Proceedings 2005

Invitation

Field Robot Event

3rd edition

Bring your own prototype

June 16/17, 2005

Wageningen, The Netherlands

Workshops

Party

Prototype contest

www.fieldrobot.nl

WAGENINGEN UNIVERSITY

WAGENINGEN UR

WAGENINGEN UNIVERSITY

WAGENINGEN UR

Proceedings of the 3rd Field Robot Event 2005

Wageningen, June 16 & 17, 2005

ISBN	:	90-6754-969-X
Date	:	November 2005
Publisher	:	Farm Technology Group
Address	:	Technotron (Building 118) Bornsesteeg 59 6708 PD Wageningen The Netherlands Tel: +31 317 482980 Fax: +31 317 484819
Contact	:	info@fieldrobot.com

Sponsored by:

Wageningen UR
Postbus 9101
6700 HB Wageningen
The Netherlands
Tel: +31 (0)317 489 111
Fax: +31 (0)317 483 999

AMAZONEN-Werke
H. Dreyer GmbH & Co. KG
Am Amazonenwerk 9-13 - D-49205
Hasbergen - Germany
Tel: +49 (0)5405 501-0 –
Fax: +49 (0)5405 501-147

Agrotechnology and Food Innovations
Bornsesteeg 59
6708 PD Wageningen
The Netherlands
Tel: + 31 (0)317 - 475029
Fax: + 31 (0)317 - 475347

National Instruments
Postbus 124
3440 AC Woerden
The Netherlands
Tel.: +31 348 - 433466
Fax: +31 348 - 430673

Plant Research International
P.O. Box 16
6700 AA Wageningen
The Netherlands
Tel: +31 317 47 70 01
Fax: +31 317 41 80 94

Rockingstone IT
Bergstraat 18a
6701 AD Wageningen
The Netherlands
Tel: +31 317 425550
Fax: +31 317 412783

PREFACE

In summer 2003, when the 1st Field Robot Event was “born” at Wageningen University, it has been an experiment to combine the “serious” and “playful” aspects of robotics to inspire the upcoming student generation. Specific objectives have been:

- Employing students creativity to promote the development of field robots
- Promoting off-curriculum skills like communication, teamwork, time management and fundraising
- Attracting public interest for Agricultural Engineering
- Creating a platform for students and experts to exchange knowledge on field robots

Driven by the success of the 2nd Field Robot Event in 2004 Wageningen University organised in June 2005 the 3rd Field Robot Event. This event was accompanied by a workshop about robots and a fair where the teams presented their robots. The teams also had to write a paper describing the hard- and software design of their robot. The papers collected in this *Proceedings of the 3rd Field Robot Event* are a very valuable source of information. This edition of the proceedings ensures that the achievements of the participants are now documented as a publication and thus being accessible as basis for further research. Moreover, for most of the student team members it is the first (scientific) publication in their career - a well-deserved additional reward!

Wageningen, September 2005

Jan Willem Hofstee,
Chairman 3rd Field Robot Event 2005

INDEX

The field robot Cornhoolio

Design and constructional overview

Daan Blaauw, Wouter van Gulik, Jorn Kommer, Dennis de Koning

Hogeschool van Amsterdam, The Netherlands
Amsterdamse Hogeschool voor Techniek
E-Technology

Internet: http://www.rttc.tk
Contact: dennis.de.koning@hva.nl

27-06-2005

Abstract

Cornhoolio is an autonomous vehicle, developed for the Field Robot Event 2005, and designed for navigating through a field of maize and counting the plants along the way. A technical overview of the design of the robot and the technology that is used is presented in this document.

Keywords

Autonomous agricultural robot, navigation systems, sensor technique, maize.

1 Introduction

Field Robot Event, organised by Wageningen University, The Netherlands, is an event in which international and interdisciplinary teams compete by building a field robot. In 2005, the objective of the robot was to move autonomously through a field of maize and to count the plants along the way. Moving through the field includes navigating through straight and curved rows of maize and turning at the end of a row. A freestyle session was also part of the competition.

Cornhoolio is the robot developed by the Riders Through The Corn, a team that consists of four E-Technology students from the Amsterdam School for Higher Education. Design, building and testing of all the hard- and software was done by the team. A chassis was obtained by sponsorship.

The main objective of the Field Robot Event, from the team point of view, has been learning to cooperate as a team and to work together on technical solutions for complex problems. Real problems, because agricultural robots are thought to make an important contribution to the future of farming.

2 Materials and methods

2.1 Hardware

2.1.1 Chassis

The base for our self-developed electronics is formed by The Stadium Raider model car, sponsored by Conrad Electronics. The Stadium Raider is based on a Tamiya TL-01 chassis. The vehicle is electrically driven and features four-wheel-drive and servo steering. Spike wheels are mounted tot provide maximum grip.

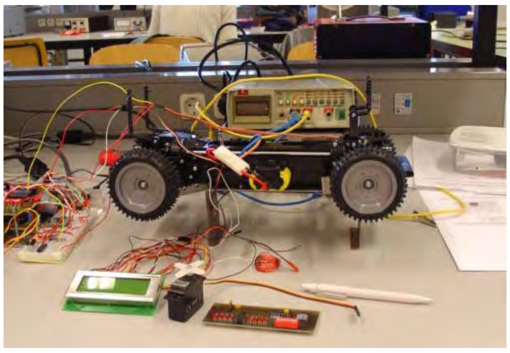


Figure 1 - The Tamiya Stadium Raider chassis in development stage

2.1.2 Sensors

To detect maize plants we make use of six Sharp GP2D12 infrared sensors. Four of them are mounted on the front of the vehicle and are used to navigate through the maize. The other two are placed on both sides of the vehicle and are used to count the plants. The sensors feature a range of approximately 10-80 cm.

Figure 2 - The Sharp GP2D12 infrared sensor

To turn our vehicle at the end of the row we make use of a digital compass, the Devantech CMPS03. The compass determines the average heading of the vehicle while driving through the maize field. This heading is used to calculate the opposite of it, which forms the heading to reach during the end turn.

Figure 3 - The Devantech CMPS03

To determine and control the current speed of the vehicle we use very small infrared sensors that are placed near the front and back left axle. Dark slopes in the reflective aluminium material are used to detect rotation and thus speed. Furthermore, voltage sensors are used to measure several system voltages. By doing this, we can easily monitor the battery conditions.

2.1.3 Supply circuit

The vehicle makes use of two independent power sources. A model car racing pack of 7,2V is used to drive the electrical engine of the car and also powers an electronic display and the GP2D12 infrared sensors. The power is distributed via a Tamiya electronic speed regulator. Four rechargeable NiMH batteries of AA-size (penlite), power an analogue circuit, which delivers a stable 5V supply voltage. This is used for the digital compass, temperature sensors and the microprocessor.

2.1.4 Main board

The main board is the place where the all the hardware comes together. The heart of the vehicle is formed by an Atmel ATMega 32 microprocessor. The main board features a UART-connection and a parallel programmer connection with the PC. Also, an electronic display on which information from the microprocessor can be displayed is connected to the board. Furthermore, a steering servo and speed sensors can be connected. For human-machine interfacing, three switches can be connected to the board. Connectors for the six infrared sensors and the digital compass can also be plugged onto the board. Finally, the board has a connection for two digital temperature sensors that can measure the inner and outer temperature of the vehicle.

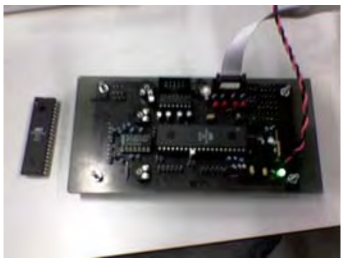


Figure 4 - The main board

2.2 Software

Below is a schematic of the software. We will describe some parts of the software in the following texts. For some parts there meaning and relevance is obvious and they will not be discussed.

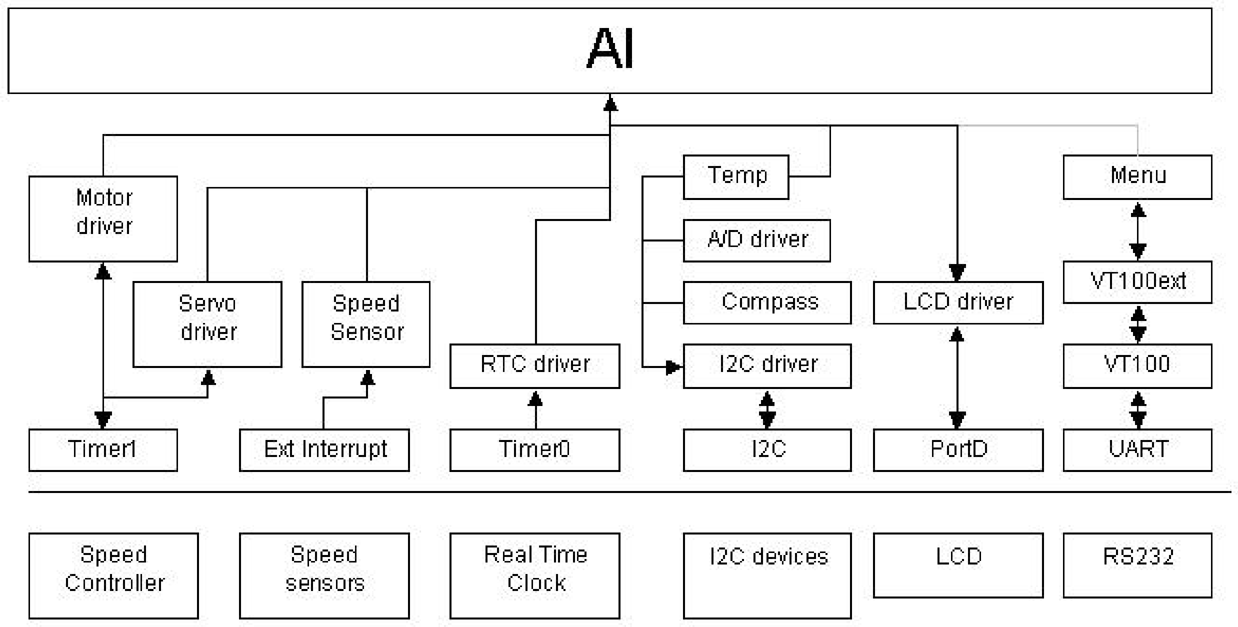


Figure 5 - Schematic of the software

2.2.1 VT100 terminal

Cornhoolio is equipped with a terminal emulator. Using this terminal emulator we are able to control en measure all our sensors within the robot. This is an easy way of making our car accessible and it runs on almost every computer. Through the menu's we have created in the robot we are able to drive him manually while observing the sensors or watching the speed sensors and starting several test routines.

2.2.2 I^2C bus

To access the different devices in the system the I^2C bus is used. The I^2C bus consists of only two wires. This reduces the use of wires and reduces the complexity of the circuit boards. For most I^2C devices there is a driver available. We used the available I^2C library from Procyon. However, for the CMPS03 and the temperature sensor we had to write our own drivers. We have also rewritten the drivers for the A/D converters so it was more efficient for us.

2.2.3 Engine and steering control

The speed control for the engines and the steering control are done by the same unit. The type of interfacing is a PWM signal with a 50Hz cycle; this is a typical servo steering signal. The speed controller uses the same type of control and thus reducing the complexity and the amount of software. All the control is now done by a simple timer, running at a 50Hz cycle with two compare outputs, toggling at the desired moment.

2.2.4

Speed control

The car has two speed sensors, one at the front left axle and one on the rear left axle. This enables us to make a feedback loop on the speed. We simply check if the vehicle is moving at the correct speed. If not we correct it. By testing in the field we noticed it takes quit sometime for the engine to provide enough power to get over obstacles. So we decided to make the car speed up fast enabling it to free it’s self from obstacles for instance. If on the other hand the car is to fast we gradually reduce the speed.

2.2.5 AI routine

The AI routine is the routine that represents the behaviour of the car. It starts with storing the compass value so we know what direction we should turn and the end of the row. The it starts to run for about a meter while sampling the compass to have an average which we then use to navigate on through the row while trying to avoid running in to the corn. After nine meters the vehicle will drive to a point where it sees no corn and then make the U-turn by navigating on travelled distance and the compass. Furthermore in every cycle it checks the current speed and corrects if necessary. It also samples the compass, reads the sensors and decides whether or not to go left, right or straight on. Then it waits for the sensors to update. The sensors have an update cycle of approximately 40 ms, so the AI is executed every 40 ms. The object and collision avoidance algorithms are the somewhat simple but they proved to work best in the test field.

2.3 Tactics

2.3.1 Tactics

Our tactics for driving through the cornfield is best explained in the following diagram.

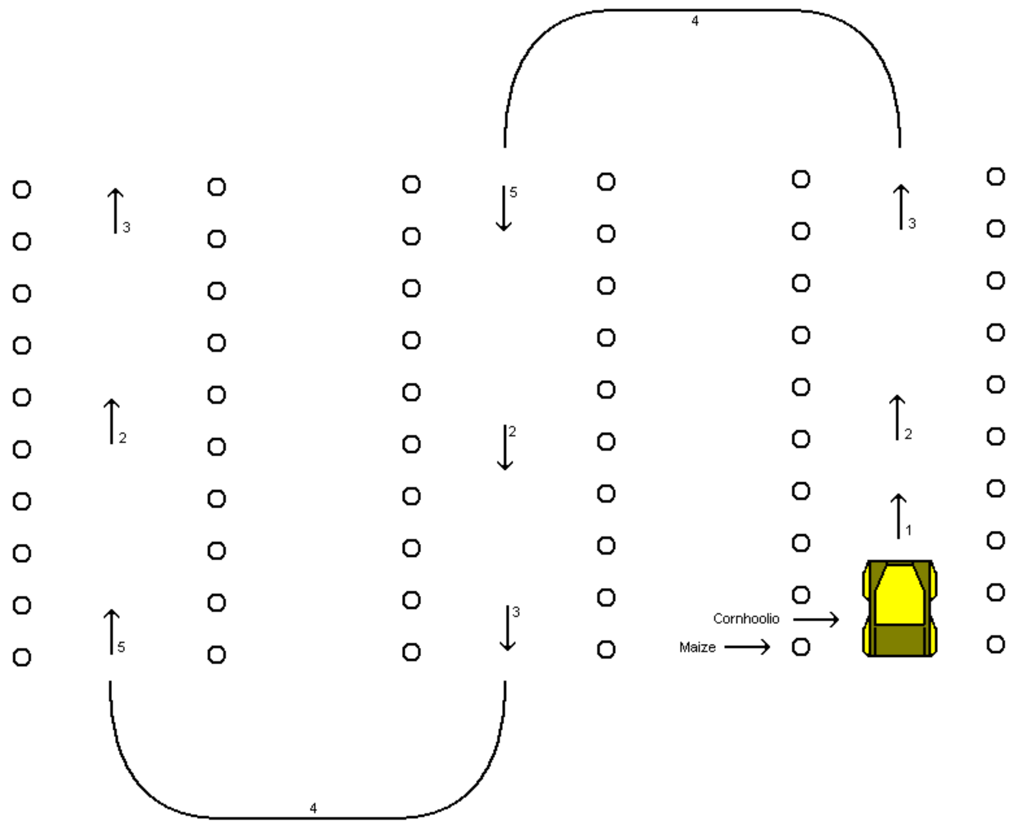


Figure 6 - Tactics for navigating through the cornfield

1. At the start of the race, the vehicle drives for about a meter and checks the scene of the maize field.
2. For the rest of the lane the vehicle checks the compass value and calculates an average value. While this is done the infrared-sensors make sure that the vehicle doesn’t drive into the plants. Another set of infrared sensors is used for counting the plants.
3. When the frontal sensors of the vehicle have no more plants in sight, the current average of the compass is saved and the value for left or right is read.
4. The saved compass value is used to make a turn of 180^o. Depending on the value for left or right the turn is made accordingly
5. If the current compass value is 180^o shifted in comparison of the saved value and the front sensors have spotted the next row, the vehicle starts driving straight ahead again. The value for the next turn is set to the opposite of the previous turn.

2.3.2 Testing in cornfield

To test our previously mentioned tactics, we used two methods. The firs testing session was indoor with some wooden sticks with green cardboard strings stapled onto them. This was done to simulate a cornfield. The test results from this test were above our expectations and proved the integrity of our tactics. For the second testing session we had the opportunity to test in a cornfield close the city of Dronten, in the province of Flevoland, The Netherlands. Here we had two days of testing with the elements as our biggest opponent. These tests were a good contribution to our tactics and after some minor adjustments to our speed control everything was looking promising.

2.3.3 Sponsoring

Since we had to develop all of our hard- and software ourselves, we thought it would be handy if we could have a sponsored chassis. Conrad Electronics was the company that gave us their helping hand and offered us a Tamiya TL-01 chassis. This is an off-road model car. Furthermore they provided us with two RC racing-packs, this are high-quality batteries made especially for RC racing cars. In return we promoted Conrad during the fair and the race, by means of stickers and catalogues. Our second sponsor, MEP, provided us during the development stage with a test board, with an onboard microprocessor/FPGA. Due to the problems we had with this board, we gained some delay in our schedule. Eventually, we had to cancel the use of this development board. The electronic display is also sponsored by MEP.

2.3.4 Team organisation

Our team consists out of four electronics and engineering students from the Amsterdam School for Higher Education. Internal allocation of tasks is as follows:

1.	Team leader and main hardware developer	Dennis de Koning
2.	Main software developer	Wouter van Gulik
3.	Assistant hardware developer, responsible for chassis mounting	Jorn Kommer
4.	Assistant software developer, responsible for VT100 interfacing	Daniel Blaauw

3 Results and discussion

3.1 Race results

The final race result was a disappointing ninth place. Cornhoolio once in motion drove pretty straight between the rows in both the straight and the curved rows. It also did count the plants although this was a difficult thing, because the vehicle had to be resetted a number of times during the race. When the vehicle stopped or slowed down too much and had to accelerate again then the loose soil was a problem. We actually forgot to implement a special feature so this was programmed after the meandering row race, it worked quite well since it only had to go fast and uncontrolled, like Cornhoolio in the movie Beavis and Butthead do America is.

3.2 Performance

During the test hours on Thursday we already saw that the loose soil was going to be a problem for us. Cornhoolio was to eager to go and accelerated too much which resulted in a cloud of dust behind him and the vehicle digging itself in the sand. We also had some problems with our batteries, we had three good batteries total, but only one had been used before, the other two arrived Thursday morning, so there was no time to train de batteries, in other words repeatedly charge and discharge them, which resulted in two potentially good batteries performing very moderate. Due to this problem the car was too weak to make a run on one battery and therefore we had to change the batteries very often. There were also some minor problems with the electronics for instance the LCD-screen would only work now and then, but for most of the time the electronics worked fine. The software as it was worked fine, there are always things that could be improved, but that would also be true if we had won the race. The only problem with the software was that because of other problems we had, things were overlooked. For instance, must we go left or right when we leave the row? It turned out that it was not an actual problem because the soil on the headlands was so loose we could not turn at all. We also tested in a maize field before going to Wageningen; this was a maize field near Dronten in Flevoland. There it drove very well, because it was a clay surface, which gave us a lot more traction than the loose soil in Wageningen.

Conclusion

At the Field Robot Event 2005 in Wageningen, a rather disappointing ninth place was obtained. Problems were caused mainly by the small size of the wheels of the vehicle. However, several test runs through a maize field with a more solid soil showed a steady behaviour of the vehicle while navigating through the field. This proves the reliability of our algorithms in the software and shows the working of our hardware.

The development of the vehicle as a team, working witch each other, and working with a second, colleague team, certainly improved our teamwork skills. Although the final result in Wageningen proved not to be as good as we hoped, we believe to have completed a very successful project.

References

- Franco, Sergio. *Design with operational amplifiers and analog integrated circuits, Third Edition*. Mc Graw Hill, New York, 2002.

- Stang, Pascal. *Procyon AVRlib, C-language function library for Atmel AVR processors*. Available at: http://hubbard.engr.scu.edu/embedded/avr/avrlib/ Visited several times since last update at January 30, 2005

- Field Robot Event Website. Available at http://www.fieldrobot.nl

Acknowledgements

We hereby would like to thank our sponsors four providing us with the opportunity to develop our vehicle beyond our tiny budget.

Conrad Electronics
http://www.conrad.com

MEP
http://www.mep-telecom.com/

Corn **O**riented **R**obot **N**avigation

Development of the wheel-based autonomous robot CORNickel*

** "CORNickel" resembles "Karnickel", a german word for rabbit.*

Stephan Fretzschner, Martin Grismajer, Tim Klusmeier, Kim Listmann

Abstract

Four students of the Technical University of Dresden developed the autonomous mobile robot 'CORNickel' to compete in the Field Robot Event 2005 in Wageningen, Netherlands. The vehicle is based on the drive concept of a Swiss mountain tractor. Its electronic system consists of multiple microcontrollers and is equipped with ultrasonic and infrared sensors. They are used for navigation between the rows of a maize field and for recognizing and counting plants. Decisions, details and results of the development are described in this paper. Moreover an additional freestyle task is chosen and illustrated.

Keywords

autonomous robot, navigation, Cornickel, ultrasonic sensor, electronic compass, agriculture, maize plants, counting, infrared sensor, rigi-trac, microcontroller, wheel-based, Tamiya Super-Clod-Buster, ATmega 128

Introduction

Analyzing the specifications of the contest resulted in its division in four basic subtasks: Constructional design, electrical design, navigation and counting plants. The first considerations concerned the general robot concept. Basically, there are two different approaches possible: a track based or a wheel based robot. Although there are a lot more possibilities such as walkers or hovercrafts, we decided to choose one of these standard concepts because they seemed to be easier to handle, particularly thinking of the fact that this would be our first robot. Because of the greater number of moving parts and the more complex build up a track based model seemed to be more difficult. Moreover, working at the Institute of Agricultural Machines [1] of our university, we wanted our robot to look like a tractor. So we decided to make it a wheel-based robot.

To provide a basis for navigation and counting plants an electrical architecture had to be developed. Driving autonomously needs both enough processing power to meet real-time requirements and an electronic basis, that complies to the demands of mobility: low weight, small dimensions and low power consumption, for the robot has to drive as fast as possible between two rows of maize plants. A reliable detection of the row end must be guaranteed to enable the headland turns into the next or next but one row. In addition to that, solutions for smooth navigation in the case of missing plants must be included. For the counting of plants it is required to distinguish single plants with an intra-row distance of approximately eight cm and a distance of 30 cm to the robot. Therefore appropriate sensor systems, computing power, programming methods and interfaces to the drive train had to be chosen. Development and adaption of fast and reliable algorithms were necessary to fulfil all tasks.

Materials and methods

Construction design

Focusing on wheel based vehicles many different build-ups are possible. Most of them use springs and dampers for axle suspension to ensure wheel grip. But with respect to dynamic vehicle behaviour a badly balanced carriage is the worst case. An alternate concept is used by a Swiss mountain tractor, the Rigi-Trac [2]. The non suspended axles are fixed to the frame, while a central joint in the middle of the vehicle takes care of ground adaption. By avoiding the application of springs and dampers the number of moving parts can be reduced significantly .

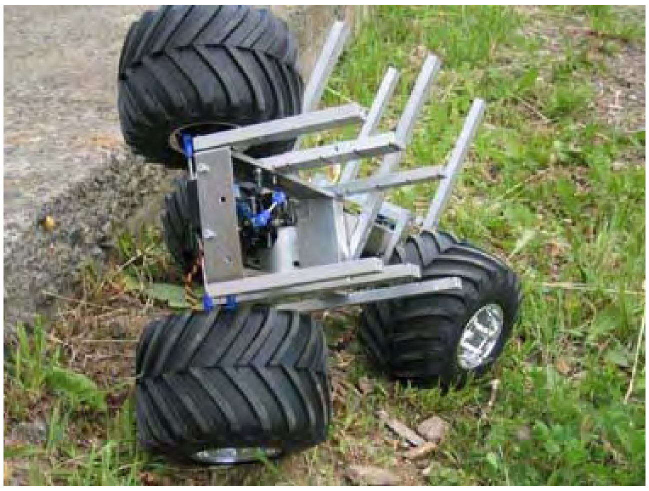


Fig. 1 The central joint of CORNickel.

As shown in figure 1 this joint secures wheel grip and has no big influence on the dynamic vehicle behaviour. In addition to that, the center of gravity can be lowered and together with 4WD an astounding all-terrain capability is achieved. To reach the same level of manoeuvrability as track based models, four wheel steering is necessary. The Rigi-Trac even enlarges its manoeuvrability by working with two independent steering shafts, resulting in three different steering modes.

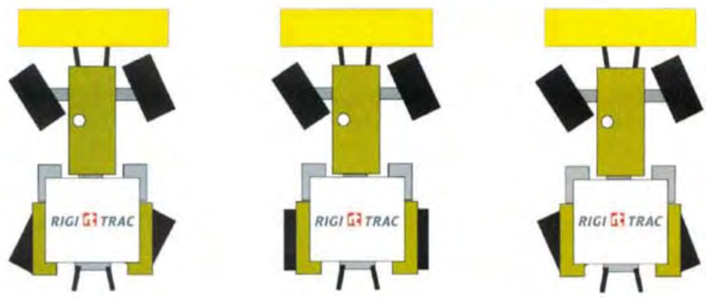


Fig. 2 Steering modes: all wheel, front wheel and crab steer (f.l.t.r.).

The possibilities arising from that, shown in figure 2, convinced us to use this concept for our robot. To avoid building every part of the vehicle on your own, an RC-car with these requirements was needed. We finally found that the Tamiya "Super Clod Buster" [3] fulfilled our wishes the best way and chose it to form the basis of CORNickel.

A steel frame, whose parts are screwed together, builds the chassis. To carry the load this frame is clinched on both sides (C-profiles) to increase the buckling resistance. In order to get above the axles to unplug electronics to the front side of the vehicle the frame is about 80 mm above the axis of the joint and the drive shafts. The drive axles and the joint axis must be in-plane to avoid a torque load, which would result in a deadlock of the central joint. Special materials were chosen for the central joint. The rocker bearing is made of aluminium while the connector is of red bronze. To improve the lubrication of the red bronze silicon grease was applied. The joint is screwed into the rear half of the chassis and the link to the front is held by a circlip. As both parts must be tight together washers are used in the space between the two vehicle parts.

The joint, both axles and the steel frame compose the basis of the robot. On it all other parts like sensors, electronics, power supply etc. can be fixed and it carries the load. The design of the attachment parts holding the interior was the next step on the way to build the robot. Because of our controlling concept based on four microcontrollers, which are integrated in three boards, and an additional board for power electronics we needed a special fitting for the circuit boards. Therefore aluminium columns were screwed to the steel frame with slots in them to insert the boards from the front side (front boards) and from above (rear board).

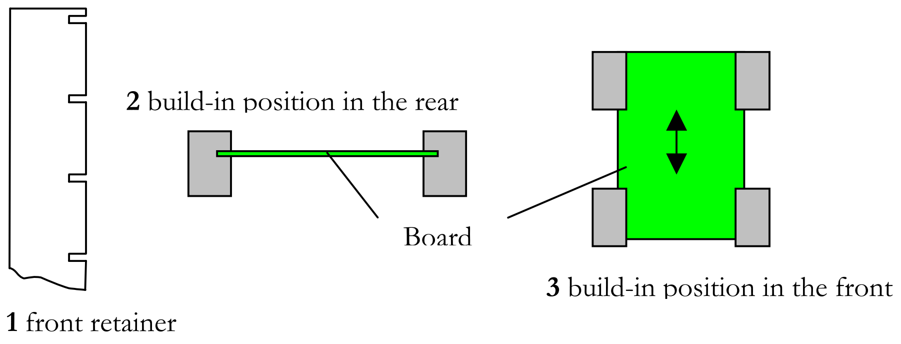

2 build-in position in the rear
Board
3 build-in position in the front
1 front retainer
Fig. 3 Retainer for electronics.

Figure 3 shows the front retainer with its slots and the build-in positions for the board in the rear and one of the boards in the front part of the vehicle. Both are shown from above.

The front axle is clamped to the foremost columns and an additional support between the gearbox and the steel frame holds the torque load. For power supply three rechargeable battery packs are mounted in a U-shaped steel part in the back of the car. They are fixed with three rubber bands so that they cannot move while driving.

To protect the interior of the robot from rain and dirt, a plywood car body is put around the steel frame and the columns. The wood is coated so it will not let water in. Moreover we used the coat for design reasons and trying to give CORNickel a good look. At the back of the robot, the car body carries the LCD and the switches for the different programm modes in the competition.

One of the most important parts of a wheel based vehicle is the construction of the steering rods and the quality of steering. The most interesting parameter a constructor can set up is the turn circle, because it is decisive for the speed of the headland turn. The only constructional parameter influencing the turn circle diameter which could be set was the wheel base, as all the steering rods were included in the base model and should not be changed. CORNickel's wheel base was designed for reaching the next but one row, because we optimized it for the first part of the competition. Here, in the straight row field, the robot has to do a lot more turns than in the curved field.

Deciding to take part in a competition in natural environment, you always have to think of how modifiable your robot must be and how you solve this modification – using software or hardware. Looking at the Field Robot Event we could not be sure about the height of the plants and their appearance. Therefore we designed a special fitting for our sensors. Every sensor can be modified in height and in position on the outside of the vehicle and can be pivoted around three axes, which gives us a high degree of modification. The negative effect of these additional super structural parts wass that the vehicle became much heavier. To protect the sensible sensors from the rain, they had to be inserted in water resistant cases, which are sealed by silicon.

The only sensor not needing a modification was the compass. But here other problems occured. It reacted very sensible to all metal parts, even to non-ferromagnetic materials. In order to avoid great mistakes in the displayed angle of the compass, we had to put it as far away from the aluminium columns, the motors and the servos as we could. Therefore a pole was screwed to the vehicle with the compass glued on top of it. To avoid any possible disturbance we even changed the metal screws of the case to plastic ones. By these arrangements we ensured the function of all sensors, which are the most important parts of the robot.

Electronic Design

To meet the demanded processing power there were basically three options: a laptop, a microcontroller based system or an embedded PC. The laptop would have guaranteed the smallest development effort, but considering it to be the most expensive and oversized choice, it seemed unhandy. A potential alternative was an embedded PC, combining the advantages of a desktop PC and a microcontroller based system. Nevertheless it would need several interface cards and a real-time based operating system. Ensuring the least costs while giving the robot enough energy to run as long as possible, we decided on a microcontroller architecture and ended up with two ATmega128 and two ATmega16 [4]. That also gave us a wide range of possibilities in connecting sensors and actuators.

