Proceedings of the 4th Field Robot Event 2006

Stuttgart/Hohenheim, Germany
23-24th June 2006

PROCEEDINGS OF THE 4TH FIELD ROBOT EVENT 2006

STUTTGART/HOHENHEIM, GERMANY
23-24TH JUNE 2006

ISBN :	978-90-8585-480-7
Date :	March 2009
Publisher :	Farm Technology Group
Editors:	Timo Oksanen & Eldert van Henten
Address :	Technotron (Building 118)
	Bornsesteeg 59
	6708 PD Wageningen
	The Netherlands
	Tel: +31 317 482980
	Fax: +31 317 484819
	Contact : info@fieldrobot.nl

Editorial

The Field Robot Event has been held since year 2003. It all started at Wageningen University, the Netherlands, where this competition was held three times in a row from 2003 until 2005. As it has always been the intention to create an international event travelling through Europe, for 2006 it was decided that hold the competition at Hohenheim University, Stuttgart, Germany.

The FIFA World Cup 2006, simultaneously hosted by Germany, inspired the organizers of the Field Robot Event 2006 in defining the tasks of the competition. In the first task the robots had to draw a straight white line on the grass while navigate to a corner flag. Obviously, this was a demonstration of how these smart field robots might be used in other than purely agricultural applications. In the second task, robots had to navigate through a row crop and simultaneously count yellow golf balls representing Dandelions. It was the intention to make the competition field more challenging by sowing corn in grass. This would challenge the vision systems used by many robots. However, due to a too cold spring in Stuttgart, the maize did not sprout and the organizers deployed a rescue plan, having the competition take place between the rows of a barley crop. The third challenging task, also inspired by the FIFA World Cup, involved finding and indicating holes in a grass lawn. The robots had to scan the whole field, find the holes in the grass and indicate them. This task was followed by a speed race in a barley crop. The performance of the robots in view of fast and accurate navigation was tested in that task. Last but not least, the competition ended with the usual free style session in which teams were given total freedom to demonstrate the (special) abilities of their robots.

Publication of this Proceedings of 4th Field Robot Event 2006 was delayed due to numerous reasons. However, it was finally possible to publish these valuable reports of the robots that participated during this event. This proceedings contains 11 articles, each of them describing one robot that took part into competition.

Timo Oksanen and Eldert van Henten
Editors
Wageningen, March 2009

Index of Robots

Sponsors

The Field Robot Event 2006 was sponsored by:

UNIVERSITÄT HOHENHEIM 1818

AMAZONE

FENDT

CLAAS STIFTUNG

DAIMLERCHRYSLER

NATIONAL INSTRUMENTS

JOHN DEERE

MÜLLER Elektronik ... wir regeln das!

Junior Business Team e.V.

Disciplines and Rules for Field Robot Event 2006

For the first time in 2006, there was a twofold competition - a basic competition for teams on budget or "newbies" and an advanced level.

Basic competition

- The line - can you draw a straight white line towards a given location?
- Dandelion detection - how many dandelions can you count navigating between rows?
- Speed race "all in a row" - can you outspeed your competitors in an open race?

Advanced Competition

- Hole detection in grass - can you detect damage, litter, etc.?
  The perfect training to perform in the "Gottlieb-Daimler-Stadion".

Freestyle (all)

- What is your specialty? Present your own ideas.

Rules

1. An implement will be provided or bring your own.
2. A "corner flag" will be the target - recognize/detect it, use the bearing, or find your own solution.
3. Dandelions will be simulated using yellow golf-balls.
4. Standard, curved maize rows, 75 cm wide, on "advanced terrain": Expect some parts to be "hilly" and some to be "rainy"...
5. Standard, straight maize rows, 75 cm wide, all robots start at the same time. Collision-free driving required ...

More Rules ...

- Anybody can participate: No limitations regarding age, education, and/or profession.
- All kind of autonomous vehicles are welcome. Wanna fly? Fly!
- Professionals and/or teams on a high budget will start in an own league.
- A paper has to be submitted before the contest, documanting the construction of the robot.
- A scientific committee will judge the vehicles

Cornickel 2

S. Ebert, M. Großer, T. Haase, K. Mögle, D. Müller, M. Schulz and
academic advisor M. Grimsel

Technical University of Dresden
Faculty of Mechanical Engineering
Chair of Agricultural Machinery
email: grimsel@landmaschinen.tu-dresden.de

Abstract

Based on the robot Cornickel (Field Robot Event 2005) six students from the Technical University of Dresden developed a new concept for a robot, which takes part in the Field Robot Event 2006 in Hohenheim.

In this international contest an autonomous vehicle should navigate through wheat rows as fast as possible or count yellow golf balls during navigate through curved rows. Because of the FIFA FOOTBALL WM 2006 in Germany there were 2 football tasks. The robot draws a straight white line to a corner flag and also detects damage in grass.

Keywords

autonomous robot, microcontroller, Atmega128, agriculture robot, ultrasonic sensor, infrared sensor, radio, CMOScam2, navigation, electronic compass

1. Introduction

After analyzing the tasks for this year, we decided to develop a new robot, because 3 competitions were completely different to the last year.

One important part is the bonding of a camera module, which is necessary to count the golf balls and to find the flag. During the first weeks we did a lot of brainstorming on the different competitions and their technical solutions. Also we deliberate on the general problems last year and what can we do to avoid these problems. So we used cruise controller from model making and electronic from industrial applications.

2. Materials

2.1 Mechanics

A central joint in the middle of the vehicle replaces springs and dampers. Together with two independent steering shafts it can drive in every difficult terrain. A steel frame carries the driving axles, the two DC motors and the two servo motors for steering.

In the front and the end of the robot a L-beam carries all distance sensors. The rest of the body consists of wood. It carries the accumulators and the controllers. Wood has a lot of advantages for this task. It is very cheap, stable and easy to work. This is a competition for field robots, so Cornickel got a cover to look like a tractor.

2.2 Sensors

2.2.1 Ultrasonic sensors

The DEVANTECH SRF04 (fig. 1) measures the distance in the rows. The advantage of these sensors is the great dispersion (fig. 2), which makes it independent from small gaps in the row.

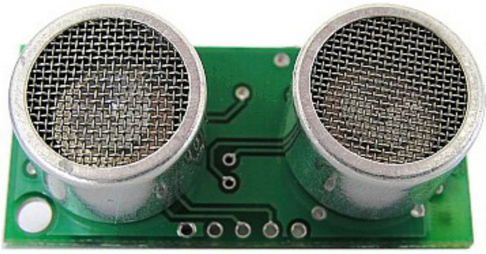


Figure 1. DEVANTECH SRF04

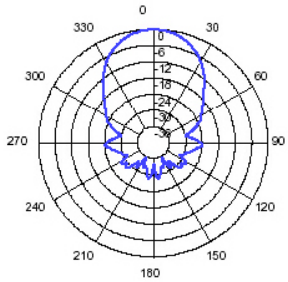


Figure 2. Detection angle

The output signal is a pwm signal with a linear relationship between the time the ultrasonic needs and the distance to the rows. After a start impulse with a length of 10µs the sensor starts the output impulse. The length of the impulse is measured with a 8bit timer and a resolution of 100µs.

2.2.2 Infrared sensors

2 different types of infrared sensors are needed.

2 SHARP GP2D12 (fig. 3) left and right behind the robot detect the end of the rows. These sensors give an analog voltage value. The microcontroller converts the value in a digital value and calculates the distance with a special function (fig. 4).

Figure 3. SHARP GP2D12

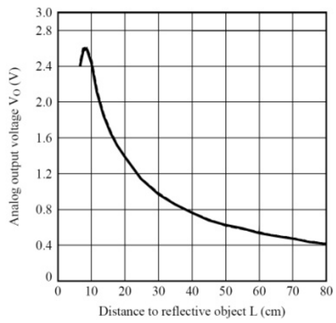


Figure 4. function to calculate the distance

28 infrared diodes OSRAM LD274 and sensors SHARP IS471F work like a switch, which switch when the distance to an object under-run 10cm. This is used to found holes in the grass.

2.2.3 Camera

Why is a camera necessary?

One important fact is the size of the golf balls. They are very small and the subsoil is very bumpy. The yellow golf balls are distinguishable from the surrounding area and easy to count with a special camera.

The other fact is the distance of the corner flag. In the description to this competition was written that the distance will not exceed 10 meters. The most other distance sensors are not able to detect objects in this distance and the camera can also detect the exact direction, where the corner flag is.

The camera needs onboard image processing, because the interface to the microcontroller was not fast enough to transmit a complete picture shortly. TheOV6620 OMNIVISION CMOS camera with a SX52 microcontroller was a cheap and reliable solution. The communication with the microcontroller proceeded with a RS232 interface.

Here is a list of used commands and a short explanation of their functions:

L0	switch a red LED
L1	switch a yellow LED
\r	send a carriage return/ end of a command
RS	reset of the camera
HR	change the resolution from 88x143 to 176x255
VW [x1 y1 x2 y2]	create a virtual window with the corners 1 and 2 the smaller the window, the faster the camera
TC [Rmin Rmax Gmin Gmax Bmin Bmax]	the camera searches the color with the given RGB value and answers with T packets until a new command is transmitted
T mx my x1 y1 x2 y2	m is the middle of mass value and x/y are the corner from a box with all pixels with this color
SF	send a whole picture
SV	send the servo motor positions

2.2.4 Compass

The DEVANTECH CMPS03 is necessary to know the exact direction during the turns at the end of the rows. The electronic compass measures the earth magnetic field with 2 magnetic field sensors, which are mounted at right-angled. They detect every change in the horizontal component of the earth magnetic field by comparing the values of the sensors.

The output signal is a pwm signal with a signal time that is equivalent to a direction (18ms for south and 36ms for north).

2.2.5 Photo interrupter

In each wheel a photo interrupter SHARP GP1S23 is mounted. These sensors have a resolution of 10 degree rotation of the wheel. With these sensors the driven distance and speed can be measured.

2.2.6 GPS

For exact localization a NAVILOCK NL-303P gps receiver is used.

In this way we can save the positions of the golf balls or the hole in the grass.

This receiver supports the NMEA protocol. This means, that it uses the RxD/TxD connections from RS232 interface to transmit the information. The NMEA protocol supports different input and output messages. For getting the position data the RMC message is read each second.

The following table explains the format of the message:

Name	Example	Description
Message ID	$GPRMC	RMC protocol header
UTC Time	123843.123	hhmmss.sss
Status	A	Data valid: A not valid: V
Latitude	5107.1234	ddmm.mmmm
N/S Indicator	N	north: N south: S
Longitude	01234.1234	dddmmm.mmmm
E/W Indicator	E	east: E west: W
Speed over ground	0.25	Speed in knots
Course over ground	123.45	Course in degrees
Checksum	*10	to test the transmission
<CR> <LF>		End of message

2.3 Radio

One important fact for the radio connection to a computer is the possibility to get the information from all sensors during the test runs and also the possibility to drive the robot remote controlled. Later it was easy to display the number of golf balls and a map with the location of the hole.

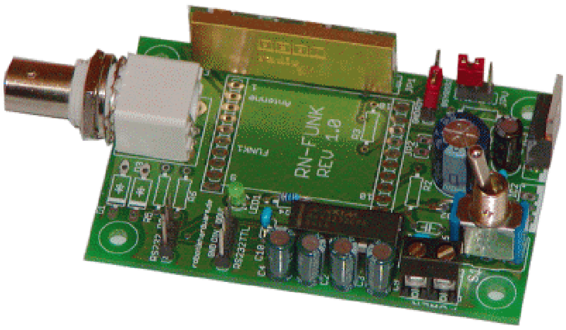


Figure 5. radio module

2 RN radio modules (fig. 5) can displace a RS232 cable and transmit information over a distance up to 200m. The baud rate can be changed with jumper.

2.4 Microcontroller

Our sponsor Werner Industrielle Elektronik offered us the Atmega128 on different board types. These controllers have a lot of advantages, which were useful for the competitions.

The Atmega128 has 2 communication interfaces so we can built a bus system and have still one interface free for the radio, camera and gps receiver. Another advantage is the price. When the controllers are cheap, we can use more of them and have more connections for sensors and more computing power.

These controllers work with a frequency of 14,7456 MHz and have a lot of peripheral features:

- 2x 8bit and 2 16bit timer
- 2x 8bit pwm channels and 6 16bit pwm channels
- 8x 10bit analog/digital converter

- I²C interface
- 2x serial USARTs
- 53x programmable I/O ports

It is necessary for the controllers to exchange data. A simple solution is a board with the hardware for a bus system (fig. 6).

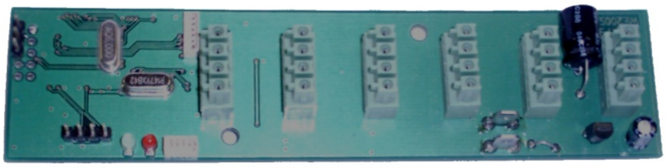


Figure 6. Atmega128 with hardware for a bus system

The microcontroller is mounted under the board. On one site of the board are 6 slots for more boards. As bus system the RS485 standard is used. It is a fast and reliable system and gave us the possibility to program our own protocol for data transmission. The board also supplies electricity for the other boards.

The camera, the radio and the gps receiver require a board with a RS232 interface and a RS485 interface (fig. 7).

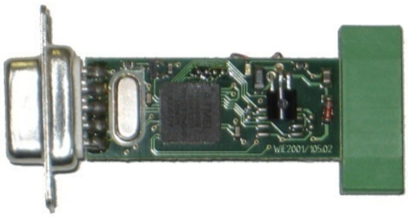


Figure 7. Atmega128 with 2 serial interfaces

Both interfaces supply a transfer rate up to 115,2kb/s for a reliable data transfer. They have a buffer of 2 registers for each receive register. When they are interrupt controlled the processor is fast enough to process every byte.

The main task for these controllers is to collect information from one interface and send them to the other interface. Both interfaces have different protocols and transfer rates. But the Atmega128 has 2 separate serial connections which makes it perfect for this task. The last board type (fig. 8) has a lot of I/O ports for the sensors, switches, servo motors and the cruise controller.

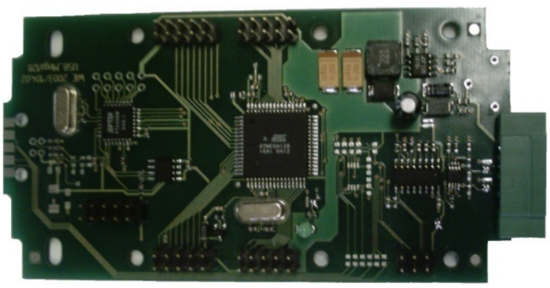


Figure 8. Atemga128 with all ports

2.5 Power management

As power source the robot uses two parallel 7.2V accumulators and one 12V accumulator. To allocate the power with the right voltage to every part, we built a board with voltage regulation for 12V, 7.2V and 5V. The controllers and the camera get their power direct from the

12V accumulator. The cruise controllers use also the current direct from the 7.2V accumulator. The other components use the voltage regulator.

2.6 Software

2.6.1 GUI

We developed a graphical user interface (fig. 9) for a pc with Visual Basic. When the pc is connected with a radio module the robot can be controlled without a wire.

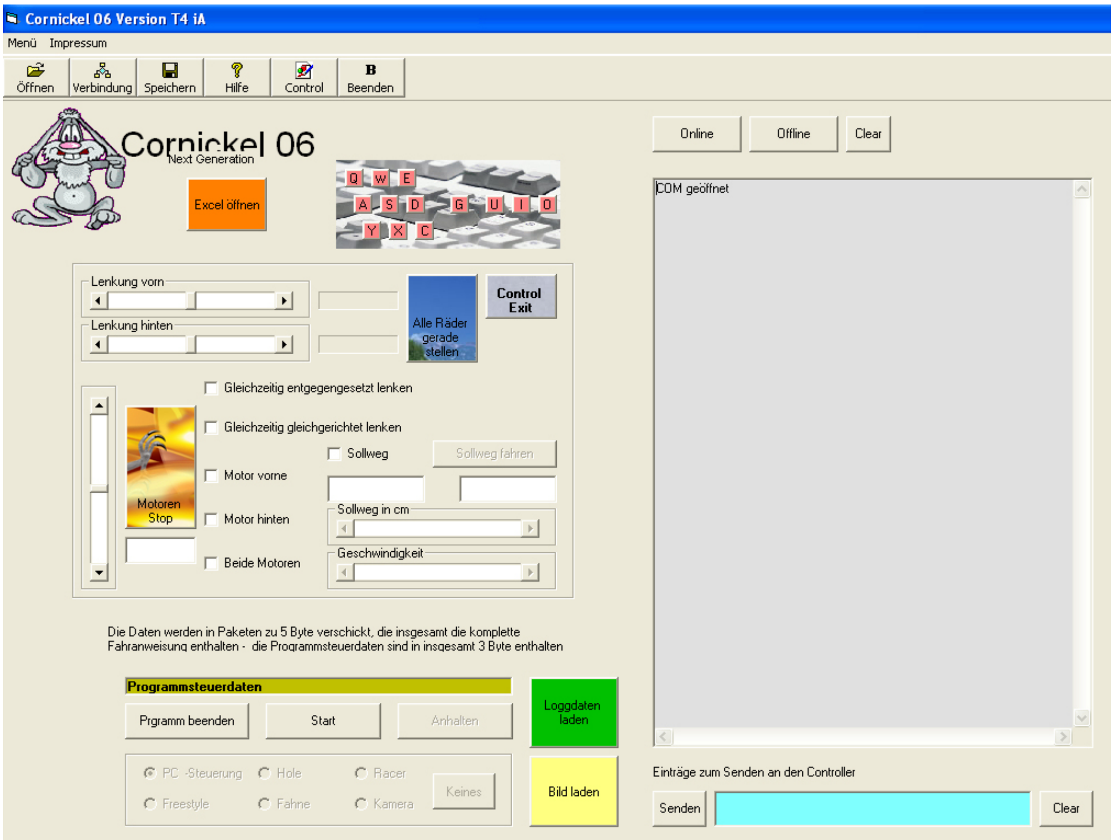


Figure 9.GUI programmed for Cornickel 2

This GUI has a lot of features. One important is the remote-control-mode. During the tests we could drive it remote controlled and analyze the values of the sensors. The GUI can also save all information like speed, driving direction and distance, gps position and sensor values in a table and can make a chart with the interesting values.

The GUI is also the interface to choose the program on the microcontrollers for the different competitions.

Every command from the GUI to the radio controller is send back to pc and printed in the right part of the program.

2.6.2 Programs on the microcontrollers

All 7 microcontrollers have different programs for their task.

Communicator

The communicator handles the complete bus system. It sends the program choice to the other controllers, collects all sensor and driving data and sends them to the task and driver.

Radio controller

The radio controller is the interface between the bus system and pc. It routes the commands from the pc to the communicator and sends data to pc.

GPS controller

The gps controller analyzes the gps and compass values and sends them to the communicator.

Sensor controller

The sensor controller analyzes the signals from the distance sensors and the photo interrupters. The photo interrupters and ultrasonic sensors are interrupt controlled.

Driver controller

The driver controller gets new driving information from the bus system and sets the output signals for driving direction and speed.

Task controller

The task controller can get sensor data from bus system and the 28 infrared sensors are direct connected during the hole detection competition. Each task requires other sensor data which are provided by the communicator. After calculations the controller sends commands back for the other controllers.

Camera controller

The camera controller gets the program choice from bus system. It sends the necessary commands to the camera and routes the data from the camera to the communicator.

3. Methods

3.1 The line

The task was to draw a straight white line towards a corner flag. The track color feature of the CMUcam2 is perfect for this task. After sending the TC command with a defined color range to the camera it answers with a T packet. This packet contains the middle of mass of the defined color and the most left and most right pixel. The driving direction depends on the position of the middle of mass.

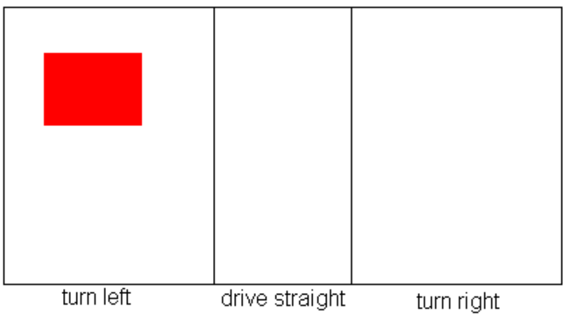


Figure 10. partition of the picture with a found flag

Depending on the x value of the middle of mass there are 3 different driving directions as shown in fig. 10. After start the robot turns left until it sees the corner flag the first time. This makes it independent from the starting direction and gives also the possibility to find the flag again after loosing it one times. The two ultrasonic sensors, which are necessary to navigate through the rows, are turned forward as the stop condition. If the value of one sensor under run 10cm the robot stops. A simple trailer with chalk is used to draw the line while driving to the flag.

3.2 Dandelion detection

This competition can be divided in three tasks. The first task is to navigate between wheat rows. Another task is to make turns at the end of each row and while driving through the rows the robot counts yellow golf balls which simulate dandelions.

3.2.1 Navigation between rows

Two ultrasonic sensors in the front measure the distance to the rows. The carriageway is divided in 5 parts depending on the distance to the rows. The closer the distance to the rows is the bigger is the steering angle to the other direction (fig. 11).

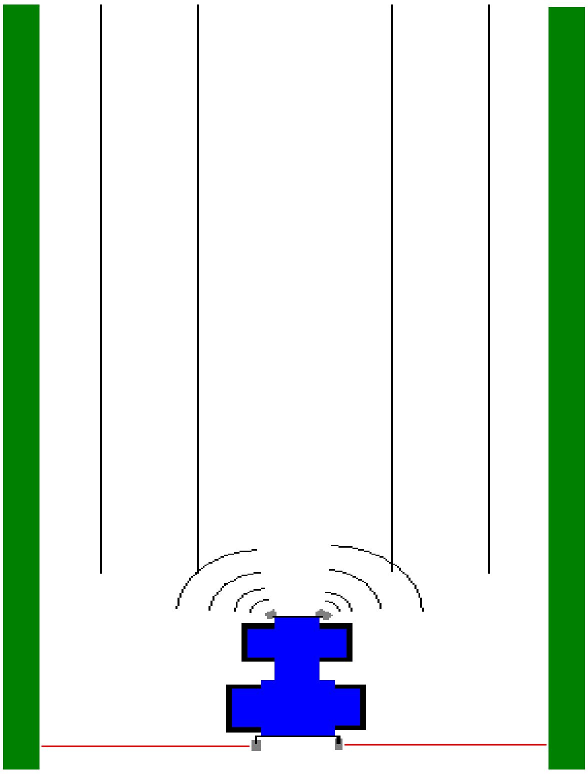


Figure 11. function of the ultrasonic sensors between the rows, two infrared sensors behind the robot detect the end of the rows.

