GPS.lib* file

```
/*** BeginHeader */
#ifndef __GPS_LIB
#define __GPS_LIB
/*** EndHeader */

/* START LIBRARY DESCRIPTION
*********************************************
gps.lib
ZWorld, 2001
Gunnar Hamann, 2003

Functions for parsing NMEA-0183 location data from a GPS receiver.
        Also has functions for computing distances, bearings and to calculate
        base points:

        gps_get_position
        gps_get_utc
        gps_ground_distance
        gps_bearing
        gps_basing_point

        END DESCRIPTION
*******************************************************/

        /*** BeginHeader */

        //This structure holds geographical position as reported by a GPS receiver
        //use the gps_get_position function below to set the fields

        typedef struct {
                int lat_degrees;
                int lon_degrees;
                float lat_minutes;
                float lon_minutes;
                char lat_direction;
                char lon_direction;
                char sog;                       //speed over ground
                char tog;               //track over ground
        } GPSPosition;

        //in km

        #define GPS_EARTH_RADIUS 6356           // in km
        #define dbp 0.25                        // distance to next bp in km
```

```
/*** EndHeader */

/*** BeginHeader gps_parse_coordinate */

int gps_parse_coordinate(char *coord, int *degrees, float *minutes);

/*** EndHeader */

//helper function for splitting xxxxx.xxxx into degrees and minutes
//returns 0 if succeeded

nodebug int gps_parse_coordinate(char *coord, int *degrees, float *minutes)
{
auto char *decimal_point;
auto char temp;
auto char *dummy;

decimal_point = strchr(coord, '.');
if(decimal_point == NULL)
        i.   return -1;
temp = *(decimal_point - 2);
*(decimal_point - 2) = 0; //temporary terminator
*degrees = atoi(coord);
*(decimal_point - 2) = temp; //reinstate character
*minutes = strtod(decimal_point - 2, &dummy);
return 0;
}

/*** BeginHeader gps_get_position */

int gps_get_position(GPSPosition *newpos, char *sentence);

/*** EndHeader */

/* START FUNCTION DESCRIPTION
*******************************************
gps_get_position                    <gps.lib>

SYNTAX:                    int gps_get_position(GPSPositon *newpos, char
*sentence);

KEYWORDS:                  gps

DESCRIPTION:               Parses a sentence to extract position data.
                     a.  This function is able to parse any of the following
                     b.  GPS sentence formats: GGA, GLL, RMC
```

```
(54)      PARAMETER1:              newpos - a GPSPosition structure to fill
(55)      PARAMETER2:              sentence - a string containing a line of GPS data in
    NMEA-0183 format

(56)      RETURN VALUE:            0 - success
                              a.  -1 - parsing error
                              b.  -2 - sentence marked invalid

(57)      SEE ALSO:

(58)      END DESCRIPTION
    *******************************************************/

(59)      //can parse GGA, GLL, or RMC sentence
(60)      nodebug int gps_get_position(GPSPosition *newpos, char *sentence)
(61)      {
(62)      auto int i;

(63)      if(strlen(sentence) < 4)
               i.  return -1;
(64)      if(strncmp(sentence, "$GPGGA", 6) == 0)
(65)      {
               i.  //parse GGA sentence
              ii.  for(i = 0;i < 11;i++)
             iii.  {
                       1.  sentence = strchr(sentence, ',');
                       2.  if(sentence == NULL)
                               a.  return -1;
                       3.  sentence++; //first character in field
                       4.  //pull out data
                       5.  if(i == 1) //latitude
                       6.  {
                               a.  if( gps_parse_coordinate(sentence, &newpos->lat_degrees,
                                   &newpos->lat_minutes)
                               b.  )
                               c.  {
                                       i.  return -1; //get_coordinate failed
                               d.  }
                       7.  }
                       8.  if(i == 2) //lat direction
                       9.  {
                               a.  newpos->lat_direction = *sentence;
                       10. }
                       11. if(i == 3) // longitude
                       12. {
                               a.  if( gps_parse_coordinate(sentence, &newpos->lon_degrees,
                                                         &newpos->lon_minutes)
                               b.  )
```

```
                    c.  {
                            i.  return -1; //get_coordinate failed
                    d.  }
                13. }
                14. if(i == 4) //lon direction
                15. {
                    a.  newpos->lon_direction = *sentence;
                16. }
                17. if(i == 5) //link quality
                18. {
                    a.  if(*sentence == '0')
                            i.  return -2;
                19. }
          iv.  }
(66)      }
(67)      else if(strncmp(sentence, "$GPGLL", 6) == 0)
(68)      {
           i.  //parse GLL sentence
          ii.  for(i = 0;i < 6;i++)
         iii.  {
                1.  sentence = strchr(sentence, ',');
                2.  if(sentence == NULL)
                3.  {
                    a.  //handle short GLL sentences from Garmin receivers
                    b.  if(i > 3) break;
                    c.  return -1;
                4.  }
                5.  sentence++; //first character in field
                6.  //pull out data
                7.  if(i == 0) //latitude
                8.  {
                    a.  if( gps_parse_coordinate(sentence, &newpos->lat_degrees,
                                                 &newpos->lat_minutes)
                    b.  )
                    c.  {
                            i.  return -1; //get_coordinate failed
                    d.  }
                9.  }
                10. if(i == 1) //lat direction
                11. {
                    a.  newpos->lat_direction = *sentence;
                12. }
                13. if(i == 2) // longitude
                14. {
                    a.  if( gps_parse_coordinate(sentence, &newpos->lon_degrees,
                                                 &newpos->lon_minutes)
                    b.  )
                    c.  {
```

```
                              i.   return -1; //get_coordinate failed
                         d.   }
                    15. }
                    16. if(i == 3) //lon direction
                    17. {
                         a.   newpos->lon_direction = *sentence;
                    18. }
                    19. if(i == 5) //link quality
                    20. {
                         a.   if(*sentence != 'A')
                              i.   return -2;
                    21. }
               iv.  }
(69)      }
(70)      else if(strncmp(sentence, "$GPRMC", 6) == 0)
(71)      {
               i.   //parse RMC sentence
               ii.  for(i = 0;i < 11;i++)
               iii. {
                    1.   sentence = strchr(sentence, ',');
                    2.   if(sentence == NULL)
                         a.   return -1;
                    3.   sentence++; //first character in field
                    4.   //pull out data
                    5.   if(i == 1) //link quality
                    6.   {
                         a.   if(*sentence != 'A')
                              i.   return -2;
                    7.   }
                    8.   if(i == 2) //latitude
                    9.   {
                         a.   if( gps_parse_coordinate(sentence, &newpos->lat_degrees,
                                                            &newpos->lat_minutes)
                         b.   )
                         c.   {
                              i.   return -1; //get_coordinate failed
                         d.   }
                    10. }
                    11. if(i == 3) //lat direction
                    12. {
                         a.   newpos->lat_direction = *sentence;
                    13. }
                    14. if(i == 4) // longitude
                    15. {
                         a.   if( gps_parse_coordinate(sentence, &newpos->lon_degrees,
                                                            &newpos->lon_minutes)
                         b.   )
                         c.   {
```

```
                                    i.   return -1; //get_coordinate failed
                              d.   }
                    16. }
                    17. if(i == 5) //lon direction
                    18. {
                              a.   newpos->lon_direction = *sentence;
                    19. }
(72)      if(i == 6) //speed over ground, knots
                    1.   {
                              a.   newpos->sog = *sentence;
                    2.   }
(73)      if(i == 7) //track over ground, degrees true
                    1.   {
                              a.   newpos->tog = *sentence;
                    2.   }
(74)      }
(75)      }
(76)      else
(77)      {
               i.   return -1; //unknown sentence type
(78)      }
(79)      return 0;
(80)      }
```