Counting Plants

The plants are counted by using Sharp GP2D12 infrared sensors [5]. They seemed to be suitable because of their approximately five cm wide beam. The GP2D12 has a measuring range from 10 to 80 cm and the change of the analog output voltage in this area is big. So these facts should simplify the detection of single plants. The output voltage depends on the distance of the reflective object and on the position within the beam. The maximum time of one measuring is approximately 53ms.

For counting plants we subdivided the rows in small sections containing one measurement. If more than one measurement is done in one section only the maximum value is saved. After sufficient measurements are taken we start with the evaluation of little groups of sections, trying to identify single plants. This method has two advantages. First, we do not use up too much of our microcontroller's 4 kBytes of SRAM although every measurement is a 16-bit value. And second, this method makes it easier to evaluate the sections as they are independent of the robot's speed and equally spaced.

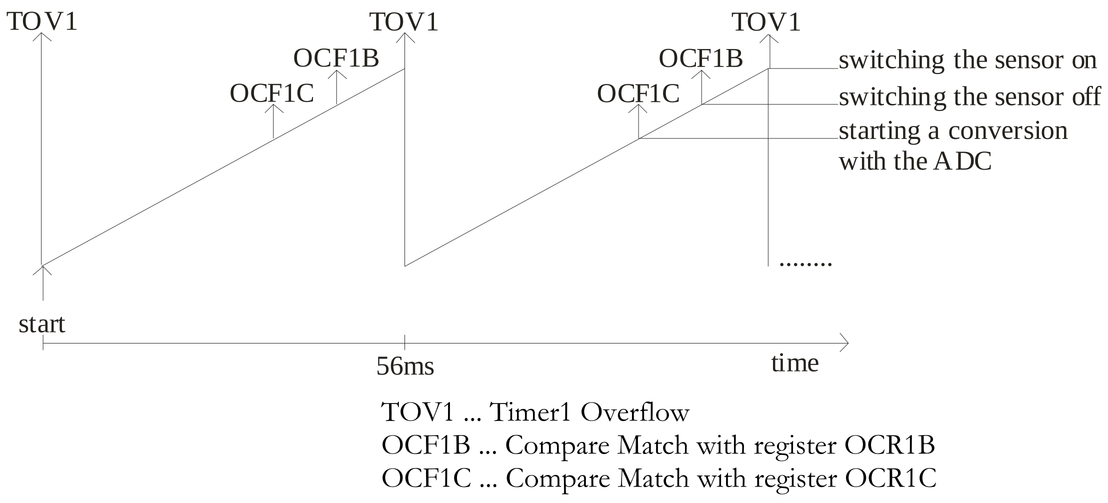


Fig. 4 Timing scheme of the infrared sensors.

The sensors are controlled by using the Output Compare Units and the PWM modes of the controller's two 16-bit timers. The Output Compare Unit continuously compares the Output Compare Register (OCR) with the Timer/Counter Register and signals a match. A match will set the Output Compare Flag (OCF) which can generate an output compare interrupt if it is enabled. This is used for starting a conversion with the Analog to Digital Converter (ADC). The Output Compare Unit in combination with the PWM Mode is used to switch the sensors on and off.

For counting plants we use two sensors in front of the robot and two additional ones in a defined distance behind them. The front sensors are started continuously by the PWM of Timer 1. The different measuring times from minimum 32,7ms to maximum 52,9ms require the above-mentioned switching on and off (Fig. 4). So we get the defined section in which the measuring is started. The allocation of a measurement with the accompanying section is realised by counting the pulses of a wheel encoder. This means, the allocation is independent from the speed of the robot. Figure 4 also shows which respective flags are set when a compare match occurs. Because of the long time that one measurement takes, it is possible that one section doesn't get a measurement at all, if the robot drives to fast. In this case the rear sensors are started specifically to fill this gap. A special number of pulses (five) from the encoder are counted and then the sensors are started (Fig. 5). This allows the robot to drive at a higher speed without loosing information.

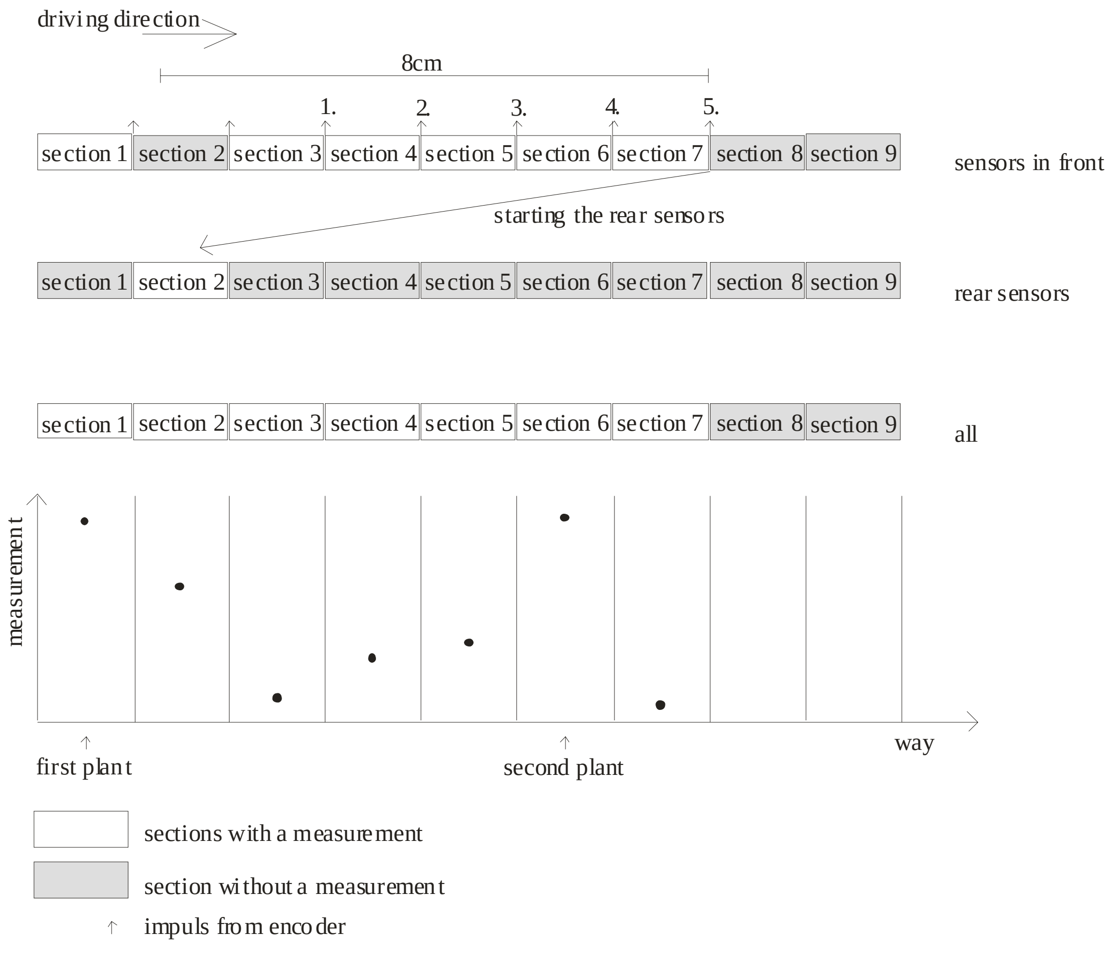


Fig. 5 Recognition of plants within the sections of covered distance.

The sensors can't be started by a PWM because the time between two measurings is different. To solve the problem we used the Force Output Compare (FOC) function of the Timer 3. The FOC-bit can by set by software when five encoder pulses are counted. It simulates a real output compare match and updates the associated pin. For evaluation the single measurements are compared in small groups and an algorithm decides if it is a plant or not.

Navigation

Some of the conditions influencing the robot navigation while driving through the corn rows are not exactly known before the contest or the practical usage, as there are: underground, row width between the leaves, height of the plants, light conditions or wind. Because it is not possible to test the robot under all circumstances, our navigation concept was developed to work as independent as possible of these influences.

Furthermore it seemed better not to have too many different sensor principles or even redundancy because every sensor-technology has its own sources for errors which must be taken into account. If information is necessary about the environment and two redundant sensor-systems (e.g. infrared and ultrasonic) give opposite answers, the robot has to be programmed to have more confidence in one of them. So it should be enough to have one trustworthy sensor-system for every subtask with known errors, which may be filtered for improvement.

As this was our first autonomous robot project and we wanted to avoid expensive and very sophisticated sensor-systems (e.g. camera or laser scanner), we decided to start it simple. So we used four Devantech SRF04 ultrasonic modules[6] for navigating through the rows (the microcontroller measures the time between sent and received signal and calculates the distance in the range of 3cm to 3m) and a Devantech CMPS03 electronic compass module [7] for the headland turn (it outputs a PWM signal for its orientation which can be processed by the microcontroller). All the calculations and decisions are done by one of the ATmega128 microcontrollers, resulting in new speed values for the drive system and steering information for the two steering servos. These microcontrollers offer enough resources for analyzing the data of the chosen sensors and for communicating with one another. They are easy to be programmed in C and there is a lot of information and examples available about them, which were important criteria for us.

At the front of the robot there are two ultrasonic sensors oriented at 45° to the left/right, which are to obtain information about the row distance in front of the vehicle (to allow early reaction). Two additional sensors at the rear are oriented at 90° to the left/right (for the distinction between heading or position mis-alignment). In this configuration the sensor signals also don't influence each other, which is important because all sensors are sending their signals at the same in order to get as much information about the environment as possible.

Because of the unknown conditions mentioned above, these four sensors are mounted low and tilted up to avoid ground reflections (the cone can be up to 60° wide) and to get the plant distances independent of their leave length. To avoid influences from fields of the power electronics, the compass module is mounted high above the robot on a pole.

Two different basic programming methods were used:

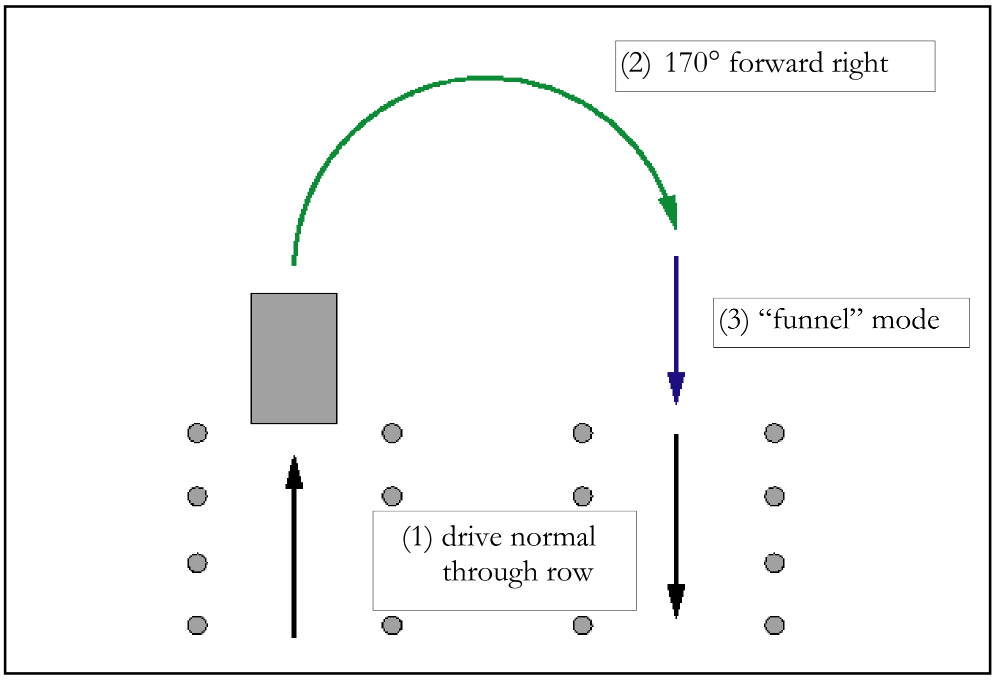
(a) The so called Subsumption method [8] for driving through the rows: The program is running in a loop which consists of reading the distances measured by the four ultrasonic sensors, comparing them with predefined situations, choosing one of them and setting new speed and steering values. If none of these situations fits, the robot is driving in the standard mode – straight ahead at full speed.

(b) Step by step at the headland turn: The robot is doing something until a special predefined sensor (compass) configuration occurs and continues with the next step.

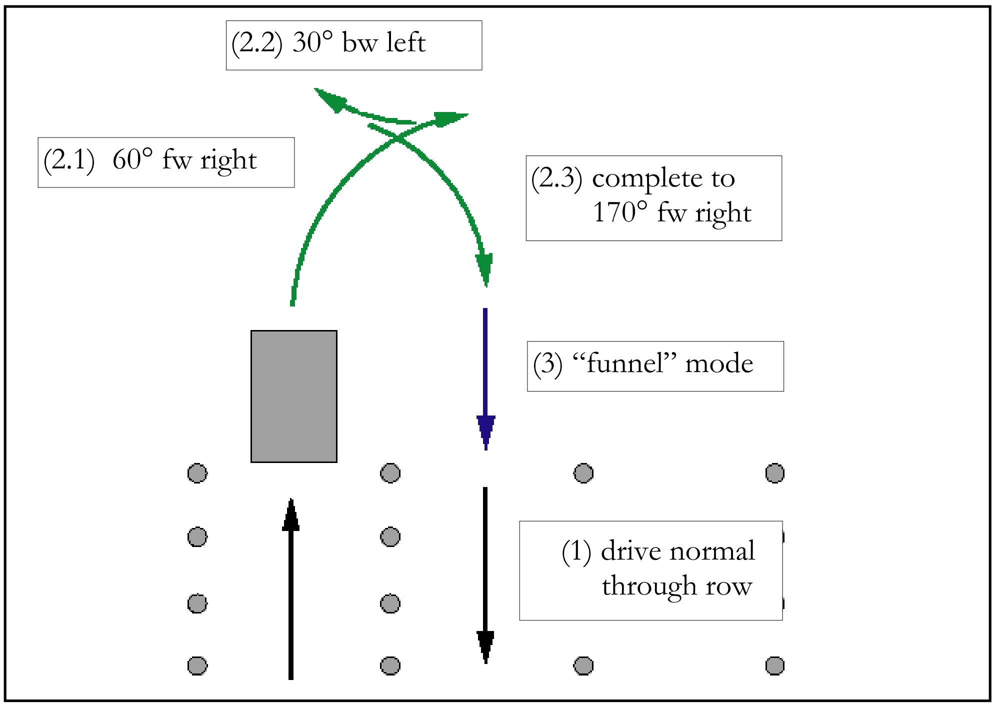
Some problems and their solution:

- Bad sensor data:
  Every ultrasonic distance value is checked against the previous and the next value and sorted out if impossible. Unfortunately this can result in the usage of data which is already one circle old, but as the sensors are fast enough this doesn't matter.
- Smooth navigation:
  A stack was programmed to have always access to some old sensor values which can be used to make the navigation smooth. Like driving a car by a steering wheel the steering is used almost all of the time , but with very small values. Only front steering mode is active for navigating through the straight and curved rows, while for the headland turns both steering axles are used.
- Independence of row width:
  Distance values from both sides are only compared to each other, not compared to absolute predefined distances.
- End of row or missing plants:
  If one or more sensors report too long distances the program changes into an "unsure" mode and navigates as well as possible with the available data. If all sensors can't see enough anymore, speed is reduced and the headland turn starts.
- Headland Turn:
  Using the steering of both axles the robot is able to do the headland turn with predefined compass-values directly in one step (two rows further) or in three steps (next row). After completing a turn of approximately 170° the program switches into a 'funnel' mode, trying to find the entrance to the new row. When all ultrasonic sensors have distances which indicate a position in the

row, the program switches back to the "normal drive" mode. The turning procedure is illustrated in figure 6.



(a) turn two rows further



(b) turn into the next row

Fig.6 Headland Turns

Freestyle Session

For the freestyle session of the contest we chose a problem that real tractors often have to compete with. When a forage harvester is used on the field, the crop is collected on a tractor-drawn trailer. So the tractor driver has to maintain a fixed position with respect to the harvester, which is often driving at a considerable speed.

To achieve following an ahead driving vehicle two tasks have to be solved. First, a constant distance has to be maintained and second, the direction of the vehicle ahead has to be recognized. An ultrasonic sensor at the front of the robot was used to control the distance with a simple proportional feedback controller. To recognize the direction more effort had to be spent. An infrared source modulated with 36kHz frequency is put on an RC car (the harvester) while a binary IR-receiver array is placed on CORNickel's front. As the IR-receivers we used were designed for signal transfer they detect failuremode when receiving the same signal for longer than two seconds and stop working. To outsmart them they had to be turned off once every second. The resulting error must be eliminated by software. The function of the receiver–array is shown in figure 7. The two outer receivers have a broader window for incoming beams to ensure never losing the car, while the inner receivers only have a small window to be more precise in detecting the direction.

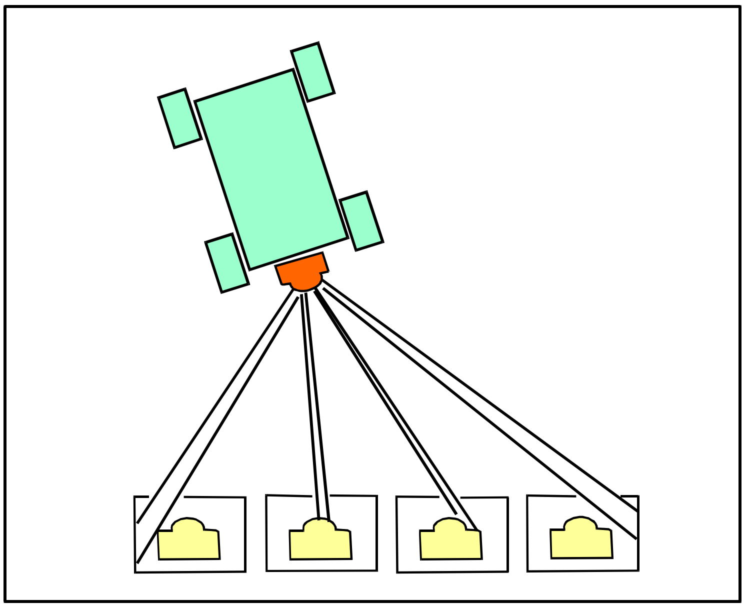


Fig. 7 Receiver-array for the freestyle session.

Results and Discussion

Construction Design

Looking at our robot right after build up, it was much heavier than we supposed it to be. Particularly the tires were heavily overloaded and needed to be filled with special foam, which was winded around the rim until it filled out the inner-wheel-space. Another severe problem with the weight was that CORNickel could not be steered while standing still, because the servos could not apply enough torque.

This was a big testing problem for our programmer. Every time he wanted to see some steering reactions, he had to lift the vehicle up. In addition to that, also the driving performance is inflicted by the heavy weight. Especially starting up or driving slowly on uneven ground can lead to problems, unless an additional torque control is implemented. Generally, the chassis concept with the central joint proved to be good, as all four wheels had permanent ground contact and delivered optimal traction.

Electronic Design

Cornickel is equipped with a multiprocessor unit. We found dividing tasks would give us two main advantages: First, real-time tasks like navigation and counting plants would not interfere with one another if realized on different controllers, and second, autonomous controller boards would make testing easier as the programmers can work independently on their specific problems.

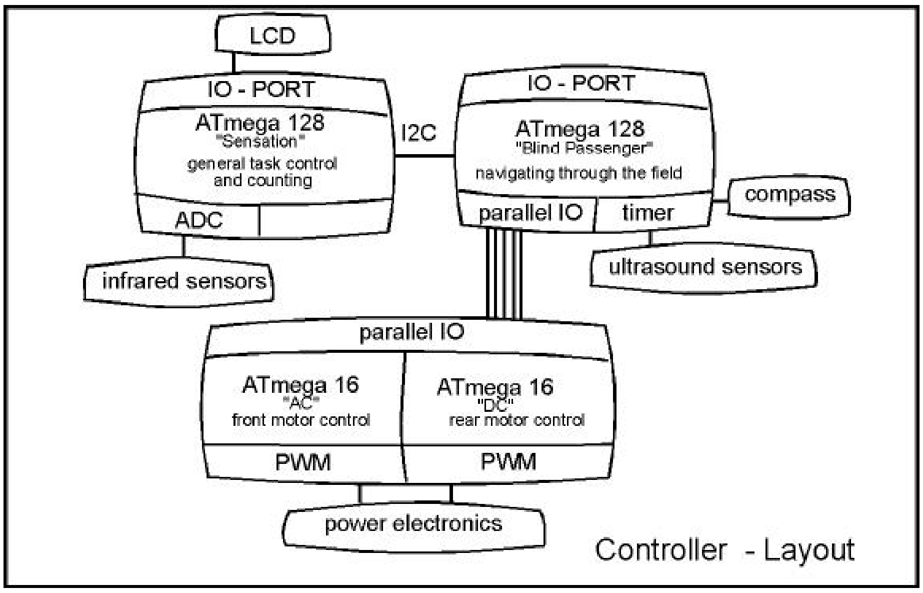


Fig. 8 Controller Layout

As can be seen in figure 8 CORNickel's multiprocessing architecture is based on four microcontrollers. There is one ATmega128 controller designed for navigation and user interface, another one doing the plant counting while two simillar ATmega16 are controlling the DC Motors. The controllers are clocked on 16 Mhz and connected through parallel I/O ports.

Power supply is realized through a 7.2 V / 3.5 Ah NiMh battery and a voltage regulator that guarantees 5 V onboard voltage. The two 90W DC motors need an independent power system using another two 7.2V NiMh batteries. The motors are driven through an H-Bridge using power MOSFETs whose pulsewidth control is provided by the two motor controllers. To avoid damage they are galvanically isolated from the power-electronics. Originally, the controllers where designed for feedback control using incremental encoders in every wheel to make traction control possible, but the implementation failed due to lack of time.

Navigation
During the contest, the robot completed almost all of the mentioned navigation-subtasks at least once and drove through the rows quite smooth and fast. There were some contacts with the plants and sometimes it needed help, so obviously some more testing would have been helpful for better adjustment of sensor-beam-directions and predefined parameters in the program.
Generally, there were some problems with the unexpected rough ground and rather small plants which the navigation concept was not perfectly prepared for. Sometimes spectators in the headland area disturbed the sensors and confused the program while turning.

Conclusions

For improving CORNickel, one of the first goals must be reducing its weight. This can easily be done by using lighter materials like aluminium instead of the steel parts. Also the attachment of the sensors on the outside of the car does not need to be that large as soon as the sensor positions are known. As all these plates are made of steel some kilograms can certainly be saved.

The multicontroller concept generally seemed to work fine and was worth a try. Most problems occurred with the implementation, as we soldered all circuits by ourselves. Therefore we had several contact problems and a general problem with the power electronics. Looking back, we would recommend not to spent too much time into electronics and rather buy complete parts and circuits if available.

For the preparation of the freestyle session we only had very little time, which led to problems during the competition, but the concept still seems to be robust and mature.

Looking at the plant counting results of the competition, the number of counted plants was only about half of the real number. This may result from the intra-row distance of the plants beeing smaller than expected. The robot could not detect the gaps between the plants and so it was difficult to distinguish single plants. Because of the small plants, we also could not use the designated attachments for the sensors on both sides of the robot. So the sensors had to be placed at the rear of the robot to get as close to the ground as possible.

The navigation concept worked quite well but had problems with some special conditions. Subsumption is generally a good method to avoid storage of big data, but a simple world model (global coordinates) might be useful e.g. to notice a failed headland turn. A camera can improve the navigation between rows in the case of smaller or damaged plants while some additional sensors for realizing contact with plants and algorithms for returning to the row would be helpful as well. Problems with rough underground can only be solved with a speed and torque control in the drive system. This can easily be implemented as CORNickel is already equipped with four independent wheel encoders.

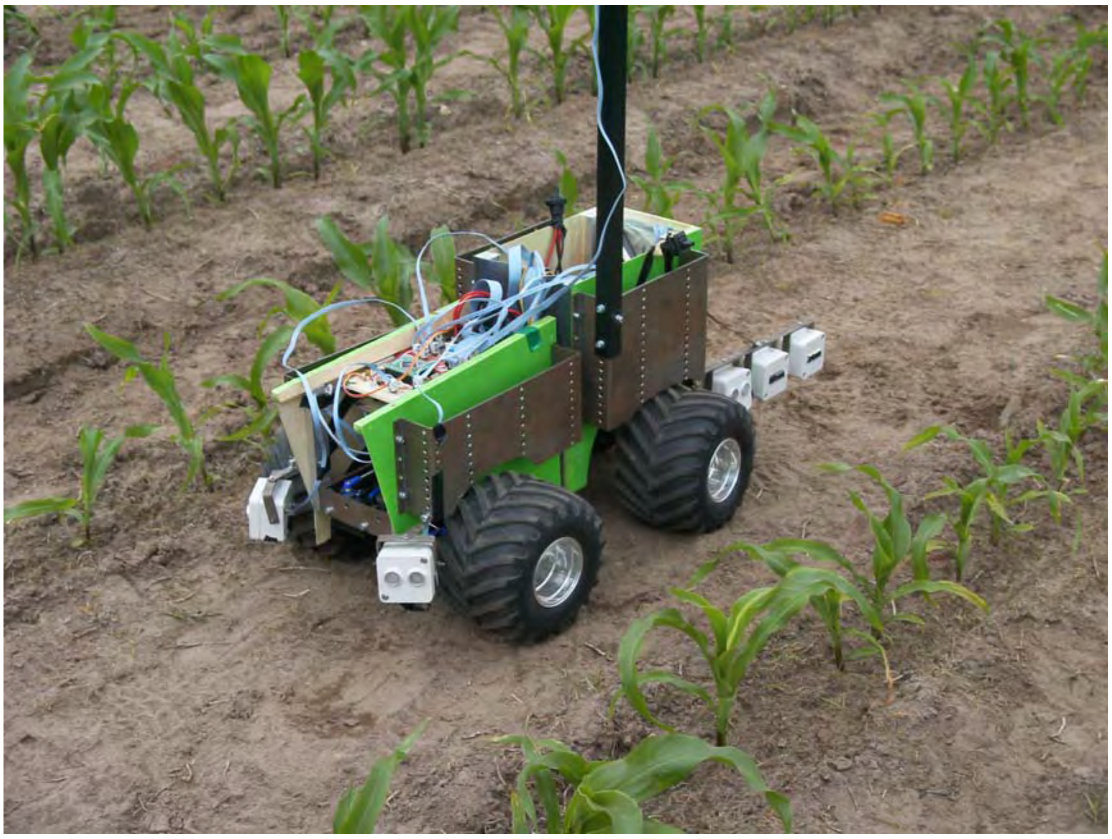


Fig. 9 CORNickel during a test drive.

References

[1] http://www.landmaschinen.tu-dresden.de

[2] http://www.rigitrac.ch

[3] http://www.tamiya.de/
[4] http://atmel.com/
[5] http://sharp-world.com/products/device/lineup/selection/index.html
[6] http://www.robot-electronics.co.uk/htm/cmps3doc.shtml
[7] http://www.robot-electronics.co.uk/htm/srf04tech.htm

[8] Jones, J. L., Flynn, A. M.. 1995. Mobile Robots: Inspiration to Implementation

Field robot “Eye-Maize”: Still on the field !

Ralph KLOSE, Martin MEIER, Andres LINZ, Arno RUCKELSHAUSEN

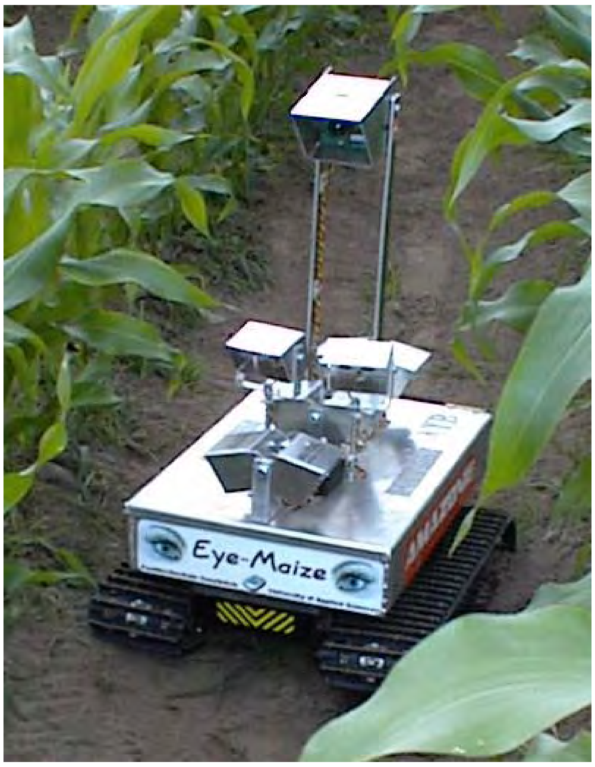
Email: a.ruckelshausen@fhos.de

University of Applied Sciences Osnabrück
Faculty of Engineering and Computer Science
Albrechtstr. 30, 49076 Osnabrück, Germany

ABSTRACT

In 2004 a group of undergraduate students attending a course of electrical engineering with emphasis on electronics decided to take part in the Field Robot Event 2004 in Wageningen. With total costs of about 1.400 € the robot “Eye-Maize” was developed, based on sensor fusion and a microcontroller platform. At the Field Robot Event 2004 Eye-Maize successfully drove between the straight and curved rows with reasonable speed. Moreover, the U-turn worked well for dry soil. Some problems occurred at the U-turn for wetted soil due to slip.

The technology has been described in the proceedings of the Field Robot Event 2004 [EYE MAIZE 2004] and has been presented at the AgEng [LEUVEN 2004]. The 3rd place – as best student team – resulted in new research ideas, moreover continuous interest of print media and TV came up. From the technical aspects, the implementation of a low-cost and high performance camera was of broader interest, especially the application for fast row guidance purposes [CMUCam 2005]. The positive experiences with Eye Maize resulted in the development of the new field robot optoMAIZER (described in these proceedings). However, since Eye-Maize was quite successful, the idea came up to start again with Eye-Maize. The software was adapted to the new tasks of the contest (see below), else Eye Maize is still the same.



Field Robot Eye-Maize

KEYWORDS

Field robot, optoelectronic sensors, sensor fusion, CMOS camera, plant count, maize

Software Add-on

The navigation between straight or curved rows already worked fine, thus no changes were necessary for these tasks. As compared to the Field Robot Event 2004, however, new tasks occurred for the competition in 2005:

A) The robot has to turn at the end and enter 2 rows further again.

B) Counting the number of maize plants in a row.

The concept of a microcontroller-based sensor fusion has the advantage that both new tasks can in principle be solved without any hardware changes. The interpretation of sensor information (for B) as well as engine control (for A) can be realized via software changes:

Solution for A: The U-turn was realized with a turn of 90°, driving some distance and another 90° turn. If the driving distance is changed, entering 2 rows further can be realized. Via the touch display, both options – entering 1 or 2 rows further – can be selected.