3.2.2 Make a turn

At the end of each row the compass value is stored. While driving slowly with biggest steering angle the current compass value is compared with the stored value. If the difference is around 180 degree the robot drives straightforward until the ultrasonic sensors detect the rows.

3.2.3 Count golf balls

While driving through the rows, the robot should count yellow golf balls. The camera uses the color track function similar to detect the flag. The difference is the virtual window function of the camera, with which the camera only analyzes a small display window. This is necessary to distinguish one from two balls. Every time the camera detects a new ball, the camera controller counts it and the communicator writes the number of balls to the logging data.

3.3 Speed race

The task is similar to navigation between rows from dandelion detection. The differences were a smaller steering angle, because the rows were straight, and more speed to be fast enough. The two ultrasonic sensors measure the distance to the rows and the infrared sensors detect the end of the rows.

3.4 Hole detection

The robot should detect a hole in a 5x5 m² grass field. After a lot of tests with a camera and mechanical sensors a trailer with 28 infrared sensors was build (fig. 12).

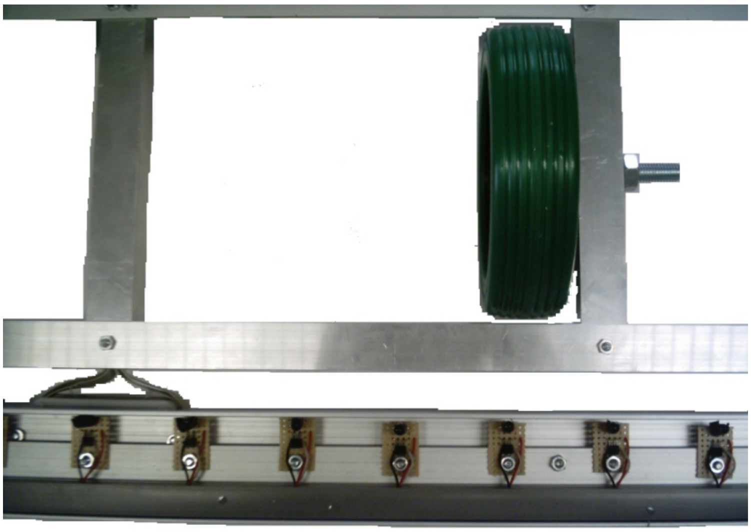


Figure 12 part of the trailer with infrared sensors

The distance between the sensors is 5cm so that the trailer reaches a resolution of 2,5cm, which is high enough for this task. These simple infrared sensors work like a switch. They turn on when something is nearer then 10cm. If there is a hole under a sensor it will turn off and a controller, which is connected with all sensors, saves the sensor, which found the hole. Together with the value from the photo interrupter (driven distance) the controller knows the exact location of the hole. The bus master sends the location together with the other logging data to the pc. Another program makes a chart from these data so that it is easy to find the hole.

4. Conclusion

The navigation through the rows worked well, as long as the plants were big enough. Some plants in the curved rows did not grow well and the ultrasonic sensors did not recognize them. Maybe it is useful to include the camera in the navigation concept.

The headland turns worked only one time well, depending on the compass. The compass is to accident-sensitive for such an environment. A gyroscope could be another solution to measure the rotation angle.

The camera had some problems with the infrared part of the sunlight and the vertical movement while driving over the soil.

Another attachment for the camera could solve the vertical movement problems. A black box with a small hole protected the camera against the sunlight. This can also be done by infrared filters.

A speed regulation was missing in hole detection competition.

The grass was not as expected. But a speed regulation can easy be implemented. All necessary data are available in the bus system.

The bus system has done all tasks well. It is fast enough and has all necessary functions for data input and data output implemented. It also had no problems with the temperature and dust.

Acknowledgment

We would like to thank our sponsors:

Werner Industrielle Elektronik (http://www.wie.werner-electronic.de/)

GWT: (http://gwtonline.de/)

References

http://www.roboter-teile.de/datasheets/cmps03.pdf

http://www.roboter-teile.de/datasheets/srf04.pdf

http://www.sharp.co.jp/products/device/lineup/data/pdf/datasheet/gp2d12_j.pdf

http://www.roboter-teile.de/datasheets/IS471F.pdf

http://www.roboter-teile.de/datasheets/LD274_OSR.pdf

www.navilock.com

http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf

http://www.wie.werner-electronic.de/

Cropscout II, a modular mini field robot for precision agriculture

J. Hemming, B.AJ. van Tuijl, J. Balendonck, EJ. van Henten, H.J.J. Janssen

Wageningen UR
Greenhouse Horticulture
P.O. Box 644, 6700 AP Wageningen, The Netherlands
e-mail: jochen.hemming@wur.nl

Abstract

In this paper a small agricultural robot named Cropscout II is described. Besides the objective to participate in the annual Field Robot Event competition Cropscout II operates as a modular test bed for autonomous robot control using sensor fusion techniques and artificial intelligence. The main challenge in this aspect is to cope with the poorly structured environment and the variation in shape, size and color of biological objects encountered in the open field. The very flexible and modular design of the system in both the electrical and mechanical way proofed to have many advantages. Unless some of the tasks to complete were solved very well the final conclusion is that it is still a big challenge to build a robot for the wide variety of different and unpredictable outdoor conditions. Future research on all aspects is essential.

Keywords

autonomous robot, field robot, navigation, agriculture, computer vision, maize, sensor fusion

1. Introduction

Cropscout II is the successor of the award winning Cropscout I (Henten *et al.* 2004), a smallscale experimental platform for research on sensors for precision agriculture. One the one hand Cropscout II was built to participate in the annual Field Robot Event competition (Straten 2004 and http://www.fieldrobot.nl) on the other hand it operates as a modular test bed for autonomous robot control. The main challenge in this aspect is to cope with the poorly structured environment and the variation in shape, size and color of biological objects encountered in the open field.

The 4th edition of the international Field Robot Event took place at the University of Hohenheim (Stuttgart) in Germany at 24th of June 2006. Inspired by the soccer FIFA World Cup held at the same time the tasks the field robots had to perform were as follows:

The line: can the robot detect a corner flag placed on the lawn and draw a straight white line towards it?

Dandelion detection: how many dandelions (simulated by yellow golf balls) can the robot count while navigating between rows? The robot will navigate through curved crop rows, spaced 75 cm. At the end of each row the robot is expected to make a turn, miss out one row, re-enter in the rows and keep going back and forth. In doing so, the robot has to count dandelions.

Speed race "all in a row": can the robot outspeed its competitors in an open race? The robot will be within a straight crop row and follow the row. The end of the row is the finish line. Collision free driving is required.

Hole detection in grass: There will be a competition field with lawn, approx. 5x5 m. The boundary is marked white – like on a soccer field. Inside, the lawn is damaged at one spot. The robot has to detect this spot. The hole will be some 10x10 cm wide and a minimum of 5 cm deep.

Freestyle: Present your own ideas. In this paper, the technicalities of Cropscout II are illustrated and the results of test-runs and of the competition are described and discussed.

2. Objectives

The objectives of the project described are as follows: The development of a robot which participates in the Field Robot Event competition and which wins the first prize. The development of a small experimental platform for research on precision agriculture (e.g. detection and control of weed and diseases). And finally the design and realization of a system which serves as a test bed for the development of autonomous robot control algorithms using sensor fusion techniques and artificial intelligence.

3. Materials and Methods

3.1. General construction of the vehicle

Cropscout II is based on a hand made wooden box containing the electronics mounted on top of a under carriage containing the motors, batteries and the tracks. After one of the gear transmissions from our custom designed under carriage broke down just some days before the contest we decided to reuse the motors and tracks from Cropscout I (Henten *et al.* 2004) which are made from a scale-model of a crawler. Sensors for navigation and orientation, including cameras are mounted around and on top of the vehicle. Two tracks, powered by electrical motors, are used as drive train. Figure 1 and Figure 2 show photographs of the robot and its components. In Figure 1 the optional spraying unit is mounted. In Figure 1 the upper box is opened to view some of the inside components.

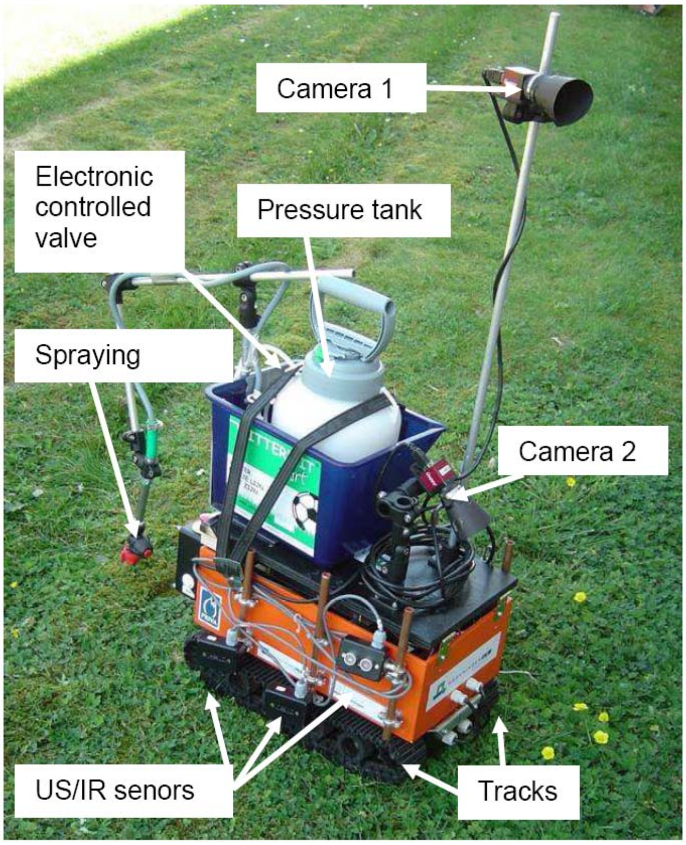


Figure 1: The components of Cropscout II (1)

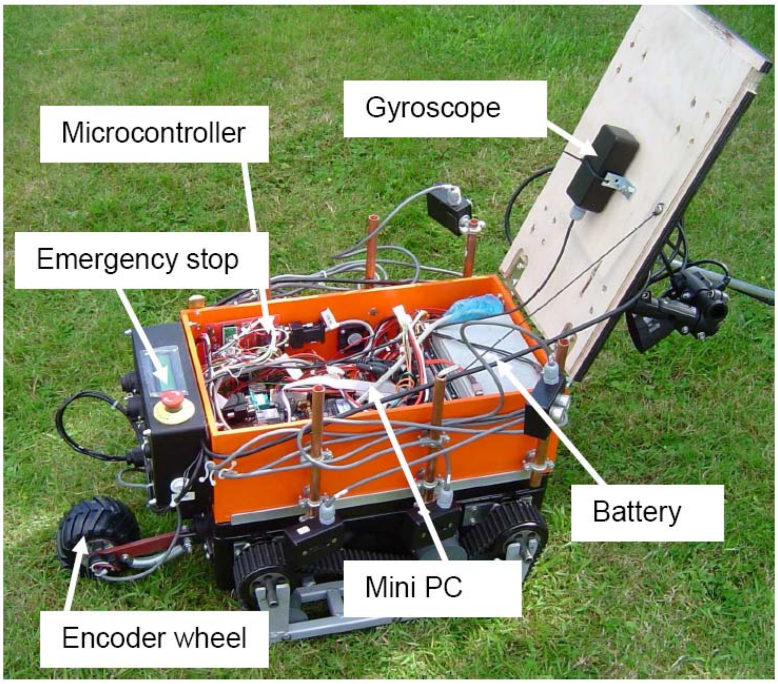


Figure 2: The components of Cropscout II (2)

Table 1 lists the mechanical dimensions of the robot and some properties of components used.

Table 1: Overview of different properties of Cropscout II

Property	Measure
Width	38 cm
Length	48 cm
Height	26 cm
Ground clearance	3 cm
Camera for corner flag detection mounted at	100 cm
Weight	15 kg
Drivetrain	2 electrical motors (Graupner BB700) with gear transmission
Conception	Track driven
Power	12 V Battery/7Ah for motors and sprayer 12 V Battery/4 Ah for electronics
Speed	mean speed approx 1.5 m/s

A 2x16 character LCD display is connected to the electronics. This user interface is used to provide information about the current state of the machine and information about the detected objects (i.e. number of golf balls found in the field). The robot is operated by a number of onoff and tip switches which can be accessed at the back side of the vehicle. The user can select an operation mode and can start, stop and reboot the system. For normal operation there is no need to attach a keyboard or monitor to the system.

3.2. Spraying unit and flash light

For the task of spraying a white line on the lawn a spraying unity was developed. This device consists out of a 5 liter plastic pressure tank, a solenoid-controlled valve, tubes and a membrane controlled minimum pressure nozzle to prevent dripping and leakage after shutting down the sprayer. The tank is filled with fluid soccer field paint and pressurized by hand. The unit can be mounted on top of the box. In place of the spraying unit also a small flash light can be mounted on the same electrical interface. The flash light was used for indicating holes and obstacles.

3.3. Sensors

The sensor concept of Cropscout II is a modular system which enables to use different kinds of sensors which can be positioned at almost any place on the vehicle. The different sensors were all mounted in the same type of housing with a standard mechanical and an electrical connector. A system consisting of several metal tubes, clamps and joints is used to position the sensors. Depending on the task, the positions can be changed quickly. The sensors used include infra-red range sensors, ultrasound range sensors, a gyroscope and two digital color cameras operating in the visible light spectrum. Sensor redundancy was implemented to increase the robustness of the system under varying outdoor conditions.

Cameras

Cropscout II is equipped with two color cameras (Allied Vision "Guppy", F-033C, 1/3" Sony Progressive Scan CCD) IEEE1394 with 6 mm lens. This very compact camera has a standard C-mount lens adapter and is able to acquire images up to a resolution of 640x480 pixels (VGA). For the tasks described in this document, an image resolution of 320x240 pixels and 160x120 pixels was used.

Infra red range sensors

Two short range infrared (IR) distance sensors (Sharp GP2D12, range between 0.10 m and 0.8 m) and two long range IR sensors (Sharp GP2Y0A02YK, range between 0.20 m and 1.80 m) can be used.

Ultrasound range sensors

Further two Devantech SRF08 ultrasound range sensors can be mounted on the robot. These sensors have a measurement range of 0.03 to approximately 6 m with an accuracy of about 0.03 to 0.04 m. The SRF08 uses sonar at a frequency of 40 KHz to detect objects.

Gyroscope

A gyroscope (Analog Devices ADXRS150) is used to measure changes in the yaw angle of the vehicle. The gyroscope produces a positive going output voltage for clockwise rotation about the Z-axis. By integrating the voltage readings over a defined period it is possible to determine (changes in) the heading direction of the device and thus of the robot it is attached to. This sensor is e.g. used for controlling the head-land turns.

Odometer

A free running extra wheel pulled by the vehicle was equipped with an encoder (Spectrol 120e, generating 128 pulses per revolution). This sensor is used as odometer. Figure 3 shows the used sensor positions for in row navigation and dandelion detection. On each side of the robot one ultrasonic, one long range infra red and one short range infra red distance sensor is mounted. All three sensors are rotated by some degrees so that they are pointing towards the direction of driving. Actually only one camera was used for the task of detecting the dandelions.

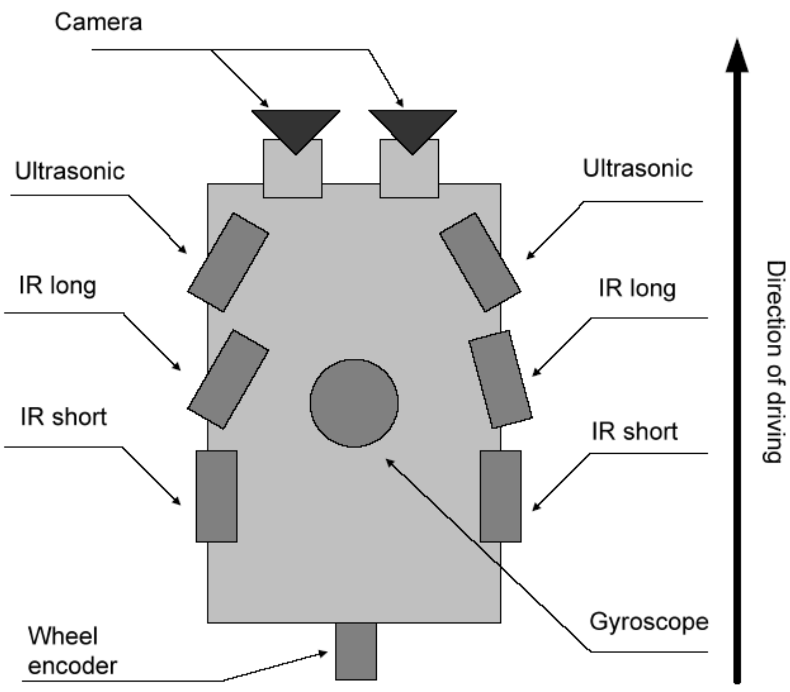


Figure 3: Sensor positions for in row navigation and dandelion detection (top view of robot)

3.4. Control hardware

The control hardware consists out of three main components:

- A mini-ITX PC mainboard with VIA Epia 1.3 Ghz CPU, 512 MB RAM and hard disk (http://www.via.com.tw/en/products/mainboards/)
- A Basic ATOM40 microcontroller (Basic Micro http://www.basicmicro.com/) for sampling sensors and switches; and
- A Rototeq (http://www.roboteq.com/) motor controller.

Via the USB port of the mini-PC a WiFi dongle is installed to exchange data with other PCs, handheld PDAs or other robots. Figure 4 shows the main schema of the electrical design. At low-level the microcontroller is used to sample the sensor values (A/D conversion) and the state of the switches on the back panel. Also the calculation of the heading direction of the vehicle based on the gyroscope values and the control of the LCD character display is done by the microcontroller. The motor controller is used for controlling the speed of the motors. The used controller has also a number of special inputs and is therefore used for sampling the wheel encoder of the odometer and to control the actuator port (spraying unit or flash light). Via serial interfaces the microcontroller and the motor controller communicate with the mini PC. This PC is used for image acquisition and image processing and for the high-level control.

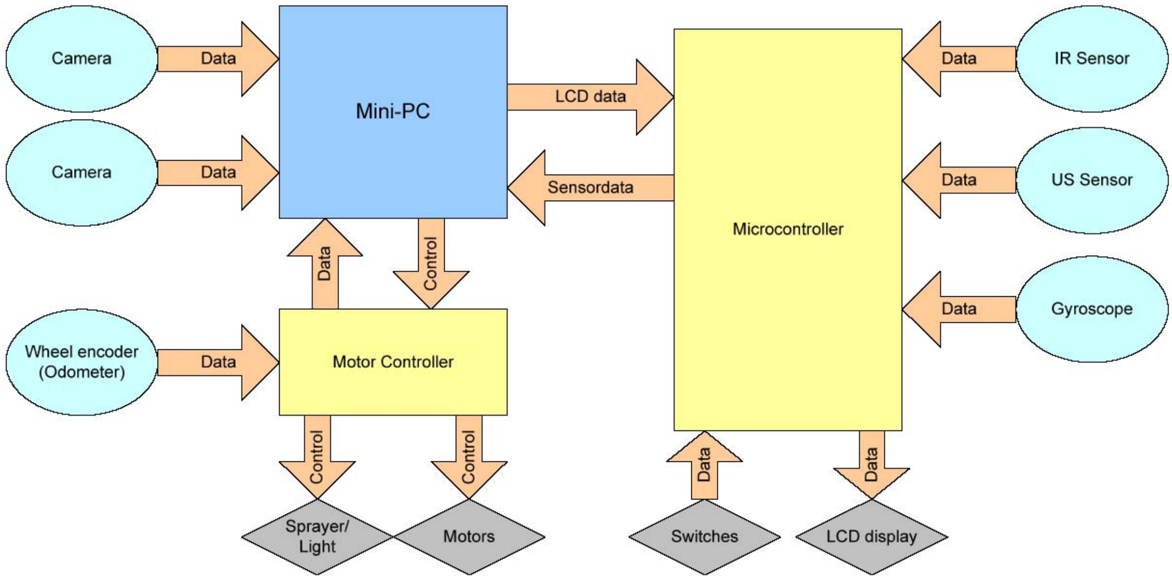


Figure 4: Main schema of the electrical design

3.5. Control software

The Mini-PC is running on the Windows XP operating system. National Instruments Labview 8.0 is used for the high level software layer and for image processing. C is used to program most of the control algorithms. Figure 5 shows a screenshot of the high level application. The ATOM microcontroller is programmed in Micro Basic 2.2. The software is running in a sequence with different steps:

Table 2: Main steps in high-level software

Step No.	Description
0	Read in configuration file and perform settings and store values in global variables
1	Perform init functions such as initialize variables and init cameras and COM ports as required
2	Main loop containing several multitasking loops. This step/loop is running until a “stop application” command is given. This main loop can handle different states or modes such as “navigate in the row” or “find corner flag” or “idle”
3	De-initialize hardware such as COM ports and IEEE1394
4	Shutdown functions such as exit Labview and shutdown/reboot PC

Once the main step is initiated several multi-tasking loops are running using a specific timing and priorities (Table 3).