Exhibit 11: Source code of *gps_ground_distance* function from *GPS.lib* file

```
/*** BeginHeader gps_ground_distance */
float gps_ground_distance(GPSPosition *a, GPSPosition *b);
/*** EndHeader */

/* START FUNCTION DESCRIPTION
 *******************************************
gps_ground_distance            <gps.lib>

SYNTAX:            float gps_ground_distance(GPSPosition *a, GPSPosition *b);

KEYWORDS:          gps

DESCRIPTION:       Calculates ground distance(in km) between two geographical points.
 (Uses spherical earth model)

PARAMETER1:        a - first point
        PARAMETER2:        b - second point

        RETURN VALUE:      distance in kilometers

        SEE ALSO:

        END DESCRIPTION
 ********************************************************/

        nodebug float gps_ground_distance(GPSPosition *a, GPSPosition *b)
        {
        float angle;
        float lat_a, lon_a, lat_b, lon_b;

        lat_a = a->lat_degrees + a->lat_minutes/60;
        if(a->lat_direction == 'S')
                i.  lat_a = -lat_a;
        lat_a = lat_a * PI / 180;
        lon_a = a->lon_degrees + a->lon_minutes/60;
        if(a->lon_direction == 'W')
                i.  lon_a = -lon_a;
        lon_a = lon_a * PI / 180;

        lat_b = b->lat_degrees + b->lat_minutes/60;
        if(b->lat_direction == 'S')
                i.  lat_b = -lat_b;
        lat_b = lat_b * PI / 180;
        lon_b = b->lon_degrees + b->lon_minutes/60;
        if(b->lon_direction == 'W')
```

```
            lon_b = -lon_b;
        lon_b = lon_b * PI / 180;

        angle = sqrt(cos(lat_a)*cos(lat_b)*pow(sin((lon_a - lon_b)/2), 2) +pow(sin((lat_a - lat_b)/2), 2));
        angle = 2*asin(angle);

        return angle * GPS_EARTH_RADIUS;
        }
```