Solution for B: The signals of the optical distance sensors are interpreted with respect to counting plants. This is realized by averaging, both from the sensors and the software.

Moreover, further extensions of the software have been carried out. Via touch display different setups for dry or wet ground can be selected (as a rough slip correction). For the freestyle session, Eye-Maize is moving straight ahead until something “red” occurs.

References

[Eye Maize 2004] “Field Robot EYE-MAIZE” ; Frank DIEKMANN, Jens FLEISCHHACKER, Johannes HENKEL, Ralph KLOSE, Torsten KÖNIG, Martin MEIER, Nicola MOCCI, Axel MÜHRING, Daniel NEGD, Tobias NOLTE, Evert NORD, Maik SCHOTMANN, Johann SCHULZ (Student project supervised by N.Emeis, A.Linz, A.Ruckelshausen); Field Robot Event 2004, Wageningen/The Netherlands, Proceedings,ISBN 90-6754-818-9, March 2005

[Leuven 2004] Student Project: Field Robot EYE-MAIZE“ ; A.Ruckelshausen, A.Linz; AgEng 2004 Conference “Engineering of the Future”, Leuven / Belgium, 12.-16.09.2004

[CMUCam 2005] „Reihenführung autonomer Roboter mit der Low-Cost-Kamera CMUCam“ ; Ralph Klose, Michael Meier, Andreas Linz, Arno Ruckelshausen ; Bornimer Agrartechnische Berichte, 2005 (ISSN 0947-7314), Potsdam-Bornim, to be published

Acknowledgment

We would like to thank all of our generous sponsors:
AMAZONEN-Werke H.Dreyer, Hasbergen-Gaste, Germany
Phytec Technologie Holding AG, Mainz, Germany
Leibniz-Institute of Agricultural Engineering/ATB, Potsdam-Bornim, Germany
Farnell inOne GmbH, Oberhaching, Germany

µCallum – an autonomous car

Constructions and results of the µCallum car

Frank Joosten, Mark Lambert, Bart Kramer, Thijs de Ridder

E-mail: webmaster@microcallum.nl

Abstract

A group of four students designed, built, tested and competed with their car. The car had to drive through a maize field. It used a compass and four infrared sensors as input. It had two motors at the rear wheel for driving and one steer servo at the front for steering.

The microcontroller which controls the car is an ATmega64. The software for it was written in C. Further there was an FPGA on the car for reading the sensors and setting the motors.

The car was very successful during the field robot competition 2005. It won the first prize.

Keywords

µCallum, autonomous car, field robot, navigating

Introduction

For several years the University of Wageningen has organized a competition in which different teams from all around the world have to compete with each other. The competition was held in a maize field in Wageningen. The cars had to drive through straight maize rows while counting the plants they passed. At the end of a row, the car should enter two rows further. Another test is navigating though curved rows. At the end the car has to go into the next row. Finally the teams get the opportunity to show whatever they like in the freestyle session.

In this document we will tell you about the techniques we used to build our car and what results our team achieved in the race. Finally we will draw our conclusion about our results.

The team

With two teams from the Hogeschool van Amsterdam we joined the Field Robot Event. The name microcallum, which is our team name, got its name by combining the name of corn in Latin and the microcontroller which we use. Corn translated to Latin gives you callum, combined with micro from microcontroller it gives microcallum. The team consists of four members, Frank Joosten, Thijs de Ridder, Mark Lambert and Bart Kramer. We are all third year students E-technology. All of us also built a robot last year. This robot had to follow a wall in our school. This made us enthusiastic to build a more complex robot to compete in the Field Robot Event.
Frank Joosten designed and built the chassis and has been writing the software to control the display.
Thijs de Ridder wrote the VHDL source code, which is the language to program the FPGA. And he made most of the software for the microcontroller.
Mark Lambert made the main PCB, on which the microcontroller and FPGA are placed. Further he designed the H-bridge and our grass-seeder for the freestyle session. Finally he was our webmaster.
Bart Kramer designed the test PCB, on which the first simple software programs could be tested and the interface PCB, on which the display and the buttons are placed.

1. Materials and methods

1.1 Mechanics

1.1.1 Properties

We designed a chassis with following properties:

- Wheels with a diameter of 16 cm
- Steering front wheels
- Independent propelled rear wheels
- Four wheel contact
- 6 cm ground clearance
- Small chassis
- Variable sensor height
- High torque motors

With these properties it is possible to drive at various kinds of soil like sand, clay, wet conditions or with big clods. It does not matter if the maize is small or big because we can adjust the height of the sensors.

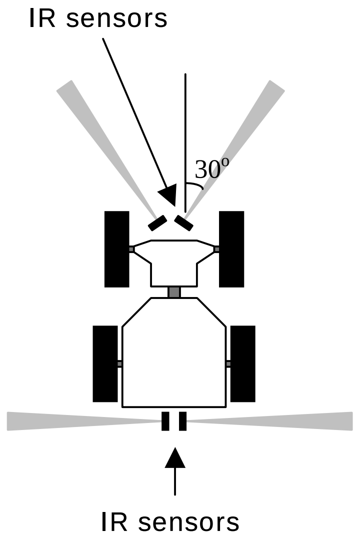


Figure 1.1: schematic topview

1.1.2 Design

3D CAD

We started our design with finding the parts we needed to build a car for our demands. With these parts we made a design in a 3D cad program. In this way we were sure everything fit well and the car remains small.
The car is approximately 380mm in length and is 350mm wide. The ground clearance is 60mm.

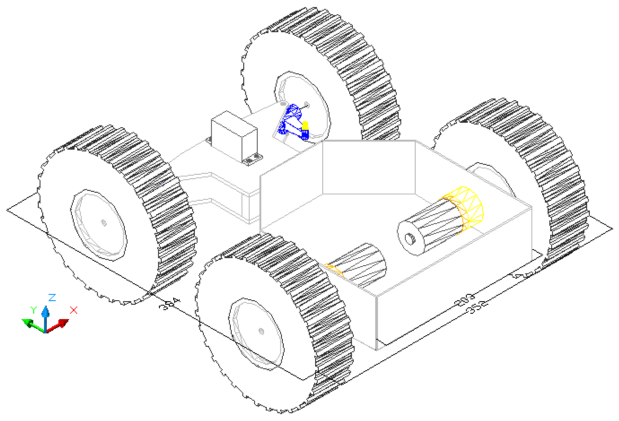


Figure 1.2: 3D view

Four wheel contact

The chassis consists of two sections, a front section and a rear section. The front and rear sections are connected with a steel axel of 12mm diameter. This axel is fixed to the rear section with two nuts. The front section is connected with the axel trough a 12mm hole in nylon. On both sides of the nylon a snap ring is mounted on the steel axel. Because of this construction the front and rear sections can rotate with respect to each other.

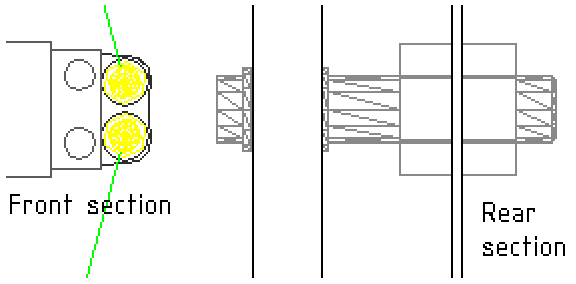


Figure 1.3: Topview of connection

Because the front and rear section can turn with respect to each other, the car has always four wheels on the ground. This makes sure the car has better grip.

Front section

The front section consists of three main parts, a nylon plate and two acryl glass plates. The nylon plate is the connection with the rear section (see section Four wheel contact). In figure 1.4 you can see the two acryl glass plates. They support the wheel suspension and the servo. The top and bottom acryl glass plates are almost identical, a hole for mounting the servo is made in the top plate.

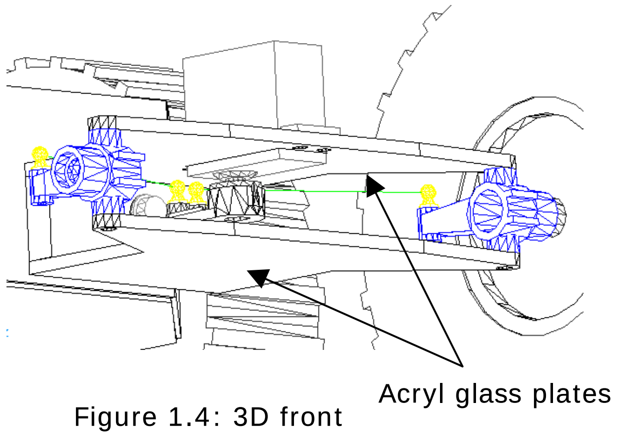


Figure 1.4: 3D front

To connect the wheels to the chassis and to the steering servo we used some parts from a Tamiya TL-01 chassis, a remote control car. To connect the wheels to these TL-01 parts we made two adapters to fit 12mm axel in a 5mm bearing.

Rear section

The rear section consists of aluminum. We folded it out of one part so it became a strong box. The motors are mounted with three screws to the side of the chassis. The two motors can be powered separately so the car can have a smaller turn radius. The battery is placed in front of the motors to get enough pressure on the rear wheels. The car has some kind of roof on the rear section. It is as large as the whole rear section and covers everything in it. This roof plate can be opened, so we can get to the electronics, battery and motors. The electronics are placed above the engines, connected to the roof.

Sensors

The car has four infrared sensors. Two are placed at the front and two are placed at the back of the car (see figure 1.5). The IR sensors can be adjusted in height. So we can adjust the height to the height of the maize. The sensors can be placed at the height of 5cm till 30cm with steps of 2cm. To protect the sensors against rain and sunlight we have mounted plates above it.
There is also a compass mounted on the car. The compass measures the earth magnetic field. That is the reason why we placed it high at the front, far away from our motors. Motors produce magnetic fields which can disturb our compass.

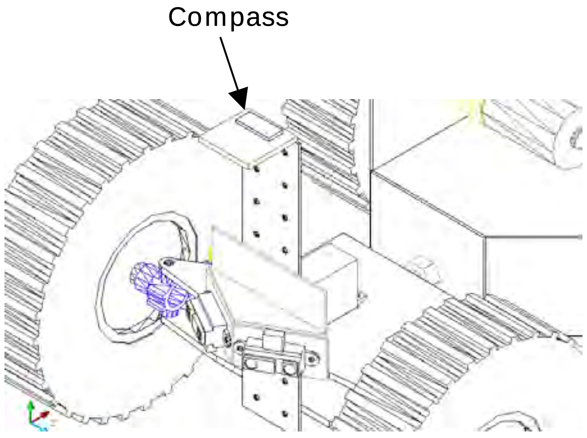


Figure 1.5: 3D view front

Motors

The car is driven by two 12Vdc motors, which will use 1 Ampere at maximum. The motors are driven by a PWM (Pulse Width Modulated) signal. Because of this signal the speed of the motor can be adjusted, but the motor keeps its torque.
Steering is done by a servo, a motor that can turn only a view degrees, depending on a PWM signal.

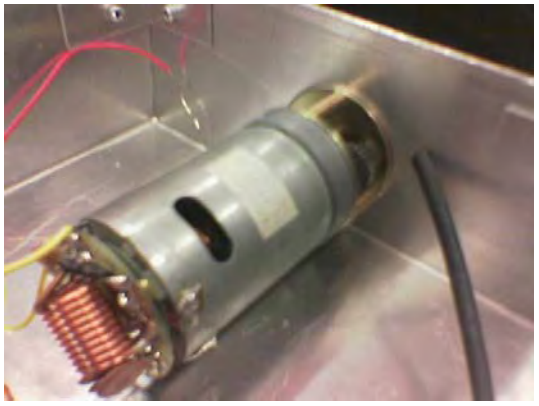


Figure 1.6: Motor

1.2. Electronics

Most of the electronics in our car are developed by ourselves. In figure 2.1 the block diagram of our electronics is shown.

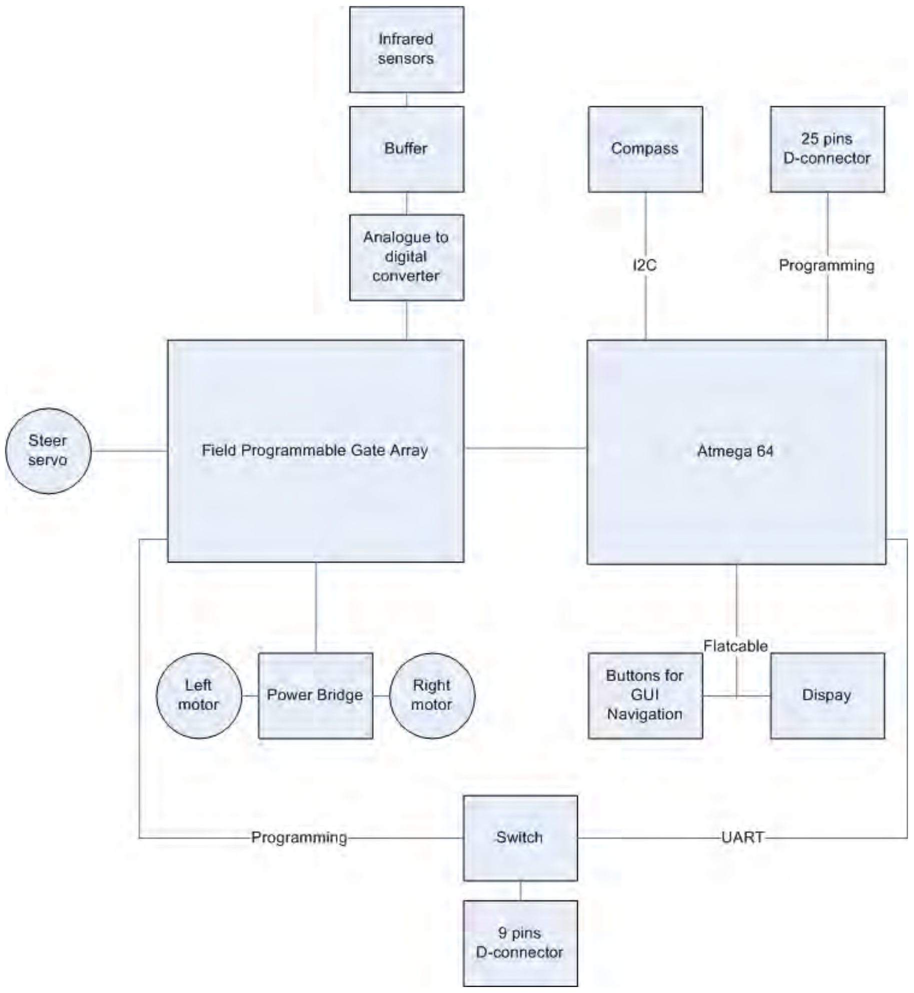


Figure 2.1: Block diagram electronics

1.2.1 Sensors

We have chosen to use four infrared distance sensors. These sensors send an infrared light beam, and calculate the distance when they receive the signal that is reflected by an object. We can read the distance from the sensor by an analogue signal. After the signal passes an analogue/digital converter we send the signal to the FPGA. With these sensors it’s possible to detect distances from approximately 10 till 80 cm.

Figure 1.7: IR sensor

We also used an electronic compass. The compass uses the earth magnetic field to determine where the north is. It sets the direction in its registers, which we can read by the use of the microcontroller. The connection between the magnetic compass and the microcontroller is the I^2C bus.

Figure 1.8: Compass

On both front wheels a sensor detects the rotation speed of each wheel. By measuring this we are able to detect the speed the car is driving, the distance it has. The sensor is based on the same principal as a mouse (with ball). Because we couldn't get the sensors reliable before the race we didn't use these sensors.

1.2.2 Microcontroller

We have chosen to use an ATMega64, because of the number of I/O ports, the availability of an I^2C bus (Two wire Interface) and the UART, to connect the microcontroller to a computer or laptop.
The microcontroller is the main part for the intelligence of the car. To control the car, the microcontroller communicates with the FPGA for reading the sensors and setting the speed of the motors. The microcontroller also handles the text on the display.
During the race the ATMega64 used a clock source of 1 MHz.

1.2.3 FPGA

The Field Programmable Gate Array (FPGA) on our car is small. We used the Pluto board from fpga4fun[1]. The FPGA on the Pluto board is an Altera ACEX EP1K10TC100. FPGAs are programmable logic integrated circuits. We developed the logic for the FPGA in the hardware describing language VHDL.

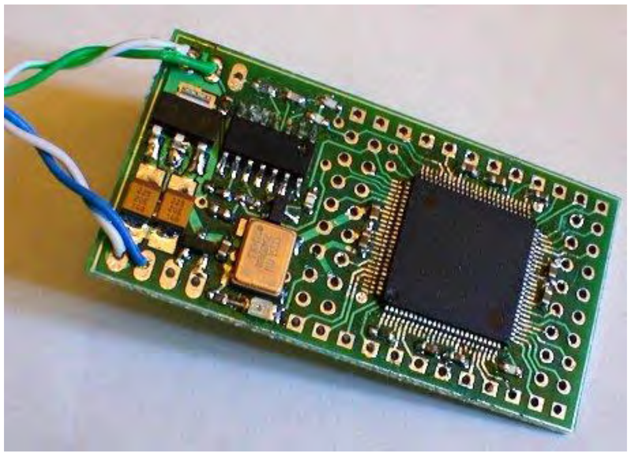


Figure 1.9: Pluto board (FPGA)

1.2.4 Main print

The main print is the connection between all components of the car. On the main print the microcontroller, the FPGA and connectors to other PCB's are placed.

[1] Website fpga4fun: http://www.fpga4fun.com/

1.2.5 User interface

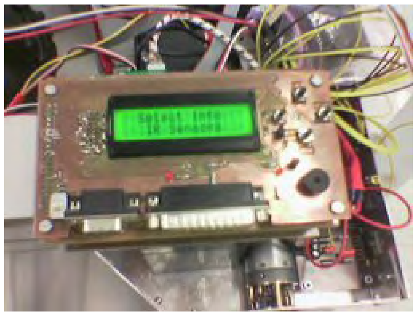


Figure 1.10: Interfaceboard

The user interface consists of five buttons and a display. The display has two lines of sixteen characters. With the buttons we can select different options.
The user interface PCB is also used to program the microcontroller and FPGA, and to read the UART for debug information from the microcontroller.

1.2.6 H-bridge

The H-bridge PCB converts the low-power signal from the FPGA to a high-power PWM signal for the driving motors.

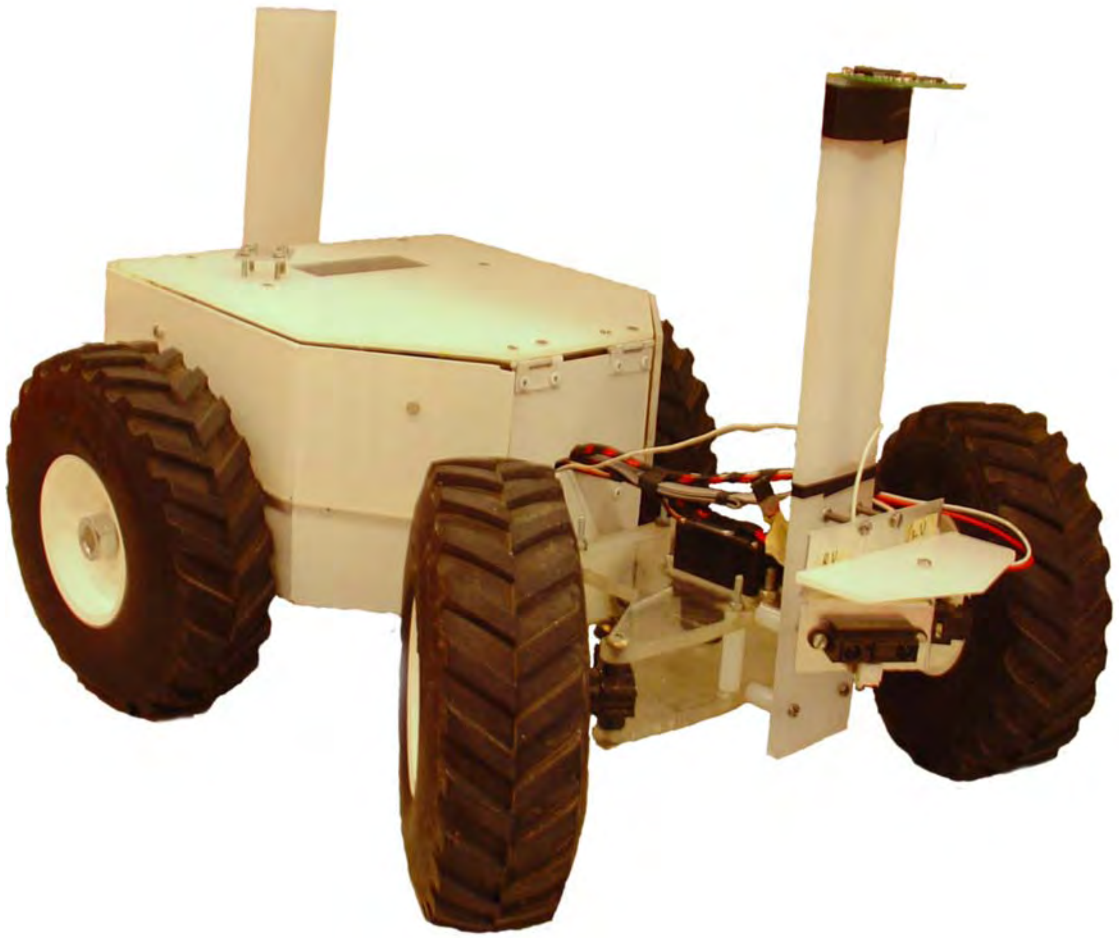


Figure 1.11: Complete Robot

1.3. Intelligence

Before we started to build the car we decided how we wanted to drive. We had 4 infrared sensors and a compass as inputs. As output we had 2 motors and one steer servo. A schematic picture of the car with the sensor placements is shown in figure 1.11. The infrared sensors at the front are placed at an angle of 30^o. We had to drive with the information from these sensors and with those motors.

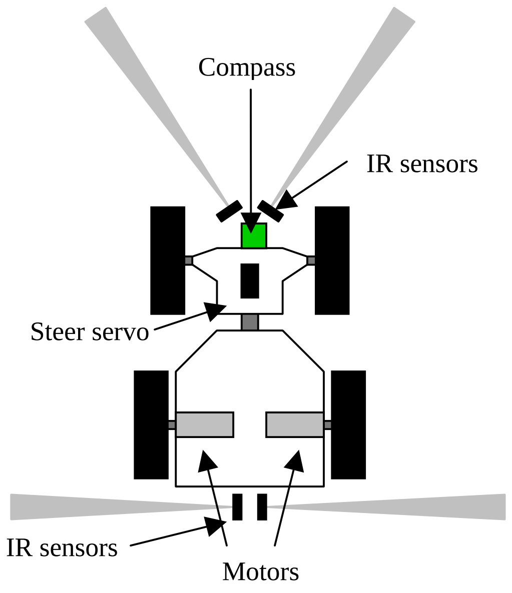


Figure 1.12: Topview car

There were two different competitions. We had to drive in a straight row and in a curved row. For both of them we had programmed different intelligence. First the straight row will be described.

1.3.1 Straight row

At the start of a race the car saves the angle of our compass. This angle is the angle the car needs to drive. When the car is driving it checks the front sensors to determine whether the car is driving in the middle, at the left or at the right of a row.

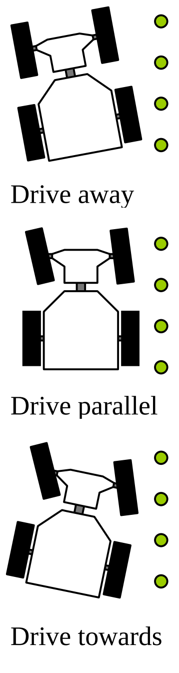


Figure 1.13: Driving style

If we are in the middle of a row the compass is checked. The car reacts as follows:

- If the angle is greater than the desired angle the car steers a little to the left.
- If the angle is smaller than the desired angle the car steers a little to the right.
- If the car is driving with the correct angle, the car steers straight forward.

When the car is closer to the right of the row, we thought of three situations. The car drives to the maize, the car drives parallel to the maize or the car drives away from the maize. (See figure 1.12). The car can determine in which situation it is, by checking the compass. By comparing the measured angle with the desired angle the right situation of these three is determined. The car handles as follows:

- If we drive to the maize, we steer away from the maize.
- If we drive parallel to the maze, we steer a little away from the maize.
- If we are driving away from the maize, we don't steer.

The reason we used this technique is to drive flowing lines. When the car is on the left of the row instead of the right, it reacts the same.

There is a situation which overrules all the situations above. If the front sensors detect a distance closer than the minimal distance, we steer a lot away from the maize and slow down a bit. This is to avoid a collision.

If we don't see objects for a certain time the car has to be at the end of a row. So the car needs to go into the next row. The car steers with a certain angle. This angle is tested so it skips one row. The rear motors are driven at different speeds. This is because the turn is quite short and the inside wheel has to turn slower than the outside wheel. While we turn the car checks the compass. If we turn 180^o the car is straight in front of the right row. When we detect an object we start the normal routine.

The rear infrared sensors are for counting. When driving in a straight row, this is the only thing we do with these sensors.

1.3.2 Curved rows

The front sensors are placed in an angle of 30^o (See figure 1.13). When drawing this a little differently the triangle of figure 1.14 is formed. When the car is parallel to the maize the distance the front sensor measures is twice the distance the rear sensor measures. With this technique it is possible to use almost the same strategy as in the straight row.

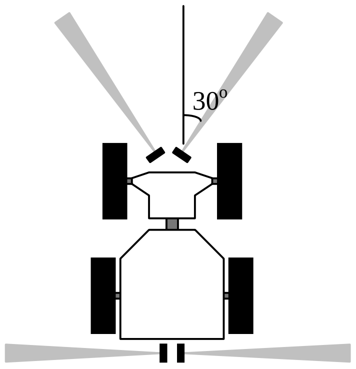


Figure 1.14: Top view car

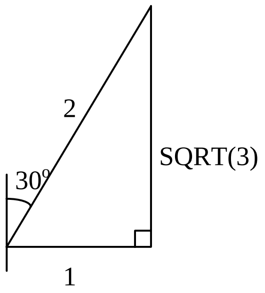


Figure 1.15: Simplifed triangle

First is determined whether the car is in the middle, at the left or at the right of the row. After the car knows on which site it is, it checks if it drives to, parallel or away from the maize. The car can do this by checking if the front sensor is twice the distance of the rear sensor. It does it as follows:

- If the front sensor is less than twice the rear sensors it drives towards the maize.
- If the front sensor is more than twice the rear sensor it drives away from the maize
- If the front sensor is the same as twice the rear sensor the car drives parallel.

How the cars steers is the same as in the straight row.

After testing this technique at a real maize field it didn't seem to work well. We couldn't determine the reason. So we tried exactly the same as with driving through straight rows, only without checking the compass. This worked a lot better. So we decided to keep it that way.

The turn at the end of the row needed to be tighter than at the straight rows. This is done by steering more and making the speed difference between the motors larger.

1.4. Software

1.4.1 Microcontroller

The software for the microcontroller of our car is written in the language C. The final software has a size of approximately 18kB. The software is built of different parts. It consists of some 'hardware drivers' to control the IO. Furthermore, there is a menu part and an intelligence part.

The microcontroller has an UART (Universal Asynchronous Receiver / Transmitter) built in. We use the UART to debug the software. The microcontroller sends all kind of debug information to the UART. With a PC connected to the car we can determine what problems are in the software.

We use a I^2C bus for reading the compass. This is the only device connected to this bus. The microcontroller is the master on this bus.

Every 20 ms the software goes through all the intelligence software. When it is ready with it the processor goes to sleep mode. It wakes itself after 20 ms. When the software awakes it checks all the sensors (compass and 4 infrared sensors) first. Then the processor checks in which mode it is: so driving through straight rows or driving through curved rows. The software then checks the measured inputs and makes decisions on these values. The problem with the infrared sensors is that they have a very small beam. So for most of the time the sensors don't see anything. It is hard to make decisions on that. So the software saves the values of the last 16 measurements in an array. Before the car makes any decision on it, it checks the shortest distance in the array. With this value the software makes its decisions. With this method we made the beam of the infrared sensor kind of wider. The microcontroller determines the speed and steering. But the microcontroller itself doesn't generate the right PWM signals. It was connected to the FPGA which generated the PWM signals. The communication between the FPGA and the microcontroller was a self made simple protocol. The microcontroller always takes the initiative for the communication with the FPGA. Besides sending which PWM signals de FPGA had to generate, the microcontroller also asked the values of the infrared sensors.

1.4.2 FPGA

The function of our FPGA was reading the infrared sensors (via an ADC converter) and setting the right PWM signals. There was an option to measure the distance we drove too, but we didn't use it, because we couldn't get the hardware reliable. A block scheme of the FPGA is shown in figure 1.15. The FPGA continuously reads the AD converters and generates the PWM signals. When the microcontroller wants to know the current value of a sensor, it gets the last measured value. When the microcontroller changes a PWM signal, the FPGA changes the result immediately on its output. There is a led on the Pluto board. It is blinked by the led blink process. This is done so we could easily see if the FPGA was programmed or not.