Table 3: High-level multitasking loops

Timer	Task	Priority	Default timing [ms]
GUI	Update graphical user interface	Very low	100
ATOM	Read/write serial data microcontroller	High	10
Motor	Read/write serial data motor controller	High	20
Enc	Request data motor controller (e.g. wheel encoder)	High	100
Ctrl	Control intelligence loop	Very high	50
Vision	Image acquisition and processing loop	Medium	100
LED	Watchdog loop	Low	1000

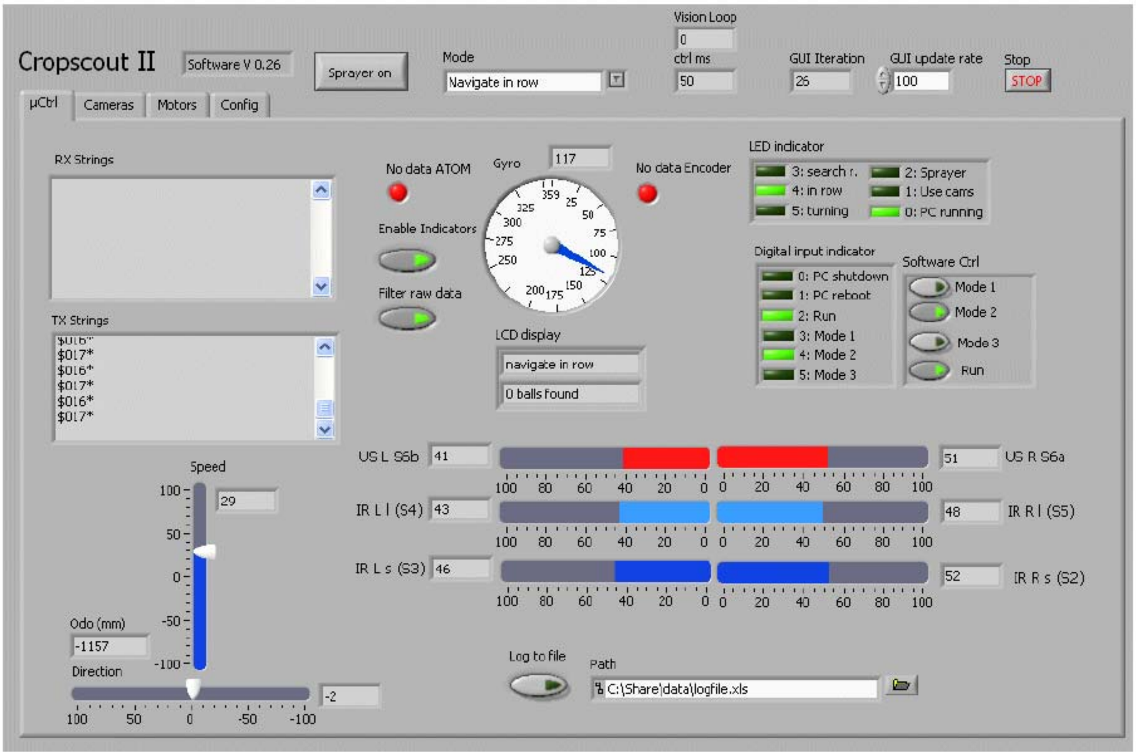


Figure 5: Screenshot of the main user interface

Strategy and control intelligence

The line

The strategy for drawing a line on the soccer field was as follows:

Search flag by computer vision while robot is running a small circle.

Once the flag is found, centre flag in front of the vehicle by steering based on the camera image.
Once centered, drive straight towards the flag while spraying the line. The straight forward drive is based on the readings of the gyroscope.
Stop when flag is reached (measure distance of approaching flag post by an ultrasonic range sensor looking straight forward).

The system can be configured to look for a specific flag color (e.g. red or yellow). To make the color detection more independent from changing light conditions the red, green, blue (RGB) color space of the image was first transformed to hue, saturation, intensity (HSI) color space. The algorithm tries to detect a post (a straight line) beneath the detected colored blob using edge detection methods. An example image is given in Figure 6 where an edge is detected (indicated by a red line) within the edge search area (green box). Only the combination edge and colored blob gives a valid flag detection result. Furthermore, the system calculates the centre deviation of the flag position in relation to the camera position. This value is used to control the motors in such a way that the flag gets centered in front of the robot.

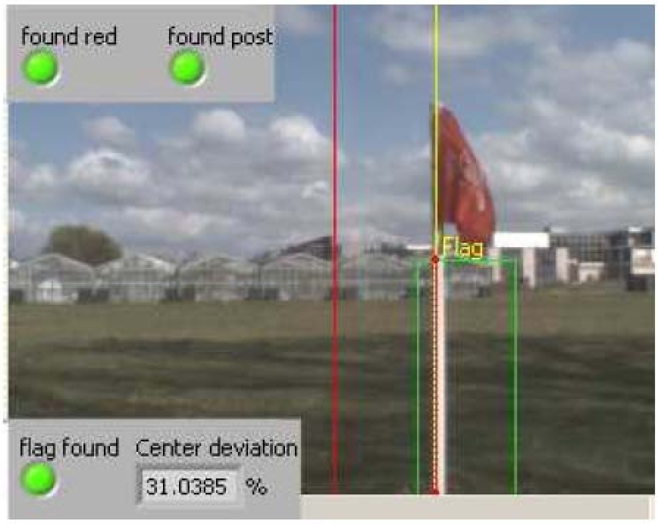


Figure 6: Locate position of corner flag by computer vision

Navigate in row and count dandelions

The strategy for this task is as follows:

Search row.
Navigate in row based on infrared and ultrasonic sensors.
Detect and count dandelions by computer vision (color and shape parameters) while driving in the row.
At the end of the row turn headland based on gyroscope.

Much of the in row navigation code used in Cropscout I was reused for Cropscout II. Refer to Henten *et al.* (2004) for an in depth discussion about this subject. One major difference is that cameras and computer vision are not used for navigation this time. This is due to the fact that the idea of the competition field this year was to have green maize plants on a green undersown crop. This will make it very difficult if not impossible to implement an image processing algorithm which can detect the crop row. It is the objective to drive Cropscout along a trajectory exactly between both rows. The offset from this trajectory is measured by the pairs of sensors mounted on each side of the vehicle. The offset is translated to a control signal to drive the individual tracks. Once the end of the rows is reached, a turn is implemented using the gyro signal. The sensor-based detection of the rows of maize plants plays a crucial role in Cropscout control. Switching from the ‘search for row’ state to the ‘navigate’ state and to the ‘turning’ state *etc.*, is fully determined by the detection of the plant rows.

For the detection and counting of the yellow golf balls (the dandelions) one camera looking ahead on the field in front of the vehicle was used. The image acquisition of the camera was triggered by the odometer in such a way that there was merely no overlap in successive images (e.g. one image per 80 cm). Each image was analyzed individually. As for the corner flag, first the RGB color image was transformed into the HSI color space. Objects of a certain size and a circular shape showing the pre-learned “dandelion color” were counted as valid object. The number of found objects was presented on the LCD display.

Speed race

For the speed race Cropscout II used the gyroscope to drive straight ahead. Before the start the robot was placed manually in the correct orientation in the row. Measured deviations in the driving direction during the run were compensated by the controlling the speed of the two tracks. Motors were set to near maximum speed.

Hole detection

The strategy for this task was as follows:

Navigate in-between white lines (based on information of the first camera).
Search holes with the second camera.
Indicate hole by flashing a light and make avoidance maneuver.

For this task both cameras were used. The task of the first camera was to look ahead some centimeters and to detect white lines on the lawn. Once a line was detected the robot is supposed to make a turn of 180 degrees based on the gyroscope in order to stay within the contest field. A line detection algorithm which could cope with incomplete and fuzzy lines was developed using color algorithms and morphological image processing operations. At the same time the second camera looked for spots which were neither green nor white (the holes).

Freestyle

For the freestyle session Cropscout II performed a spot spraying task. The strategy was as follows:

Slowly drive forward.
Search for small artificial flowers (coloured pieces of plastic).
If detected, drive robot to the flower.
Stop at flower and spray flower with water from the line spraying unit.

Navigation for this task was done by computer vision using the data of the odometer.

3.6. Contest field and weather conditions

As in the earlier editions of the event a real outdoor maize field with straight and curved rows was used for the competition this year. However, inspired by the soccer world cup some additional modifications were introduced. Some tasks should be performed on grass/lawn and some on maize rows sown on a ‘green’ bed consisting out of undersown and mowed grain.

Due to unsuitable weather conditions in spring the maize did not emerged well so that the coordinator decided to mow rows directly in undersown barley. Figure 7 show a photo of the curved rows section. One of the effects was that the remaining mown undersown crop did have a completely different color - way less green as expected. The “crop-row” was at many places also not as dense as expected, large gaps made the row detection more difficult. Also due to very dry conditions just before the competition much of the undersown crop was dried out completely.

Figure 7: Contest field with curved rows of grain

On the day of the event there was clear sky and high temperatures above 30 degrees Celsius in Hohenheim.

4. Results

4.1. The line

Navigation straight to the corner flag and drawing a white line (Figure 8) worked very well during test sessions and also during the contest. Because of the dual feature based image processing (flag and post of flag must both be detected), the detection of the flag turned out to be very reliable and was not affected by e.g. somebody wearing a red t-shirt standing behind the flag. The spraying unit operated flawlessly and a professional white line was drawn by this 10 implement. Because the implement was mounted behind the vehicle it was not possible to spray the line up to the position where the flag was plugged in the field. On the other hand this minimized the paint soiling the robot.

Figure 8: Drawing a line towards a corner flag

4.2. Navigate in row and count dandelions

At the end of the development the robot could navigate, count and turn in a satisfactory manner in the artificial test field the authors had built up indoors. Having the hot, dry and dusty outdoor conditions during the contest the performance decreased dramatically. The leaves of the grain plants of the contest field were very thin in comparison with the maize plants we expected and tested to navigate through. The IR range sensors did not give a robust signal due to lack in reflection of the infra red light. Finally they could not be used for the task of navigation. In addition to this problem the ultrasound range sensors were influenced by dust so that also this signal was unreliable. Thus the sensor fusion concept intended to use failed because most of the sensors did not give a reliable result. The overall row navigation result during the contest was less than expected.

Counting yellow balls was successfully tested indoors and outdoors. To aggravate the situation during the contest the yellow balls provided by the coordinator did have a light yellow "neon" color which turned out to show almost no color component in direct and high intensity sunlight. To stay in the given row of the field the robot had to be reset some times during the competition run. As a consequence the golf ball counter was reset at the same time so that the robot could not present the right number of balls laid out in the field.

4.3. Speed race

The robot performed very well during the contest and we ended up on the 3rd place. The robot drove very straight and stayed in the row. Only the top speed was less than the top speed of some of the competitors.

4.4. Hole detection

The contest field provided by this task was not very suitable for Cropscout II. The field was soggy and the white border line was hardly visible and turning was difficult. In addition to this, the hole detection algorithm (looking for blobs in the image which are neither green nor white) did sometimes classify shadows as holes. Due to a time consuming control algorithm the performance of the mini PC was at the edge of its possibilities causing the whole system to stall sometimes. Anyhow the whole indicating mechanism with the flash light worked perfectly.

4.5. Freestyle

The robot performed pretty well in the freestyle session. The flowers were detected by the system. However in some cases the positioning of the spraying nozzle was inaccurate. As far as we could analyze this was due to a system overload in the high level control caused by the combination of the complex image processing and the control of the vehicle. Also the high outside temperatures may have slowed down the system control.

4.6. Overall remarks

The very flexible and modular design of the system in both the electrical and mechanical way turned out to have many advantages. Such a system is very suitable as a test bed for research and can easily adapted to new tasks. The use of Windows XP as the operating system of the high level controller simplified the integration and the debugging compared to the use of a microcontroller only, as was done in Cropscout I. The mini PC could be easily integrated in and accessed over the network (both wired and wireless). Labview as graphical programming language also allowed rapid implementing of user interfaces and of a complex multitasking system. The use of the comprehensive image processing library of Labview was also favorable and time saving for the project. Drawback of the windows operating system is the lack of real time performance. Much more then expected the cycle time of time critical loops like the core control loops were affected and delayed by other tasks that caused the whole system to stall sometimes.

The used microcontroller turned out to incidentally reset itself. The reason was not found but caused unpredictable behaviors of the robot. The BASIC stamp is a low budget microcontroller which is easy to program but afterwards it would have been worth to invest in a more powerful component here.

The weight of the vehicle turned out to be almost too high to be carried and driven by the mechanics. This was also caused by the fact described above that the authors had to reuse the motors and tracks from Cropscout I. A vehicle on tracks makes the control easy; no wheels have to be steered and the traction on the field is high. In Cropscout II the setting of the track speed is implemented as an open loop control without feedback. Advisable for the future is to implement a closed loop control enabling a much better control.

5. Conclusions

The objective to develop a small experimental platform and to create a test bed for autonomous robot control algorithms was fully realized. The sensors, cameras, control hardware and software can easily be deployed and adapted for various applications.

It turned out that even for an experienced team it is still a big challenge to build a robot which can cope with the wide variety of different and unpredictable outdoor conditions. Beside these aspects most of the components used will not be able to deal with conditions encountered in agricultural practice: the system is not yet waterproof, sensors are sensitive to dust, mud and high or low temperatures. Due to the capacity of the batteries the operation time is limited to less than one hour. Future research on all aspects is essential.

Acknowledgements

Financial support by Priva, De Lier, The Netherlands (http://www.priva.nl), IMAGWageningen, Wageningen, The Netherlands and WitterWit Sport, Bekenbergen, The Netherlands (http://www.witterwit.nl/) is gratefully acknowledged.

References

Hemming, J.; van Tuijl, B.A.J ; Balendonck, J.; van Henten, E.J.; Janssen, H.J.J. (2008): Cropscout II, a modular mini field robot for research on precision agriculture. In: Conference Proceedings CD of the International Conference on Agricultural Engineering / Agricultural & Biosystems Engineering for a Sustainable World. - EurAgEng (European Society of Agricultural Engineers)

Henten EJ, Tuijl BAJ, Hemming J, Achten VTJ, Balendonck J, Wattimena MR (2004). Cropscout – a small-scale experimental platform for research on precision agriculture. In 'Proceedings of the AgEng2004 Conference'. Leuven, Belgium pp. 16-17.

Straten G (2004) Field Robot Event, Wageningen, 5-6 June 2003. *Computers and Electronics in Agriculture*, 51-58.

DEMETER – Autonomous Field Robot

M. Telama[1], J. Turtiainen[1], P. Viinanen[1],J. Kostamo[2], V. Mussalo[2], T. Virtanen[2]
and *academic advisors* T. Oksanen[1], J. Tiusanen[2]

[1]Automation Technology Laboratory
Helsinki University of Technology
email: timo.oksanen@tkk.fi

[2]Department of Agrotechnology
University of Helsinki

Abstract

An autonomous robot Demeter was developed for the Field Robot Event 2006. The four-wheel driven robot was built on a modified RC-platform. Spring suspension has been replaced with a middle joint. The robot has been build symmetric, and having all the wheels turning, is equally capable of driving into either direction. The robot has been made out of aluminum and is battery driven. The robot is controlled by two microcontrollers and a laptop that all onboard. The basic tasks for the robot are; driving towards a flag, driving between rows and hole detection in grass. The robot archives this by using machine vision and ultrasonic sensors. An electronic compass was included as an additional sensor. Machine vision uses a webcam attached to a servo that rotates the camera to point the robot's driving direction. The robot comes with a trailer that measures soil hardness and moisture. The work was done by a group of students from two universities during the semester 2005-2006. This document describes the development and technology overview of the robot.

Keywords:

field robots, ultrasonic sensors, machine vision, autonomous navigation, Simulink, OpenCV, hough transform, kinematics, simulation

1. Introduction

In August 2005 a group of six students began planning the building of a robot that would take part of the Fieldrobot competition 2006 in Hohenheim. A similar team from the same universities had taken part in Fieldrobot competition in 2005 under the name Smartwheels. After sharing information with this team, it was concluded that building the robot should start from a clean table. This time, a considerable amount of effort was put into planning and designing the robot. Number of different design possibilities were evaluated, before choosing the current design. Great care was put into building the mechanics platform to perform as well as possible. This delayed the project and it was not until April that the first tests could begin. That and various unexpected problems kept the team occupied to the last minute.

2. Materials and methods

2.1. Hardware

2.1.1. Chassis

The main task of the chassis of the robot is simple: It should be able to carry the measurement devices and provide the robot the ability to operate in field conditions. There are numerous different ways to achieve this goal and the selection of the technical solution depends on many factors. At least following properties of the chassis are desirable:

- Reliability
- Low cost
- Good off-road properties
- Good accuracy and ability to respond to instructions
- Low power consumption

In practice the low budget of student project forces to some tradeoffs and all of these properties cannot often be fulfilled in the desired way. However, it should be kept in mind that the chassis of the robot is the base on which the all the sensor and computer systems are built. It is obviously essential requirement that the chassis is able to drive all the equipment to the position defined by the controlling computer.

The position of the robot and the position defined by the computer are not the same due to measurement errors and perturbations. There are many sources of perturbations when the robot operates in the field. Error to the dead reckoning position measurement can be caused by e.g. inaccuracies in wheel angles, slip of the wheels, errors in rotation measurements of the wheels and so on. These errors cumulate when driving the robot for a longer time in the field and therefore additional position measurement and navigation systems are needed. Despite of the additional navigation systems the off road properties of the chassis and its ability to follow control signal have a significant impact on the performance of the robot. It is not reasonable to compensate the large errors caused by the chassis with a sophisticated computer control program. If the errors caused by the chassis can be kept as small as possible, the reliability of the navigation and accuracy can be improved. The key point in developing the motion system of the robot is to ensure good controllability in all conditions that can be normally expected in the field.

The chassis for the robot is based on a Kyosho Twin Force R/C monster truck. Building began with the original chassis which needed some modifications, but eventually a completely new chassis was build. Only the tires, axles and some parts of the power transmission were kept in the final version of the robot.

The chassis is newly built with self-made parts with no previous drawings, though based loosely to the original Twin Force parts. Aluminium was chosen because of its lightness. The engine rooms have been separated from the rest of the chassis and the microcontrollers have been placed into a separate slot to allow easy access. Chassis is four-wheel-driven and each one of the engines powers one axle. All of the springs have been removed, and instead there is a large joint in the middle of the chassis providing the robot manoeuvrability somewhat similar to farm tractors.

The vehicle is electrically driven with two Mabuchi 540 motors. The motors are driven by two PWM amplifiers.

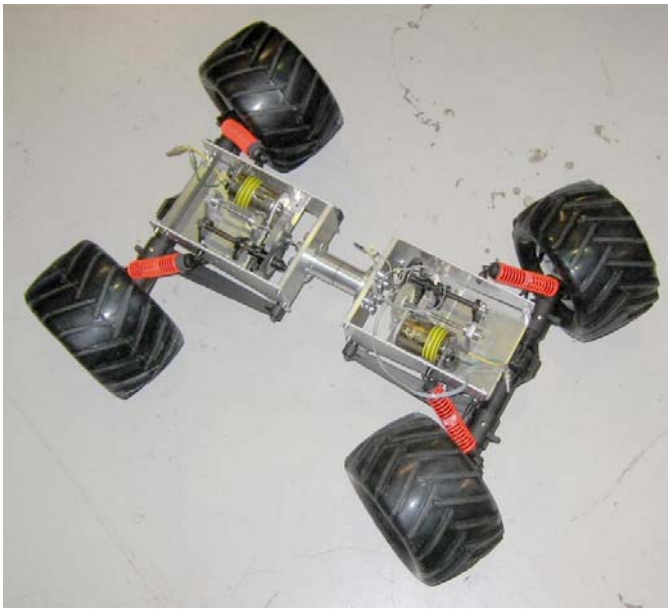


Figure 1 Motor and steering unit

The Motor unit of the Twin Force monster truck was modified to meet the requirements of the field robot. Two separate motor units were used for both front and rear axis. The motor units were identical and they were operated independently from each other. The use of two motor units enables the independent velocity control and independent steering of both front and rear axis. The main modifications done to the motor unit were:

- The maximum speed of the Monster truck was reduced to one quarter by adding an additional pair of gears.
- An optical encoder was attached directly to the shaft of the Mabuchi drive motors.
- A steering servo was attached to the motor unit

The chassis didn't originally have four-wheel-steering, so it had to be build with optional parts provided by Kyosho. Four-wheel-steering was necessary to give the robot enough turning radius. Two HS-805BB+ MEGA servos were used for steering and they seemed to provide enough steering power for the robot.

2.1.2. Sensors and machine vision

Four Devantech SRF08 ultrasonic sensors were used together with a Devantech CMPS03 I2C compatible compass. The sensors were connected to I^2C sensor bus. For machine vision the robot has a Logitech QuickCam Pro 5000 camera attached to a carbon fibre pole with a servo motor to enable a turn of 180 degrees. The compass was placed to the camera pole, high enough to avoid magnetic fields that would create interference with it.

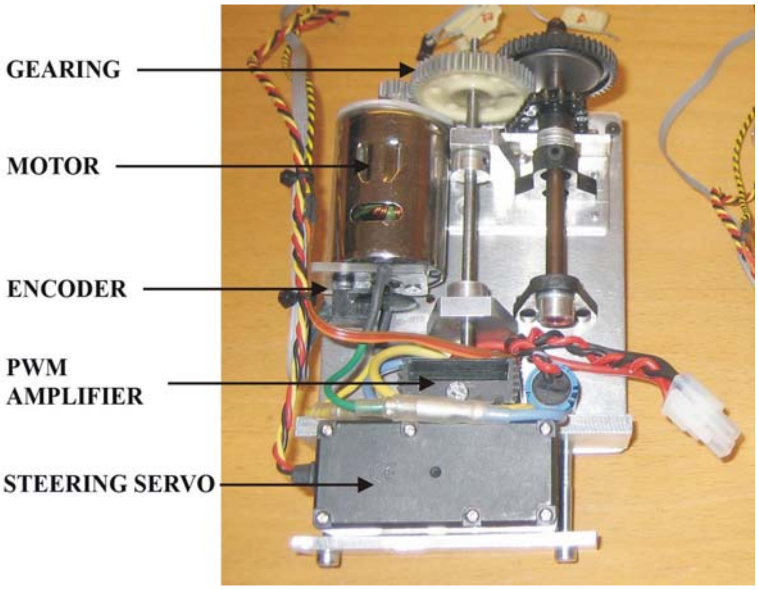


Figure 2 Power supply

Eight (8) pieces of 1.2V rechargeable Ni-Mh batteries soldered in series were used as the power supply of the robot. The capacity of each battery ranged from 7000 mAh to 9000 mAh depending on the battery package that was in use. The power from the batteries was proven to be adequate.

2.2. Processing equipment

2.2.1. Microcontrollers

The robot has two microcontrollers: ATMEGA 128 and a PIC 18F2220. The first one was used to control the Msonic power controllers for the DC-motors, steering servos, and the camera-servo. The second one was used to collect sensor input from compass, ultrasonic sensors, and to connect to trailer's inputs and outputs.