Exhibit 12: Source code of *gps_bearing* function from *GPS.lib* file

```
/*** BeginHeader gps_bearing */
float gps_bearing(GPSPosition *c, GPSPosition *d, float dist);
/*** EndHeader */

/* START FUNCTION DESCRIPTION
 ********************************************
gps_bearing                 <gps.lib>

SYNTAX:                 float gps_bearing(GPSPosition *a, GPSPosition *b, dist);

KEYWORDS:               gps

DESCRIPTION:            Calculates bearing(in degree) from one geographical point a to a
 geographical point b. (Uses spherical earth model)

PARAMETER1:             a - first point
        PARAMETER2:             b - second point
        PARAMETER3:             dist - ground distance between the two points a b

        RETURN VALUE:       bearing in degrees

        SEE ALSO:

        END DESCRIPTION
 *********************************************************/

        nodebug float gps_bearing(GPSPosition *c, GPSPosition *d, float dist)
        {
        float bearing;
        float lat_c, lon_c, lat_d, lon_d;

        lat_c = c->lat_degrees + c->lat_minutes/60;
        if(c->lat_direction == 'S')
                i.  lat_c = -lat_c;
        lat_c = lat_c * PI / 180;
        lon_c = c->lon_degrees + c->lon_minutes/60;
        if(c->lon_direction == 'E')
                i.  lon_c = -lon_c;
        lon_c = lon_c * PI / 180;
        lat_d = d->lat_degrees + d->lat_minutes/60;
        if(d->lat_direction == 'S')
                i.  lat_d = -lat_d;
        lat_d = lat_d * PI / 180;
        lon_d = d->lon_degrees + d->lon_minutes/60;
        if(d->lon_direction == 'E')
```

```
            i.  lon_d = -lon_d;
        lon_d = lon_d * PI / 180;

    a.  if (cos(lat_c) < 0.0001)  // Small number
        if (lat_c > 0)
        bearing = 180;           // Starting from N pole
        else bearing = 360;  // Starting from S pole

        dist = dist / GPS_EARTH_RADIUS; // Convert distance to radian

    a.  if (sin(lon_d - lon_c) < 0)   // Calculation of bearing
            i.  bearing = acos((sin(lat_d)-sin(lat_c)*cos(dist))/(sin(dist)*cos(lat_c)));
           ii.  else
          iii.  bearing = 2*PI-acos((sin(lat_d)-
                sin(lat_c)*cos(dist))/(sin(dist)*cos(lat_c)));

        return bearing * (180 / PI);
        }
```