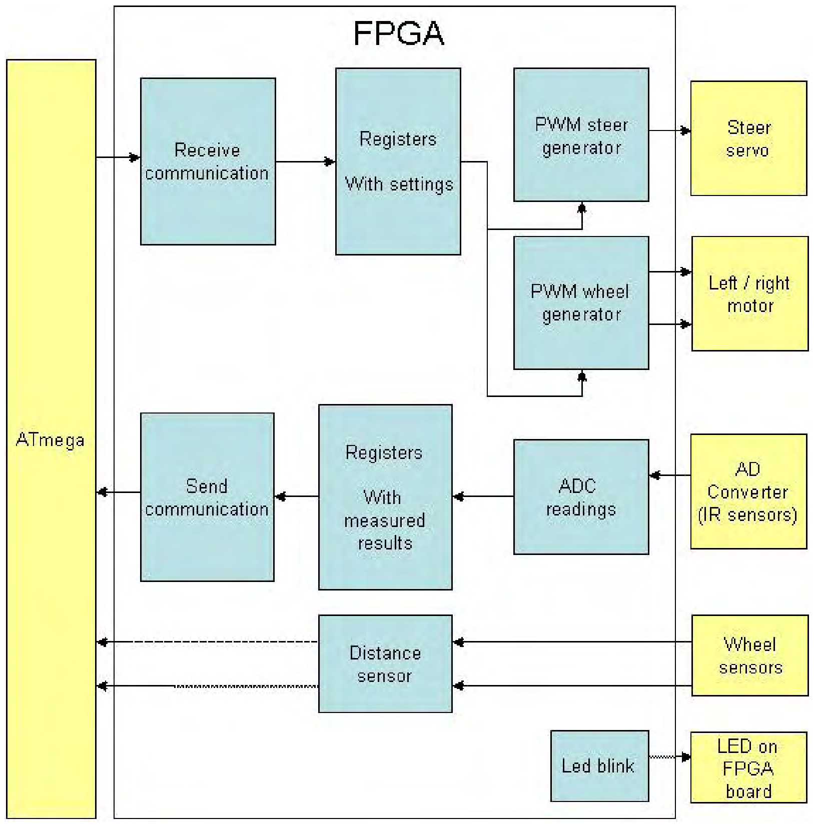


Figure 1.16: Block scheme FPGA

1.5. Freestyle session

We waited a long time before we started thinking about the freestyle session. Our first priority was to build a car that was able to drive between the rows of maize plants, without touching them. When we had just one week before the race we decided to try to build a grass-seeder. When the maize is being reaped, the minerals in the soil won't be absorbed by the plants. This is bad for the environment and causes erosion of the soil. When approximately two weeks before reaping the maize grass is sowed, the minerals will be absorbed by the grass as soon as the maize is reaped.

The grass-seeder we built is placed on a trailer that we can couple behind our car. With a special mode in the software the car drives through the rows of maize. The seeder has a tray which can be filled with grass-seed. On the bottom a rotating tube with holes is placed. When the holes are upside the tube will fill itself with grass-seed. When the holes are downside the seeds will fall out of the tube. The wheels at the back of the car will push the seeds a bit in the soil.
The car will stop seeding when the car is turning to enter the next row.

2. Results and discussion

2.1 Testing

2.1.1 Indoor

To test if the car does what we wanted it to do, we made an 'indoor test field'. We simulated the maize plants with green paper. With this test field we were able to see the car's behavior.

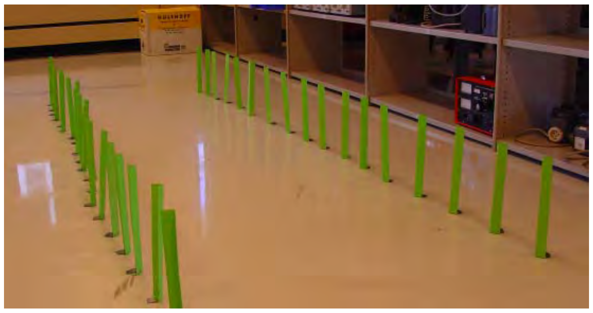


Figure 2.1: Indoor test field

2.1.2 Outdoor

The Field Robot Event is kept in a real maize field so we had to test in a real maize field. We tested in a maize field in Dronten for two days. The conditions were bad, small plants and a lot of big clods of clay.
The first day we spent on getting a better algorithm to drive straight and turn on the headlands skipping one row. The second day we made an algorithm for the curved rows and turning into the next row. These test days were very useful.

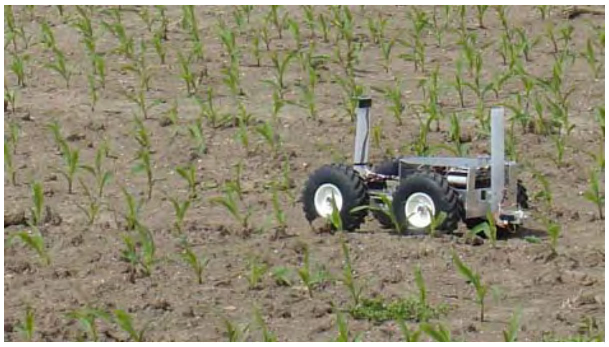


Figure 2.2: Outdoor test field

2.2 Race results

The day before the race the car didn't drive perfectly. The car still very rarely touched some plants and the headland turns were not perfectly performed. So there were still things that could be improved by changing the software. But on race day we were satisfied with the behavior of the car: it is impossible to build a car that drives precisely as the builders would like, in the unstructured environment

Finally it was time to show the performance of our car. We all were very excited. At the straight lanes, the car drove pretty nice. It only went wrong at the headland turns. The car had a big turning circle and drove into the press for two times. Beside that we needed to help the car for two more times. But the car managed to get to the end of the field, with counting the plants. Although our car didn't count all the plants (it missed 1/3 approximately) most other teams were not even close. The curved row performance went very good for us. We didn't need to help the car once! During the curved rows we counted the plants too. At the end of the day we heard that we won this part of the competition.

At last the car had to perform in the freestyle session. Unfortunately the car didn't perform it perfectly. It could show that the car could seed pretty well, although it stopped two times for unknown reasons.

Conclusion

We never expected to win the Field Robot Event. We had a great chassis and very good software, but we had a small budget and compared to some other teams not so much time and not as many team members.
These things might have helped us to think about simple solutions.

Because we used only 4 infrared sensors, and one compass, we didn't need to run a heavy software program. So we had more time to work on the real intelligence of the car. One of the main advantages was the two test days in the real maize field. This gave us very much information about the behavior of our car, and we had enough time to use this information to improve our car.

The weakness of our car was the headland turn. It is hard to go into the right lane, because of the way we steer. Most of the time the car drove into the right lane, but sometimes it drove into a wrong lane. One other issue was counting the plants. We still missed a lot of plants. This is because of our infrared sensors. Those had a slow refresh rate, every 30-35 ms new data was available. Because the plants were thin, and the sensors were 'slow', we missed some plants. This could be solved by using faster sensors.

Overall we can conclude that we didn't work on needless difficult solutions, but we used our time to build a simple car, which is easy to adjust to new requirements. We are very satisfied with the performance of our car.

Field robot „optoMAIZER“ : Development of a mechatronic system based on sensor fusion, a real time operating system and WLAN

Ralph KLOSE, Martin MEIER, Andres LINZ, Arno RUCKELSHAUSEN

Email: a.ruckelshausen@fhos.de

University of Applied Sciences Osnabrück
Faculty of Engineering and Computer Science
Albrechtstr. 30, 49076 Osnabrück, Germany

ABSTRACT

Based on the results of Eye-Maize (Field Robot Event 2004) a new concept based on a real time operating system has been developed. Information for row guidance, positioning for turning and counting plants is based on 21 sensors (8 different types) where the priority of algorithms is given to the low-cost camera CMUcam2. As bidirectional interfaces a display and a WLAN have been implemented. Moroever, powerelectronics for speed and steering control of 2 engines as well as power supply is integrated. WLAN is a powerful tool for developing and characterizing the field robot. The complexity of the mechatronic system has been overcome by the implementation of the real time system RTXtiny. The development of the project is combined with the diploma thesis of Ralph Klose and Martin Meier.

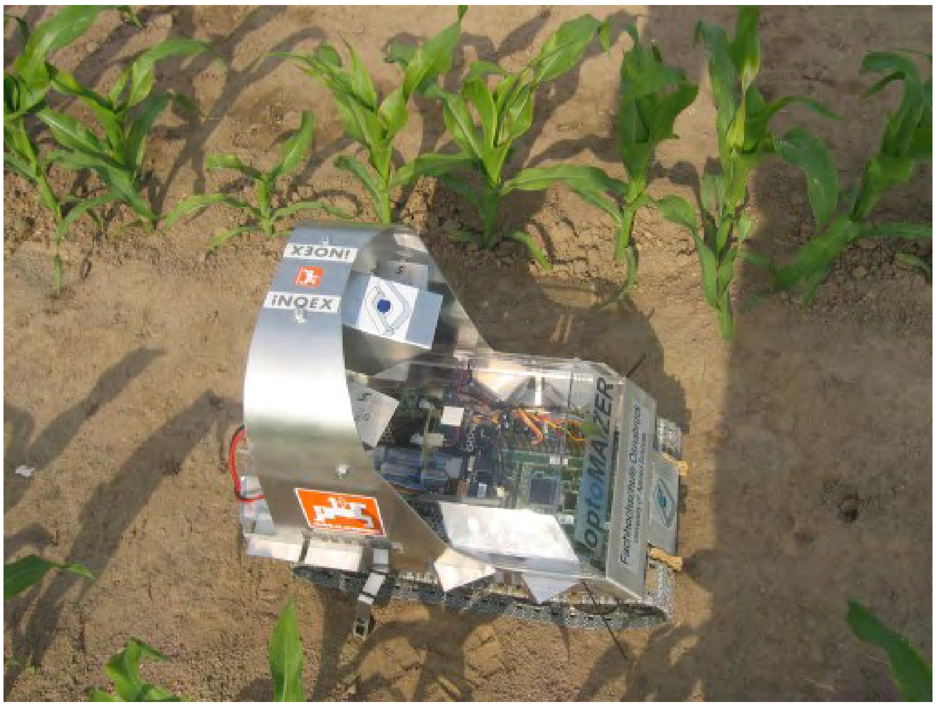


Fig. 1: Field robot *optoMAIZER*

KEYWORDS

Field robot, optoelectronic sensors, sensor fusion, CMOS camera, gyroscope, WLAN, compass, plant count, maize

1. CONCEPT

A new robot optoMAIZER based on the sensor fusion concept of the field robot Eye-Maize [Eye-Maize 2004] has been developed. The basic microcontroller-based system structure is shown in figure 2. According to the tasks of the Field Robot Event sensors for row guidance, turning and counting have been included, moreover redundant sensors for a safe operation are included. The low-cost CMOS camera system CMUCam2 is used for row guidance, the concept is described by the authors (see [CMUCam 2005]). Distance sensors of different ranges and speed support the row guidance and are used for counting the maize plants. A new gyroscope integrated sensor, a compass, mechanical sensors as well as hall sensors are used for the turning process at then of the row as well as for safety. A display serves as a user interface for activitating programs and changing parameters as well as visualizing the status of the system. A WLAN bridge has been integrated to control the system completely via an external PC. The wireless operation is extremely helpful during the development of the field robot, the PC is not used for the operation of the robot in the maize field. Two accumulator batteries are used, moreover a power electronics takes care for speed and steering control of the 2 engines.

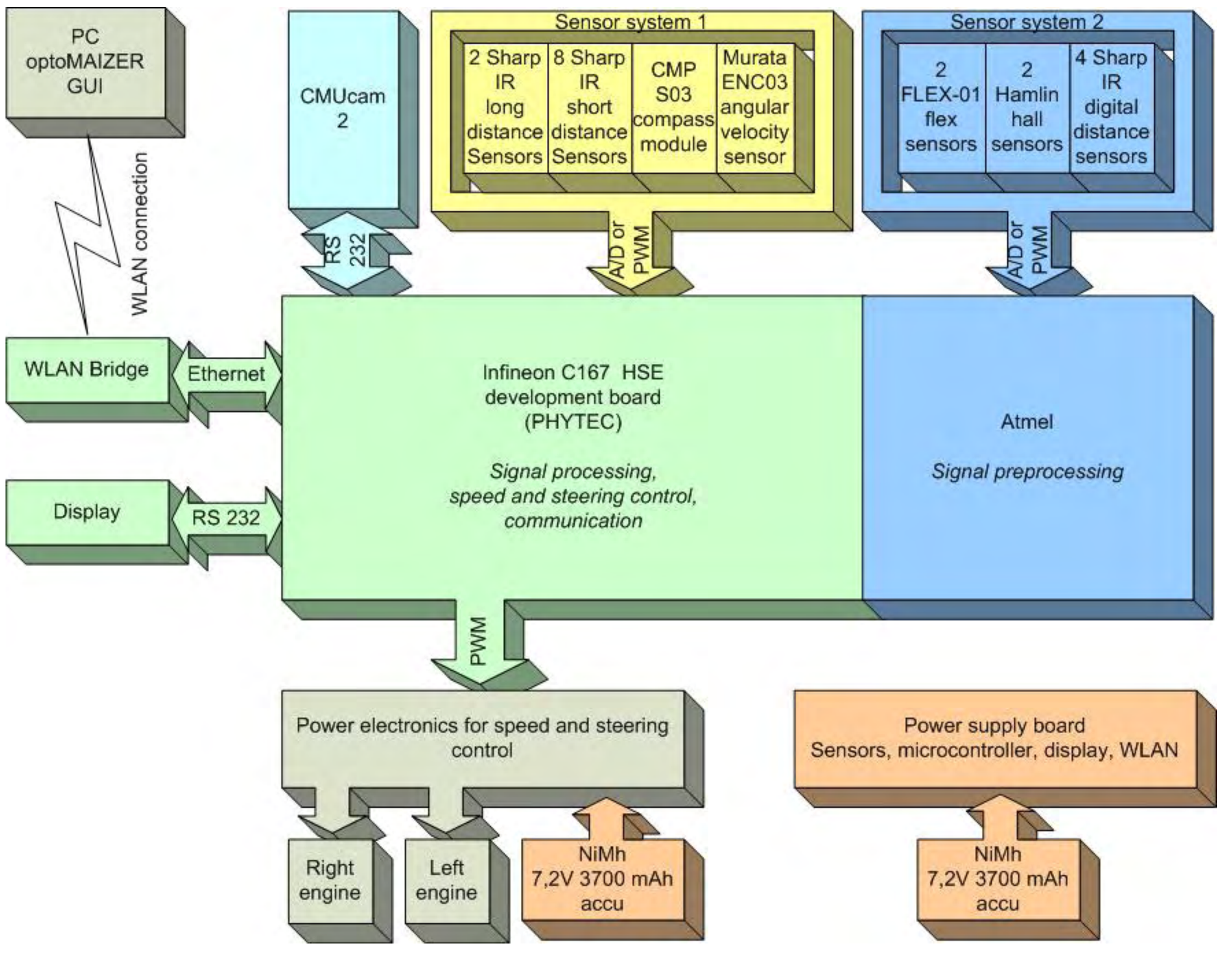

PC optoMAIZER GUI
WLAN connection
CMUcam 2
RS 232
Sensor system 1
2 Sharp IR long distance Sensors
8 Sharp IR short distance Sensors
CMP S03 compass module
Murata ENC03 angular velocity sensor
A/D or PWM
Sensor system 2
2 FLEX-01 flex sensors
2 Hamlin hall sensors
4 Sharp IR digital distance sensors
A/D or PWM
WLAN Bridge
Ethernet
Display
RS 232
Infineon C167 HSE development board (PHYTEC)
Signal processing, speed and steering control, communication
Atmel
Signal preprocessing
PWM
Power electronics for speed and steering control
Right engine
Left engine
NiMh 7,2V 3700 mAh accu
Power supply board Sensors, microcontroller, display, WLAN
NiMh 7,2V 3700 mAh accu

Fig. 2: Electrical block diagram of the concept of optoMAIZER

2. Mechanics

Base unit

The base of our robot consists of a track based model made by "Tamiya" and a self designed case to protect the hardware against rough conditions.

The base model has two electrical engines which are connected to the tracks via a gearbox (fig. 3). This combination makes it possible to control the two tracks individually and to make a turn without moving forward.

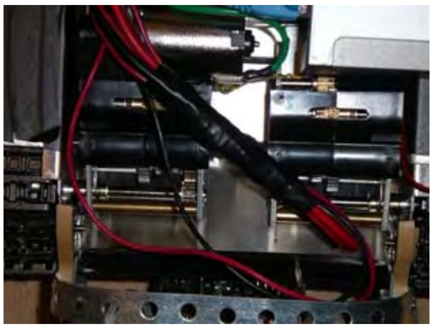


Fig. 3: Gearbox

The model also includes a power electronic module which is connected to the two engines. This power module can be linked to a microcontroller via PWM channels for steering and direction control.

For the case, we decided to use a combination of aluminium and plexiglas (fig. 4). The plexiglas gives the opportunity, for all interessted people or spectators, to have a look inside our robot.

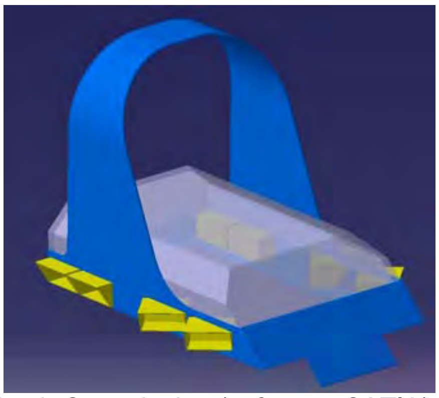


Fig. 4: Case design (software: CATIA)

3. Sensors

CMUcam2

Fig. 5: CMUcam2

The CMUcam2 (fig. 5) consists of a SX52 microcontroller interfaced with an OV6620 or OV7620 Omnivision CMOS camera on a chip that allows simple high level data to be extracted from the camera's streaming video. The CMUcam2 is connected to our controller board over a RS232 link (fig. 6). We have decided to use this type of camera because of the many useful features.

- Track user defined colour blobs at up to 50 Frames Per Second
- Find the centroid of any tracking data
- Gather mean colour and variance data
- Gather a 28 bin histogram of each colour channel
- Transfer a real-time binary bitmap of the tracked pixels in an image
- Arbitrary image windowing

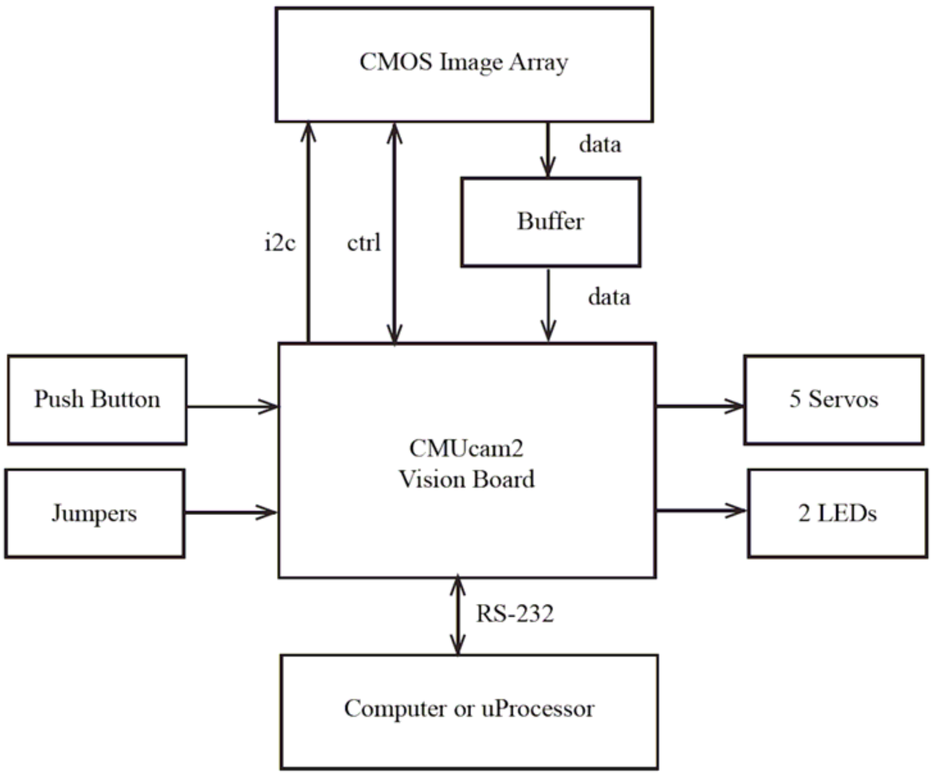


Fig. 6: Functional block diagram CMUcam2

The most important feature of the cam is the colour-tracking functionality (fig. 7). It can be used easily by sending a defined colour-track command over the serial link to the cam. With the use of this command the cam sends back a T-packet. This T-packet contains:

- centroid of the traced colour (X, Y)
- a frame around the pixels of this colour (Xmin, Ymin, Xmax, Ymax)
- number of the tracked pixels
- the occurrence of tracked pixels in the selected area

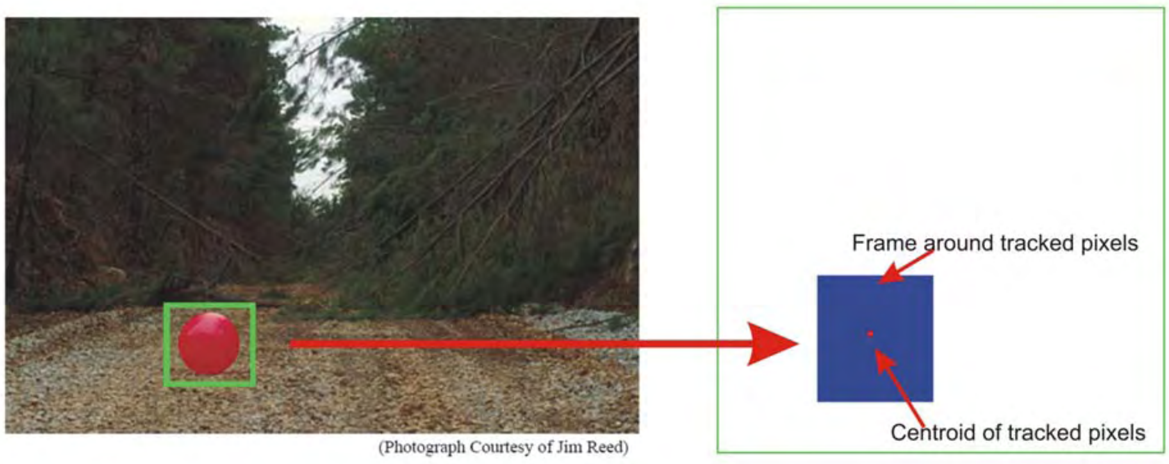

(Photograph Courtesy of Jim Reed)
Frame around tracked pixels
Centroid of tracked pixels

Fig. 7: Colour-tracking of the CMUcam2

The CMUcam2 offers two different colour spaces: RGB and YCrCb (fig. 8). It turned out, that the YCrCb colour space is more robust against illumination changes. The following pictures show what the robot sees when colour-tracking is set to green.

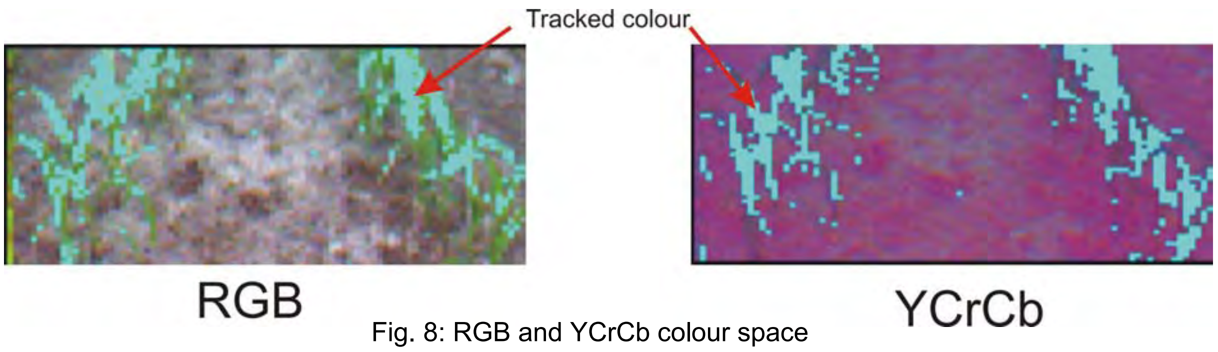


Fig. 8: RGB and YCrCb colour space

The camera offers the possibility to divide the picture into virtual windows (Fig. 9). The track-colour function can be used within each window separately. With this functionality it is possible to search for the colour centroid in different areas of the picture.

We have divided the picture vertical to examine the left and the right row of the field separately.

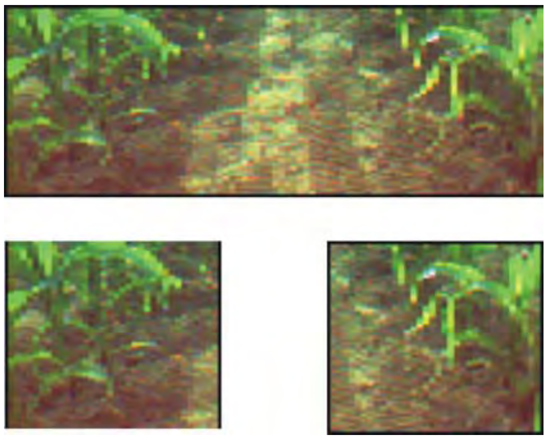


Fig. 9: Virtual windows

With this positon information of the centroid in the different windows a reliable steering decision can be made. We gave the CMUcam2 the highest priority in our multi sensor concept.

Distance sensors

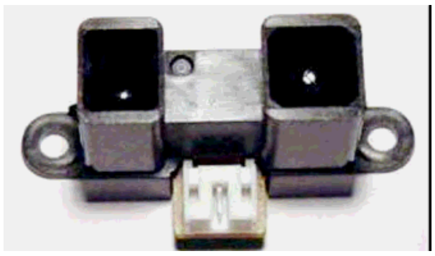


Fig. 10: Sharp GPY0D02YK

For the detection of the maize plants we decided to use two different types of IR sensors. The first type is the Sharp GPY0D02YK long distance IR distance sensor (fig. 10). It can measure distances between 20 cm and 150 cm. Because of their possibility to look far ahead, we have mounted two of them on top of our robot. It is their job to look ahead into the row for the detection curves and to avoid collisions.

The second type is the Sharp GP2D12 short distance IR sensor. It can measure distances between 10 cm and 80 cm. We placed eight of these sensors around the aluminium case. With the information gathered from these sensors, we are able to calculate the position of the robot in the row.

These sensors generate a voltage representing the distance measured. All the distance sensors are connected to the A/D-converter channels of the microcontroller board. One negative aspect about using these kind of sensors is the slow internal conversion speed of about 50 ms. This aspect makes it very difficult to detect the plant while the robot is moving fast.

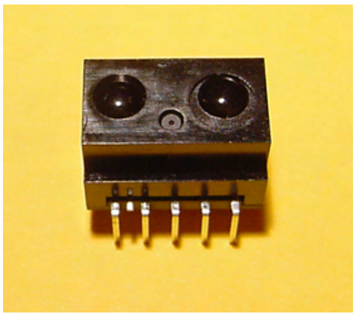


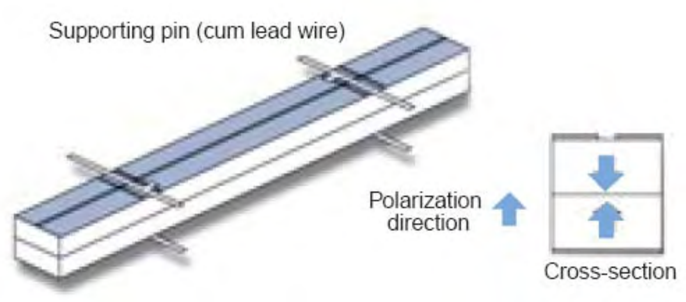
Fig. 11: Sharp GP2Y0D340K

Additionally, with the need to count the plants, we have integrated two Sharp GP2Y0D340K digital IR distance sensors (fig.11). They have a 1 bit digital output which toggles at a defined distance of 60 cm (distance can be changed).

To place these sensors as deep over the ground as possible, we have mounted a holder next to the tracks.

Gyroscope

Because of the need to make an exact turn at the end of each row, a sensor measuring the rotation angle of the robot had to be integrated. After a search on the internet we decided to use the Murata ENC-03 angular velocity sensor. The basic operation of the chip is as follows (Source: ENC-03 datasheet, see fig. 12):



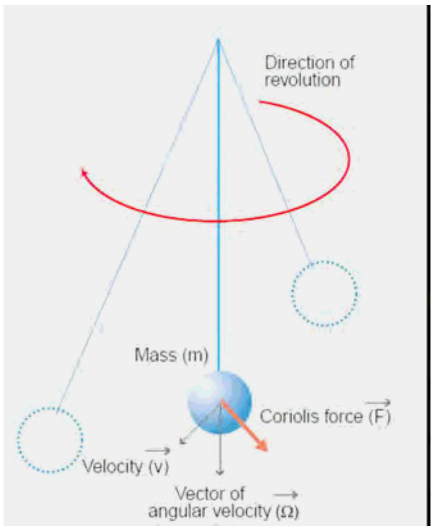

Direction of revolution
Mass (m)
Coriolis force $(\vec{F})$
Velocity $(\vec{v})$
Vector of angular velocity $(\vec{\Omega})$

Fig.12: Basic structure and principle of the angular velocity sensor (Murate ENC-03)

"This gyroscope utilizes the coriolis force. Coriolis force operates in a direction perpendicular to the direction of the motion of the pendulum and is proportional to its velocity. The piezoelectric vibrating gyroscope has its tuning bar vibrator by use of piezoelectric ceramic – this corresponds with the pendulum's vibration – and if this vibrating system is

given a revolving angular velocity, Coriolis force is generated in a direction perpendicular to the original vibration.Since this gyroscope uses a piezoelectric ceramic, Coriolis force can be detected and transformed to electric signals by the basic principle of piezoelectric ceramic."

"The structure of a ceramic bimorph vibrator is shown. It is structurally characterized by a ceramic plate adhered to another, and these two plates are arranged so that their polarized direction are reversed. When voltage is applied to these electrodes, a curvature movement is effectively excited in which one of the plates expands while the other shrinks. In the case of the ceramic bimorph vibrator, voltage is applied to the right and left electrodes formed on the flat upper surface to drive the vibrator. Vibrations are detected by the right and left electrodes on the upper surface of the vibrator, in the same way as the vibrator is driven."

Fig. 13: ENC-03

The ENC-03 (fig. 13) generates a voltage representing the angular velocity in mV/°/s. This voltage is A/D-converted by the microcontroller and integrated to calculate the rotation angle of the robot.

To avoid the effect of temperature drift or to suppress the noise components in the sensor element, a circuit board with a high-pass- and a low-pass-filter had to be developed. The schematic is shown below.

Compass

Fig. 14: Compass module CMP S03

As an additional system to determine the direction of the robot, an electrical compass module was integrated. The compass module Devantech CMP S03 (fig. 14) is the most common module. It can be connected to a microcontroller in two different ways:

1. Use of a PWM channel
2. via I²C bus

The module has an resolution of 0,1° and a calibration functionality which allows the modules to extract other magnetic fields interfering the earth magnetic field.

Mechanical sensors

Fig. 15: Flex sensor

For security reasons we have integrated two flex sensors (fig. 15) at the front of the robot to avoid collisions with plants/walls. They consist of strain gauges which change their resistance for 10kΩ to 20 kΩ when pressed.

4. User Interface

WLAN converter

We decided to integrate a WLAN converter to give the possibility to change the settings of the robot or to transmit the cam pictures to a remote PC.