2.2.2. Laptop

A P4 laptop was used to process all necessary tasks along with the microcontrollers.

2.3. Hardware for the freestyle

2.3.1. Trailer

The trailer was built from scratch. The platform had to be made big enough to fit all of the equipment and with enough ground clearance to give the linear motor enough room to function. Old Tamiya Clod Buster wheels were used together with a new spindle that was made to increase manoeuvrability.

The main equipment used in the trailer is a linear motor provided by Linak. All the electronics have been fitted into two boxes. The linear motor is operated by two relays. The primary idea for the freestyle was to measure the penetration resistance of the soil (cone-index) and simultaneously measure the moisture of soil by measuring the electrical conductivity of the soil.

The linear motor was used to thrust two spikes into soil. Soil penetration resistance could then be measured from the amount of current consumption changes in the linear motor. Simultaneously the electrical conductivity of the soil was measured with an input voltage of 5V directly from the microcontroller of the robot. This was an experiment and it has to be remembered that the soil conductivity measurements rely heavily on other properties of the soil, especially on the amount of different salts there are in the soil.

The linear motor operates with a 12 V lead-acid battery, which, although being quite heavy, doesn't give the trailer enough mass to thrust into hard soil. However the trailer cannot be too heavy for it has to be pulled by the robot. The weight of approximately 9 kilograms was rather easily pulled.

3. Camera & Machine vision

3.1. Camera

Logitech QuickCam Pro 5000 was used for machine-vision. It's a higher end webcam, which is still quite cheap. Camera is capable of 640*480 resolution, but 320*240 resolution was used to improve performance and this doesn't require so much data transfer. Camera sends about 25 images per second to laptop. Laptop processes about 20 images per second and calculates parameters and image convertions for each frame.

3.2. Machine vision

Machine vision software was developed on Microsoft Visual C++ 2005 and Open Source Vision Library (OpenCV [1]). OpenCV provides libraries and tools for processing captured images.

3.3. Preprocessing

Image processing was done by EGRBI [2] color transformation (Excess Green, Red-Blue, Intensity). At the beginning of algorithm, image is split into 3 spaces: red, green and blue. In EGRBI transformation a new image-space is created. Components are Green, Intensity and Red-Blue (cross product of green and intensity). Intensity is 1/3(R+G+B value).

EGRBI is calculated by matrix product.

$$\underset{[320\text{x}240\text{x}3]}{\text{Image}} \quad * \quad \underset{[3\text{x}3]}{\text{mask}} \quad = \quad \underset{[320\text{x}240\text{x}3]}{\text{result}}$$

By changing mask Excess Red can be calculated easily.

The main idea of using this method was to detect green maize rows from dark soil. Green pixels could be detected from camera image by adding more weight on green and by ‘punishing’ red and blue.

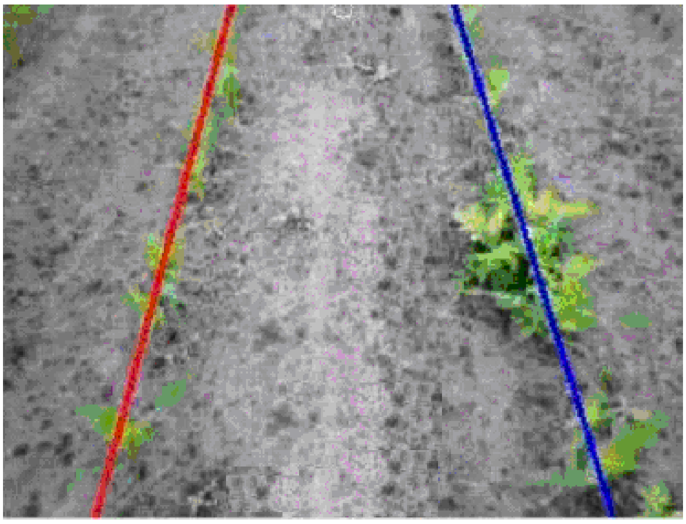


Figure 3 Calculated Hough lines from binary picture.

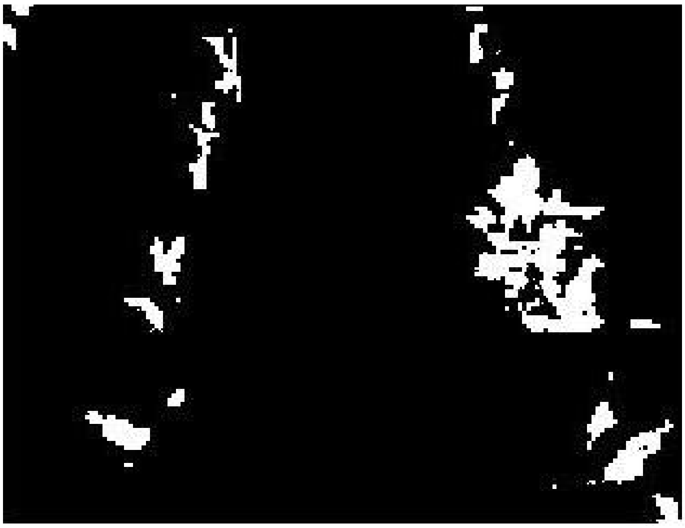


Figure 4 Binary picture, used EGRBI transformation.

Estimate of robot's positioning and angle between maize rows was made by Hough transform. Hough transform is quite heavy to calculate. For each pixel (in binary image) 2 parameter values (angle, distance) are calculated. So the pixels in binary image have to be kept as low as possible. Hough transform returns many lines that fit with the pixels. 10 best lines from each side are taken and mean value of the best lines is calculated. As a result left and right lines are gotten. From these the positioning and angle errors are calculated. The information is send to controller which then calculates correct steering controls.

3.4. Dandelion detection

While driving between maize rows, dandelions must be detected and counted. EGRBI with a yellow mask was used to detect yellow. This was made by finding proper intensity levels and different weighting in R/G/B-values. After binarization each dandelion became a contour. Each contour has position and area. Each dandelion should be detected only once so the positions of contours between image frames were compared.

3.4.1. Hole detection in grass

This section was performed by using inverse green detection. 10cm x 10cm hole covers a well-known proportional area in a picture, so hole can be detected in binary-image with correctly calibrated threshold value.

3.4.2. Driving towards flag

In this section, red flag was detected, in a way similar to detecting the green maize, except that this time a red mask was used instead. Center of the flag was calculated as the center of mass from the threshold binary image. Error was sent to controller.

4. Software

The microcontrollers only function as I/O devices and do very little processing to the data. One of them has the speed controller but apart from that all the intelligence is found in the main program which runs on a laptop PC. The main program was implemented in C++ .NET with MS Visual Studio. Camera- and machine vision functions were done using OpenCV-library.

Controllers are designed with Simulink and compiled into DLLs which are loaded into the main program.

The program's architecture is seen in the picture below. All the blue classes represent different threads and run in 50ms loops.

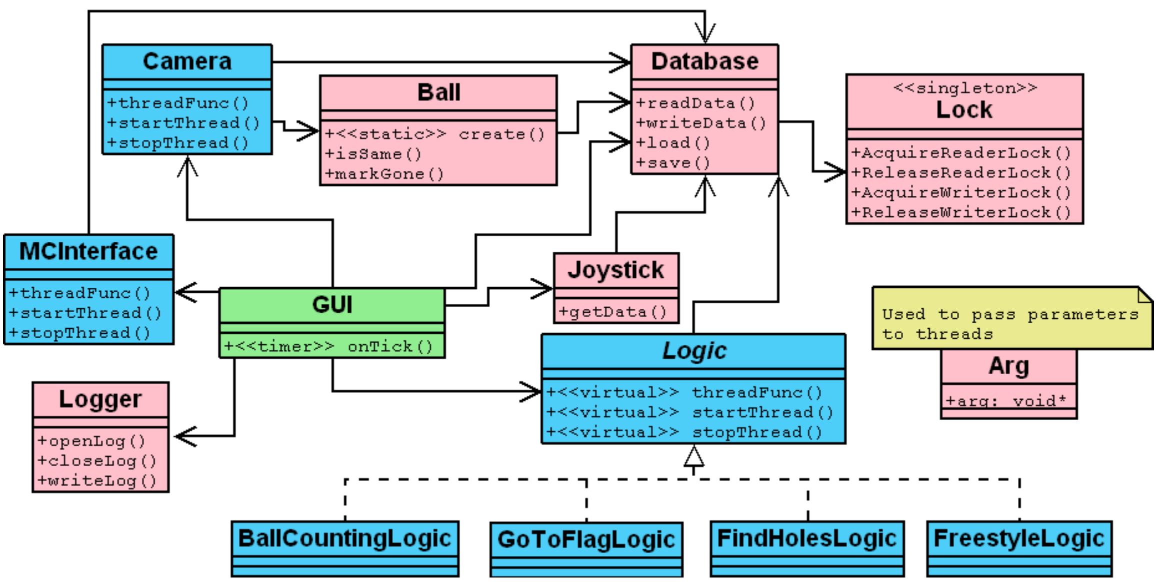


Figure 5 UML class diagram of the main program

GUI: Handles communication between user and the program. Has a 100ms loop.

Database: Contains all the data and takes care of synchronization. Have also methods to construct input arrays to different controllers.

Lock: Used by Database to prevent access to same data by different threads simultaneously. Implemented using Singleton pattern, only one instance allowed.

Joystick: Reads joystick using DirectX-interface.

Logger: Writes all input- and control data to a log-file.

Camera: Gets picture from camera and processes it. Runs in it's own thread.

MCInterface: Communicates with both microcontrollers. Runs in it's own thread.

Ball: Used to keep track of seen balls, so that each ball is only counted once. No public constructor. Only way to create new instances is to use createBall() method, which checks whether there already exists a ball at the given coordinates.

Arg: A managed class used to pass unmanaged object to Threads, which require managed parameters.

Logic: Reads the DLL for the controller. The base class for all the logics. This is an abstract class and all the real logics are derived from this. The derived classes have different routines to do different tasks.

5. Simulator

Simulator was implemented to better design the controllers. Implementation was done using Matlab and Simulink. It can be used to simulate the use of ultrasonic sensors. The simulator has an editor that can be used to create test tracks. The tracks also contain a correct route that is compared to simulated route to calculate error values. These values can be used to measure how good the controller is.

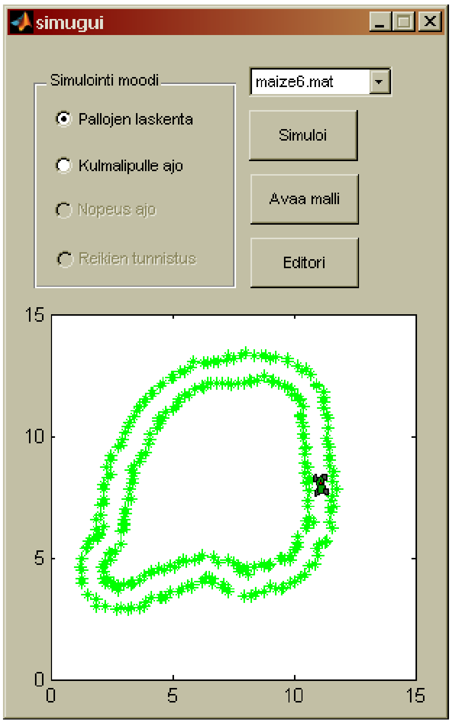


Figure 6 UI of the simulator

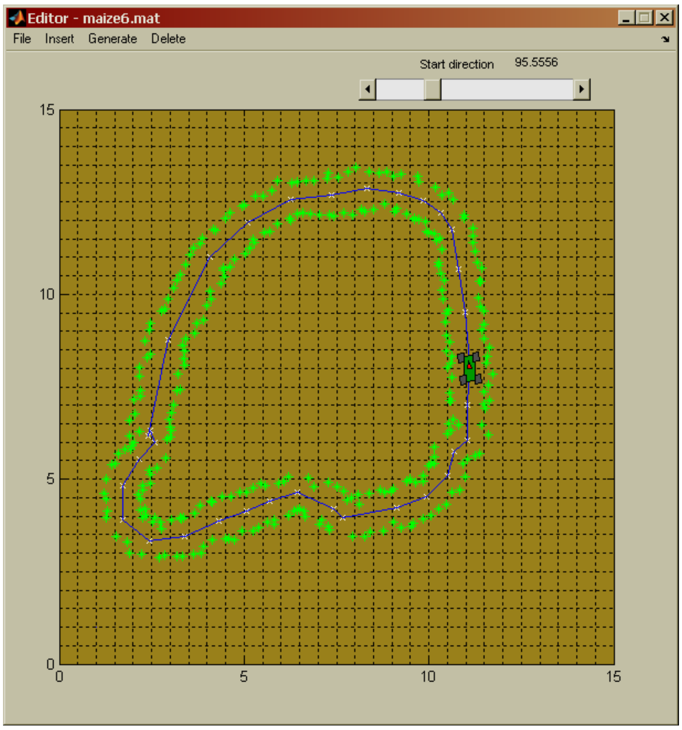


Figure 7 Editor for the simulator

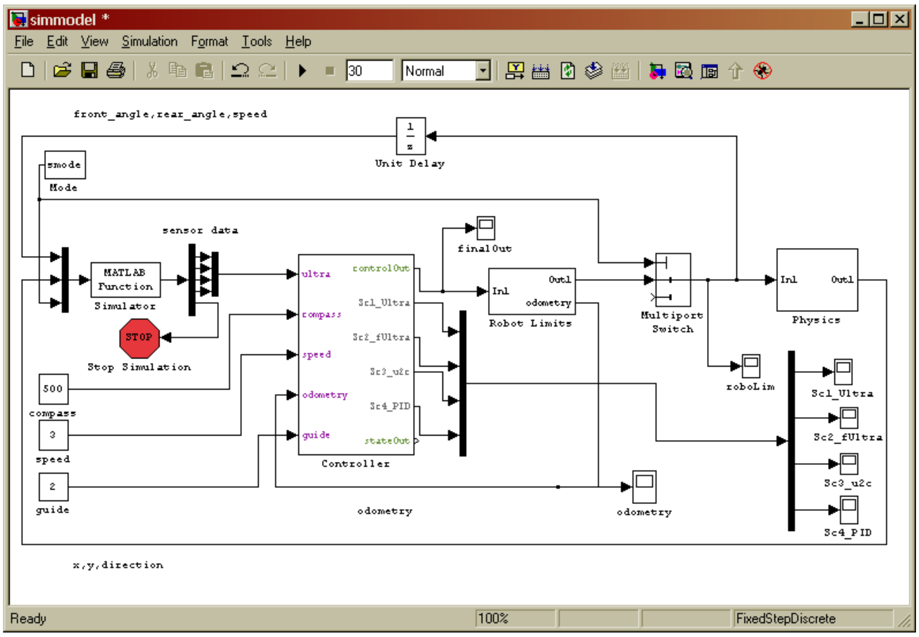


Figure 8 Simulink model of the simulator

6. Control logics

Control logics were developed in Simulink / Matlab. Each task was given an unique controller. Controllers were developed as a set of Simulink library blocks that could simultaneously be used in simulator and controller-export models. C code was generated from the Simulink model using real-time workshop. The final product from the Simulink model was a Dynamic Link Library file. Reason behind the idea was to make the imports to the main program easier. The DLL came with a function that was called from main program every 50 milliseconds. The function was given all the sensor information (ultrasonic, camera, odometry), as well as some parameters as an input. The output returned steering angles for front and rear wheels and speed, together with some other controls and debugging information depending on the task.

6.1. Driving between the rows

Of the four different controllers, Rowcontroller was the one used for driving between rows. The controller had four basic parts: filtering and processing of the sensor information, sensor fusion, control, and state flow, shown in figure 9. Filtering and processing was mainly done for the ultrasonic sensors and compass as the image processing had already been done in the camera class. Two methods were used to get position and direction errors from readings of the ultrasonic sensors. One method used direct filtering, that calculated the position and directional error from past 5 readings, filtering off any miss readings. The second method used backward simulation of the robot movement. Axes were drawn from the robots current coordinates and the robot movement was simulated backwards for a short distance while the sensor readings were plotted. After that a least square roots line was fitted over the plots to estimate the rows on both sides of the robot, figure 10. The sensor fusion part was used to combine the position and direction error values from camera and from the two methods with ultrasonic. The error values were taken to a control block that had two discrete PID controllers. One PID to control the position error and the other one for directional error. The outcome from the PID controllers was then transformed to front and rear steering angles. Final part in the controller was a state flow block that was responsible for the turn in the end of the row. Simulink's Stateflow block was used for the state flow. Two different turning methods were implemented. First method, took use of the robots crab like movement by turning both front and rear wheels into the same direction. The second method was a regular turn at the end of the row with pre-estimated parameters. The 'crab' turn proved to be more reliable as the robot was less likely to lose its direction, although by having to skip one row, the robot had to drive relatively long way out to get enough side movement. (The wheels were turned to an angle of 0.3 radians.)

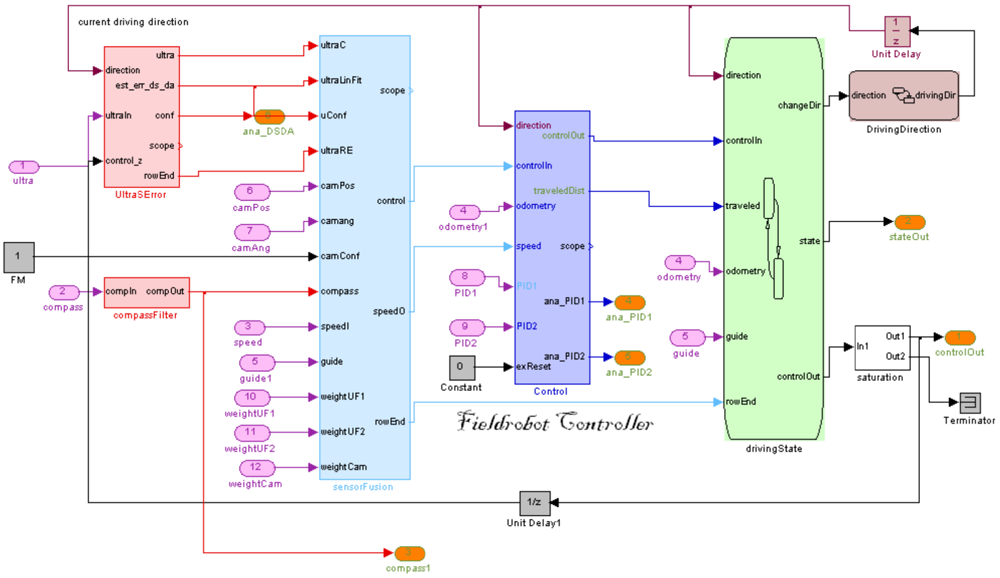


Figure 9 Simulink model of the rowController, used for driving between rows.

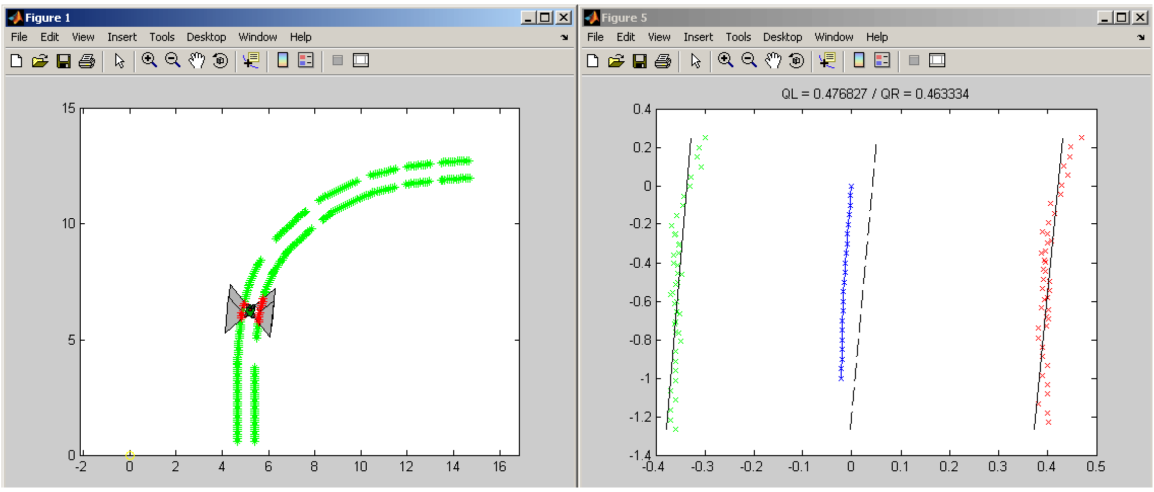


Figure 10 Simulation of driving in a maize field, right side shows the result of a backward simulation with least squares lines drawn.

6.2. Driving towards the flag

For driving towards the flag, there was a controller that only had a single PID controller. This time there was only the directional error to deal with as the position error could not be measured. Thus whenever the robot was turning, both wheels were turned the same amount to opposite directions. The directional error was gotten from the camera class. Flagcontroller also had a state flow block with couple stopping states. State changes were trigged based on information gotten from an ultrasonic sensor in front of the robot. Some additional rate and other Simulink blocks were used to smooth the robot movement.

6.3. Searching for holes

Our strategy for hole-seeking was to cover the whole area by driving straight and moving little sideways in both ends. As the robot had a 180 degrees turning camera and was equally capable of driving in to either direction the idea was to keep the robot heading direction same all the time. This was mainly done by just giving same steering commands to both front and rear wheels, but also with a help of a PID controller that used filtered compass reading to prevent robot from totally losing its heading. The controller came with two state flows, one for driving, and one for actions when a hole was seen.