Exhibit 13: Source code of *gps_basing_point* function from *GPS.lib* file

```
(1) /*** BeginHeader gps_basing_point */
(2) gps_basing_point(GPSPosition *c, GPSPosition *bp, float tc1);
(3) /*** EndHeader */

(4) /* START FUNCTION DESCRIPTION
    ******************************************
(5) gps_bearing                <gps.lib>

(6) SYNTAX:                   float gps_bearing(GPSPosition *c, GPSPosition *bp, tc1);

(7) KEYWORDS:                 gps

(8) DESCRIPTION:              Calculates lat and lon of a base_point bp at a certain distance dbp
    from starting point c with the initial bearing tc1

(9) PARAMETER1:               c - starting point
(10)        PARAMETER2:       bp - base_point
(11)        PARAMETER3:       tc1 in true degrees
(12)        RETURN VALUE:

(13)        SEE ALSO:

(14)        END DESCRIPTION
    *******************************************************/

(15)        nodebug gps_basing_point(GPSPosition *c, GPSPosition *bp, float tc1)
(16)        {
(17)        float lat_c, lon_c, lat_d, lon_d, dist, dummy2;
        a.  int dummy;

(18)        lat_c = c->lat_degrees + c->lat_minutes/60;  // transform lat and lon to radians
(19)        if(c->lat_direction == 'S')
(20)        lat_c = -lat_c;
(21)        lat_c = lat_c * PI / 180;
(22)        lon_c = c->lon_degrees + c->lon_minutes/60;
(23)        if(c->lon_direction == 'E')
(24)        lon_c = -lon_c;
(25)        lon_c = lon_c * PI / 180;

(26)        tc1 = tc1 * (PI / 180);      // algorithm for base point lon and lat
(27)        dist = dbp / GPS_EARTH_RADIUS;

(28)        lat_d = asin(sin(lat_c) * cos(dist) + cos(lat_c) * sin(dist) * cos(tc1));
        a.  if (cos(lat_d) == 0)
                i.  lon_d = lon_c;  // endpoint a pole
               ii.  else
```

```
                  iii.  lon_d = (lon_c - asin(sin(tc1) * sin(dist) / cos(lat_d)) + PI);
                   iv.  dummy = (int) ((lon_d) / (2 * PI));
                    v.  dummy2 = (float) dummy * (2 * PI);
                   vi.  lon_d = lon_d - dummy2;
                  vii.  lon_d = lon_d - PI;

(29)        lat_d = lat_d * 180 / PI;      // assign lat value to base_point stucture
(30)        if(lat_d < 0)
(31)        bp->lat_direction = 'S';
(32)        else bp->lat_direction = 'N';
(33)        dummy = (int) lat_d;
(34)        bp->lat_degrees = dummy;
(35)        lat_d = (lat_d - (float)dummy) * 60;
(36)        bp->lat_minutes = lat_d;


(37)        lon_d = lon_d * 180 / PI;      // assign lon value to base_point stucture
(38)        if(lon_d < 0)
(39)        bp->lon_direction = 'W';
(40)        else bp->lon_direction = 'E';
        a.  dummy = (int) lon_d;
(41)        bp->lon_degrees = dummy;
(42)        lon_d = (lon_d - (float)dummy) * 60;
(43)        bp->lon_minutes = lon_d;

(44)        }
```

Exhibit 14: Source code of *TimeDelay.c* file

```
(1) // TimDly.c

(2) Gunnar Hamann 2004
(3) Z-World 2004

(4) /***************************************************************************
    ******
(5) fnMsDelay - delay some number of milliseconds

(6) input parameter:      int number of milliseconds to delay

(7) return value:         none

(8) errors:               none

(9) Note: this function does NOT check for MS_TIMER overflow
(10)        *********************************************************************
    ***********/
(11)        void fnMsDelay ( unsigned int iDelay )
(12)        {       auto unsigned long StopTime;
(13)        StopTime = MS_TIMER + (unsigned long)iDelay;
(14)        while ( (MS_TIMER - StopTime) < 0L );
(15)        }

(16)        // program to test fnMsDelay

(17)        main () {
(18)        WrPortI ( SPCR, NULL, 0X84 );             // port A = output
(19)        while (1)
(20)        {       WrPortI ( PADR, NULL, 0 );
                 i.  fnMsDelay( 10 );
                ii.  WrPortI ( PADR, NULL, 1 );
               iii.  fnMsDelay( 5 );
(21)        }
(22)        }
```