Fig. 16: D-Link DWL 810+

The D-Link 810+ WLAN converter (fig. 16) is connected to the Ethernet interface of the microcontroller board.

It has the following features:

Standards: IEEE 802.11b IEEE 802.3 Ethernet; Adapter Type: IEEE 802.3 Ethernet to IEEE 802.11b Wireless; Data Security: 64-bit and 128-bit WEP (Wired Equivalent Privacy) Encryption

Network Architecture: Supports Ad-Hoc Mode (Peer-to-Peer without Access Point) or Infrastructure Mode

Data Rate: 11 Mbps

Touchdisplay

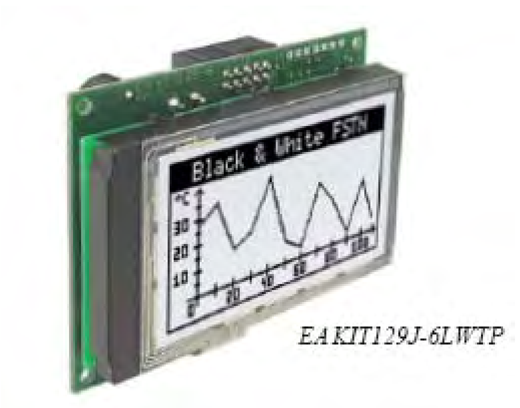


Fig. 17: Tochdisplay made by Electronic Assembly

Additionally we have integrated a touch display to give the user an easy way to control the functions of the robot.

5. Microcontroller system

Atmel controller board

For signal preprocessing we created another board, where the signals of flex-sensors, hall-sensors and plant-count-sensors were analyzed (see fig. 18). The flex-sensors are parts of two comparator-circuits. If the electrical resistance of the sensor is higher than an adjustable level (potentiometer) the comparator will give a “high”-signal, otherwise a “low”-signal. These signals are directly led to the C167-board. The digital signals of the hall-sensors and the plant-count-sensors were analyzed with an Atmel AT902313 microcontroller. As a result of the magnetic field of the electrical engines there are short unwanted peaks in the hall-signals. Because of the relatively huge length of the hall-impulses it is possible to filter them out.

The length of plant-count-impulses depends on the speed of optoMAIZER. In case an an impulse is more or less long than expected, it is probable that the reason for the impulse wasn´t the trunk of a maize-plant and thus it will be ignored. The analysis of the two hall-signals and the up to four plant-count-signals is realized on an atmel AT90S2313 microcontroller. If an impulse is realized as correct it is given to the C167-board.

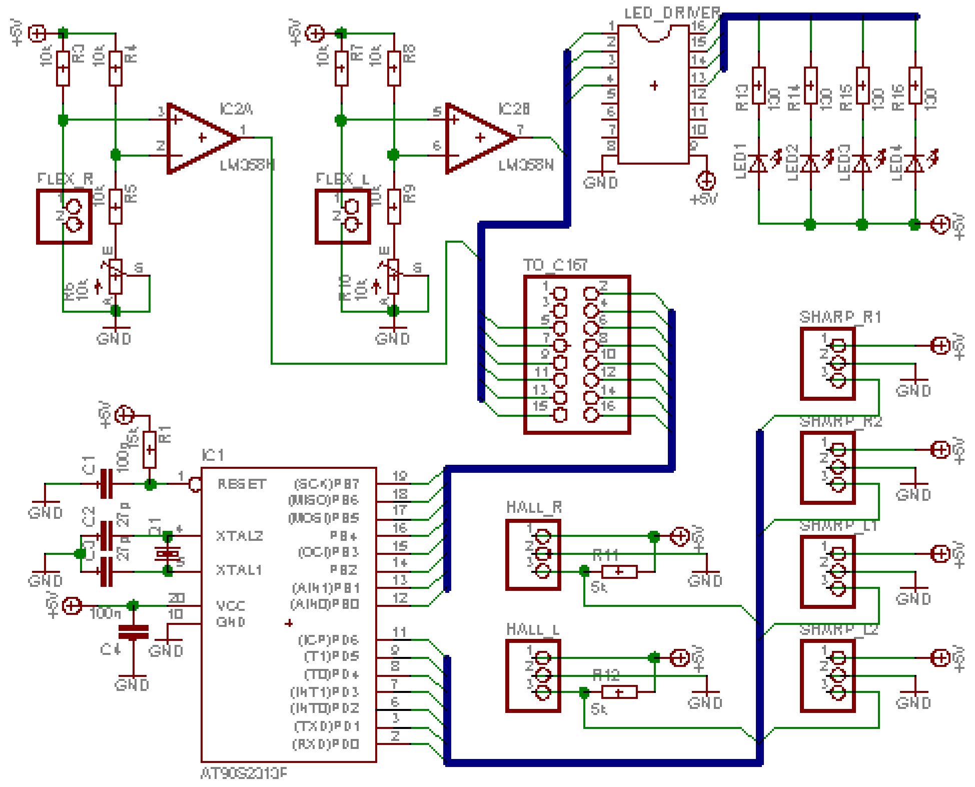


Fig. 18: Schematic of the Atmel board

Phytec development board with Infineon microcontroller

As the main controller board the Phytec development board “phyCore-167 HSE” (fig. 19), equipped with an Infineon C167CS, is used. It handles the information of the IR distance sensors, the cam, the display, the gyroscope and the compass. With these information it can make speed and steering decisions.

Another main task of the controller is to handle the communication with the optoMAIZER graphical user interface over the WLAN link.

The microcontroller's features include:
• High Performance 16-bit CPU with 4-Stage Pipeline
• 80 ns Instruction Cycle Time at 25 MHz CPU Clock,
• up to 40 MHz crystal speed
• 400 ns Multiplication (16 ´ 16 bit), 800 ns Division (32 / 16 bit)
• Enhanced Boolean Bit Manipulation Facilities
• Additional Instructions to Support HLL and Operating Systems
• 16-Priority-Level Interrupt System with 56 Sources, Sample-Rate down to 40 ns
• 3 KBytes On-Chip Internal RAM (IRAM)
• 8 KBytes On-Chip Extension RAM (XRAM)
• 256 KBytes On-Chip Program Flash (Endurance: 100 Program/Erase Cycles min.)
• 4 KBytes On-Chip DataFlash/EEPROM (Endurance: 100,000 Program/Erase Cycles min.)
• On-Chip Peripheral Modules
• 24-Channel 10-bit A/D Converter with Programmable Conversion Time down to 7.8 ms (used for the distance sensors)
• Two Multi-Functional General Purpose Timer Units with 5 Timers
• Two Serial Channels (Synchronous/Asynchronous and High-Speed-Synchronous) used for touchscreen and camera.

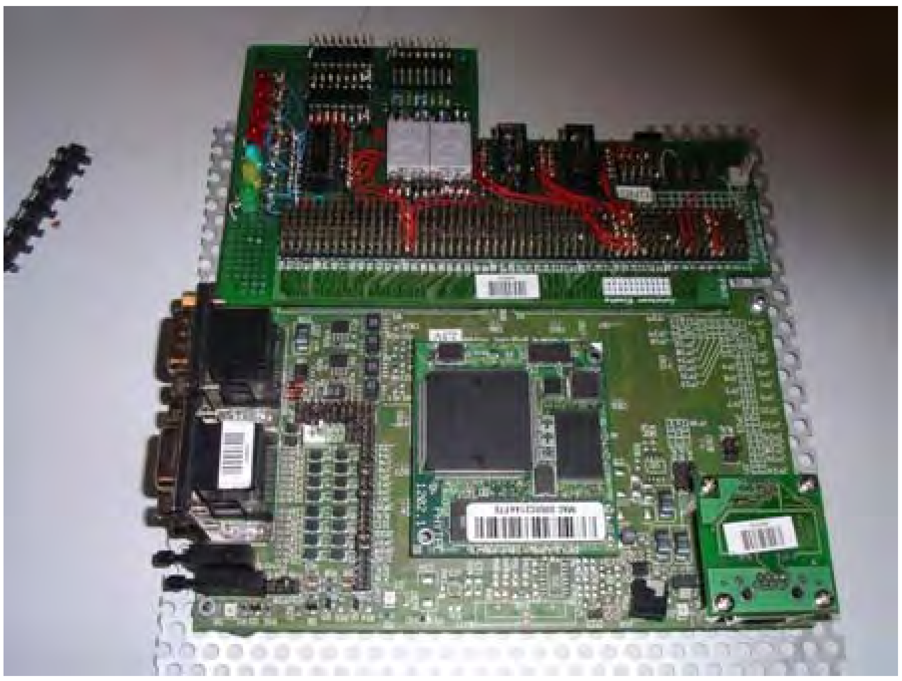


Fig. 19: Phytec controller board

6. SOFTWARE

Real time operating system

Because of the different task the controller has to handle, a real time OS was implemented (Fig 20). We used the RTXtiny real time OS. This OS is made by Keil especially for microcontrollers. It uses round-robin switching and cooperative multitasking. The OS supports the following kernel routines:

- **os_create_task**: Start a new task
- **os_delete_task**: Stop a task
- **os_running_task_id**: Return the ID of the task that is running
- **os_send_signal**: Send a signal from one task to another task
- **isr_send_signal**: Send a signal from an interrupt service routine to a task
- **os_clear_signal**: Clear a previously sent signal
- **os_wait (K_SIG...**: Wait for a signal
- **os_wait (K_TMO...**: Wait for a specified number of OS clock ticks
- **os_wait (K_IVL...**: Wait for a specified interval (since the last **K_IVL** call)

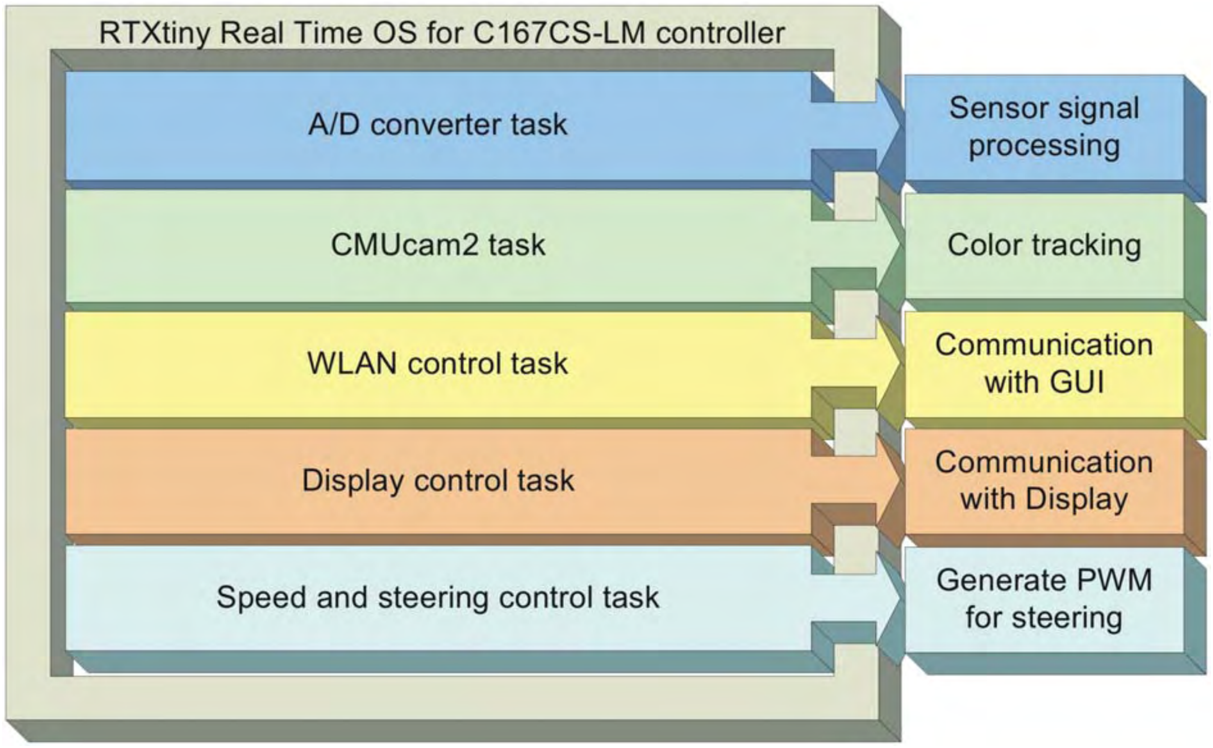
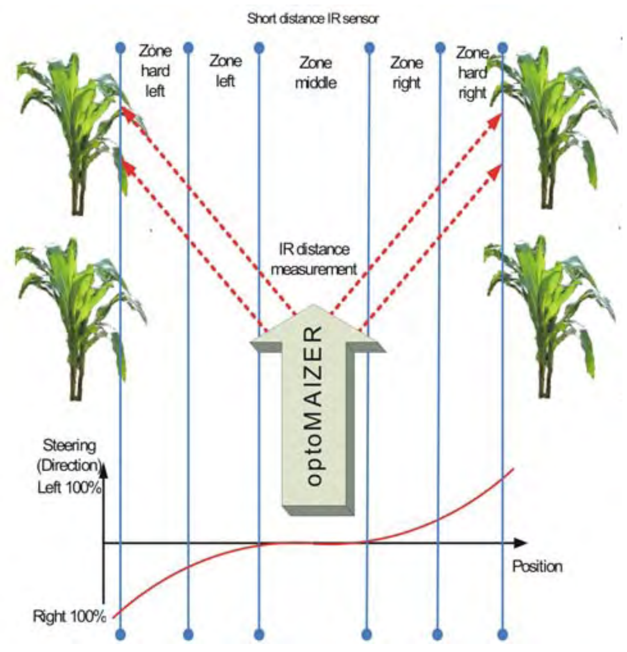

RTXtiny Real Time OS for C167CS-LM controller
A/D converter task
CMUcam2 task
WLAN control task
Display control task
Speed and steering control task
Sensor signal processing
Color tracking
Communication with GUI
Communication with Display
Generate PWM for steering

Fig. 20: Real time OS tasks

The A/D converter task includes the processing of the sensor signals, especially those of the IR distance sensors. It also includes the algorithms using the IR distance sensors. The CMUcam2 task handles the communication with the cam over the serial link as well as the calculation of the direction, using the CMUcam2 algorithms. The WLAN control task includes the whole TCP/IP software stack. In the Display control task only the communication with the touch display is handled. Another important role has the Speed and steering control task. Its main function is to generate the PWM calculated by the different algorithms.

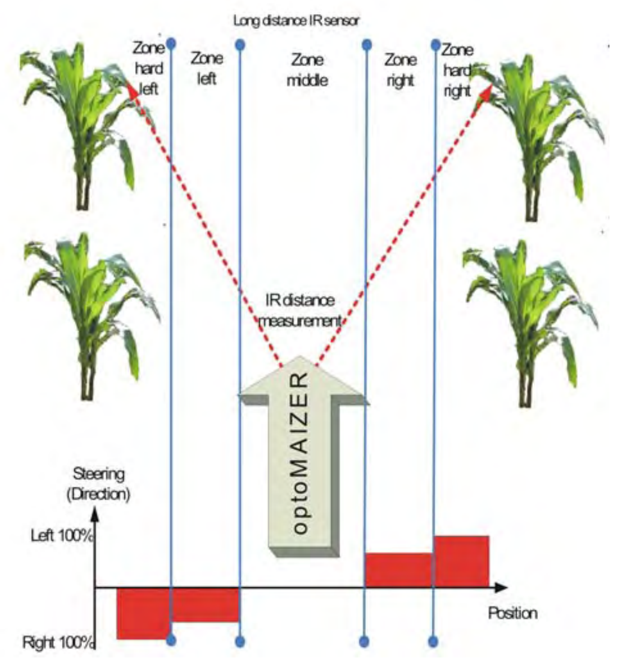
Usage of the short distance IR sensors



The short distance IR sensors are used to determine the exact position of the robot in the row. The position of the robot is divided into different zones in the row (see fig. 21).
These zones are used to calculate the speed and the direction the robot has to drive next.
We used two of the short distance IR sensors at each side of the robot to be sure that at least one of these sensors has measured the distance to a plant.
Additionally we have created an algorithm to detect reliable values.
With these algorithms it is still possible to determine the position of the robot if there are only plants available on one side of the row.

Fig.21: Usage of short distance IR sensors

Usage of the long distance IR sensors



The long distance IR sensors (fig. 22) were integrated for security reasons. Because of their ability to measure long distances, they have the important job to look forward into the maize row. That gives us the possibility to recognize changes in the field like curves or missing plants.

Another important aspect of using these sensors is to avoid collisions with plant.

In this algorithm, the row is also divided into zones which stand for a defined steering direction.

Fig. 22: Usage of long distance IR sensors

Priority of the algorithms

The priority of the algorithms is defined by safety reasons (flex sensors), the presence of plants (long distance IR sensors), first priority row guidance (CMUCam2) and second priority row guidance (short distance IR sensors):

1. flex sensors
2. long distance IR sensors
3. CMUcam2
4. short distance IR sensors

optoMAIZER GUI

The optoMAIZER GUI communicates over the WLAN link with the robot. In this graphical user interface parameters can be changed, algorithms - to be used by the robot - can be selected and the picture of CMUcam2 can be displayed. Another interesting feature is the ability to remote control the robot over the WLAN link.

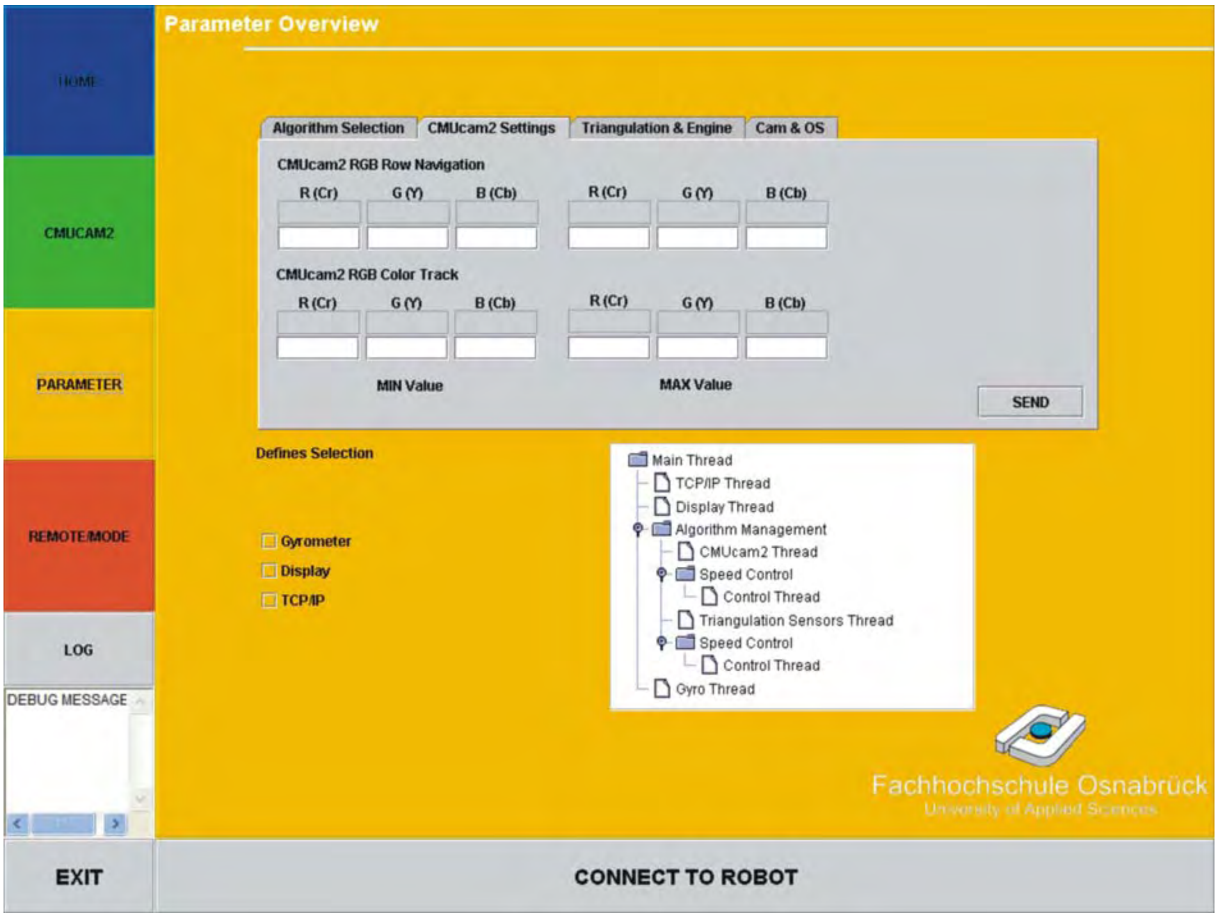


Fig. 23: Screenshot of a GUI window

References

http://www.sharp.co.jp/products/device/lineup/data/pdf/datasheet/gp2y0d02yk_j.pdf
http://www.sharp.co.jp/products/device/lineup/data/pdf/datasheet/gp2d12_j.pdf
http://www.sharp.co.jp/products/device/lineup/data/pdf/datasheet/gp2y0d340_j.pdf
http://www.phytec.de/phytec/fuer_16-bit_module/rapid_development_kit_phycore-167hs_e.html
http://www-2.cs.cmu.edu/~cmucam/cmucam2/
http://www.roboter-teile.de/datasheets/cmps03.pdf
http://www.lcd-module.de/deu/pdf/grafik/kit129-6.pdf
http://www.hamlin.com/images/upload/ByCatalogue/pdf/55100.pdf
http://www.murata.com/sensor/index.html
ftp://ftp10.dlink.com/pdfs/products/DWL-810+/DWL-810+_ds.pdf
http://www.atmel.com/
http://www.keil.com/

[Eye Maize 2004] “Field Robot EYE-MAIZE” ; Frank DIEKMANN, Jens FLEISCHHACKER, Johannes HENKEL, Ralph KLOSE, Torsten KÖNIG, Martin MEIER, Nicola MOCCI, Axel MÜHRING, Daniel NEGD, Tobias NOLTE, Evert NORD, Maik SCHOTMANN, Johann SCHULZ (Student project supervised by N.Emeis, A.Linz, A.Ruckelshausen); Field Robot Event 2004, Wageningen / The Netherlands, Proceedings, ISBN 90-6754-818-9, March 2005

[CMUCam 2005] „Reihenführung autonomer Roboter mit der Low-Cost-Kamera CMUCam“ ; Ralph Klose, Michael Meier, Andreas Linz, Arno Ruckelshausen ; Bornimer Agrartechnische Berichte, 2005 (ISSN 0947-7314), Potsdam-Bornim, to be published

Acknowledgment

We would like to thank all of our generous sponsors:

AMAZONEN-Werke H.Dreyer, Hasbergen-Gaste, Germany
Phytec Technologie Holding AG, Mainz, Germany
Farnell inOne GmbH, Oberhaching, Germany
INOEX GmbH, Bad Oeynhausen, Germany
Electronic Assembly GmbH, Gräfelfing, Germany

Padvinder
an autonomous field robot

Pad·vin·der
~(m) iemand die paden zoekt door onbekend terrein
Path·find·er
~[n] someone who can find paths through unexplored territory

Team:	Arjan Vroegop
	Gerwin ten Brinke
	Martijn Holtman
	Mark Nijenhuis
Contact:	Leon Bronckers
	Hogeschool van Arnhem en Nijmegen
	Faculteit ICA
	Ruitenberglaan 26
	6826 CC Arnhem
	brn@ica.han.nl

Abstract

This paper describes the "Padvinder" robot, a small autonomous robot built to participate in the fieldrobot event organized by the Wageningen University. This paper contains technical design details.

1. Introduction

Once upon a time there were four students at the Hogeschool van Arnhem en Nijmegen (HAN), who needed to think of a graduating project. They've read about the fieldrobot event, and so the "Padvinder" was born.

2. Materials & methods

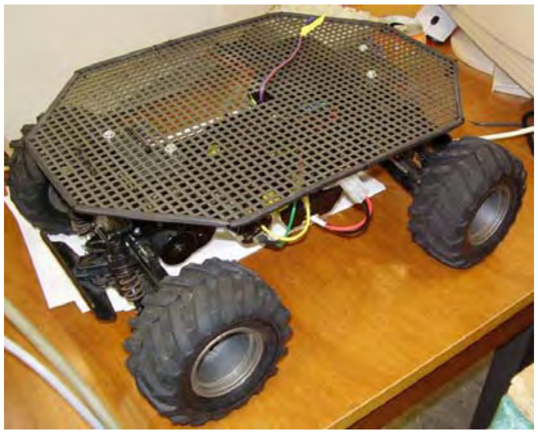
Since we are a group of students, our school has given us a budget of €500 to spend on hardware. Although costs would be the main limiting factor, we also had to think about the time that we could spend on building the robot. Therefore we have mainly chosen for existing parts that would fit in our budget, instead of building them from scratch.

The chassis

The first step was to find a suitable chassis for the robot. At first we were looking for a tracked Kyosho Blizzard chassis, but soon we would find out that the chassis wasn't sold anymore and was rare to be found second hand. Soon after that we found an advertisement of a used Tamiya Wild Dagger. This 4x4 r/c car seemed rigid and suitable enough, so we decided to buy it as a chassis for our field robot.

Specifications:

- 4 wheel drive with differentials
- 2 Mabuchi RS540 motors
- Large 125mm wheels
- 3-step speed control
- 2 wheel servo controlled steering, 4 wheel optional with conversion kit
- Gearbox with changeable ratios
- Size: 40x30x30cm
- Base weight: around 3KG with battery pack



We changed a few things after purchasing the chassis:

- Replaced plastic cover for a metal mounting platform
- The stock springs were replaced for stiffer ones, to support the weight of the extra components
- Changed the gear ratio so the top speed would be lower, however speeds in excess of 20km/h are still possible

A 4-wheel steering conversion was optional, but we couldn't do that because of our budget, the conversion would cost about €50 on additional parts.

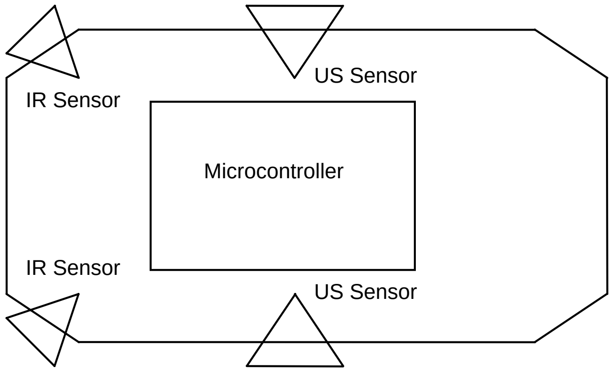
The chassis was fitted with a 3-step mechanical speed controller, which was very imprecise. Therefore we replaced it for an electronic speed controller which operates on a 500Hz PWM signal.

The chassis was further enhanced by adding a bumper and a mounting platform for our maize tickers. The bumper is for mainly for safety reasons and to prevent damage while testing the robot. The bumper is functional, when the robot rams an object it will stop immediately.
The controller and other sensitive electronics are housed in a weather-proof plastic case, placed on top of the mounting platform. The top panel of the case has a large Plexiglas window, so the LCD and status leds can still be seen.

Sensors

There are numerous sensors used on the robot:

- 2x Devantec SRF04 ultrasonic distance sensor
- 2x Sharp GP2Y0A02YK infrared distance sensor
- Heavy duty switch behind the bumper
- 2x Zippy microswitch for the ticker beams
- 2x Reed contact for the wheel revolution counters



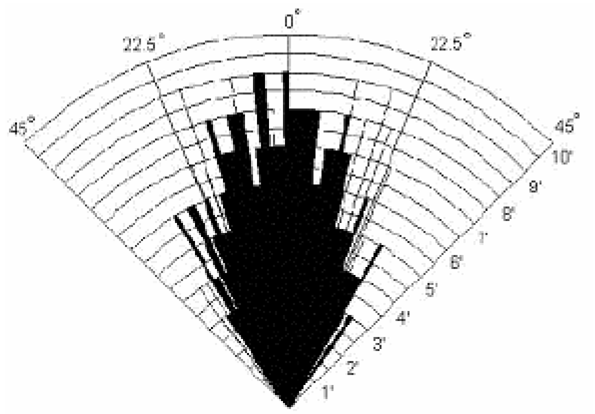
The ultrasonic and infrared sensors are the most important. They are used for navigating in the rows of maize, in fact they are the eyes of the robot.

Devantec SRF04 ultrasonic distance sensor

This distance sensor is placed on both sides of the robot, to measure the distance between the robot and the rows of corn. It has a wide detection angle, so it will always return the distance to the nearest maize plant.