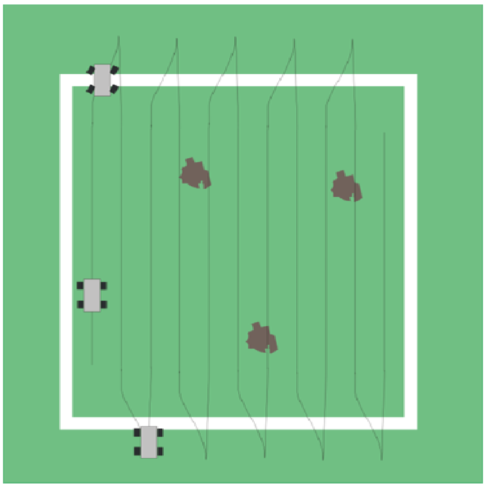


*Figure 11 Robot is facing the same direction at all the time in hole-seeking while camera is rotated 180 degrees to point into the current driving direction. Robot navigates trough the 5m * 5m area, and checks whether there are any holes in the grass.*

7. Conclusion

Quite many algorithms and other possibilities were researched and constructed, yet never used in the final product. They have been a bit of wasted time in making the robot, but might not be such a waste as far as the learning goes. While there were little problems with the laptops, they might still be considerable choice for this kind of projects. Laptops provide enough processing power for machine vision, and can also be utilized in the development part. It was however noticed that the UI should have been on a remote computer. The machine vision should be made more adaptive to reduce any need for manual calibration, to really get use of this kind of application. The use of Matlab and Simulink turned out to be very useful in the development of control algorithms, especially in the initial testing and debugging. Being able to first test the controller with a simulator and the directly export it to the main program was a great help. The use of a Stateflow block was also found useful as it made the understanding of the process easier for other team members, and it made the debugging faster. The mechanics performed well, especially having the middle joint instead of spring suspension has seemed to be a good choice.

The Team

Miika Telama, Jukka Turtiainen, Jari Kostamo
Pekka Viinanen, Tuomas Virtanen, Ville Mussalo
Trailer + Demeter

Acknowledgements

This work has been made possible by the following sponsors: **Henry Ford foundation Finland, HP, Linak, Koneviesti, OEM Finland**

References

[1] OpenCV, Open Source Computer Vision Library, Intel Corporation http://opencvlibrary.sourceforge.net/

[2] Nieuwenhuizen, A.; van den Oever, H.; Tang, L.; Hofstee, J.W.; Müller, J. 2005. Color-Based In-Field Volunteer Potato Detection Using A Bayesian Classifier And An Adaptive Neural Network. In: ASAE Annual International Meeting, Tampa, Florida, 17-20 July 2005.

[3] Koolen A.J. & Kuipers H. Agricultural Soil Mechanics. Springer-Verlag 1983. 241 p.

[4] Srivasta A.K., Goering C.E. & Rohrbach R.P. Engineering Principles of Agricultural Machines. ASAE Textbook Number 6, Revised printing 1995

[5] Rial, W. S., Han, Y. J. Assessing Soilwater Content Using Complex Permittivity. Transactions of the ASAE. VOL. 43(6): 1979-1985

FieldMeister, An autonomous vehicle – Analysis of the construction and achievements of a crop scouting autonomous vehicle.

H. IJken, A.van der Kamp, W. Middel, H. Stols, R. Werkman, P. Werkman

Students of Agrotechnology
Wageningen University
email: peter.werkman@wur.nl

Abstract

This paper gives information about an autonomous vehicle build by several students Agrotechnologie. The robot is able to navigate autonomous through the row, detect yellow balls in maize and count them. The robot is also able to drive to a red corner flag.

Keywords

crop scouting, autonomous steering, field robot, navigation, detection

1. Introduction

Every year in June there is organized a Field Robot Event. In June 2006 the Agricultural University in Hohenheim organized the FRE. The robots have to navigate through the row, detect a count yellow balls and navigate to a corner flag.

Six Agrotechnology students from Wageningen University decided to participate at this event. In March 2006 they started building their FieldMeister.

2. Material and methods

2.1. Chassis

The chassis of the FieldMeister is for a part built from aluminum. A part of the chassis is used three times in earlier robots. In 2003 the chassis was used in the robot Agrobot 2, in 2004 in Challenger and 2005 in Rowbo. By using a part of the existing chassis, we could build the robot faster and cheaper. Big different with the chassis last years is the number of wheels. This year there is chosen for a three wheel driven vehicle, with a steering front wheel.

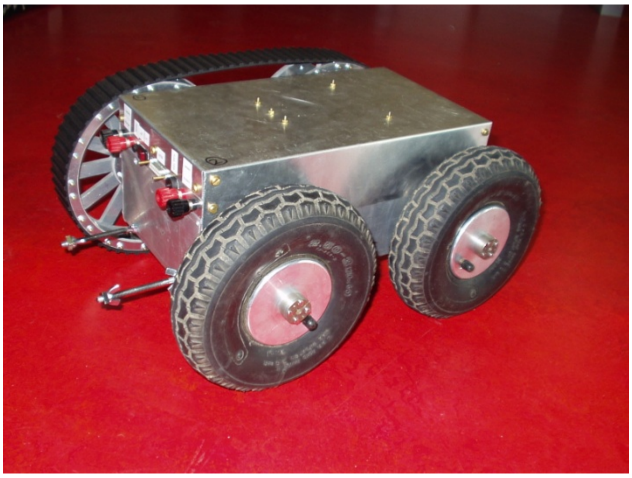


Figure 1: The chassis from earlier years

Figure 2: The FieldMeister chassis

Looking at the competitors of last years, we were thinking about the most ideal steering system. Last year we saw at the Field Robot Event many robots on tracks, or robot on wheels, without a steer, steering like tracks. We decided to look for an other steering system than last year, because skid steering is a very unpredictable steering system. A skid steered vehicle can drive many different curves with the same engine rotations, due to different slip of the wheels.

After looking to many steering systems we started to change the skid steered vehicle into a trike, for better steering control. With a driven front wheel steered trike we wanted to realize a vehicle that can drive curves with less slippery wheels.

We removed one wheel and put one wheel in the front of the robot. After that we put the engine that drives the front wheel at the bottom of a vertical axle. On the top of the vertical axle the engine used for the fourth wheel previous years was assembled. To protect the engine against high forces on the axle of the engine and to enlarge the steering power, the vertical axle is not directly mounted on the steering engine, but on a second axle. This axle with strong bearings, make the front wheel of the FieldMeister very solid. So it can resist high forces from several directions. The both axles are connected by gears, so a decrease in rotations per minute to increase the steering power was possible. So a solid driven front wheel with a steering translation of 85 degrees to the left and to the right was realized.

2.2. Control Hardware

The control hardware of FieldMeister exists of a microcomputer,

RoboteQ's AX3500. This microcomputer is a dual channel DC motor controller capable of directly driving up to 60A continuous on each channel up to 40V.

The AX3500 is used for several applications in 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.

The RoboteQ is able to work with 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. (Bruggen, R. van *et al*, 2005)

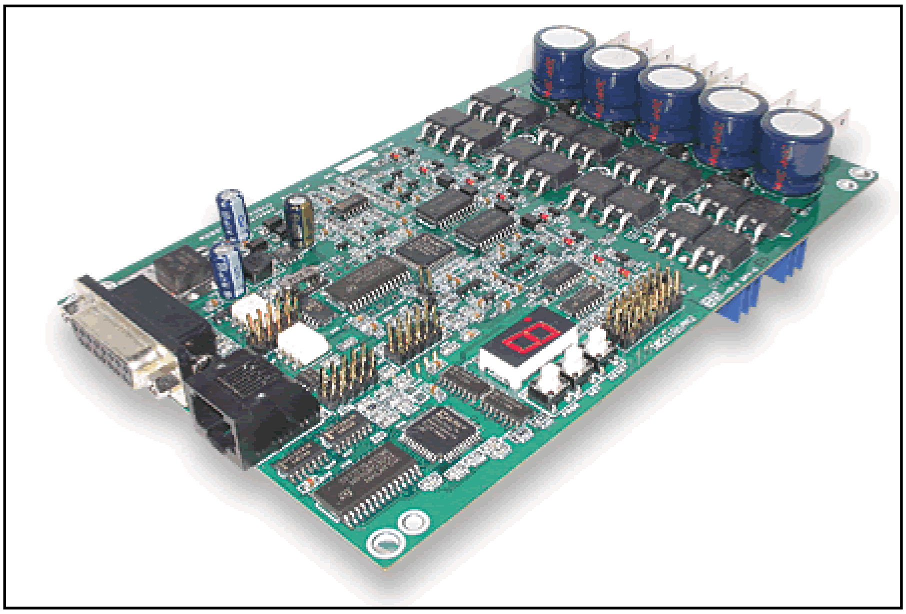


Figure 3: RoboteQ's AX3500 motor controller (RoboteQ 2005)

2.3. Sensors

2.3.1. Ultrasonic

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 traveling 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.

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 were fired and read.

Including sending the sensor information to the computer, the Basic Atom fires each sonar about 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 were plant stalks. (bron: Bruggen, R. van, et all, 2005, page 4)

2.3.2. Compass

For turning at the headland, a compass is used to determine the angle of the robot at the end of the row. The robot turns in a preset speed and direction until the angle of the robot is 180 degrees different from the angle at the end of the row.

2.3.3. Camera

For driving to the corner flag and counting the yellow balls, a camera is used. A Logitech webcam (1.3 mega pixel) is connected to the USB-port of the computer on the FieldMeister.

Figure 4: Ultrasonic sensors

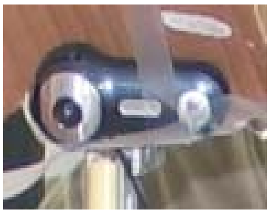


Figure 5: Camera

Figure 6: Compass

2.4. Control Software

2.4.1. Microcontroller

For control of the FieldMeister National Instruments Lab View 7.1 is used.

With the distances of all sensors given by the Basic Atom, the direction of the robot in relation to the rows and the offset of the robot from the middle of the rows is calculated. From this angle and offset, the required direction of the steering wheel is calculated linearly.

2.4.2. Motor controller

This direction is sent to the RoboteQ controller. Also the driving speed is calculated. With a higher steering angle (far from the ‘zero’ point), the driving speed is reduced to reduce the error of a possible overshoot.

2.4.3. Software navigating through the row

The software to define a straight line to the corner flag is also prepared in LabView.

For driving curves the angle of the steering wheel, the driving speed and the angle of the curve can be installed. After steering the curve the robot drives straight for a while to be sure it is in the row and then goes back to navigation mode.

2.4.4. Software navigating to corner flag

Lab View is used to find the biggest red spot in the image, and calculates the horizontal angle of the center of the red spot compared to the center of the image. With this angle, the steering direction is calculated linearly.

2.4.5. Software counting yellow balls

For the detection of the yellow balls, also the webcam is used. Lab View grabs the image from the webcam and counts the round yellow spots in the image. By tracking back to the number of yellow spots in the previous image, double counting can be prevented.

2.5. Steering Mechanisms

To measure the steering angle of the front wheel a variable resistance is placed on the top of the vertical axle. The resistance can rotate about 270 degrees and varies from 0 till 10KΩ. Our steering angle from left to right is 170 degrees, so the resistance is very suitable.

After checking what the value was for the resistance when the front wheel goes straight on, the PID control system on the RoboteQ controller was calibrated for steering, by first calculating the values for P I and D. After fine-tuning by trial and error of the calculated values a system emerged that is able to steer very fast to the 'zero' point. However, when the front wheel reaches almost his zero point the steering speed will be decreased by the PID control system, so steering goes very smoothly.

3. Results and discussion

FieldMeister acted reasonably well during the Field Robot Event. However, there were some last minute problems a few days before the contest. On an indoor test track, the navigation through the rows as well as the turning on the headlands were nearly without errors. Outdoors however, the navigation through the rows was still good, but the headland turns were very unpredictable. For the headland turns, a preset speed, direction and time were used, which worked good indoor, but very bad outdoor. Therefore we decided to put the compass of last year's RowBo on FieldMeister, to make sure the turns would be 180 degrees. The programming of the software and wiring the hardware were finished in time, but due to the steering engine on top of the front axle, the compass was not nearly as accurate as in the previous year. This was improved much by raising the compass much higher, but it was not sufficient to make all headland turns work properly.

During the contest there were also some problems with the detection of the end of the row, which should start the headland turn mode. This was due to some grass on the headlands. The ultrasonic sensors, which are very sensitive, determined some of this grass also as a row, which prevented the robot from going to turning mode. Also there where some holes and bumps in the headlands, which made the robot tilt a bit. This intensified the problem of detection of the end of the rows.

Also there were some problems with the parallel working cycles of the navigation through the rows and the detection of the yellow balls. This caused the detection cycle to fail sometimes, which, unfortunately, also happened during the contest.

In the contest, driving to the corner flag went well, with a good score from the jury. With the navigation and yellow ball detection the detection failed to work, and there were some problems with the headland detection, so the jury scores were a bit lower. In the speed race we were second in the first heat, which put us in the run for 5th place, but in the second run the computer jammed, which resulted in the 7th place.

Overall FieldMeister became 8th out of 13 competing robots.

Conclusions

FieldMeister can navigate quite well through the row. It showed that an autonomous vehicle can navigate through maize rows without using visual data. However, the driving speed during the Field Robot Event was low, compared with other vehicles. By increasing the number of corrections the driving speed should probably be increased.

Further, the compass should be made properly, so that is not influenced by the engine or other electromagnetic or steel disturbances.

Also the software problems with the webcam should be solved, to make the program more reliable.

Literature list

- Bruggen, R. van, Burgwal, E. van de, Dongen, B. van, Oudman, R, Rijpkema, W. 2005, Rowbo – an autonomous field vehicle

Gaia – Autonomous Field Robot System

Jörn Knaup, Robert Meyer, Michael Meinecke,
Markus Robert, Jan Schattenberg, Jörg Schlott

University of Braunschweig – Institute of Technology
Institute of Agricultural Machinery and Fluid Power, FREDT
Langer Kamp 19a, 38106 Braunschweig
http://www.fredt.de, email: info@fredt.de

Abstract

FREDT - This is the Field Robot Event Design Team of the Technical University of Braunschweig. Our working group joined in December 2005 in order to develop and build an autonomously navigating vehicle. We derive from different fields of mechanical engineering like automotive engineering, general mechanical engineering and mechatronics. The basic idea was to develop a simple and fast realizable robot, to take part at the Field Robot Event even in 2006. So we developed our current vehicle named GAIA based on the Tamiya RC-Monstertruck Juggernaut II. GAIA is a Greek goddess personifying the Earth. Her offspring was the chaos (Well, some of these parallels to our robot can't be denied).

Keywords

Robot, precision farming, autonomous, FREDT

1. Introduction

Our working group joined in December 2005 in order to develop and build an autonomously navigating vehicle. We derive from different fields of mechanical engineering like automotive engineering, general mechanical engineering and mechatronics. The basic idea was to develop a simple and fast realizable robot, to take part at the Field Robot Event even in 2006.

2. Sensors

Main functions of robot GAIA are presented in Figure 1. Sensors and other parts are listed in Table 1.

To navigate within the corn rows GAIA uses three ultrasonic sensors. Two of them are located at both sides of the robot to measures the distance to the corn rows (see Figure 2). The third one is mounted in the center in order to stop the robot in case of an upcoming collision.

The ultrasonic data is transmitted by an I²C-Bus to a microcontroller (see Figure 4). With the help of a digital control the vehicle is kept preferably in the center of the row.

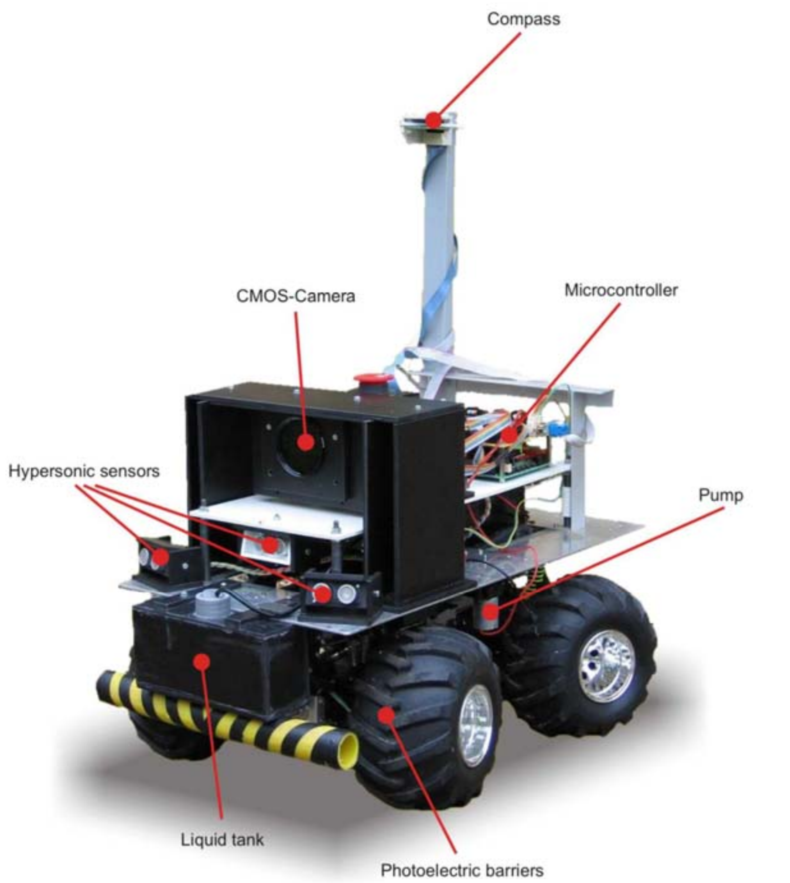


Figure 1: Hardware

Table 1: Hardware parts

Chassis: Kit *Tamiya Juggernaut II*	**Sensors:** 2x ultrasonic sensors SRF05 1x ultrasonic sensor SRF08 2x speed sensors CNY 70 compass-modul Devantech CMPS03 CMUcam2 CMOS-Camera
Drive: 2x 12V-Engine *Truck Puller II* 6100 U/min, 60W, 361Nmm, 170g	
Servo: High-Torque 0,75Nm-Servo	**Power supply:** 2x batteries: 12V 7200mAh lead-acid battery 7,2V 3300mAh NiMH
Steering: 4 wheel steering, centered servo	**Dimensions:** WxLxH: 360mm x 620mm x 620mm
Logic: 2x ATmega32-Mikrocontroller	**Weight:** 12,3kg
Control interface: Palm M105	

A camera which is located at the back of the robot films permanently the edges of the rows during navigation. Whenever a golf ball comes into the camera´s field of vision it is detected by the graphical analysis of its colour.

Because of economic reasons only one camera for the right AND the left side has been installed. The field of vision is directed to both sides by a mirror system (see Figure 3).

For the discipline "The LINE" the camera is installed at the front of GAIA to focus on the flag (see Figure 1). The control tries to detect the flag by its colour and finally drives straight towards the flag. A pumping system sprays a coloured liquid through a broad nozzle on the lawn. The centrically installed ultrasonic sensor acts as the "end switch" to prevent the robot touching the flag.

For the discipline “Hole Detection” we used another optical system to focus the cameras field of vision on the ground in front of the vehicle. A hole is indicated by an acoustic signal. For this task we used additionally a compass and two revolution sensors which are placed at the front axle to drive off the given test field after a fixed plan.

In all disciplines a Palm Pocket PC is used as input and output device (Human-Machine-Interface), to control the vehicle by hand between the different tasks. Furthermore information about speed, driven distance or the number of counted balls (“Dandelion Detection") can be called up. It is also possible to adjust individual control parameters during the testing phases.

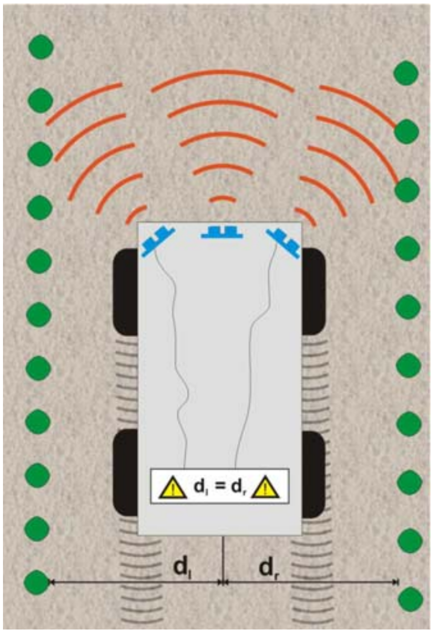


Figure 2: Row navigation

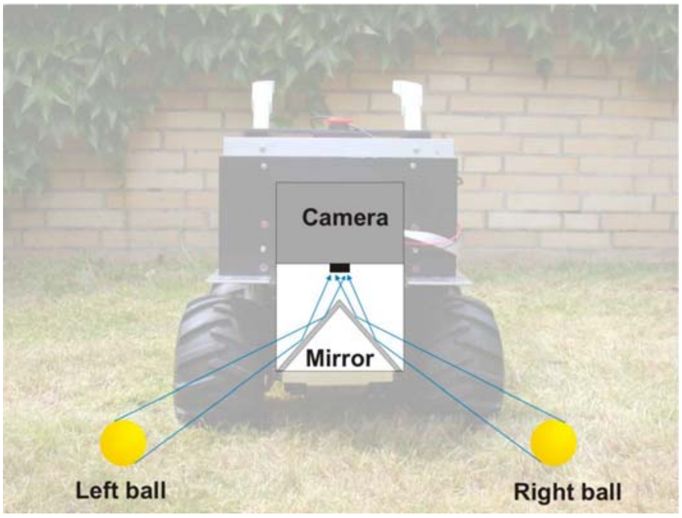


Figure 3: Optical system

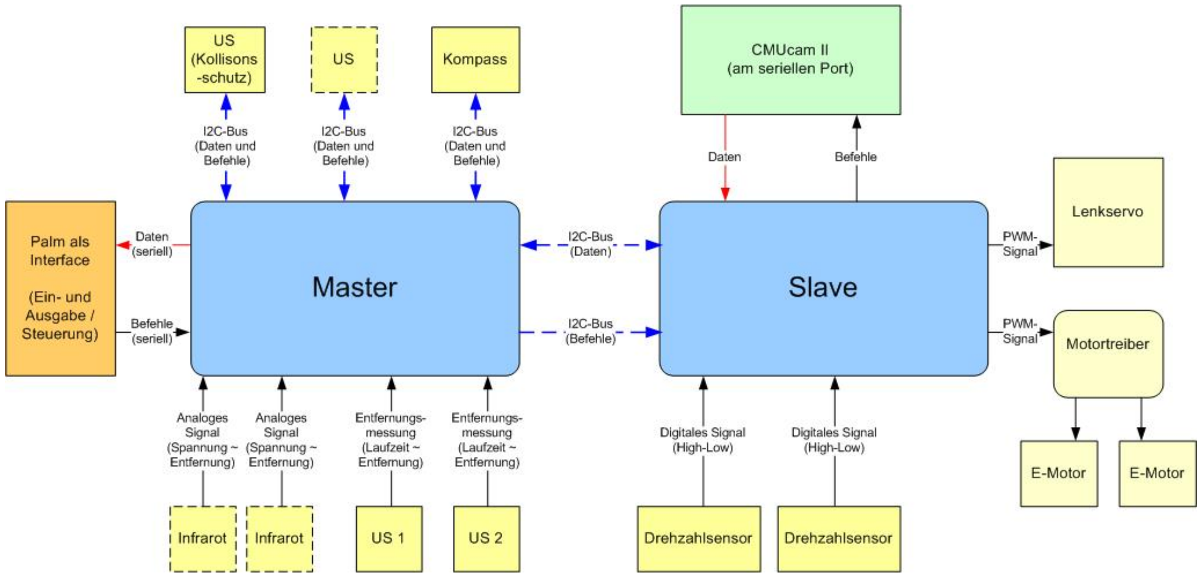


Figure 4: Electronic concept

2. Results and Discussion

We finally ended up on rank 6 and got a special price for our "Hole Detection" concept. Considering the short development time for the robot we've been really lucky and are happy with these results.