Exhibit 15: Source code of *GPS.c* file

```
/***
gps_test.c
Z-World, 2001
Gunnar Hamann, 2003

Sample program using GPS.LIB to retrieve time and position from the Motorola
GPS receiver.

The GPS receiver operating in standard NMEA-1803 mode (4800 baud, 8N1) must be
connected to serial port C.

The program will first demonstrate the use of the gps_ground_distance(),
gps_bearing() and gps_basing_point() function by calculating with two sample
points. The gps_ground_distance() function gives back the distance in km. The
gps_bearing() function takes the distance and gives back the bearing in degrees.
The gps_basing_point() function takes the starting point and the bearing and
calculates the Lat and Lon of a defined distance to the starting point (dbp). The
distance to the base point is defined in the GPS library and can be adjusted
any time.
Subsequently it calculates the bearing between to sample points and the offset
off the track of the second sample point.


The program will then start to listen for GPS data on port C. Any sentences
received will be parsed with both gps_get_position() and gps_get_utc().
When either of these functions parses valid data, the results are printed
to the STDIO window. The Motorola receiver is currently set to parse the
RMC (Recommended Minimum Navigation Information) which contents the time, the
position in Lat, Lon, the speed over ground, degree true and the date.

****/
#class auto

#use "gps.lib"

#define CINBUFSIZE 127
#define COUTBUFSIZE 127

#define MAX_SENTENCE 100

// names of days of week
const char dayname[7][4] = {"Sun","Mon","Tue","Wed","Thu","Fri","Sat"};
const char monthname[12][4] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun",
"Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
```

```
GPSPosition current_pos;
GPSPosition basing_point;

struct tm current_time;

void main()
{
char sentence[MAX_SENTENCE];
int input_char;
int string_pos;
char dir_string[2];
float distance, distance0, distance2, bearing, tc1, b2bp;
float alpha, basing_p, offset;

//calculate distance from known coordinates of two sample positions
GPSPosition starting_p, end_p, way_p, way_p0;

// all samples taken from straight line test on Jan 09 2004

// sample initial waypoint

way_p0.lat_degrees = 43;
way_p0.lat_minutes = 42.4877;
way_p0.lat_direction = 'N';
way_p0.lon_degrees = 72;
way_p0.lon_minutes = 17.2934;
way_p0.lon_direction = 'W';

// sample first point
starting_p.lat_degrees = 43;
starting_p.lat_minutes = 42.4928;
starting_p.lat_direction = 'N';
starting_p.lon_degrees = 72;
starting_p.lon_minutes = 17.2893;
starting_p.lon_direction = 'W';

// sample second point

end_p.lat_degrees = 43;
end_p.lat_minutes = 42.5033;
end_p.lat_direction = 'N';
end_p.lon_degrees = 72;
end_p.lon_minutes = 17.2807;
end_p.lon_direction = 'W';

// sample waypoint
```

```
way_p.lat_degrees = 43;
way_p.lat_minutes = 42.5605;
way_p.lat_direction = 'N';
way_p.lon_degrees = 72;
way_p.lon_minutes = 17.2288;
way_p.lon_direction = 'W';


serCopen(4800);  // open serial port C
string_pos = 0;
dir_string[1] = 0;


// calculation of initial bearing between waypoints tc1


distance0 = gps_ground_distance(&way_p0, &way_p); // calculate ground dist

printf("Way_p0 to Way_p: %f km\n", distance0);

tc1 = gps_bearing(&way_p0, &way_p, distance0);  // calculate bearing

printf("Initial bearing tc1 in true degrees: %f degrees\n", tc1);



// sample calculation of base point

gps_basing_point(&end_p, &basing_point, bearing); // calculate Lat Lon of bp

printf("Sample Basing Point Latitude: %d %f' %c\n",
                                    i.  basing_point.lat_degrees, basing_point.lat_minutes,
                                   ii.  basing_point.lat_direction);

printf("Sample Basing Point Longitude: %d %f' %c\n",
                                    i.  basing_point.lon_degrees, basing_point.lon_minutes,
                                   ii.  basing_point.lon_direction);


// bearing between two points of ten seconds distance

distance = gps_ground_distance(&starting_p, &end_p); // calculate ground dist

printf("Distance from sample Starting_Point to End_Point: %f km\n", distance);

bearing = gps_bearing(&starting_p, &end_p, distance);  // calculate bearing

printf("Bearing between sample points in true degrees: %f degrees\n", bearing);
```

```
// Calculation of offset from track

distance2 = gps_ground_distance(&end_p, &way_p); // calculate ground dist

printf("Distance from End_Point to Way_P: %f km\n", distance2);

b2bp = gps_bearing(&end_p, &way_p, distance2);  // calculate bearing

printf("Bearing in true degrees to Way_P: %f degrees\n", b2bp);

if (b2bp > tc1)
{
        alpha = b2bp-tc1;
}
else if (b2bp < tc1)
{
alpha = tc1-b2bp;
}
alpha = alpha * PI / 180;
offset = sin(alpha) * distance2;

printf("Offset from initial track: %f km\n", offset);

//receive and parse GPS data
while(1)
{
        input_char = serCgetc();
        if(input_char == '\r' || input_char == '\n')
        {
                sentence[string_pos] = 0; //add null
                printf("%s\n", sentence);
                if(gps_get_position(&current_pos, sentence) == 0)
                {
                        dir_string[0] = current_pos.lat_direction;
                        printf("Latitude: %d %f' %s\n",
                                current_pos.lat_degrees, current_pos.lat_minutes,
                                dir_string);
                        dir_string[0] = current_pos.lon_direction;
                        printf("Longitude: %d %f' %s\n",
                                current_pos.lon_degrees, current_pos.lon_minutes,
                                dir_string);

                }
                if(gps_get_utc(&current_time, sentence) == 0)
                {
                        printf("UTC: %s %d-%s-%d %02d:%02d:%02d\n",
                              i.   dayname[current_time.tm_wday],
                             ii.   current_time.tm_mday,
                            iii.   monthname[current_time.tm_mon - 1],
                             iv.   1900 + current_time.tm_year,
                              v.   current_time.tm_hour,
                             vi.   current_time.tm_min,
                            vii.   current_time.tm_sec );
                    8.   }
                    9.   string_pos = 0;
            iv.   }
             v.   else if(input_char > 0)
            vi.   {
                    1.   sentence[string_pos] = input_char;
                    2.   string_pos++;
                    3.   if(string_pos == MAX_SENTENCE)
                           a.   string_pos = 0;        //reset string if too large
           vii.   }
(112)      }

(113)      }
```