Pros:

- Wide detection angle
- Quite accurate



Cons:

- Needs to triggered for every measurement
- Reflection on smooth surfaces
- Possible interference with other robots

Voltage - 5v only required
Current - 30mA Typical. 50mA Max.
Frequency - 40KHz
Max Range - 3 m
Min Range - 3 cm
Detection angle: approx 54°
Sensitivity - Detect 3cm diameter broom handle at > 2 m
Input Trigger - 10uS Min. TTL level pulse
Echo Pulse - Positive TTL level signal, width proportional to range
Small Size - 43mm x 20mm x 17mm height

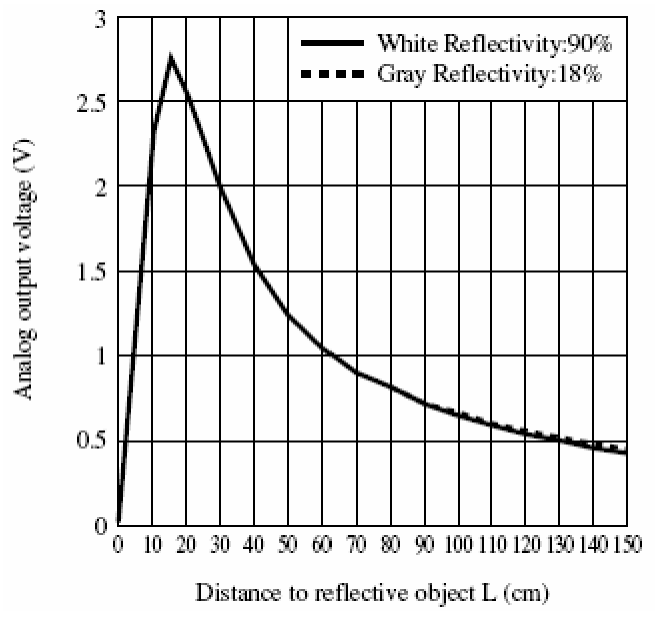
Sharp GP2Y0A02YK infrared distance sensor

These distance sensors are mounted on the front sides of the robot. They have a reasonably narrow and aimed beam, so they can look forward into a row to detect upcoming curves. Used together with the ultrasonic sensors, they are also used to determine the orientation of the robot in a row.
Pros:

- Can measure an exact point
- Fast reaction time

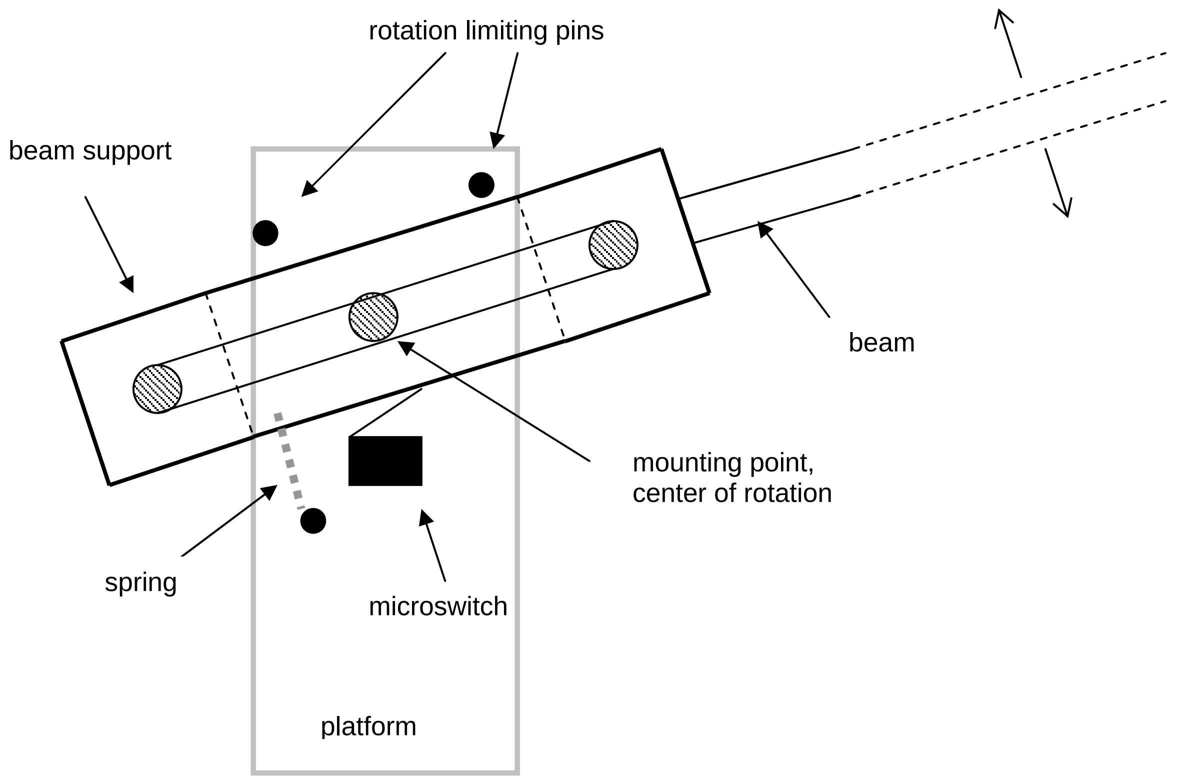
Cons:
- very limited measuring width
- black surfaces absorb IR light
- returned values are quite unstable, also because of limited measuring width

Voltage – 4,5v – 5,5v only required
Current - 33mA Typical. 50mA Max.
Frequency - 40KHz
Max Range – 1,50 m
Min Range - 20 cm



Ticker beams

We have constructed two ticker beams, which can be used to count the maize plants that pass alongside the robot. The beams were constructed in such a way that they have a total width between 50 and 85cm. There is a 10cm overlap (rows are 75cm wide) since the robot won't always be in the exact center of a row. A spring is used to keep the beam pointing outward.



Wheel revolution counter

We have also constructed a basic wheel revolution counter, to monitor the robots driving speed and to aid while turning at the end of a row. It consists of two small magnets and a reed contact switch. The magnets are glued inside a tire and the reed switch is mounted next to it. Every time a magnet passes the reed switch a pulse is given to the microcontroller. Every 2 pulses is a wheel revolution.

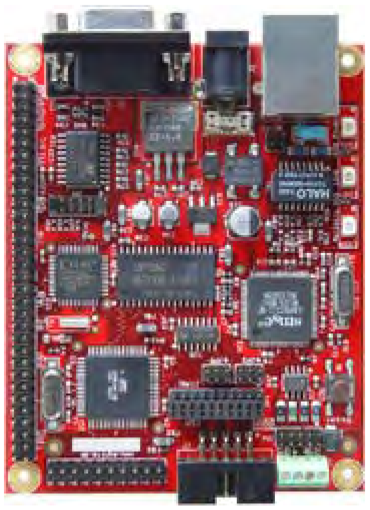
Bumper switch

The bumper switch is nothing more than a simple switch connected to the microcontroller. Every time the switch is activated an interrupt is generated on the microcontroller, so the robot stops driving as soon as possible.

Controller

We were looking for a complete microcontroller board, where price and number of I/O's would be the most important features. We've estimated that we needed at least 16 digital I/O ports and 3 analog I/O ports. After some searching we came across the Ethernut 1.3 microcontroller board. It has the following features:

- Atmel Atmega 128 microcontroller
- 32KB flash ram
- RJ-45 connector for 10MBit Ethernet
- 2 serial connectors (1x RS232)
- 22 Digital I/O's
- 8 Analog I/O's with 10bit A/D convertor
- 2x 8bit timer and 2x 16bit timer, good for 4 separate PWM channels
- Size: 78x98mm
- Price: €139



Pros:

- many I/O channels for all of the sensors
- small footprint
- there's many documentation on the Atmega 128

Cons:

- price

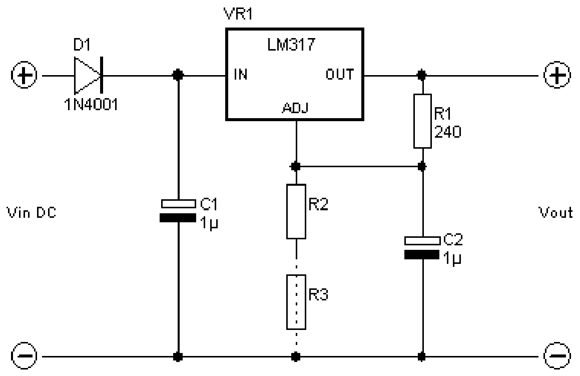
After some negiotation with Egnite Software Gmbh (the producer of the Ethernut boards) we could get a student discount and a starter package would cost us €115.

Electronics

The different systems of the robot all have different demands when it comes to the power supply. A quick overview:

- The Ethernut needs a voltage between 9-12V
- The electronic speed controller needs a voltage between 6-12V
- The sensors need a voltage of 5V exactly

We decided to split the circuitry into three groups. One group is fed from the 7.2v battery pack of the r/c car, this includes the motors and the electronic speed controller. The other 2 groups are fed from a small 1100mAh 9.6v battery pack inside the microcontroller housing. One 9.6v line goes directly to the ethernut board, another goes to a



homemade voltage regulator, based on a LM317 chip. This regulator lowers the 9.6v line to 5v and functions as a power supply for the sensors.

References

Hogeschool van Arnhem en Nijmegen
http://www.han.nl

Ethernut Microcontroller
http://www.ethernut.de

Active-robots webshop
http://www.active-robots.com

RowBo – an autonomous field vehicle

Analysis of the construction and achievements of a crop scouting autonomous vehicle.

Rene van Bruggen
Erik van de Burgwal
Barry van Dongen
Rene Oudman
Willem Rijpkema

Abstract

This paper is about the construction, steering mechanisms, sensors and results of an autonomous field robot designed by a group of students in order to drive through maize rows without causing any damage.

Keywords

Crop scouting, autonomic navigation, field robot

Introduction

For the 2005 field robot event, a team of five students made the RowBo autonomic field robot. These students all study either the BSc or MSc program Agricultural and Bioresource Engineering at Wageningen University. The produced field robot uses ultrasonic sensors to detect its position within the row, and by analyzing this information it adapts its speed and direction. As soon as the end of the row is detected it steers towards the direction in which the next row is expected to be found with the use of an electronic compass.

Materials and methods

Frame

The RowBo chassis is built from aluminium. This chassis has been used by student teams from Wageningen University in the Field Robot Events of 2003 and 2004 as Agrobot 2 and Challenger respectively. By using this existing chassis we did not have to build one ourselves. The steering is realized by skid steering. All four wheels are connected to their own DC engine.

Looking at the competitors of last year, tracks seemed to have advantages over wheels. The winner and also some other upper-region teams used tracks and these tracks seemed to be easier to control then wheels. Last years student-team from Wageningen University planned to use wheels, but problems with slippery wheels on the hoed and wet underground forced them to make primitive tracks around their wheels at the day before the field contest.

We looked for existing suitable track types, but they where not found. There are tracks used for vehicles from model kits, but we could not find tracks that would be able to carry the weight of our robot. Conversations with model building experts learned that solid tracks for our robot type simply do not exist.

We decided to make tracks ourselves. Therefore we used an inside-out indented belt which is normally used for urging. The first try was to make pins on the big wheels and holes in the belt. This forced the track to turn together with the wheels. The approach seemed to work fine in the beginning, but, while still testing it, the pins suddenly missed some holes and the wheels got badly blocked.
The next try was to make edges round the wheels so the belt could turn in between that edges. Special shaped protection parts where made to prevent stones, branches and clods to come in between the belt and the wheel and block them. Testing in very loose ground with many little hard clods showed a new problem. The little clods came in between the wheels and the belt and because of that the tension on the belt became very high, causing a tension that would be harmful for the wheel axles and bearings.

We had bad experiences with the tracks and lack of time prevented further development of this concept. Since it was proven by several robots in previous Field Robot Events that it is also possible to perform well without tracks, we decided to abandon the idea of using tracks, and chose for wheels instead. A difference this year with last year is that this years conditions where a lot less humid. With a good motor controller and not too much rain, the control over the speed of the wheels should be good enough.

We learned that the problem of using tracks for a little vehicle is that the clods are relatively hard compared to the weight of the vehicle. Big machines that use tracks are so strong and heavy that the clods are easily flattened between the wheel and the track. For little robots the relative forces necessary for this are a lot higher.

Control Hardware

The control hardware that was used last year by the Challenger was an open loop control using Pulse Width Modulation (PWM) and H-bridges. This control hardware needed improvement because field conditions are too unpredictable for an open loop control. To close the loop, the control software should not only know what the desired speed and direction are, but also the actual speed and direction. This would be very difficult because to achieve this not only the actual wheel speed has to be known but also the slip between wheels and surface. To make life easier we neglected the slip and used the actual wheel speed as feedback parameter.

To measure the wheel speed optical encoders were attached to the DC motors. The motors have a maximum speed of 7220 rpm and the optical encoders have an output signal of 500 pulses per rotation. This results in a very precise speed measurement, but it also means that the control hardware has to handle a feedback signal of more than 60 kHz! For electrical engineers this might sound as a challenge, for agricultural engineers like us it is a bit too much. Therefore we decided to buy an 'off the shelf' solution. After some research on the internet we found a solution that fitted our problem very well: the RoboteQ AX3500 motor controller.

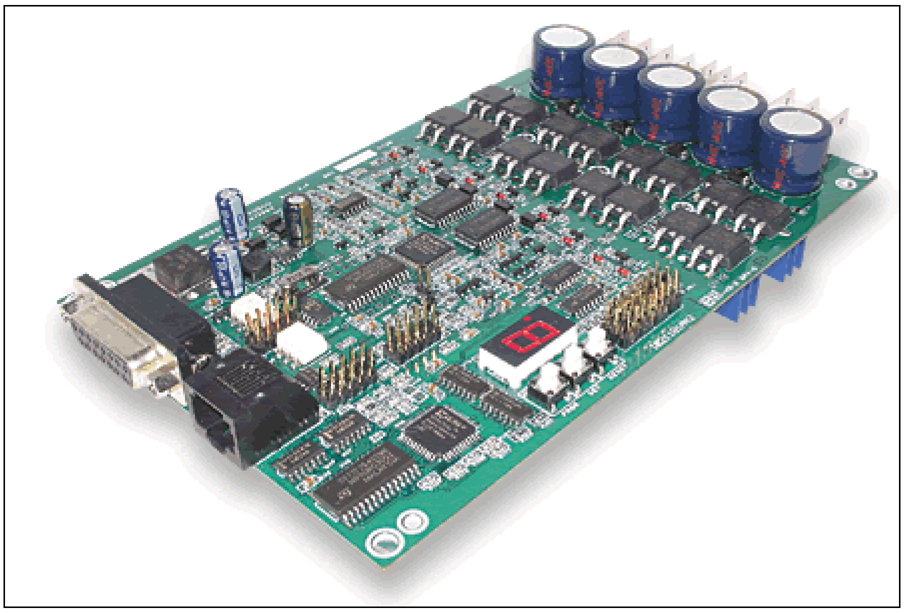


Figure 1. RoboteQ's AX3500 motor controller (RoboteQ 2005)

RoboteQ's AX3500 is a microcomputer-based dual channel DC motor controller capable of directly driving up to 60 continuous Amps on each channel at up to 40V. The AX3500 is targeted at designers of mobile robotic vehicles including Automatic Guided Vehicles (AGV), Underwater Remote Operated Vehicles (ROVs), and mobile robots for exploration, hazardous material handling, and military and surveillance applications (RoboteQ 2005).

Fitted on a compact 4.2" x 6.75" board, and targeted primarily to OEMs, the controller accepts commands from either standard R/C radio, for simple remote controlled robot applications, or serial port interface. Using the serial port, the AX3500 can be used to design fully or semi-autonomous robots by connecting it to single board computers, wireless modems or wireless LAN adapters. The AX3500 is fitted with a dual encoder input module.

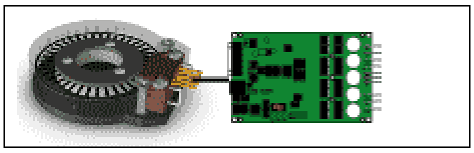


Figure 2. Dual encoder input module (RoboteQ 2005)

Use of sensors

In a maize field, there are several ways to detect plant rows. It is possible to recognize the maize either as an object or based on the colour difference to its background.

Colour based detection.

Quite some effort was put in investigating the possibilities of using a camera for both path and end of row detection. The problem we encountered was that real time conversion of raw visual data by algorithms created in Visual Basic was to slow for direct steering purposes (about 1 frame/s). Therefore we chose not to use a camera.

Obstacle based detection.

For obstacle based detection we focussed on three types of sensors;

1. Mechanical whiskers
2. Infrared distance sensors
3. Ultrasonic distance sensors

Mechanical whiskers

Obstacle based detection can be done by simply putting whiskers at both sides of the robot. Also there is a simple possibility to measure the distance between the robot and the plants. This can be done by a potentiometer and an A-D converter. A disadvantage of using whiskers is the contact to the plants. In the early stage of plant development, the impact of the field conditions to the plants is unknown. A very light whisker will be influenced by leafs of the plants, instead a heavier one may oversee some plants. Furthermore it can damage plants, particularly when plants are in an early growing stage. In our project we decided to avoid any contact between plants and robot. This decision is mainly based on the uncertainty of the field conditions.

Infrared distance measuring

Figure 3. The Sharp GP2D12[1]

[1]http://**www.junun.org**

On our robot, two kinds of infrared sensors where used. The Sharp GP2D12, used in many (small) robot projects. The GP2D12 has a useful range of almost 35 cm. Due to its very narrow beam width (3-5 cm) many measurements are necessary to effectively detect maize plants during driving. For example a speed of only 3 m/s needs a sample speed of more then 100 Hz per sensor. If for example four sensors are used the sample unit should be able to sample at 400 Hz. Thereby also calculations have to be done.
Another problem will come up because the sensor refreshes the output only every 38.3 ± 9.6 ms, this means 25 Hz. Without skipping any places the maximum speed of the robot will be below 1 m/s. For our goals this speed is too low.
An advantage of this narrow beam width is the possibility to count the number of plants if the high sample speed is reached.
On our robot the GP2D12's are placed at the front on both sides of the robot. Unfortunately it was not possible for us to get rid of the high sample speed in combination with reasonable calculations. So in the end the sensors remained unused at the robot.
Time is also been invested at the GP2Y0A21YK. The beam width of this sensor is approximately twice the beam width of the Sharp GP2D12. In this way the sampling speed can be reduced. Hereby the possibility to count the number of plants is more difficult. Because we mainly would like to use the infra red sensors for counting the number of plants, the GP2Y0A21YK is not implemented at our robot.

Figure 4. The Sharp GP2Y0A02YK[2]

The Sharp GP2Y0A02YK is a long range (1.80 m) infrared sensor. Reasonable data is given up to 80 cm distance. Two of these sensors are mounted at the front side of the robot. These sensors were placed here in order to give information about the crop plants at a larger distance of the robot.
With this sensor the same problem occurred as we had with the GP2D12 sensors. The sampling time of hard- en software was too low for using these sensors on our robot.
A disadvantage of all infrared sensors is the influence of material and sunlight to the reflection. Accurate calibration and suitable weather are needed for proper functioning.

[2] http://www.junun.org

Ultrasonic Sensors

Figure 5. The SRF08[3]

Steering of the robot is mainly based on six ultrasonic sensors. These sensors are mounted at six locations at both sides of the robot.
The SRF08 is a very accurate active sensor (max 3 cm deviation, max distance 6 meter). The SRF08 uses sonar at a frequency of 40 kHz to detect objects. A 40 kHz pulse is transmitted and the receiving device listens for reflections. Based on the travelling time of the transmitted pulse the distance to the objects can be estimated. All sensors are connected to an I^2C bus. This improves the communication with the sensors. The first output fires four sensors. These are two front sensors and two back sensors. They all send a pulse at the same time. In this way interference of the signal is prevented.
After 6.5 ms the maximum distance (60 cm) should be reached. Then it is possible to read the sensors. After this process the middle sensors where fired and read. Inclusive sending the sensor information to the computer, the Basic Atom fires each sonar 14 times a second. An advantage of the SRF08 is the wide beam width of almost 40 degrees. In the maize field a single sensor is able to detect almost everything that is ahead of him. This strongly decreases the possibility of "missing" plants. Sometimes a disadvantage of the sensor is the accuracy. When leafs are hanging in the rows, they are detected as if they where plant trunks.

Control software

In the row the navigation strategy of the robot is totally based on the six ultrasonic sensors. The basis of the software is designed for navigating between two straight walls. Although maize rows are not exactly walls, the software handles them like walls.
These walls are virtually built by regression of sensor information. Therefore both the place and direction of the sensors have to be known exactly. These places are transformed to a coordinate system where the middle front of the robot is point 0.0, See table 1.

Sensornr	Location	X	Y	Heading
0	Left front	-5	0	-60
1	Right front	5	0	60
2	Left middle	-13	-3	-90
3	Right middle	13	-3	90
4	Left rear	-10	-55	-90
5	Right rear	10	-55	90

Table 1: RowBo's coordinate system

[3] http://www.junun.org

Adding the measured distance on the sensor position in the specified angle gives three points at both sides of measured obstacles. By linear regression, two obstacle 'walls' are created. The position and the angle of the robot are calculated compared to both walls. In this step information of previous positions is also taken into account. Based on a predefined relation between position and angle, steering values are calculated.
With this method it is possible to use several and different sensors. This method also gives a complete view; there is not only information of the front side of the robot but also from the rear side (see figure 6). The middle line represents the robot. The left and right lines, build up out of the measured points, simulate the both rows. The red line shows the 90 degrees turned angle between both rows and the robot. Over that line also the distance from the row to robot is been calculated.

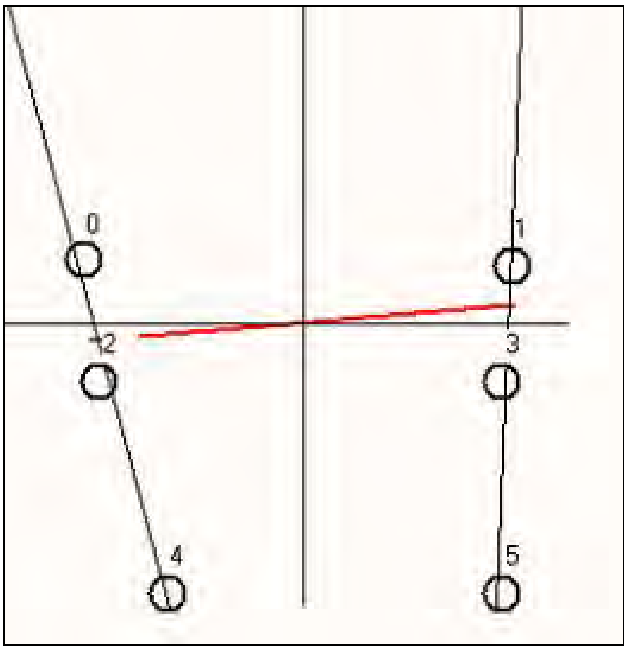


Figure 6. Overview of calculation output

A disadvantage of the use of this method in combination with the ultrasonic sensors is that it also reacts on plant leaves. Besides, the regression method is not very accurate when the vector angle is high (>100). At first sight the calculations were made for at least 10 sensors. Afterwards it was better to have a more simple calculation for the direction and place of the rows.

Steering mechanisms

The steering system is based on information provided by the ultrasonic sensors via a Basic Atom 40m to an Epia mini ITX micro PC. The PC calculates the angle at which the RowBo is positioned compared to the maize row. This determines the ratio between the left and the right wheel speed. This angle also affects the driving speed: a bigger angle will result in a lower driving speed. Another thing affecting the driving speed is the distance between the vehicle and the plants detected. A wider path results in a higher driving speed compared to a closer passage.

Several functions have been programmed to improve performance, like:
- Ignore missing plants
- Ignore erroneous measurements
- Relative sensor-weights
- Accuracy dependent speed.

The end of row detection of RowBo is based upon the same infrared information. It is calculated by an algorithm in which front- and rear-sensor information is used. If RowBo notices that it has reached the end of the row, it looks back at the driving direction of the last few meters. This direction was calculated and stored from the compass data. With this data RowBo is able to accurately and smoothly turn 90 degree, drive over the headland, and make another 90 degree turn. After this it drives back into the next row. There the drive-row algorithm starts again, until it reaches the other end of the row.

Results and discussion

Control Hardware

The control hardware, with the RoboteQ AX 3500, receiving its orders from the higher intelligence levels, is now very effective. The motor controller is continuously trying to get the measured speed as close to the desired speed as possible with a PID feedback loop. Despite the relative high costs of this motor controller we really think it was worth the money. During the tests we only encountered two minor problems.

In the first place the communication between the AX3500 and the Epia mini ITX motherboard got messed up sometimes. This expressed itself with RowBo accelerating to full speed with no way of stopping it but disconnecting the batteries. We can be happy about the fact that the maximum damage done was just some bruised shin-bones. Later on we found that some of the communication cables made short circuit through the aluminium chassis. After fixing this the problem seems to be solved.

Furthermore we used the R/C radio control of the AX3500 motor controller for test purposes a few times. Then we discovered that switching between RS232-control to R/C control is not very easy. To do this the motor controller must be programmed through a serial cable with the PC Configuration and Updating Utility 'RoboRun'. It is clear that the motor controller's software still has some minor bugs. It really takes some 'Fingerspitzegefühl' to successfully switch between the different control modes.

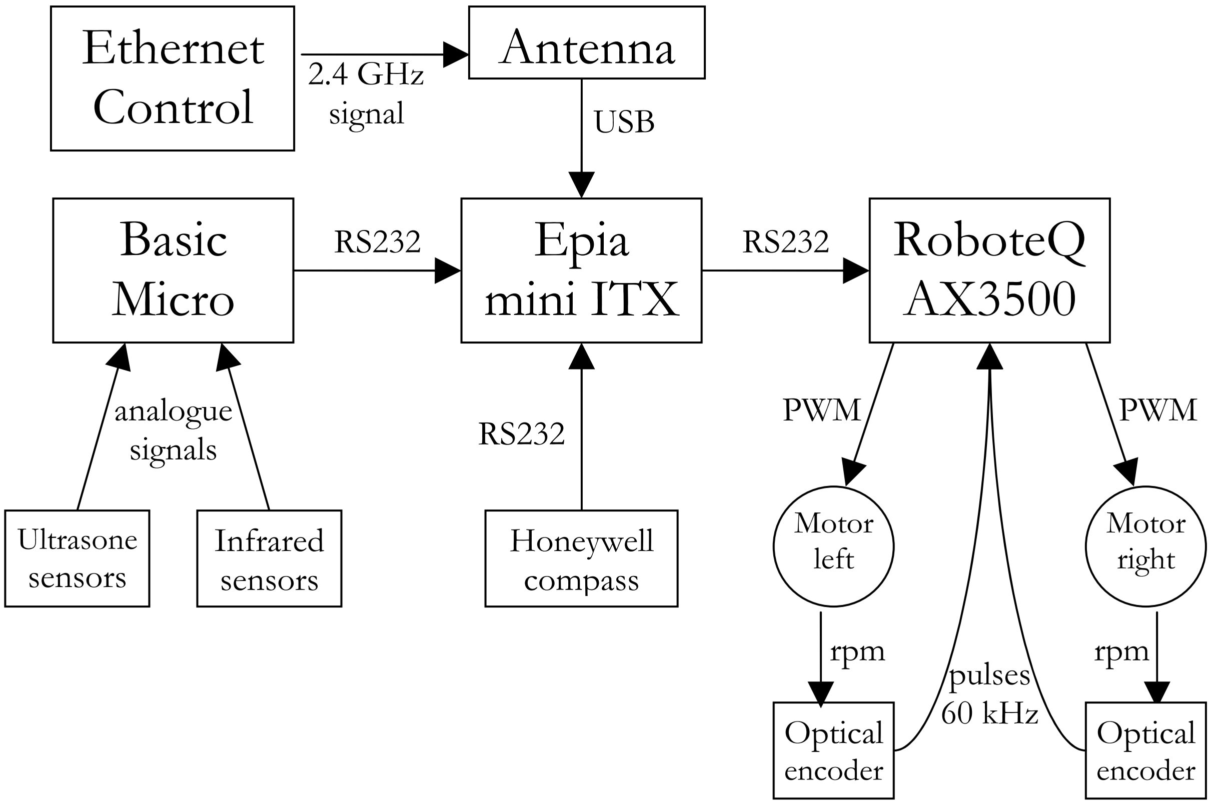


Figure 7. Information traffic between different components

Sensor, Algorithm / software

Although RowBo proved to detect the crop row properly, it sometimes nearly hit the maize rows. It seems that for some reason, the steering algorithm is a little to 'slow', and correcting only happens when it is almost too late. In the contest no problems occurred, as we set the maximum speed quite low. To safely increase the maximum speed RowBo should drive more in the centre of the row.
A problem that occurred with our sensor systems was that when RowBo is very close to the maize plants the sensor beams can easily miss the plants, and therefore the robot will steer in the direction of the plants. An even wider beam of the sensors could fix that problem.
Furthermore the use of image processing software would be very useful. Team RowBo did not find time to implement this, but it definitely should be possible with software packages like Labview (National Instruments), which currently supports Logitech web cams.

Conclusions

RowBo performed quite well in the field robot event. Compared to most other robots, it offered a pretty safe ride through the maize rows. It showed that an autonomous vehicle can navigate through maize rows without using visual data. The driving speed was not very high, which resulted in a lower score on some of the programme items. Despite this handicap it was still capable of scoring a reasonable fourth place in the event from a total of eleven teams. If we had used a camera, RowBo could have watched further ahead, and thus a higher driving speed would be possible. However, we are satisfied with this result.

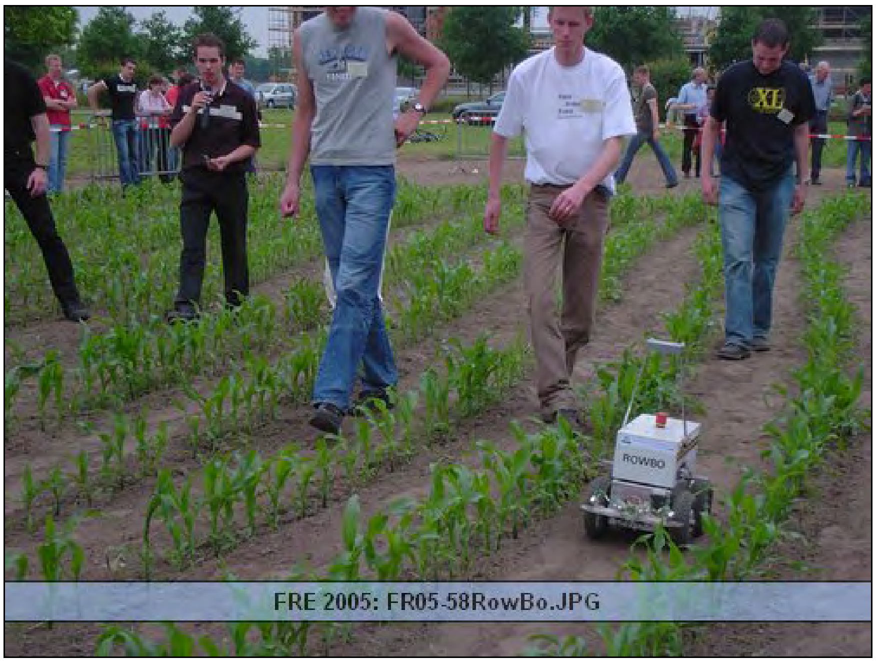


Fig 8. RowBo in autonomous movement[4]

[4] http://www.fieldrobot.nl

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY

HELSINGIN YLIOPISTO

The Development of an Autonomous Robot for Outdoor Conditions

M. Honkanen [1], K. Kannas [1], H. Suna [1], J. Syvänne [1], T. Oksanen (advisor) [1]
H. Gröhn [2], M. Hakojärvi [2], A. Kyrö [2], M. Selinheimo [2], J. Säteri [2], J. Tiusanen (advisor) [2]

[1]Automation Technology Laboratory
Department of Automation and Systems Technology
Helsinki University of Technology

[2] Department of Agrotechnology
Helsinki University

Abstract

The autonomous robot SMARTWHEELS described in this document was built up in collaboration by two student groups during the semesters 2004-2005. The purpose was to learn project working and to test the usability of minimal cost camera vision system with laptop in outdoor conditions. RC-platform modules were used to enable easy assembly and integration and to achieve high speed. Basic web-cam, compass and ultrasonic sensors are used for navigation, servos and drives for motor control. Portable computer works as the main processing hardware and the windows based software has all the higher functionality and intelligence. The PC-software handles the camera interface and communicates through a serial connection with the microcontroller that is used to read the sensor data and to control the motors and servos.