During the test phases the weakness of the mechanical hardware has been one of the main problems. Too much play in the steering elements and in the drive strand made it extreme difficult to control the robot. The bad material quality led again and again to problems.

For the coming period a complete self-construction of the chassis is planned. In addition some studies at the Institute of Agricultural Machinery and Fluid Power are already running. The next step will be the optimizing of the graphical analysis.

Acknowledgements

Finally FREDT wants to thank the Institute of Agricultural Machinery and Fluid Power for the excellent assistance and, of course, the organizing team of the Field Robot Event for making all this possible. We spent great days in Hohenheim and we hope the public interest for this event will continue to grow in the next years.

Maizerati – Conception & realisation of a multi-purpose autonomous field robot

R. Klose, J. Klever, H. Markus, O. Meyer, W. Niehaus, A. Rahenbrock, V. Schwamm, M. Urra-Saco, K.-U. Wegner

Faculty of Engineering and Computer Science
University of Applied Sciences Osnabrück
Albrechtstr. 30, 49076 Osnabrück, Germany
Email: a.ruckelshausen@fhos.de

Abstract

The Field Robot Event 2006 came up with some tasks known from the last year but also with new challenges and modifications. They were the initial point for the development of the new autonomous field robot Maizerati. Every part from the chassis to the controller was screened for compliance with the defined requirements.

Over a period of 4 months the team of 9 students build up this mechatronic low-cost system parallel to their studies with the assistance of 3 further students and their lecturers (see chapter 0). They took it as an important experience in project management and came in contact with new technologies. The making of decisions was guided by the results of the former competitions so futile options could be discarded at an early state.

The required information about the environment is acquired by a total of up to 19 sensors and is processed by two micro controllers. Sophisticated algorithms interpret the incoming signals to compute steering and speed variations.

Keywords

Field robot, student competition, optoelectronic sensors, CMOS camera, sensor fusion, gyroscope, WLAN, compass, dandelion count, hole detection, real time operating system, CAN

1. Introduction

This year the annual competition of autonomous navigating robots called „Field Robot Event" was organised for the first time by the University of Hohenheim. The Field Robot Event is a happening were interdisciplinary teams mainly from universities all over the world compete by letting a self-build robot accomplish a variety of different tasks on an outdoor field. A group of nine Mechatronic Systems Engineering students from the University of Applied Sciences in Osnabrück, developed in a four month period a small four-wheeled robot called "MAIZERATI" to participate in this competition.

The five main tasks to be performed by the robots were as follows:

1. Drawing a white line

The robot is placed on an open field from where he had to navigate towards a target (= corner-flag) and draw a (preferably straight) white line. Criteria: straightness and visibility of the white line; distance between robot and flag after stop

2. Dandelion detection

The robots had to navigate through curved plant rows. At the end of each row they were expected to turn and re-enter the rows. This ought to be repeated until the time target set by the committee or the end of the field was reached. While navigating through the rows the robot had to count dandelions (yellow golf balls). Criteria: difference from accurate count, mileage, style

3. Speed Race

Three robots placed next to each other on separate plant rows pulled out at the same time. The quickest one of each race advanced to the next round. The robot that crossed the finishing line in the final round first was declared as the winner of this task. Leaving the row led to disqualification.

4. Finding holes

The committee set up a lawn with holes dug into it. The boundary was resembled by a white line – comparable to those on a soccer field. The objective was to let the robot detect these holes autonomously. Criteria: correct hole localisation, time, technology, style

5. Freestyle session: Watering flowers

The idea the team made up for this competition was to let the robot water the flowers which stood in a straight line with irregular gaps between them. The robot had to detect the flowers and water had to be pumped from the trailer to be sprayed onto the flowers. Criteria: creativity of the idea, complexity, realisation

The money spent for the robot was also considered in the overall standings. The cheaper the system was the more points were assigned.

2. Concept

In order to develop the new field robot Maizerati with improved characteristics and a better performance the last year's competitors optoMAIZER [optoMAIZER 2005] and Eye-Maize [Eye Maize 2004] were analysed.

The decision to alter from a track based concept to a wheel based chassis was driven by the purpose to reach a higher maximum speed. As good experiences were made with the established main microcontroller and the steering algorithms an evolution rather than a revolution were applied to this division. Now it is supported by a second microcontroller which is responsible for the data collection from the increased number of sensors. The unsteady conditions on an open field and the keyword "sensor fusion" led to a redundant system of sensors which are taking advantage of different physical effects.

To control the robot and to visualize the system's status a touch screen is integrated which allows choosing among different operation modes during the contest. For the surveillance and the recording of sensor data in the test stage a WLAN-Module is connected to the system. It also enables to acquire images from the camera and to control the robot via an external PC.

The high power consumption of the electric drives during acceleration raised the need of separated energy sources. One accumulator battery only provides the energy for the motors. Another one supplies the control electronics with current.

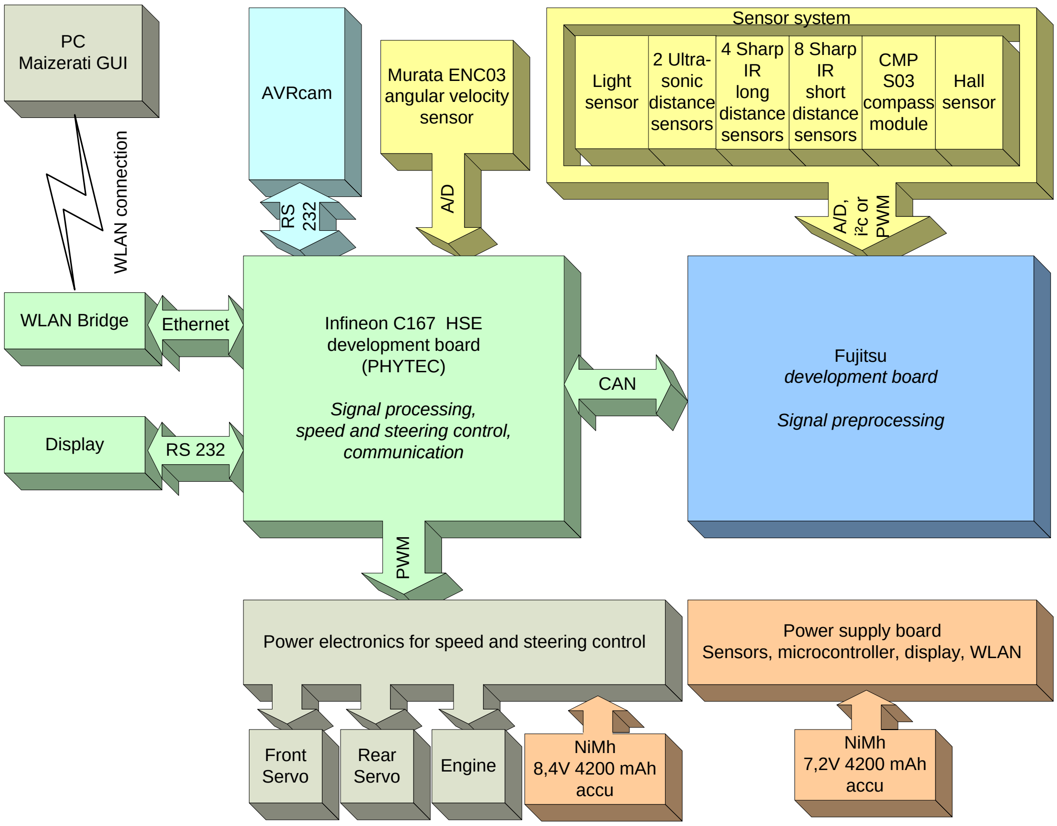


Figure 2. Block diagram of the electric components

3. Hardware

3.1. Mechanics

Virtual Product Development

During the conception phase different models were considered as the basis for the new field robot. Own concepts were developed in addition to commercial vehicles. The main arguments for the decision were speed, agility and capacity as a lot of hardware has to be carried along. Needless to say that financial aspects were also regarded. The extensive examination with modern means of computer aided engineering led to the decision for the Tamiya model TXT-1.

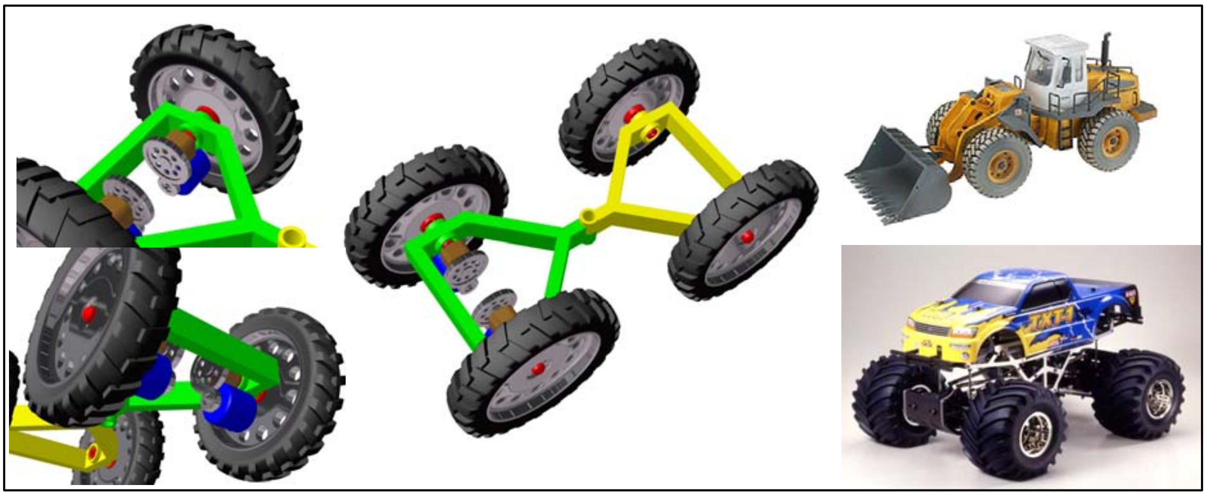


Figure 3: Own CAD sketch (left) in comparison with commercial vehicles (right; source: www.graupner.de, www.tamiya.com)

Based on the CAD model of the TXT-1 the body was designed. At this stage it was important to regard the intended structure of the control units and the concluding space consumption. Furthermore the mounts for the sensors especially the camera had to be realised.

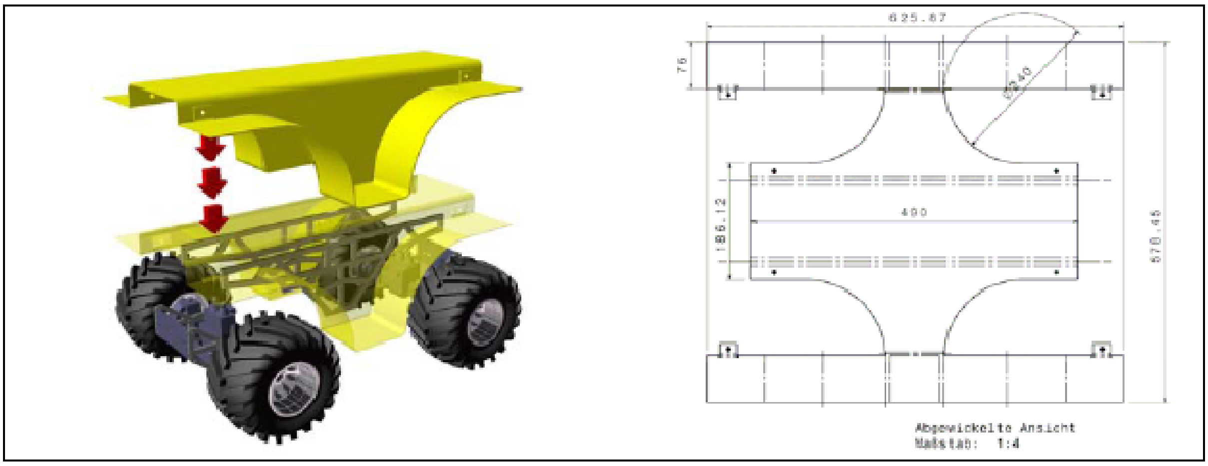


Figure 4: Virtual Tamiya TXT-1 model with underbody. Blank forming part.

Base Unit

The basis of our robot consists of a modified Tamiya TXT-1 RC-Monster truck model and a self designed case to protect the hardware against rough conditions.

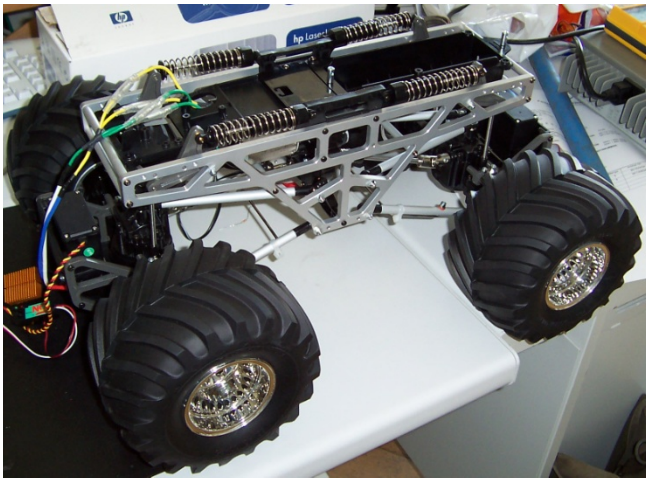


Figure 5: Tamiya TXT-1 base model

Figure 6: Case design

The 4WD base model (fig. 3) with an aluminium ladder type frame is equipped with two RS-540 electric engines which are connected to the wheels via a four-step gearbox. The all-wheel steering is driven by two servo motors. This combination allows controlling the front and rear axle individually and decreases the curve radius. The suspension is a solid axle multi link with plastic coil-over oil filled dampers.

The model also includes a no limit power electronic drive controller which is connected to the two engines energised with a seven cell battery stack. This power module can be linked to a microcontroller via PWM channels for speed control. The two servo motors for the steering are also controlled by a microcontroller via PWM and have their own independent power supply with a five cell battery stack.

For the case, we decided to use an approved combination of aluminium and Plexiglas (fig. 4). The Plexiglas provides the opportunity to interested people or spectators to have a look inside our robot. The aluminium case design also gives a high degree of security for the electronic components and handling. The whole self - made construction of the Maizerati was designed with the CAD software CATIA V5.

3.2. Sensors

The following table shows all major tasks of the „Field Robot Event 2006" and the Maizerati´s sensors dedicated to them.

Robot tasks	AVRcam	IR Sensor	Flex Sensor	Ultra Sonic Sensor	Gyroscope	Hall Sensor	Photodiodes	Light sensor
Navigation	X	X	X	X		X		X
Counting yellow balls	X							X
Drawing white line	X							X
Speed race	X	X	X					X
Finding holes					X	X	X	
Watering flowers		X	X					
Turn on end of row		X		X	X			

Figure 7: Tabel of tasks and dedicated sensors

AVRcam

Figure 8: AVRcam (source: jrobot.net)

The AVRcam is a small, real-time image processing engine capable of tracking multiple objects with different colours. Its hardware is based on the Atmel AVR mega8 microcontroller and the Omnivision OV6620 CMOS image sensor.

The AVRcam is capable of tracking 8 different user-defined colour blobs at 30 frames per second. This makes the camera suitable for a variety of applications needed for the contest.

The AVRcam is the most important sensor system because of its ability to distinguish different colours. This feature is used for navigation between plant rows, counting yellow balls and tracking the red corner flag.

Compared to other low-cost systems the AVRcam is small, quick and easy to setup. To reduce the influence of infrared light which hampers proper colour detection the camera is refitted with a filter lens. The communication between the camera and the microcontroller is done via the AVRcam's RS232 interface.

Sharp IR Sensor

Figure 9: Sharp IR Sensor (source: acroname.com)

The IR sensors are used in combination with the AVRcam for navigation. The IR sensors are fairly accurate at intermediate range and fast enough for our purposes (50 ms conversion time). They are also relatively robust and affordable compared to the next IR sensor class. The sensor itself is capsulated by self-made aluminum cases to minimize the negative effects of infrared radiation from direct and reflected sunlight.

The long range IR Sensors are in comparison to the short range sensors prioritised because this lets the robot react more smoothly to boundaries. It was necessary to install 10 IR Sensor to ensure that gaps between plants and disturbances from other external source were not misinterpreted by the robot. [optoMAIZER 2005]

Flex sensors

The two flex sensor can be considered as safety systems. If one of them is bent over limit due to mechanical contact, the robot tries to steer into the opposite direction to prevent itself from colliding with an object. The response time is very short and as it is a mechanical system its not affected by other environmental influences. [optoMAIZER 2005]

Ultra sonic sensors

The Ultra sonic sensors main purpose is to detect obstacles. Compared to the IR snsors the ultra sonic sensor scope is much greater and the waves are not influenced by light. The disadvantages however are the lack of accuracy and speed (the sonic waves spread out on distance, sound waves travel at 330 m/s). The I²C protocol can be used to communicate with the sensor. Figure 4 shows the principle of an ultra sonic sensor.

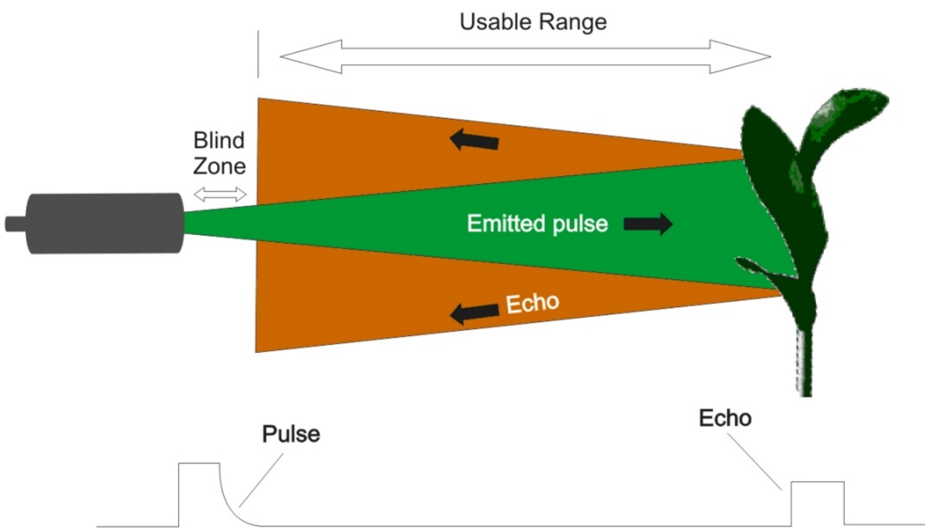


Figure 10: Ultra sonic sensor for long range detection

The emitted pulse is reflected when it hits an object. The sensor uses the time between the pulse and its echo to calculate the distance between itself and an object.

The Maizerati uses the two ultra sonic sensors to decide whether a plant row has ended and a turn must be initiated.

Gyroscope

The gyroscope is merely used for measuring the rotation angle of the robot during turning action. The sensor is needed to ensure the robot is in line with the next row. This feature is also important for hole detection where the ground must be scanned in parallel lines to guarantee that the whole field is covered by the robot. For further information refer to the specification of the gyroscope used for the Optomaizer [optoMAIZER 2005].

Hall sensor

The Hall plate is pervaded by the magnetic field of the magnet. Electrons driven through the sensor are deflected to one side of the sensor plate. The switch cuts off the plate from the source so that the comparator can compare and amplify the voltage level difference.

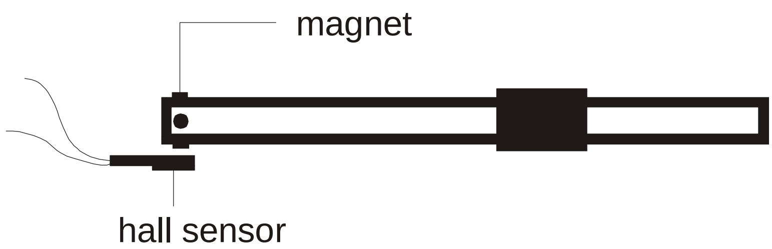


Figure 11: Hall sensor measuring crankshaft rotation

The Hall sensor is used to record the way the robot has already covered in steps of app. 5 cm. The concept behind this idea is the attachment of a neodymium magnet to the crankshaft of the Maizerati. The inertia of the magnet is small enough so that the rotation of the shaft is not disturbed. This system works well on even ground with the absence of mechanical slip. Because of the Maizerati´s differential gear the wheels can rotate at different speeds. Therefore measuring the wheel rotation would be futile. The sensor though is wheel rotation speed independent.

During the hole detection event there were no control points for orientation. The only information useful for navigation were the distance already covered and the angle during turning.

Photo sensor array

The Photo sensor array consists of 22 photo diodes combined as two sensor systems on each circuit board. To measure a hole in the ground a contact less system seemed to be a good approach.

The simplified design (Fig. 12) shows two photo diodes places adjacent to each other. The yellow diode functions as an emitter of infrared light. The design of the sensor bar reduces the interferences of environmental radiation as the point of measurement is shielded. The emitted radiation is received and transformed into current by the green. The current gets amplified by the circuit and the proportional voltage is measured over a resistor.

When the robot is placed on normal grass (Fig. 12: left picture) a large amount of light emitted by the yellow diode reflected. If a hole in the ground (0: right picture) appears under one of these sensors, the amount of reflected light hitting the receiver decreases because of the distance (light gets dispersed) and absorption (dark soil). The preset threshold gets transgressed and the system registers a hole (Fig. 13).