3. Contact information

Jackrabbit Controller:

Larry Chicchinelli, Applications Manager

Technical Support ID 302179

Z-World
2900 Spafford Street
Davis, California 95616

Phone: 1.530.757.3737
E-Mail: lcicchinelli@zworld.com
Web: www.zworld.com

Motorola GPS receiver :

Shawn Brennan, Applications Manager
E-Mail : shawn@synergy-gps.com

Randy Warner, Senior Applications Manager
E-Mail: randy@synergy-gps.com

Synergy Systems, LLC
PO Box 262250
San Diego, CA 92196

Phone: 1.858.566.0666
Web: www.synergy-gps.com

NAL Research Corporation Iridium Modem:

Trent Duewer, Applications Manager
E-Mail: tad@nalresearch.com

Ngoc Hoang, Applications Manager
E-Mail: nth@nalresearch.com

NAL Research Corporation
8798 Sudley Road
Manassas, VA 20110

Phone: 1.703.392.5386 x200
Web: www.nalresearch.com

Billingsley Magnetometer:

Eva Billingsley, President
E-Mail: eva@magnetometer.com

Billingsley Aerospace & Defense
20936 Theseus Terrace
Germantown, Maryland 20876 U.S.A.

Phone: 1.301.540.8338
Web: www.magnetometer.com

Maxim IC DAC and ADC converter:

Maxim Integrated Products, Inc.
120 San Gabriel Drive
Sunnyvale, CA 94086 USA

Phone: 1.408.737.7600
Web: www.maxim-ic.com