Keywords

Autonomous robot, Vision-based navigation, Hough transform, RC-platform

Introduction

A lot of research is currently focused on service and field robotics and the number of commercial applications is rising. Automatic vacuum cleaners and lawnmowers are already in the markets and changing everyday life. Autonomous field robots are not far from becoming a part of farmer's tools for simple operations such as fertilizing, crop and environment monitoring and reporting. The Field Robot Event 2005 organized by Wageningen University gathers teams from universities and companies around the world to demonstrate their solutions for autonomous operation in the field.

The development of SMARTWHEELS robot started at October 2004 by a group of nine students. After a few start-up meetings, one half of the team started working with the robot platform and the other with the control architecture. All the necessary information between the two teams was shared in monthly meetings and by constant email reporting. The development of each group's design tasks and implementation was monitored in weekly basis. The platform was ready for testing and instrumentation in February 2005 after which the main focus was set to software development and continuous testing.

The idea from the start was to use existing commercial modules to achieve easy assembly and integration and to give more time for developing efficient camera vision, navigation and control algorithms. RC-platform parts such as power transmission, motors, servos and motor drives were used. Due to the problems related to the hardware and partly to the software the development and testing took more time than expected. Some areas of the software were implemented from scratch just before the event and it kept the team occupied.

Hardware construction

Platform

The platform consists of a self made aluminum mainframe and RC-power transmissions (Figure 1) attached to it with 16 suspension springs. The 69 centimetres long mainframe functions as a base for control equipment, PC and other additional apparatus.

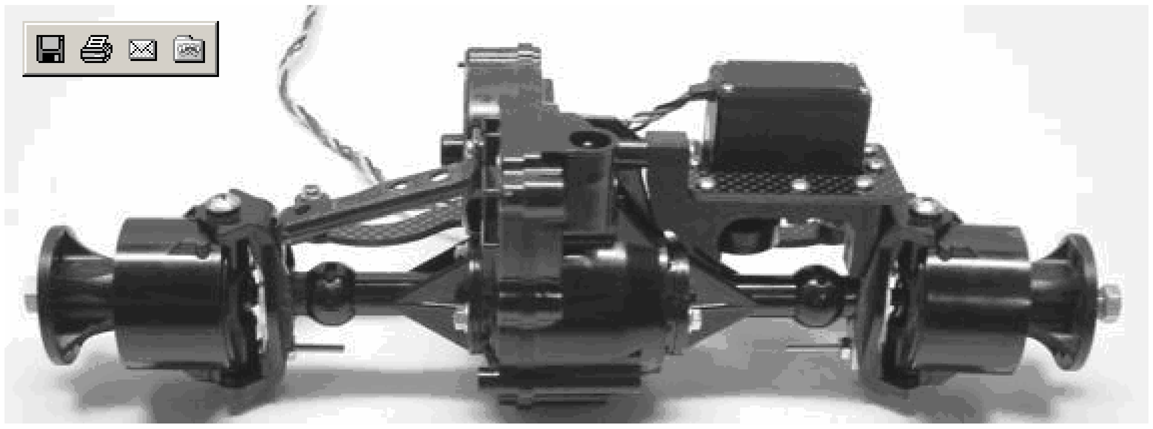


Figure 1 Power transmission

The robot is a four wheel drive with separately steered and controlled axels. The wheel turning radius of the robot is 69 centimeters, which is a bit too wide considering the 75 centimeters row width. The large tires filled with foamed plastic are especially suitable for use in rough terrain and to carry the total robot weight (12 kilograms).

Motors and gearbox

Two Mabuchi RS-540 electric motors give the expulsive force for the robot. Gearboxes with a gear ratio of 30.1:1 are used to scale down the enormous rotation speed of the DC motors. The speed measuring is realized with Hall-sensors attached to the side of the gearbox measuring the magnetic field generated by a magnet in the cog wheel.

Steering servos and motor power controllers

The weight of the robot and the size of the tires set a certain standards for the steering servos. Steering servos HS-805BB+ MEGA, manufactured by Jameco, produce a torque of 19.8 kgcm, which is just enough for this application. The plastic joints between the steering arms and the whole construction are really under a huge stress.

Both traction motors have their own power controllers. It would be simpler to control the motors with only one power controller but the current flow would be detrimental. The Msonic power controllers are manufactured by Mtroniks and they can handle currents up to 60 amperes.

Instrumentation and electronics

Web-cam

The camera used was a Logitech QuickCam Pro 4000 (Figure 2). It’s a higher end webcam and it costs about 80€, which is still quite cheap. It has a CCD image sensor with maximum video resolution of 640 x 480. However, a resolution of 320 x 240 was used for the sake of limited processing power. At the used resolution, it is able to feed up to 30 images per second. With our robot’s computer (a laptop P4) we were able to process about 10 images per second. The camera was connected to the laptop with USB 1.1 interface.

The camera was on top of a long aluminum pole and it was facing down in a sharp angle. This arrangement provides a relatively easy and informative view for processing.

Figure 2 Logitech QuickCam Pro 4000.

Ultrasonic sensors

Our selection for the ultrasonic sensor was the Devantech´s SRF08, because of its price and adequate performance. The sensors were used to measure the distance to the rows of maize plants. The same information was acquired with camera but to make the system more reliable it was necessary to use other sensor data for assurance. The ultrasonic sensors were attached to the front of the robot in an angle of 45 degrees to achieve the best measuring result.

The sensors were connected to I^2C sensor bus, which enabled continuous configuration and adaptation to circumstances. SRF08 ultrasound sensor has operating range up to 11 meters, but in favor of shorter response time the sensors are tuned to work in one meter range.

Infrared sensors

The robot has two infrared sensors for counting the maize plants. These sensors function as switches and do not give any information about the distance. GP2D15 infrared sensors are manufactured by SHARP and were again chosen because of reasonable price and performance.

Electronic compass

The vehicle is also equipped with I^2C compatible electronic compass. Its main purpose is to provide additional angle information while turning at the end of each row. Devantech's CMPS03 compass is widely used in similar applications. By specification the compass has an angle resolution of 0.1degrees but in reality it is closer to several degrees because of external electromagnetic noise.

Hall sensors

Both power transmissions have two hall sensors attached to the gearbox that monitor the rotation speed. Inside the gearbox one plastic gear has a magnet attached to it and Hall switches react to the change of the magnetic field when the magnet passes by. Microcontroller counts the rotation speeds of both axles based on the pulses of the Hall switches and sends the data to the laptop.

Power supply

Eight (8) pieces of 1.2V rechargeable Li-ion batteries soldered together in a series were used as the robots power supply. The capacity of each battery was 9000 mAh, which was proven to be adequate.

Processing equipment

The processing equipment consists of a laptop and a microcontroller circuit.

Laptop

Laptop was chosen because of its familiarity to the group of inexperienced programmers and the processing power required by the camera. The laptop has 512MB RAM and 2.8 GHz P4 processor, and the image processing could gladly use more power.

Microcontroller

Every sensor and device except the camera is connected to a microcontroller. Laptop and microcontroller communicate via a serial port using an application specific communication protocol. Microcontroller mainly operates as an I/O-card and does not contain much higher intelligence. It does some primitive counting and converting of sensor values.

PIC 18F2220 microcontroller was used in the robot. It is not unique in features, but it has what it takes to implement the application. The microcontroller and its surroundings were not “ready to use” embedded system, rather everything was designed for the set up instead (Figure 3 & Figure 4). The board was designed with CadSoft Eagle layout editor and manufactured in Automation Technology Laboratory by group members.

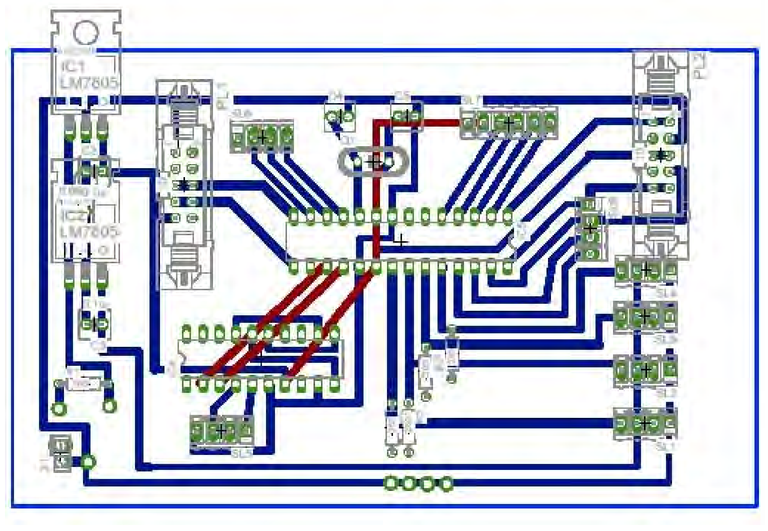


Figure 3 Layout model of the microcontroller board

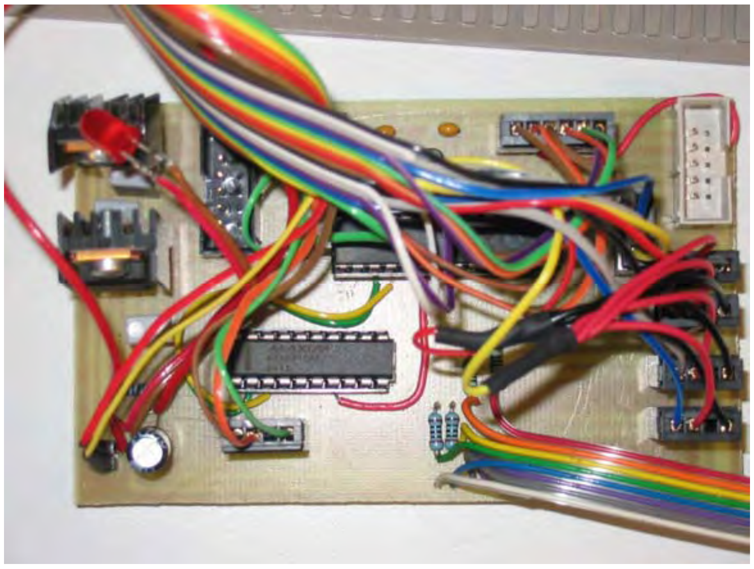


Figure 4 Picture of the actual constructed board

Additional hardware (Freestyle)

The primary idea was to develop a system for measuring the soil density (cone-index) value but in absence of time the calibration and testing remained undone. The principle was to penetrate the surface with a probe and to measure the force.

In the freestyle session a sprayer system was developed instead. It was build up of a car windscreen washer and fixed to spray just on both sides of the robot. It can also be manipulated to spray pesticides between the plant rows to control weed plants. The idea is to spray fertilizers in the ground just around the plant. It could be called surgical fertilizing.

PC Software architecture

The PC application program has all the intelligence of the robot and it is divided into nine functional classes. Graphical interface is separated from other areas and it functions as the main class of the program. It initializes the basic instances, forwards user input and shows the program status. Sensor-class has the communication protocol and interface towards the microcontroller. The software has separate classes for speed and direction control that are used in the logic to control the actuators. Information is shared trough a Database-class that has the most recent data from the devices and classes. More detailed features of the most important classes are described below.

Programs class diagram is presented below (Figure 5). The diagram consists of only the main classes and functions. Collaboration between classes is presented with arrowed lines.

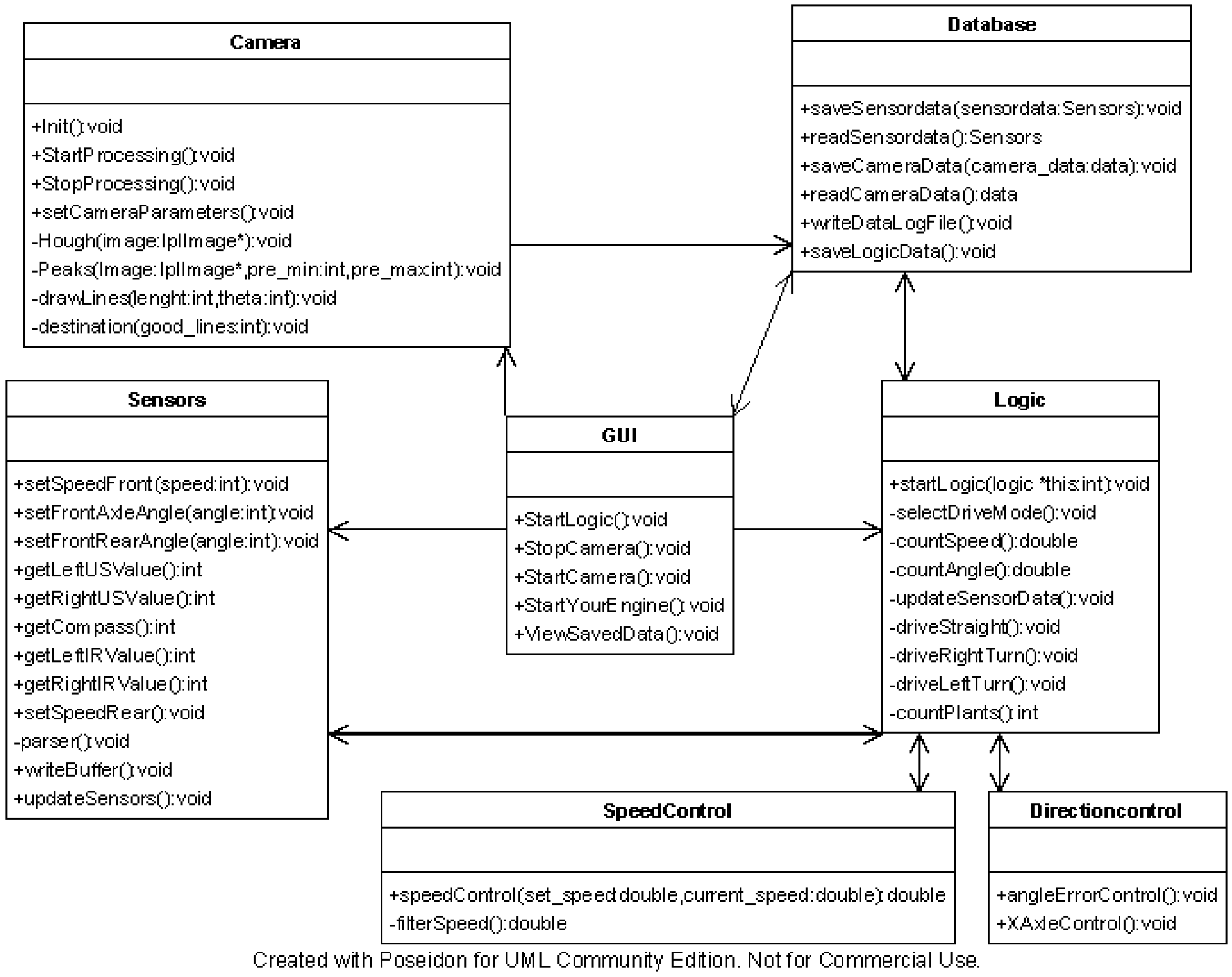


Figure 5 Class diagram

User interface

FieldrobotDlg-class is the user interface of the program. The user can give initial values, configure the system and see real-time information of the robot's status.

Camera

Camera-class has the functions and parameters needed for camera and image handling. It communicates only with the database class. The functionality and the structure of the Camera-class are described later in the document.

Database

Database-class has two tasks. Its primary function is to act as a data-storage for the control variables and sensor data. The secondary task is to provide interface and synchronization to different threads. Database-class uses semaphores to avoid reading and writing of variables simultaneously by multiple threads.

Interface to sensors

Sensor-class provides an interface from application program to microcontroller application. Application specific communication protocol implemented here is used to read sensor data and to write PWM to motors. The microchip sends the new sensor values to laptop every 50ms via serial port and updates the values for speed and direction control.

Program logic

The Logic-class of the robot described below (Figure 6) is the soul of the application. Simple logic decides what to do next (stop, go straight, turn, set speed) and how to do it. Logic contains four different driving modes: straight mode for driving inside the rows, turning modes to the right and to the left and the stop mode. It has its own functions for setting the driving speed adaptively, deciding the next driving mode and implementation for those modes. In straight mode both ultrasound sensors and camera are used to control the turning.

When information from the camera indicates that the robot has arrived to an end of a row it changes to turning mode. Turning mode is based on preordered driving sequences and it consists of one or several steps. Three-dimensional table has the turning angles, length of each step and estimated compass value in the end of a step. Distance and compass are used to acknowledge that the necessary steps were taken. In the last quarter of the turn the robot starts to search the next row with the camera. If a row is detected the drive mode changes back to straight mode.

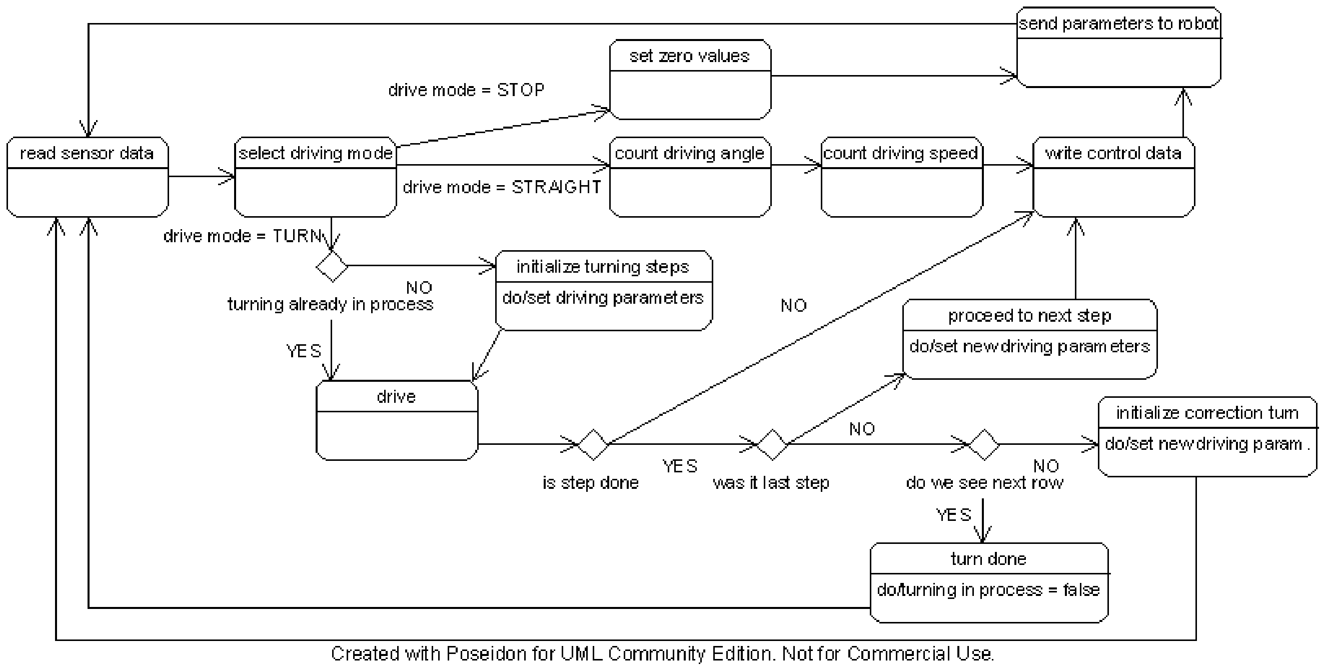


Figure 6 Logic's state diagram

Speed controller

Control-class has a PID controller for controlling the robot's speed. Gaussian filter is used to balance the values acquired from the Hall-sensors before feeding the reference to the controller.

Three different parameters have an influence to the set point of the driving speed: angle error, which is obtained from the camera, angle of the wheels and the estimated reliability of the camera given value. Set point and current speed are used as inputs to the PID controller.

Direction controller

DirectionControl-class has a PD controller for controlling the direction of the robot. Two variables are controlled: the angular error and the distance error from the mid-line of a plant row. The position in the plant row is monitored with the ultrasound sensors. PD controllers are used to predict the turning angle.

There are some other classes to provide functions for synchronization and socket communication not described in this document.

Program sequences and threads

Program contains three threads: one for the user interface (in 500 ms loop), one for the camera (in 130 ms loop) and one for the logic (in 50 ms loop). Threads communicate with each other through a database with synchronized functions. The thread state diagram is described below (Figure 7).

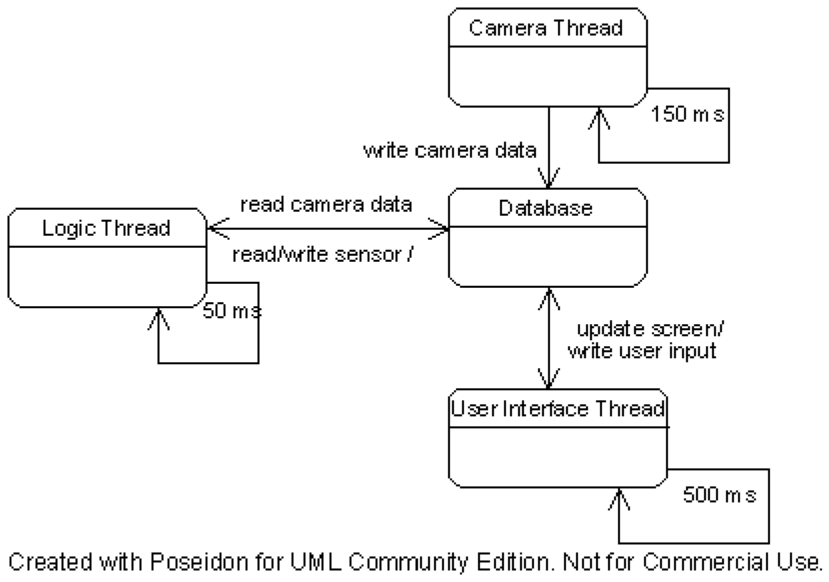


Figure 7 Thread state diagram

Plant counting

Infrared sensors scan the plants. Rising edge of the IR-sensor's signal is used to indicate a plant. Simple filtering is required to find out the actual number of calculated plants.

Motor Control with Microcontroller

Microcontroller's main tasks are to control the motors and servos and to acquire information from the sensors. Servos and motors can be controlled by generating 50Hz PWM (Pulse Wide Modulation) and altering the length of the pulse.

Microcontroller functions as I^2C bus master and it communicates with the US-sensors and the electronic compass via I^2C bus. Hall-switches, IR-sensors and acceleration sensor are connected directly to the chip's I/O ports and monitored through the interface.

Machine vision

The computer vision software was developed on Microsoft Visual C++ .NET 2003 and Open Source Computer Vision Library (OpenCV) by Intel was used for interfacing with the camera and for several parts of the algorithms.

Approaches

Two different approaches were tried for the computer vision. The first one, referred here as *slicing*, was in development long before it could be tested with the actual robot. So for a long time it was tested only with video clips showing maize rows. And so far it seemed to perform well enough. When the robot reached a physical state where it could be used as a test bed, it was realized that the *slicing* approach was inadequate for the demanding real world situations. So another approach using *Hough transform* was developed. It was used in the final version of the robot. Both approaches use similar color segmentation methods to extract the essential information from the raw images.

Preprocessing

Preprocessing is done by color segmentation. First the RGB (Red, Green, Blue) color spaced images are transformed to HSV (Hue, Saturation, Value) space. HSV space is visualized in Figure 8. It is a more natural space for color segmentation as it represents the way humans see more closely than RGB space.

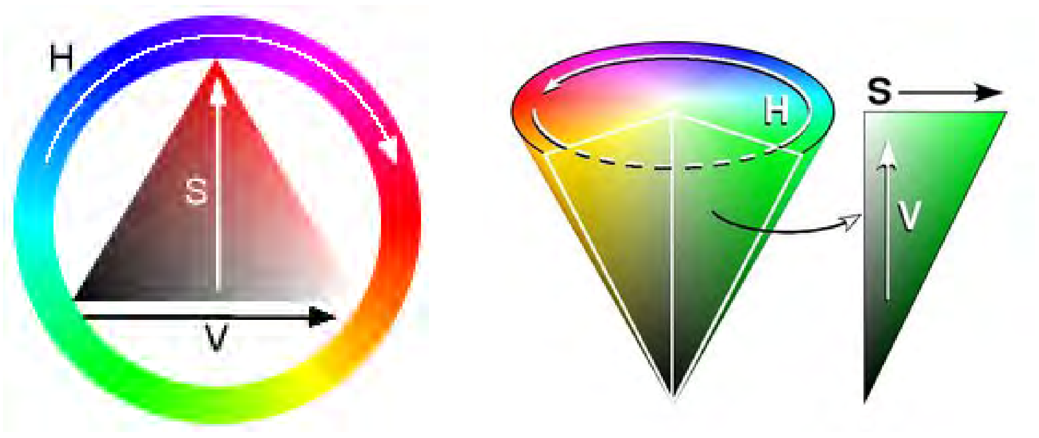


Figure 8 HSV color space

Hue is the type of the color (like green or red). Saturation is the vibrancy or purity of the color. Colors seem grayer when saturation is lowered. Value is the brightness of the color. It would be mathematically more correct to display the HSV space as a cylinder rather than a cone. But in practice the visually distinct saturation and hue levels decrease as value approaches zero (Wikipedia, 2005).

The idea is to restrict each of the three color dimensions so that only most natural plant green remains. When the original image is color segmented this way, the result is a binary image containing only the plants. This is visualized in Figure 9.

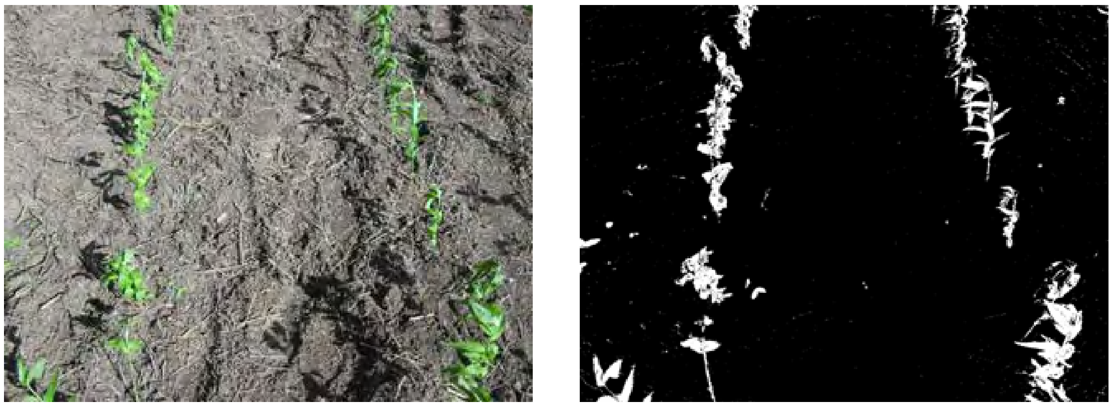


Figure 9 Color segmentation

Method 0: Moments

Actually there was a third method that was based on moments. It was tried out in the very beginning of the project. There's really no ground for any intelligent results. It was just a test to see if moments calculated from the image could be utilized for something useful.

First, moments were used to determine the center of gravity for the entire data. When that point is drawn on a scene where both maize rows are equally strong, it gives a quite good suggestion for a target point. But generally it's useless because it requires symmetry.

The moments were further used to calculate an orientation angle with another unjustified method (Kilian, 2001). Moments can be used to form a kind of inertial tensor analogy. Then the main inertial axes can be calculated. They correspond to axes of an ellipsoid which approximates the actual object (arbitrary maize row view in this case) (Figure 10).

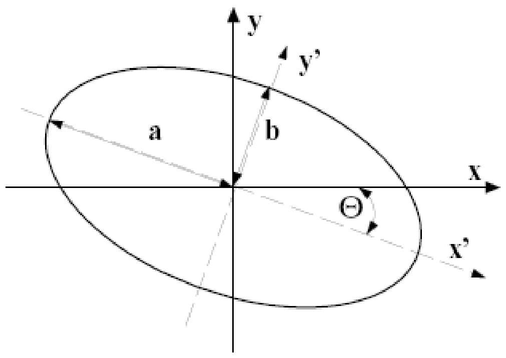


Figure 10 Orientation angle from moments.

This orientation angle was drawn on top of a video clip and it seemed to point in the right directions. It is not tested with the actual robot. There is no reason why this method should work. It was just a little experiment. Still it performed amusingly well for being unjustified.

Method 1: Slicing

It should be noted that additional preprocessing was used with this approach. Morphological erosion and dilation with masks of various sizes and shapes were used to shape the quite arbitrary data into nice continuous streaks (Figure 11) (Bock, 1998).

The goal with this first approach was to get a simple and fast algorithm capable of providing the needed navigation information. The method is called *slicing* because it slices the image in a couple of ways. First of all, it cuts the image in two directly from the vertical midline. Then it slices both halves in twelve parts and moves a 5 x 5 mask from midline to left and right borders along those slice edges (Figure 12).

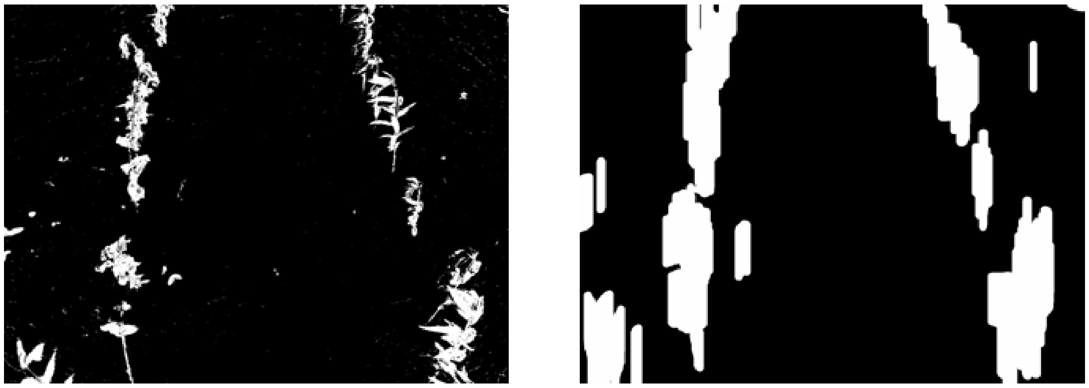


Figure 11 Morphological filtering

If there is enough white pixels inside the mask (we have a binary image), the center of the mask position is marked as a maize row border. After a full sweep, a least squares line is fitted to the marked points. If there are less than four marked point in either of the sides, then no line is fitted on that side. The least squares fitting is done with basic matrix operations (1). Weighting matrix **W** is used to filter out slices that have no marked points. The normalized variances of the fitted lines and number of detected rows (0-2) are used to describe how reliable the data might be. Main logic could use this reliability figure to decide how to weight different sensor inputs.

$$\theta = \left(X^T W X\right)^{-1} X^T Y \qquad (1)$$

Slicing method determines a target point between the detected rows and calculates an angle that points there. Main logic uses this angle for steering decisions. If only one row is visible, then the target point is guessed some distance away from the sole detected row.