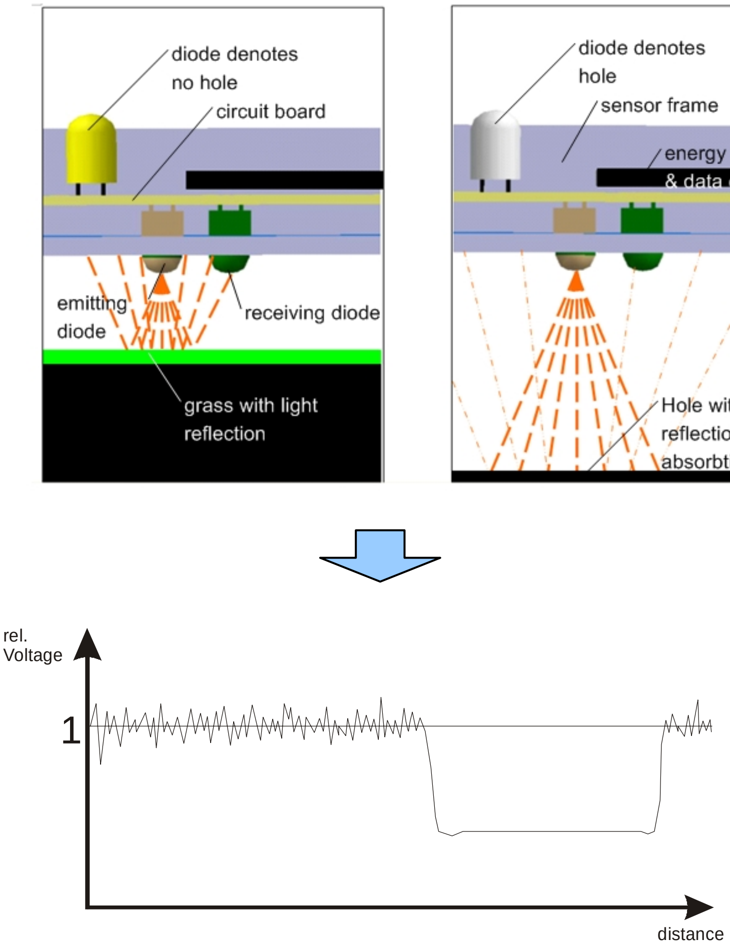


Figure 12: Photo sensor array schematic

The camera could not be used for hole detection because of the poor detail of the pictures made. The low contrast between the grass and a hole in combination with the large area to scan raised the need of a new detection method. The decision for such a long sensor bar was motivated by the turn radius of the vehicle. A complicated turning maneuver is not necessary any more. The distance between the sensors was adapted to the hole size given by the competition supervisors.

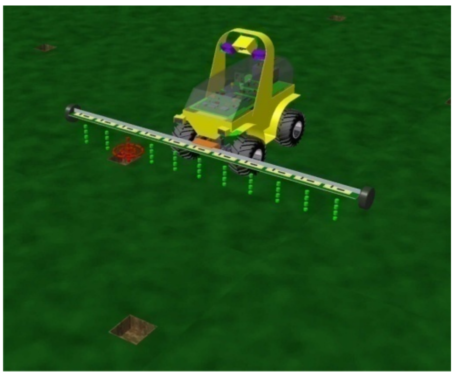


Figure 13: Photo sensor array for hole detection

Light sensor

The light sensor embedded in the Maizerati´s bonnet is used for measuring the intensity of the surrounding illumination. This information is needed to recalibrate the CMOS camera due to changing repartition of wavelength shares. The measured analog value from the photodiode must be digitalized using an A/D converter. Further processing includes the comparison of the measured value with the data sets for new calibration settings which are stored in a lookup table. When the right data set is found the associated parameters are sent to the camera to initiate the readjusting.

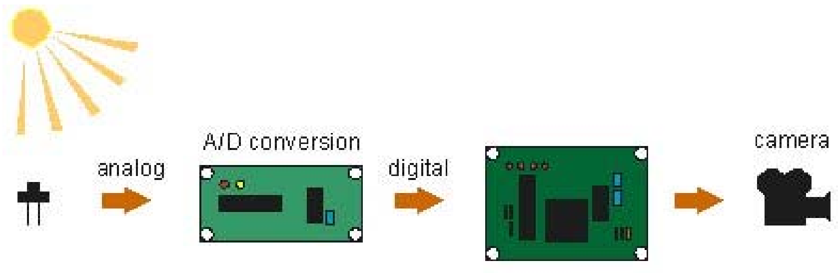


Figure 14: Light sensor schematic

Due to the late completion of the light sensor and problems in combining it with new camera it was not possible to get the two systems working together. This could be a task to be tackled in the next robot generation.

3.3. User interface

Touch screen

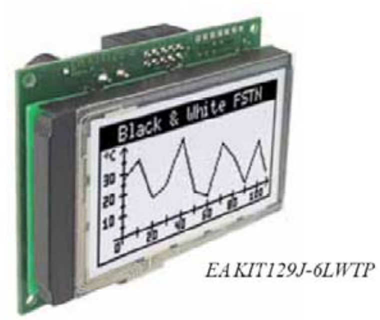


Figure 15: Electronic Assemblv touch display (souce: www.electronic-assembly.de)

The touch display is used to show the actual status of the robot system and to change between the different operation modes. It is supplied by Electronic Assembly and can be programmed with included software in a special macro language. Via RS232 the display is either connected to the PC to transfer the developed code or to the microcontroller board to send and receive data. The display area is divided into a 3x5 matrix of fields sensitive on pressure. The resulting 15 keys can be combined to set up a large button. On a user input, an ASCII sign which represents the pressed button is transmitted. In return, a new display page can be chosen by sending the macro’s number. [optoMAIZER]

WLAN-Bridge

The integrated WLAN-Bridge emerged as very helpful equipment during the test stage. In combination with the graphical user interface it assumes the tasks of the touch display and allows us to observe and record the sensor data including the acquisition of camera pictures. Furthermore it is used to quickly modify parameters in the field.

The Maizerati is equipped with a D-Link DWL-G810 WLAN Bridge which is similar to the suspended model used in the optoMAIZER. It holds the desired features like an ad-hoc mode but consumes little space without its housing. Via an Ethernet cable it is connected to the Phytec controller board. [optoMAIZER]

3.4. Micro controller Systems

As mentioned before two microcontroller boards are used to implement all functionalities. Beside the advantage of existing executable algorithms for the Infineon C167CS the choice for this microcontroller was its comparative low price. However the C167CS is not capable to handle all functionalities with optimal performance. Because of that the Infineon is supported by a second microcontroller which is responsible for data collection and pre processing from the increased number of sensors. Choice felt on the Fujitsu MB90F345CA on a Glyn evaluation board. The combination of this two low-cost microcontroller boards accomplishes placed requirements concerning real time operation and high working performance. The biggest advantage of the combination compared to a high quality/cost microcontroller is the possibility of parallel operation. The use of the CAN-Bus on high transfer rate makes data exchange very effective, simultaneous providing priority level assignment of relevant data.

Phytec development board with Infineon C167CS microcontroller

The Maizerati features the Phytec development board “phyCore-167 HSE” equipped with an Infineon C167CS microcontroller. This processes the data from the AVR camera, the display and the gyroscope. Furthermore it processes the values from the other sensors that are provided by the Glyn Evaluation Board via CAN-bus. Based on these figures, an algorithm computes a decision regarding course and speed variation.

Another important task which the controller has to handle is the communication with the Maizerati Testing GUI by WLAN.

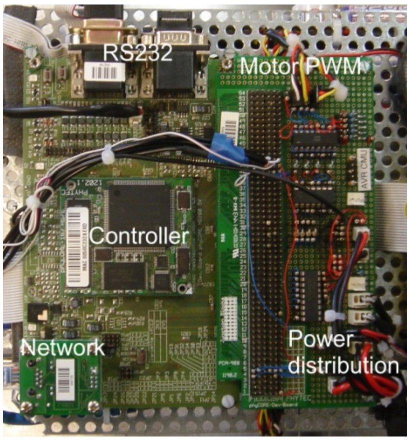


Figure 16: Phytec controller board with additional circuits

Glyn Evaluation Board MB90F340/860

The Glyn Evaluation Board processes the signals from the flex sensors, the hall-sensors, the compass-module, the ultra sonic distance sensors, the hole detection sensors, the light-intensity sensor and the IR distance sensors. All handled signals are sent to the C167-board via CAN-Bus with different priorities.

Figure 17: EVB MB90F340/860 (source: www.glyn.de)

Some main features of the Fujitsu MB90F345CA microcontroller are the following:

1xCAN Interface
UART
24MHz
512K Flash
I2C (400Kbit/s)
A/D Converter (24 input channels with 10 or 8-bit resolution)

4. Strategy

Usage of the short distance IR sensors

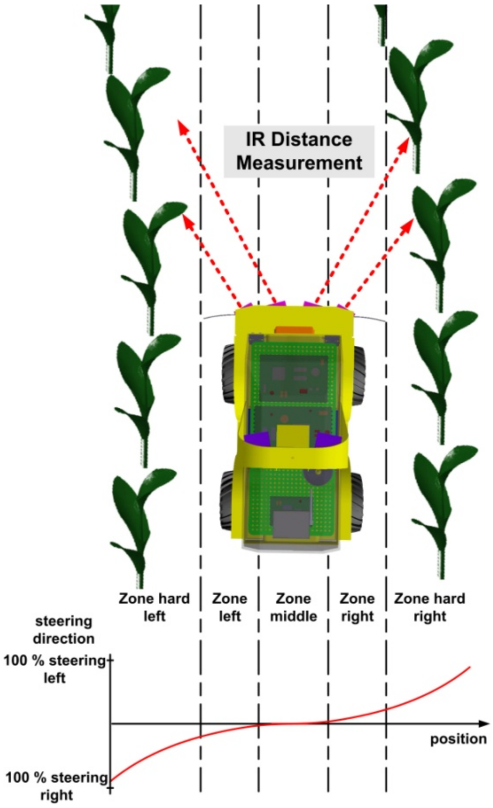


Figure 18: Usage of short distance IR sensors

The distance to the rows is separated into five zones in which the robot can be located. The exact position within the row is determined by the use of the shown IR distance sensors. With this data the robot is assigned to one of these zones. Depending on the determined location zone a decision is made concerning the steering direction. In addition the speed to set up next is calculated.

As the figure shows there are two IR sensors used for each side. The difference in position and angle is recalculated concerning the known structural differences. This redundant sensor secures that the robot is able to measure the distance to a plant in most of running time.

The implemented algorithms offer the possibility to determine the position of the robot if there are only plants available on one side of the row.

Usage of the long distance IR and ultra sonic sensors

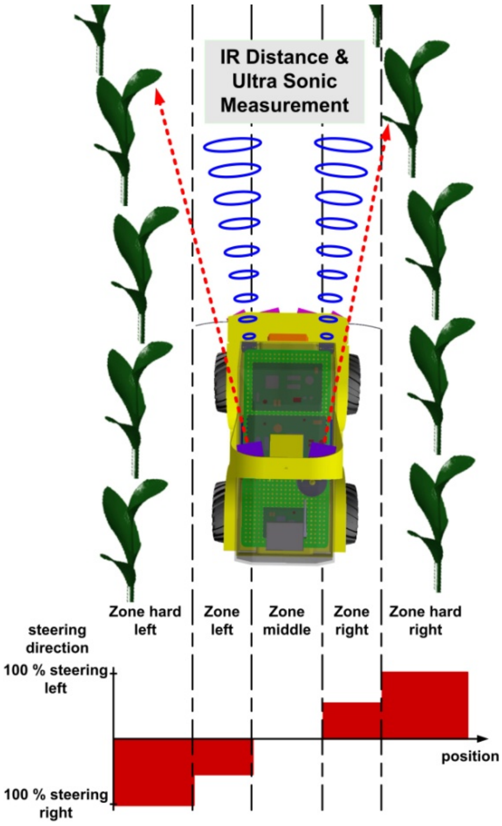


Figure 19: Usage of long distance IR sensors

The long distance IR and ultra sonic sensors were integrated for security reasons.

These sensors are able to measure long distances and offer the possibility to look a long way forward along the maize rows.

So the robot recognizes curves and sections without plant at one side very early and is able to react accordingly.

Another important aspect of using these sensors is to avoid collisions with plants and obstacles.

Also here the distances are divided in five zones, but the steering direction for a complete zone is constant.

5. Software

The software for the controller was developed with different IDEs, depending on the different controller manufacturers. The code for Fujitsu MB90F345CA was programmed with Fujitsu Softune Workbench FME. Keil µVison V3.05 is used for development code for C167CS.

Glyn B90F340 with Fujitsu microcontroller MB90F345CA

This Glyn Evaluation board has to pre process the Sensor data of the most used Sensors.

The low-cost IR Distance Sensors Sharp GPY0D02YK and GP2D12 provide a non-linear voltage. For each of these Sensors a comparison measurement with a high quality Sensor in full measurement range was made. The acquired calibration data is placed in lookup tables and used to enhance the 10-Bit digitised Sensor values.

The “sensor fusion” and the decision making for robot/actor control are done with the C167. According to this the sensor data has to be transmitted to the Phytec board. The used bus system is the CAN-bus which provides the possibility to prior messages. The determined values of the sensors are assigned to the CAN-messages as Fig. 20 shows.

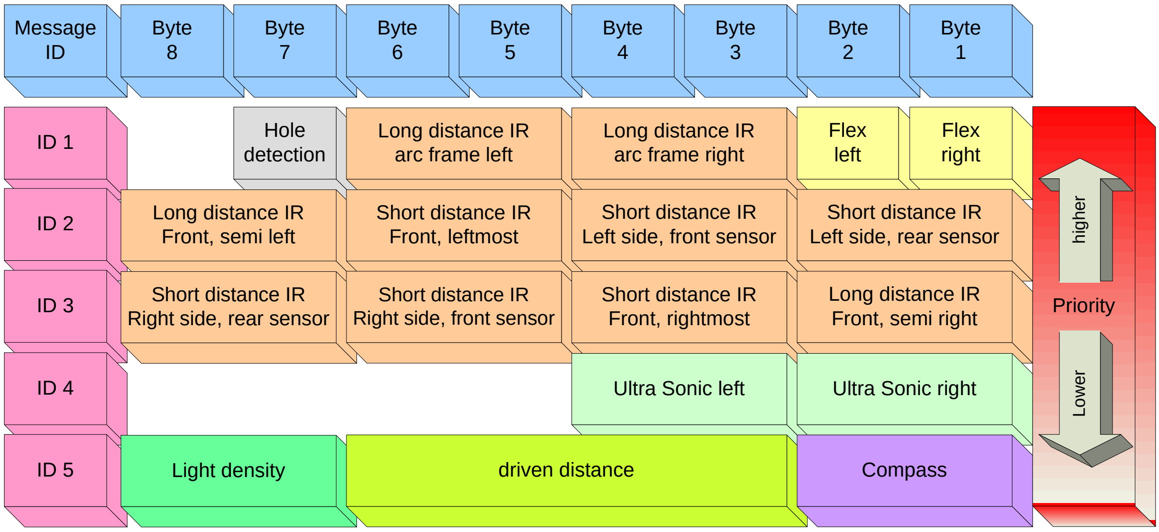


Figure 20: CAN-Messages for Sensordata transmitting

After the initialisation the program sequence of the Fujitsu is completely interrupt controlled. This ensures real-time operation, which is amongst others necessary for time measurement for PWM analysis.

The reload timer is used to keep the time interval between two ultra sonic measures, which is necessary for making a complete distance measure. External_IRQ_Handler 1 (Fig. 21) determines with help of a 16bit I/O timer the durance of high level of compass and light-density-sensor PWM, which represents the sensor values. Timer overflows are recognized in interrupt routine IO_Timer_16Bit and are considered as a pre factor in time calculating. In addition the Hall-sensor signal is used to calculate cumulative distance driven since the last reset as well as the comparator signal of left flex sensor is sent to C167 when touch event occurs. Similar External_IRQ_Handler 2 proceeds with the right flex sensor.

The A/D conversion of the IR or hole detection sensors is done in multiplexed mode in AD_Conversion. While navigation mode all CAN messages are sent to C167 when conversion of all IR sensors is done. The determined values from sensors are enhanced with defined lookup tables.

In hole detection mode the CAN message 1 is sent after a hole was detected only.

In interrupt routine CAN_RX received data from C167 is used to update used variables or switch working modes.

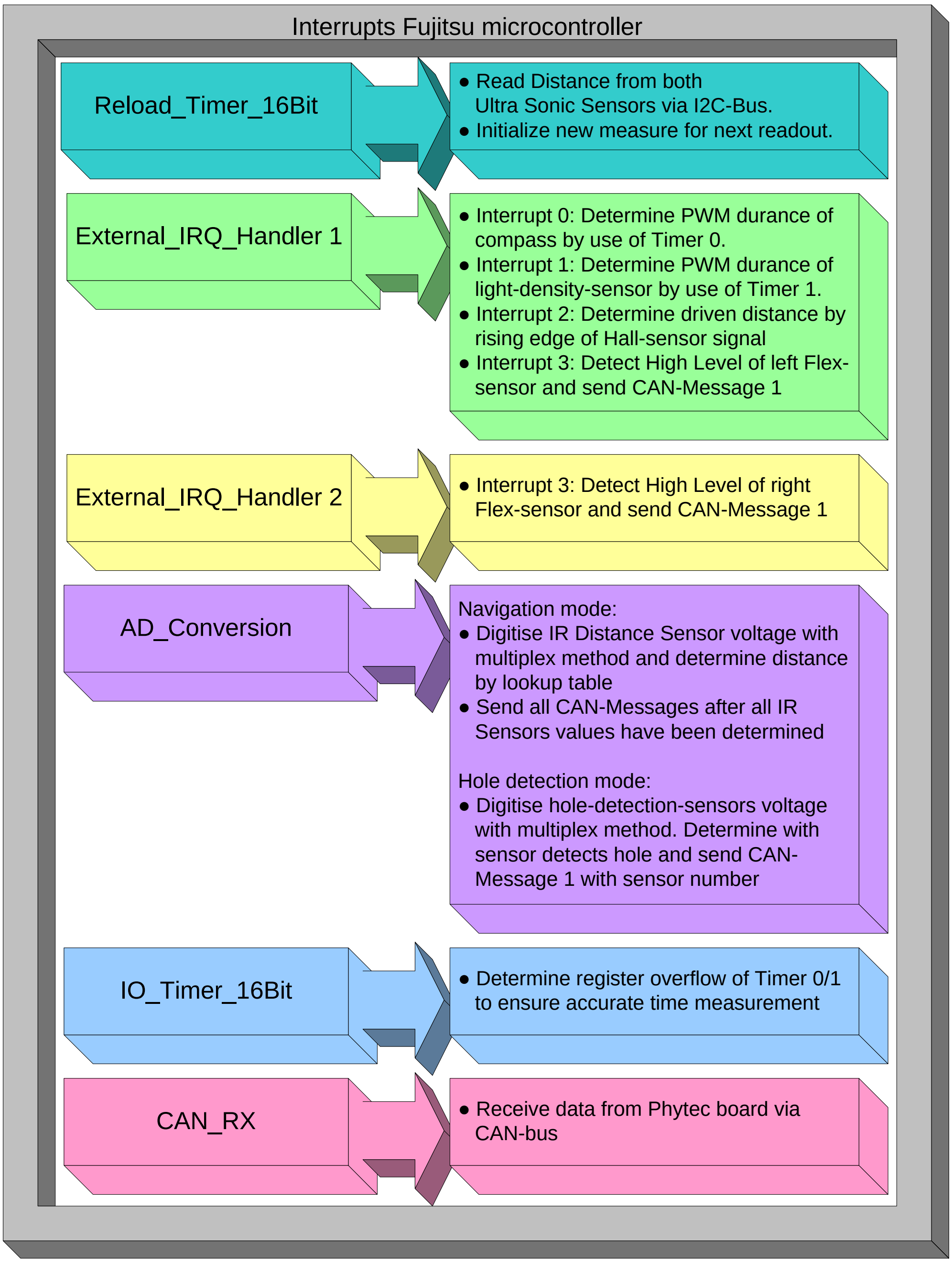


Figure 21: Interrupts implemented in Fujitsu Code

Hole detection

In hole detection mode the most A/D converters of the Fujitsu are connected to the hole detection sensors. Before beginning detection a calibration is done to calculate threshold level for the signal strength. This reduces interfering effects from environment (sun, weather, colour of grass). While detection the commonness in series of hole signal level is counted and only if the hole is big enough it is denoted to C167CS.

The defined commonness threshold depends on robot velocity and has to be modified when velocity is changed.

Phytec phyCore-167 HSE with C167CS

This controller has to handle a lot of different time-critical jobs. According to this a real time Operation System had to be implemented, which secures quasi simultaneous task handling. In contrast an interrupt controlled job handling would not provide optimal operation results. The used OS is the RTXtiny real time OS made by Keil which was developed especially for microcontrollers. It uses round-robin switching and cooperative multitasking, which guarantees mentioned requirements.

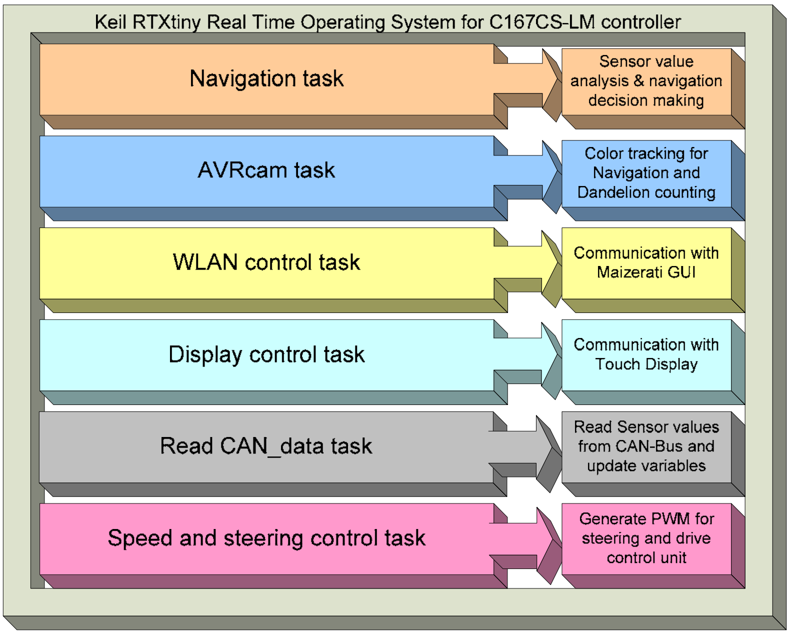


Figure 22: Real time OS tasks

The AVRcam task handles the communication with the AVRcam via a serial link as well as the calculation of the direction and ball counting, using the AVRcam algorithms. The jobs of navigation in rows and drawing a white line can be switched in this task. The WLAN control task includes the whole TCP/IP software stack and provides data exchange with client (Notebook, PC). In development phases this data link is used, but in competition robot activating and job choice is done by a touch display. In the Display control task the communication with the touch display is handled and corresponding algorithms are started. In Navigation task the sensor data is used to make decisions about steering and robot speed. In the speed and steering control task these decisions are disposed to generate the PWM calculated by the different algorithms depending. The read CAN_data task is responsible for updating used variables for sensor values. Incoming actual data is written to these variables when task is active.