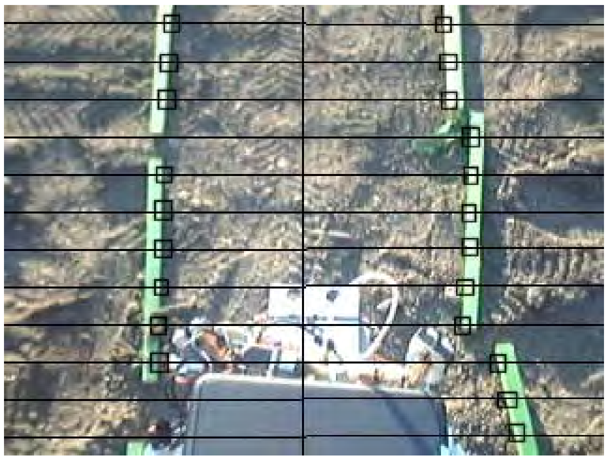


Figure 12 Slicing method principle. (mock-up)

The main problem with this approach is reliability in situations where the robot is heading strongly away from the midline of the track. Then a single maize row can cross the entire image and wrong interpretations are made from it (Figure 13). The algorithm is reliable only when the maize rows don't cross the vertical midline of the image. It was thought that the robot could rely on other sensors in those situations. In practice, the robot often sees rows from undesirable angles and reliably detecting those situations proved too difficult. Thus another approach was developed.

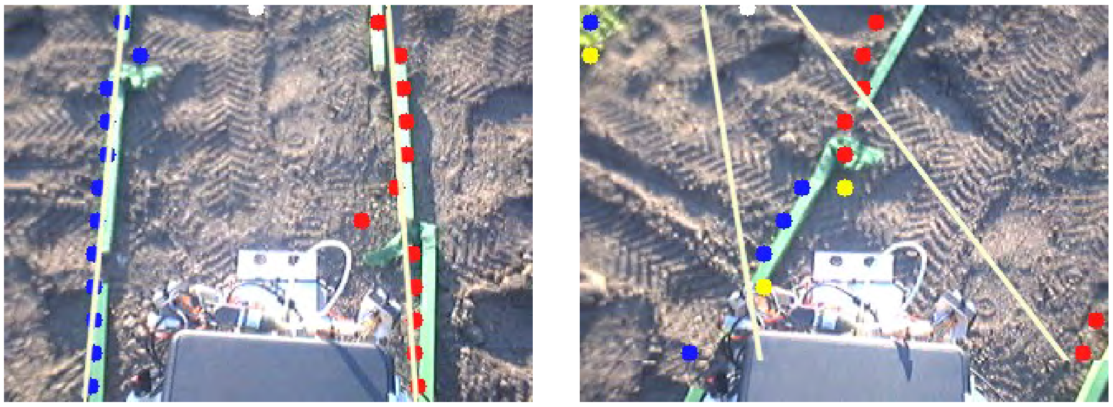


Figure 13 Good and bad cases with the slicing method. (actual algorithm)

Method 2: Hough Transform

After the failure with the first approach, a reversion to a well known and widely used method, that is the Hough transform, was made (Shapiro, 2001). Hough transform is quite CPU-intensive and gets heavier when the amount of processed pixels (amount of green in the image) increases, as the accumulator values (θ, d) (2) will be calculated for every pixel. To lower the CPU-load, the transform is first done with a very low angular accuracy. Next a more accurate transform is done for a space restricted to the area around the peaks of the first pass (Figure 14). This algorithm was made without the help of the openCV library, because the function it provided didn't offer enough customizability.

$$d = x\cos(\theta) + y\sin(\theta) \qquad (2)$$

To be considered as a peak, an accumulator array pixel must exceed a fixed threshold value and another threshold value which is 60% of the brightest pixel. One or two brightest peaks are noted. The second peak must be sufficiently far away from the brightest peak. The final peak location is determined as a center of mass for a 10 x 10 mask around the actual peak pixel.

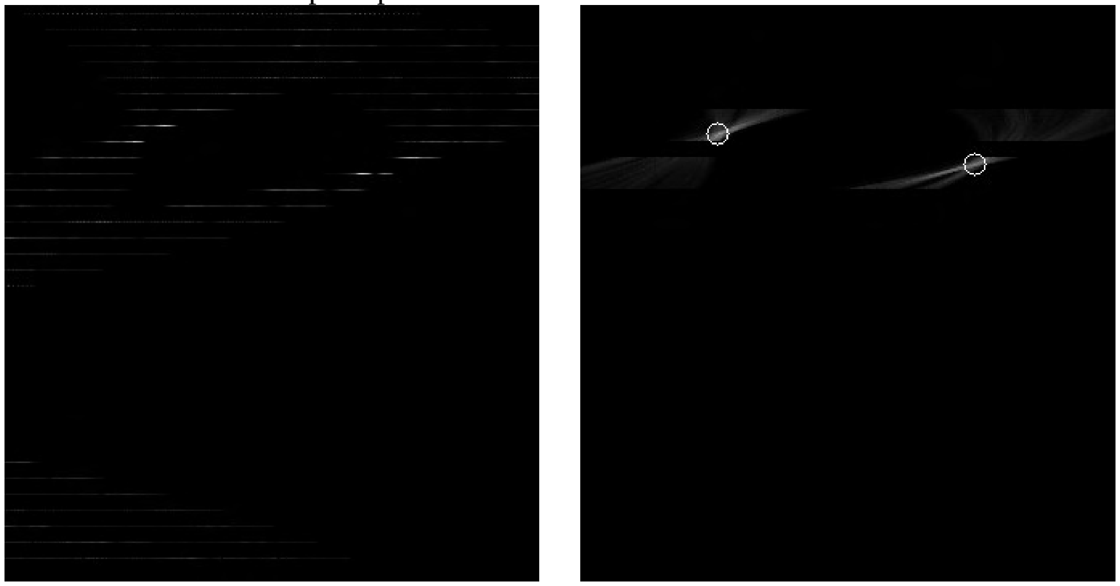


Figure 14 Two-pass Hough transform illustrated. Found peaks are marked with circles.

The target angle is determined the same way as with method 1. In the case that both accumulator peaks are found from the same maize row, the corresponding lines are checked for crossing. If they cross, only the one with the brightest peak will be taken account. A reliability figure is calculated for the main logic based on the target angle and number of detected rows (0-2).

It proved difficult to reliably determine the distance to the maize rows in both sides of the robot from the camera image. As time was running out, it was decided to use camera only for the target angle determination. There were also some difficulties with detecting the peaks from the accumulator so that they would represent different maize rows. This was partly because the openCV library lacked proper blob analysis tools. Sufficient results were achieved by hand picking some parameters. Results on a real world scene can be seen in Figure 15.

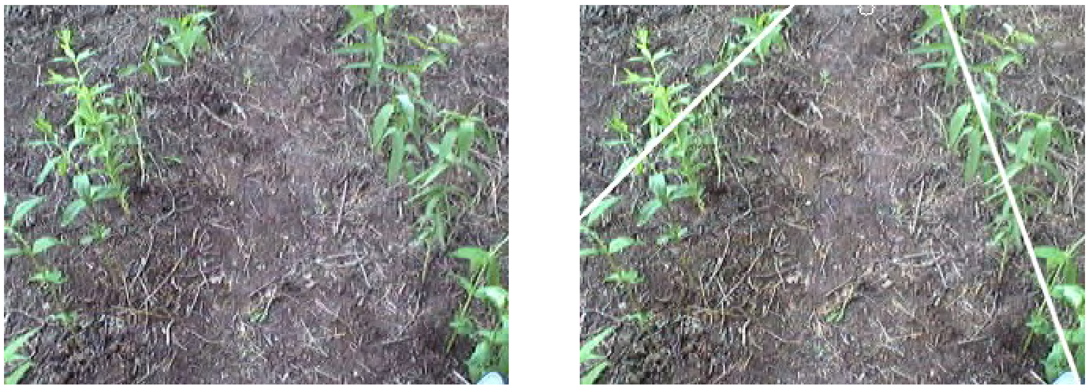


Figure 15 Results with the Hough method

End of Row Detection

For end of row detection, the upper one third of the image is monitored. If the amount of white pixels in that area in the binary image drops below a certain threshold for a long enough time, it is assumed that the row is ending. Main logic also uses other cues for end of row detection, like the traversed distance.

Conclusions

Year is a long time for developing an autonomous robot, which has just a simple logic. It is obvious that in a team of almost ten student results are hard to get without a clear organization and wide planning and specifications. It was noticed that in a project where the development group is geometrically scattered around, the information sharing and trust are the key issues. Group's members may have different goals and motivation for the project. A clear organization is not enough just by itself. It can be stated that the SMARTWHEELS would never have been ready for the event without active and skilful individuals that the team has.

It was proven that a good web camera based machine vision system can clearly be realized with minimal cost and reasonable time investment. This of course applies only if you know what you are doing. As the work was started from zero real world experience, some misleading trails were followed.

It seems that a camera really fits this kind of application, as it can be made far more error tolerant than for example ultrasonic or infrared sensors. A single gap or a misplaced plant won't matter because the camera sees the situation from a larger perspective. Also the ability to see ahead and use features like color and shape are unique compared to most other sensors. However, it's important to keep the algorithms fast. Slow machine vision will bog down the entire robot. Also different camera positions could be experimented with.

The conditions between laboratory tests and real world situations really make a difference. It's difficult to take account all the challenges that real environment pose if you don't have access to that kind of environment for testing. Adaptivity is clearly the biggest challenge here. Numerous variables affect the situation constantly and adapting to those requires a lot of extra programming and testing.

Calibration with the real world was made with crude approximations and gut feelings. The reason for this was partially the fact the robot was in a constant state of change and partially because the results seemed to be accurate enough for the most time without any sophisticated calibrations. Afterwards a more theoretical approach would have been appreciated.

All in all, the results were very promising and the team participating to next year's Field Robot Event will learn from our mistakes and have an opportunity to develop the SMARTWHEELS fieldrobot.

References

Automatic vacuum cleaner manufacturer's homepages
http://www.robosoft.fr/AutoVac-Robot.html
http://www.electrolux.com/node613.asp

Automatic lawnmower manufacturer's homepages
http://www.electroluxusa.com/node141.asp

Field Robot Event 2003, 2004, 2005
http://www.fieldrobot.nl

Bock, R., Homepage, Morphological Operations, 7 April 1998,
http://rkb.home.cern.ch/rkb/AN16pp/node178.html

Kilian, J., Simple Image Analysis by Moments, v. 0.2, 15 March 2001,
http://groups.yahoo.com/group/OpenCV/

OpenCV, Open Source Computer Vision Library, Intel Corporation,
http://www.intel.com/research/mrl/research/opencv/

Shapiro, L., Stockman, G., Computer Vision, Prentice-Hall, 2001, ISBN: 0-13 030796-3

Wikipedia, HSV color space, 25 May 2005,
http://en.wikipedia.org/wiki/HSV_color_space

Sponsors:
- Hewlett Packard HP
- VALTRA
- Kemira Growhow
- MTT
- Koneviesti

hp ®

invent

koneviesti

UNIVERSITY OF HELSINKI

MTT

Agrifood Research Finland

HELSINKI UNIVERSITY OF TECHNOLOGY
Automation and Systems

KEMIRA
GrowHow®
partnership • knowledge • solutions

VALTRA

Whirligig Beetle – a small scale hovercraft robot

Satoshi Yamamoto, Yasuyuki Hamada
OML skunk works (Private participation), Saitama-shi, Japan
E-mail: syamamot@affrc.go.jp

Abstract

As a small scale vehicle which can be applied not only to a soil field but also to a paddy field, we developed a radio controlled small scale hovercraft. This machine has an engine and a propeller which gives propulsion and floating power simultaneously. Furthermore, the navigation system which consists of a sensor unit which detects plants and obstacles, etc. while measuring the angular velocity of the vehicle, and a control unit which outputs the control signals for following rows of plants automatically based on the information from a sensor unit was developed. We could expect that the hovercraft robot will be able to run everywhere in the maize fields in the near future by adding further improvements.

Keywords

Hovercraft robot, crop scouting

Introduction

Since 2003, the "Field Robot Event" has been held every year at Wageningen University in the Netherlands. On the event, many students have been developed small scale robots vigorously, which compete for performances, such as a travel speed and operating accuracy. Inferior travel conditions, such as a muddy surface ground and a puddle, are mentioned to one of the big problems for these small scale robots. Then, we decided to adopt a hovercraft as a travel means of the small robot which overcomes such inferior conditions.

The objective is to develop the technologies as follows;

1) Small scale hovercraft which travels even on a lawn field.
2) Navigation system for following rows of plants.
3) Counting methods for the number of plants.

Especially, concerning with 2) and 3), it is thought that the following technologies are necessary;

1) Comparatively low cost sensor unit which can detect plants or obstacles while measuring an angular velocity of the vehicle.
2) Control unit which outputs appropriate signals to control the travel speed and the direction of the vehicle according to the information from the sensor unit.

3) The algorithm for following rows of plants, and turning in headlands, while counting the number of plants.

Materials and methods

1) Hardware

(1) General construction

This vehicle obtains propulsion and floating power with the engine for model airplanes, and steering by the rudder. The principles are shown in Fig. 1 and Fig. 2. Half of the airflow is used for propulsion and steering, the rest of the air flow is used for floating.

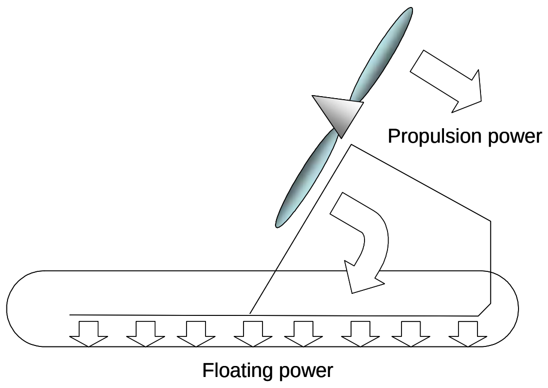


Fig. 1 Propulsion and Floating method

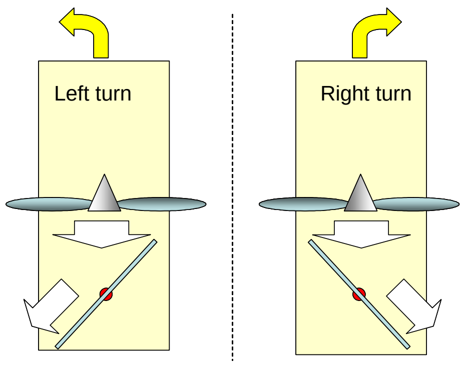


Fig. 2 Steering method

The skirt part was able to blow up uniformly by adopting the structure with which the skirt is once blown up and air flows into a high pressure air room (Fig. 3). This skill was shown in the internet HP of Tokyo Institute of Technology. We confirmed that this machine can travel even on an lawn field.

The photo of this machine is shown in Fig. 4. The size is L72cmW38cmH30cm (an antenna height for radio control of 46cm), and the weight is 2.5kg without including fuel. When the whiskers attached in right and left ahead of the vehicle is included, the width is 80cm. The chassis is made from 1mm or 1.5mm in thickness

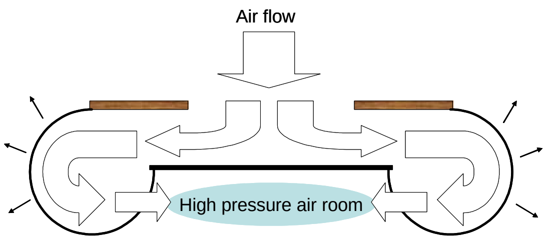


Fig. 3 Air flow for blowing up the skirt

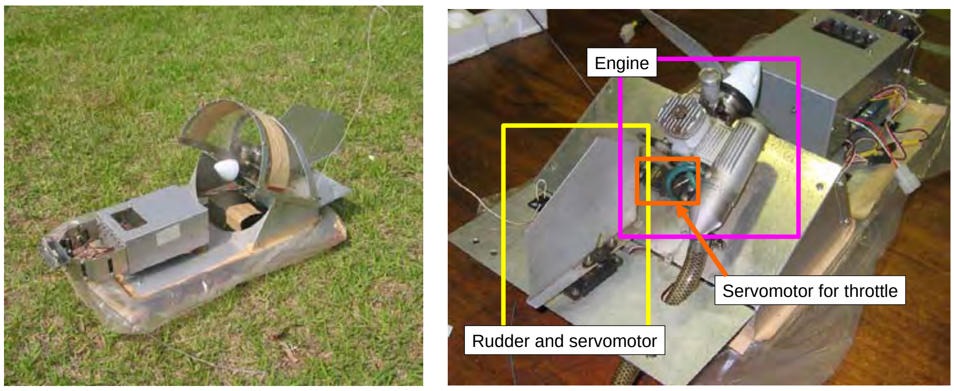


Fig. 4 Hovercraft robot

aluminum plate. For a propeller cover, the plastic plate with a thickness of 0.2mm is used. The skirt part processed and created thicker vinyl.
Servomotors for radio control driven by DC6V (four LR6 size 1.5V AA batteries) are used for operation of the engine and the rudder. Since the navigating system which consists of a sensor unit and a control unit requires DC 9V for operation, two more 1.5V batteries are connected in series to the batteries for servomotors. Moreover, if the connectors of servomotors are connected to a radio control receiver, the remote control function by a radio transmitter will be attained.

(2) Engine and accessories
The engine of a model airplane, MAX-25FX (O.S.ENGINES MFG. CO., LTD) is applied. Rated outputs are 0.62kW / 18,000rpm, and weight is 248g. As a propeller which suits this, 9x6 (with 9 inches of diameters of a propeller, 6 inches progresses by theory in 1 rotation) and a 45mm spinner are attached. The fuel tank is 220 cc of round shapes. The filter of air intake is also added for protection against dust.

(3) Microcontroller
A figure and a photo of connection between the microcontroller and peripheral equipments, such as sensors and servomotors, are shown in Fig. 5. As a microcontroller, H8/3048F (HITACHI) is applied, which has five PWM outputs and eight AD inputs (10 bits resolution). In case of making some kinds of work with combining several ON/OFF switches, some digital input ports can be available. An LCD display is used to display the number of plants.

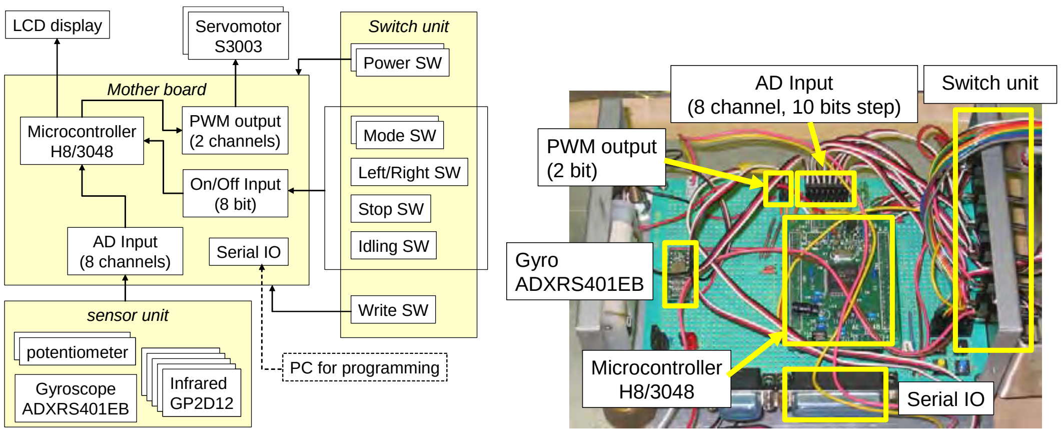


Fig. 5 Electronic hardware

(4) Servomotors
To control a travel speed and a steering, two servomotors (FUTABA Corporation S3003) are applied and attached to the throttle and the rudder. Weight is 37.2g and 0.19 seconds / 60 degrees of the rotation speed at the time of 6V, and the torque are 4.1 kg*cm at the time of 6V.

(5) Sensors
Five Infrared range sensors (sharp GP2D12, 10-80cm available) and two potentiometers are applied and arranged as shown in Fig. 6. Moreover, the angular

velocity meter (Analog Devices ADXRS401EB) was attached in the mother board of a microcontroller. Since the angular velocity meter has a measurement range to 75 degrees/second at the time of factory shipments, resistance of 100kohm was added and the measurement range was expanded suitably.

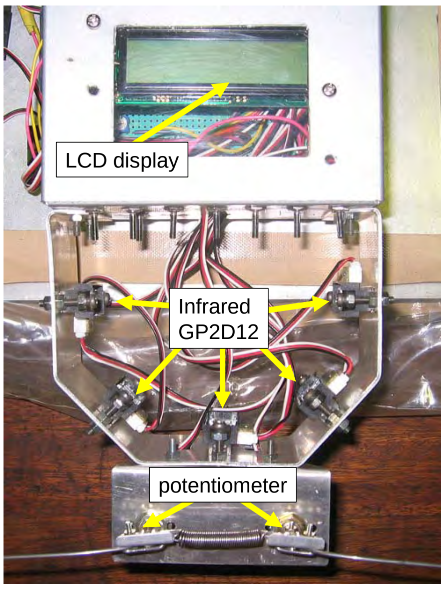


Fig. 6 Arrangement of sensors

2) Software
The operation flow figure is shown in Fig. 7. It is the work which Task1 goes through every other path between rows of plants counting the number of plants, and Task2 is work which goes through every path between rows of plants.
To evaluate the software, artificial plants was created using corrugated fiberboard with a height of 20cm.

The details of each part are shown below.

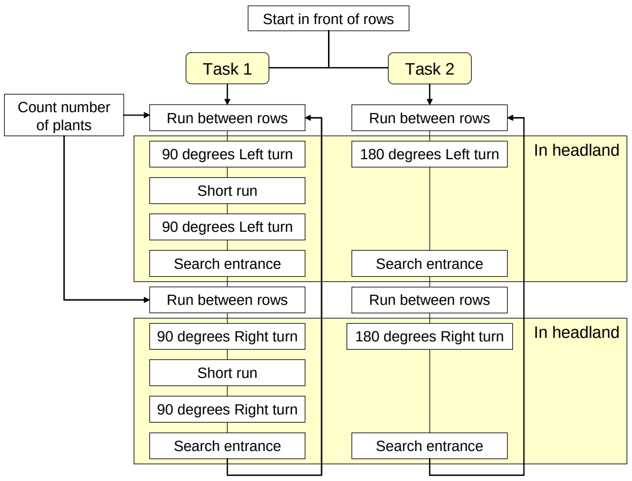


Fig. 7 Operation flow

(1) Run between rows
At the time of the run between rows of plants, as external sensors, three infrared sensors (the front and 45 degrees of right-and-left slant) and two potentiometers are used.
When the distance information from three infrared sensors are supplied, comparing the distance data of left 45 degrees and the distance data of right 45 degrees, then add the shorter distance data of them to the distance data of front infrared sensors. The result of calculating is used in the control as a control value (Fig. 8). In case of potentiometers, similarly, comparing the data of angular displacement of them, larger data is used in the control as a control value.

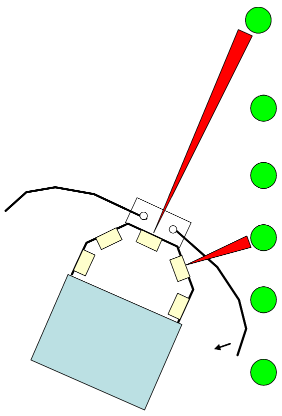


Fig. 8 Detection of rows

At the beginning, to control the rudder of the hovercraft, we adopted the ON/OFF control as an easy control method describing constant thresholds such as distance data of infrared sensors and angular displacement data of potentiometers. The amount of displacement of the angle of the rudder from the middle position were also described only several patterns. However, since it was very difficult to find thresholds and several rudder

angles for running on the middle part between rows of artificial plants, the hovercrafts could not reach the end of the rows of artificial plants.
Then, proportional control to decide the displacement of the angle of the rudder from the middle position was adopted. The formula is as follows;

$$S = Px * x + Py * y$$

S : displacement of the angle of the rudder from the middle position
Px : Coefficient of the control value of infrared sensors (constant)
x : Control value of infrared sensors
Py : Coefficient of the control value of potentiometers (constant)
y : Control value of potentiometers

Through the proportional control, when the vehicle was in the middle between the rows of artificial plants, the displacement of the angle of the rudder is small, and if it was near the row, the displacement of the angle of the rudder became large. However, when the vehicle was near the row and steered large, it was impossible to avoid to collide the opposite side row.
In order to keeps the angular velocity of the hovercraft low while the large displacement of the angle of the rudder, we decided to apply angular velocity dumping to the above-mentioned proportionality control. The formula is as follows;

$$S = Px * x + Py * y - Pz * z$$

S : displacement of the angle of the rudder from the middle position
Px : Coefficient of the control value of infrared sensors (constant)
x : Control value of infrared sensors
Py : Coefficient of the control value of potentiometers (constant)
y : Control value of potentiometers
Pz : Coefficient of the angular velocity data (constant)
z : Angular velocity data

By this method, the time to change the direction of the vehicle became very short, and the time when the vehicle was in the middle between rows was expanded remarkably.
With larger displacement of the angle of the rudder, the friction between the hovercraft and a ground increases sharply. So we adopted the throttle control which increases the rotation of the engine in case the displacement of the angle of the rudder exceeds a threshold value.
Since the travel speed is greatly depends on the ground condition, even under the constant rotation of propeller, the travel speed extremely changes. If the travel speed is too large, the steering control does not fulfill demand and it overruns headlands. Then, the function of counting number of plants is used to obtain the information of travel speed. Measuring the interval time to count the plants on right side of the machine, considering the distance between plants is almost constant, the traveling speed can be estimated. According to the interval time, if it is shorter than the threshold value, the engine speed changes to slow down.

(2) Count number of plants
By infrared range sensors attached just beside, plants of both sides are detected and the number is counted. When the distance data is shorter than the lower threshold, system waits the moment when the distance data becomes longer than the higher threshold. At the moment, the plant is recognized and the number increases.
When a hand was made to flutter in the face of infrared range sensors, they could count about 90 percent, but when thin things, such as a pencil, were passed quickly, there was a case where it could not recognize.

(3) Headland turning
a) Turning method
As conditions to recognize that the vehicle is in the headland, although the data of angular displacement of potentiometers are included at the beginning, there was a tendency for the turning timing in headland to be delayed. Then, in order to start headland turning early if possible, when infrared range sensors of the front and 45 degrees of right-and-left slant did not detect a plant, the hovercraft is decided to come out from between rows and to recognize it as having gone into headland.
As a pattern of headland turning, there are two patterns with which the hovercraft enters every path between rows of plants immediately, or every other path between rows of plants.
With the first pattern, when it has been recognized as having advanced into headland, it integrates with the angular velocity to approximately 180 degrees. In addition, a counter steering is applied and it enabled it to shift to going straight promptly, without moving in a zigzag direction until angular velocity fell within the predetermined range, since the power of considerable rotation was gained at the time of a turning end.
With the second pattern, when it has been recognized as having advanced into headland, it integrates with the angular velocity to approximately 90 degrees. Then, if the potentiometer of the side near plants or the infrared range sensor attached just beside detects plants or fixed time passes, 90 degrees turning will be performed again. At the time of 90 degrees turning, a counter steering is operated to avoid moving in a zigzag direction.

b) Searching entrance between rows
When the headland turning is finished, the hovercraft will face the objective path between rows. However, it is considered that the length of the path to the entrance is too long. In such a situation, the hovercraft will collide to plants without a steering control. Then, it was made to move forward until infrared range sensors of 45 degrees of right and left detect the plants within about 60cm, avoiding obstacles.

Results and discussion

We took part in FRE2005 in 16th -17th June with the hovercraft robot named “whirligig beetle”. It was the first time to apply it to an actual maize field. Through the competition, It turned out that the machine was effected the shape of the

ground remarkably. As shown in Fig. 9, although a hollow surface was very good, a slope, uneven or rise surface were inferior conditions.
And the engine was also affected by self-generated dusts. With the unstable power output of the engine, it was difficult to adjust the appropriate travel speed.

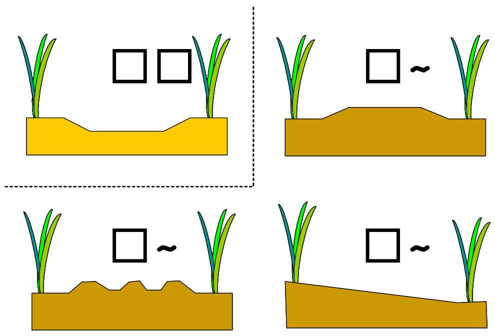


Fig. 9 Condition of ground

The function of counting number of plants did not worked in an actual maize field. We used the infrared range sensors, sharp GP2D12, however, it turned out that GP2Y0D340K, which outputs digital signal, was more suitable for counting plants. So it also disabled that controlling the travel speed with the interval time of counting plants.
We estimated that the software for following straight rows could be applicable to the similar task; following curved rows. But the robot could not follow the curved rows.

Conclusions

It turned out that the hovercraft robot needed to be improved with some points as follows;

1) Excessive maximum travel speed (excessive propulsion power)
2) Lack of floating and steering power
3) More intelligent sensors
4) Algorithm for following curved rows

To improve these points, the distribution method of the air flow from a propeller to propulsion and floating power will be changed. Moreover, since it was apprehensive about the case which damages plants with a propeller, it was thought that a propeller cover also needed to be improved.
Although whiskers and infrared sensors could work with the artificial plants, it would be better to introduce more intelligent sensors like a CCD camera to obtain more stable performance.
Through the improvements above, we estimate that practical use scenes of this machine are as follows; while running everywhere at the field which has the rows of plants which can be followed by this machine, controlling wildlife damage, monitoring of the situation of fields with newly attached camera can be carried out. The sampling for prediction of breeding of a noxious insect can also be considered.

References

[1]Proceeding of the 2nd Field Robot Event 2004, pp47-60, Cropscout
[2] SHARP, GP2D12 data sheet
[3] ANALOG DEVICES, ADXRS401 and ADXRS401EB data sheet
[4] O.S.ENGINES MFG. CO., LTD, http://www.os-engines.co.jp/
[5] HITACHI H8/3048 series hardware manual
[6] http://www.mech.titech.ac.jp/news/mechworld2k2/hoverstars/

[7] Akizuki Denshi Tsusho Co., Ltd., H8/3048 Developing manual