Priorities of robot tasks

The usage of "sensor fusion" makes it necessary to distribute priorities to different robot tasks. These priorities are defined by safety reasons (flex sensors), the presence of plants (long distance IR sensors), first priority row guidance (AVRcam) and second priority row guidance (short distance IR sensors):

1. flex sensors

long distance IR sensors
AVRcam
short distance IR sensors

Maizerati Testing GUI

Figure 23: Screenshot of Testing GUI

With help of the Maizerati Testing GUI it is possible to communicate over the WLAN link with the robot. In development and testing phases this graphical user interface in combination with a WLAN capable Notebook was used to fine tune parameters of algorithms and set up other robot data. In addition most sensor and variable values can be displayed. This eases to debug algorithms and react on errors.

Among others the number of counted balls, pictures acquired by the AVRcam and actual speed can be shown.

6. Conclusion

During the Field Robot Event the Maizerati successfully drew a white line and navigated through the straight and curved rows with outstanding speed. With regard to the dry soil and the condition of the rows, the U-turn worked very well. The second place at the speed race affirmed that the choice for a four wheel vehicle was a good one. The high performances drive however wouldn't have been functioning that well if a redundant sensor system hadn't been applied. The recognition of the thin crop spears turned out to be very difficult for all teams which relied on the measuring properties of the IR-sensors.

Problems occurred during the hole detection event. The construction of the detection bar was insufficient for the bumpy and soft grass surface. As it worked well on even ground, a rather more flexible design could have led to a success in this category.

In the last discipline which was the freestyle event, the multifunctional barrel trailer was used for watering the plants. Unfortunately the improvised procedure was less impressive than it was planned because one of the microcontrollers malfunctioned in consequence of the heat just on the verge of the start of the freestyle action.

Figure 24: The Maizerati with its painting barrel trailer during the "line"-contest

Due to the chosen strategies Maizerati became the winner of the Field Robot Event 2006. According to the overall result we were content with the performance our robot presented on the field of Hohenheim. The application of virtual development such as package analysis played a decisive role in the technical progress of the new concept. The success which progressed year by year has proven that the technical enhancements made from one generation to the next were successful and the robot processing was effected into the right direction.

Project team

*Figure 25: project team (**upper row from left to right:** Hartwig Markus, Mario Urra-Saco, Wilfried Niehaus, Frank Fender, Andreas Rahenbrock, Jörg Klever, Odo Meyer; **lower row:** Ralph Klose, Andreas Linz, Marius Thiel, Prof. Dr. Arno Ruckelshausen, Kai-Uwe Wegner, Vitali Schwamm; **missing**: Christoph Kronsbein)*

Acknowledgment

Special thanks go to the students Christoph Kronsbein, Frank Fender and Marius Thiel as well as the lecturers Prof.Dr. Arno Ruckelshausen and Andreas Linz for their support.

The whole team of the University of Applied Sciences Osnabrück would like to thank its generous sponsors:

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

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

[optoMAIZER 2005] "Field Robot optoMAIZER" ; Ralph KLOSE, Martin MEIER (diploma thesis supervised by A. Ruckelshausen, A. Linz); Field Robot Event 2005, Wageningen / The Netherlands, Proceedings, ISBN 90-6754-969-X, November 2005

About the Hardware and Software Development of Putrabot for Field Robot Event 2006

Mohd. Hamiruce Marhaban, Adznan Jantan, Samsul Bahari, Malik Arman Morshidi, Mohd, Helmi Khairan Mohd. Akik, Abdullah Sukardi, Akmar Hisyam

Dept. of Electrical and Electronic,
Faculty of Engineering,
Universiti Putra Malaysia, 43400 UPM,
Serdang, Selangor, Malaysia

Abstract

This paper describes the technical details on the hardware and software development of Putrabot. Robot vision system was used to detect corner flag, counting the dandelions (simulated by yellow golf balls), hole detection, and plant detection (for free style session). Hough transform was incorporated in the image-processing task for corner flag detection. Robot maneuvering towards a detected target was implemented using fuzzy logic controller. Infrared and ultrasonic sensors were used for navigation through curved maize rows and doing headland turn at the end of each row. Results obtained during the testing and training are also included.

Keywords

robotics, autonomous navigation, computer vision, hough transform, fuzzy logic controller

1. Introduction

For the Field Robot Event 2006, the organizing committee has defined the Contest Disciplines and Rules. They were the Basic Competition, Advanced Competition, and a Free Style session. The listings and details of the rules are as follows:

Basic Competition

The line – drawing a straight white line towards a corner flag.

Dandelions detection – counting number of dandelions while navigating between curved maize rows.

Speed race – competing against each other for the fastest robot that can navigate on straight maize rows.

Advanced Competition

Hole detection in grass – detecting a hole (10 x 10cm wide, and 5cm deep) on a lawn (5 x 5m). The boundary of the field is marked white.

Free Style

Presenting any unique specialty the robot can demonstrate.

2. Mobile Robot Structure

As shown in Figure 1, the robot was developed on top of a 4-wheeled drive all terrain CEN Genesis 46 chassis. A Vexta DC brushless motor was used to power the robot, where the shaft was connected directly to the original gearbox. An Ackerman drive technique was used, where a Futaba servomotor with 15kgs torque was used to control the swivel of front wheels. Figure 2 shows the schematic diagram of the robot architecture.

Figure 1: Putrabot chassis.

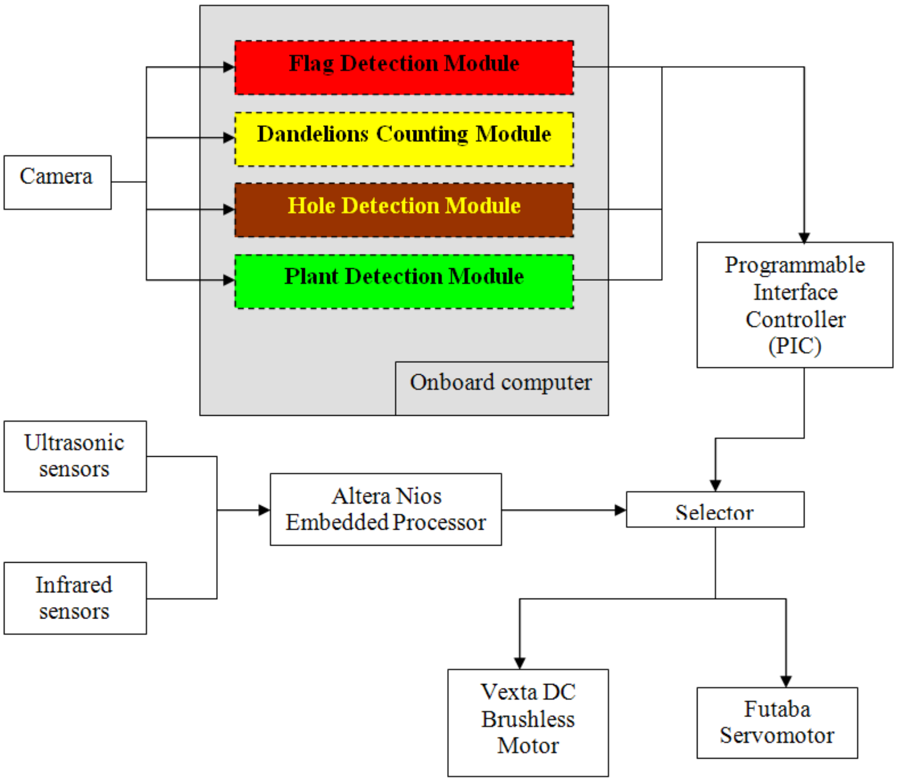


Figure 2: Schematic diagram of the robot architecture.

3. Software Development

Software development for Putrabot can be divided into three parts. They were the software for the vision system, software for robot motion control (mainly for swivel mechanism), and software for robot navigation. There is a one-way communication between vision system and robot movement control software (as illustrated in figure below), where vision system will send information in the form of characters to the robot movement control. Navigation software stands on its own, only for the purpose of robot navigation between maize rows and doing a headland turn.

3.1. Vision System Software

Vision system was developed using Visual Basic 6.0. Two types of vision systems were used for this competition. The first type was for target detection and robot maneuvering towards the detected target. The target can be the corner flag, the hole in the grass, or any object. The second type was for the dandelions counting. Only the first type of the vision system has the one-way communication with robot movement control software. Communication was done using RS232 serial port.

3.2. Robot Movement Control Software

Robot movement control software was developed using C. This software controls many parts of the robot such as its swivel mechanism, transmission, and solenoid valve (to control on-off water/paint spraying). This software was programmed into Programmable Interface Controller (PIC) chip (PIC 16F876). It communicates using serial port with the vision system software installed on an onboard laptop by receiving input in the form of characters such as ‘a’, ‘b’, ‘c’, etc. The PIC will generate Pulse Width Modulation (PWM) signal to control the Futaba servomotor. Characters were used instead of numbers because the serial port communication can only send 8 bits of data at a time. Below are the listings of characters and their corresponding controls towards the robot.

1. ‘a’, ‘b’, ‘c’, ‘d’, and ‘e’ are to control the robot to turn to the left where character ‘a’ being the maximum turn to the left.
2. ‘f’ is used to control the robot to maintain its tire at the center so that the robot can move straight.
3. ‘g’, ‘h’, ‘i’, ‘j’, and ‘k’ are to control the robot to turn to the right where character ‘k’ being the maximum turn to the right.
4. ‘m’ to trigger water/paint spraying.
5. ‘n’ to turn off water/paint spraying.
6. ‘x’ to set the transmission system to move forward.
7. ‘y’ to set the transmission system to free its gear.
8. ‘z’ to set the transmission system to move backward.
9. ‘r’ to run the motor.
10. ‘s’ to stop the motor.

3.3. Navigation System Software

Navigation system was developed using Quartus II. This software was then programmed to an Altera Nios Embedded Processor. This software acts based on the inputs received from the infrared and ultrasonic sensors.

4. Row Navigation

4.1. Navigation Sensors

The robot used four ultrasonic sensors (Keyence Digital Ultrasonic Wave Sensor) and seven infrared sensors (SunX Photoelectric Sensor). Figure 3 shows the schematic top view of the robot and the positions of the sensors.

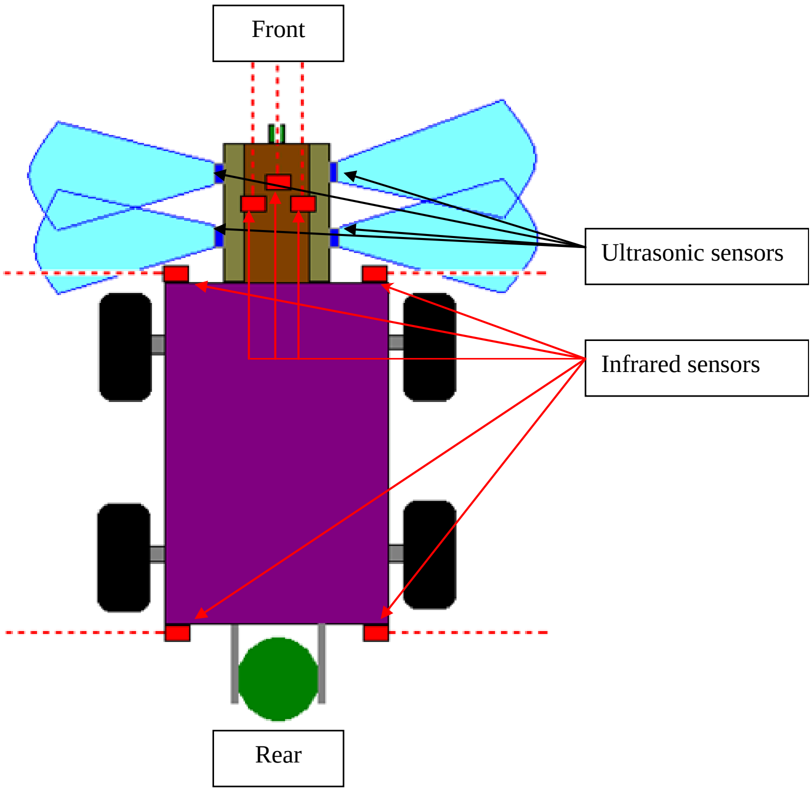


Figure 3: Sensors positioning on the robot.

All the sensors were connected to an Altera Controller board. This controller board was where all the navigation and headland turn processes were done. All sensors produced signals to the Altera Contoller board in the form of 1 (one) and 0 (zero). The four infrared sensors, each positioned close to the wheels were used to detect if the robot was still in the row. The robot was still considered in the row if at least one out of the four infrared sensors gave signal 1.

The three infrared sensors, positioned in front of the robot were used to detect if the robot was about to run into a corn tree or any other obstacles.

4.2. Headland Turn

The robot will do the headland turn whenever all sensors received no input after one second. This will be the indication that the robot was no longer in the row. The robot will make a headland turn by turning the front wheels. Upon reaching the next row, the robot will face either one of the two conditions. First, whether the robot is about to run into the corns of the next row, or second, the robot will successfully enter the row. These two conditions can be illustrated in the Figure 4 and Figure 5.

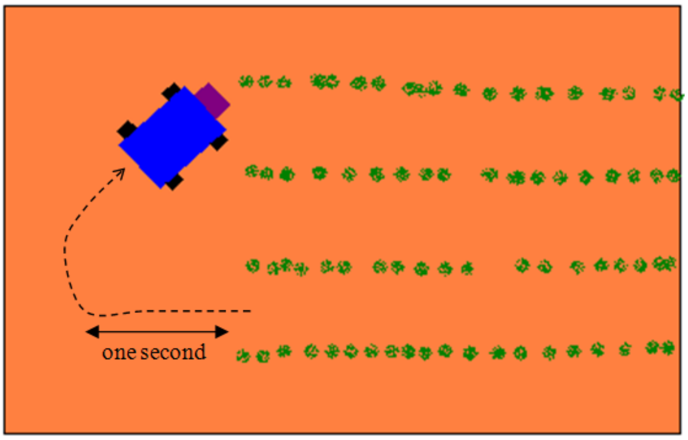


Figure 4: First condition, where robot is about to run into the corn row.

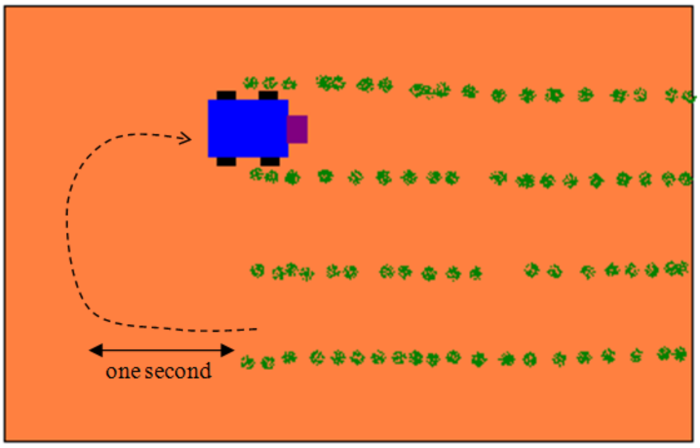


Figure 5: Second condition, where robot successfully enters the new row.

If the first condition occurred, the sensors positioned in front of the robot will detect the presence of obstacles (the corn tree) and the robot will immediately stop. Then, the robot will reverse for 0.5 second, and then move forward to enter the new row. Once the robot was in the new row, it will do the normal row navigation procedure.

4.3. Dandelion Counting

The dandelions used for the Field Robot Event 2006 were yellow golf balls. Since the color of the golf balls can be easily distinguished from the background, we decided to use color segmentation to extract the dandelions image. We used Creative WebCam Live! TM Ultra as the robot eye. The frame size was 320 x 240. As the robot navigates through the rows, the webcam continuously captures images at the rate of 3 to 5 frames per second. The original frame rate setting was set to 25 frames per second. However, when the vision system was put into the real-time test, the frame rate was reduced to 3 to 5 frames per second. This was due to the fact that there were so many image-processing processes need to be done for each frame. In addition to that, Visual Basic plays a role in reducing the speed and performance of the vision system.

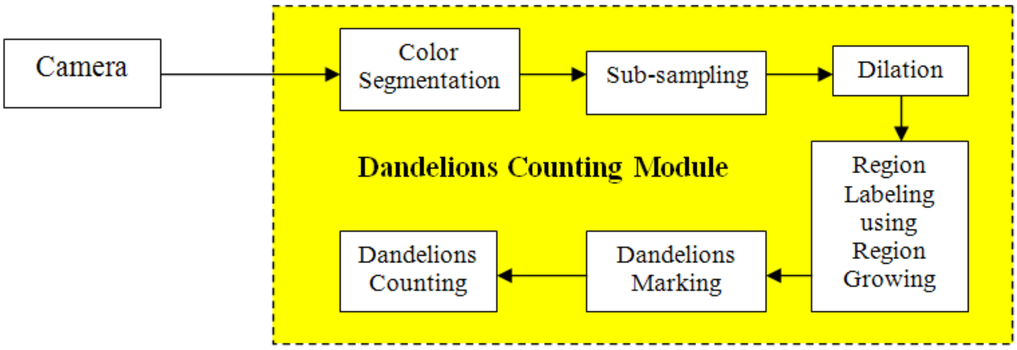


Figure 6: Schematic diagram of Dandelions Counting Module.

Figure 6 shows the schematic diagram of the dandelions counting process. Before the dandelions counting process was done, an image of the dandelion in the real environment was captured. The color of the image of the dandelion in the frame will be sampled by clicking on the image. As illustrated in Figure 7, the RGB components of the center pixel and the surrounding pixels will be sampled. From there, the average of the RGB values R_{ave}, G_{ave}, and B_{ave} will be calculated.

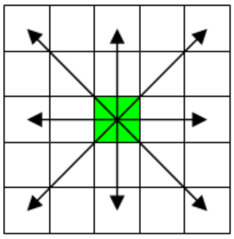


Figure 7: RGB sampling.

After the color sampling and calculation of the average RGB values, the robot will be ready for the dandelions counting task. While navigating between maize rows, the webcam continuously capture images, where each frame will be processed to count the dandelions. Next figure shows the user interface of the dandelions counting software.

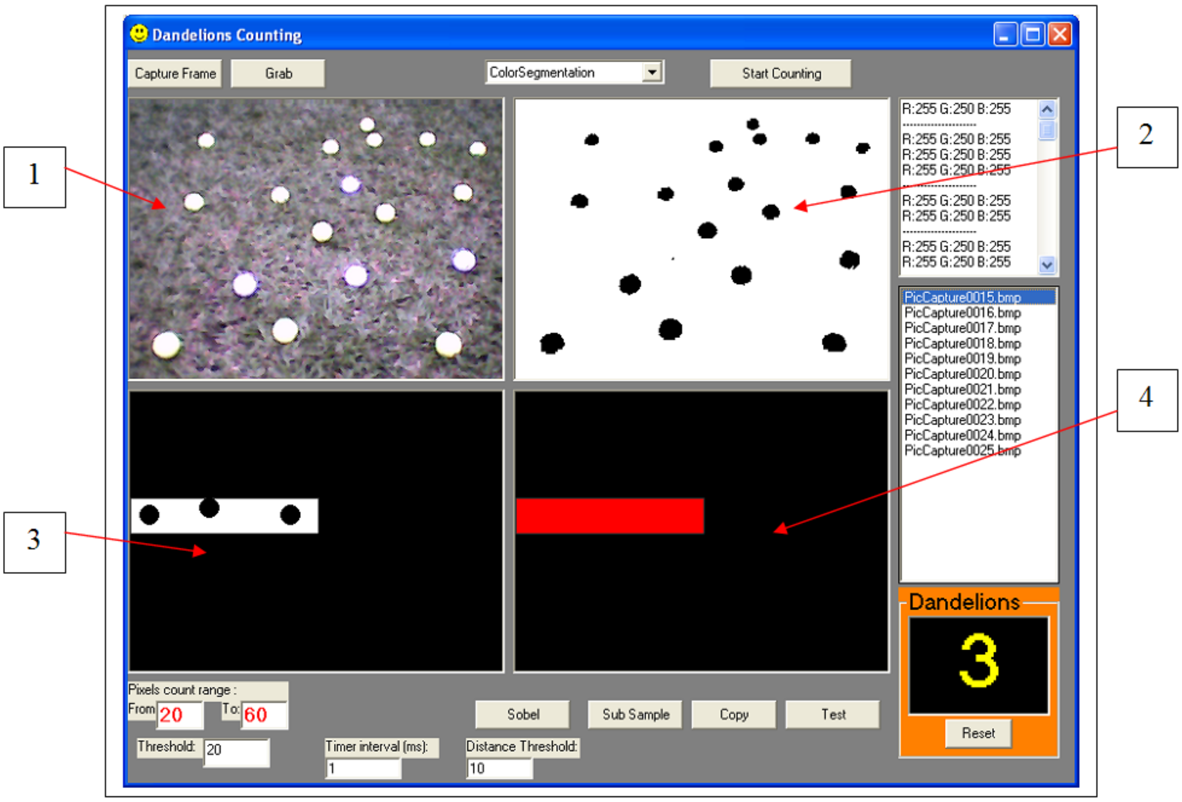


Figure 8: User interface of the dandelions counting software.

As can be seen from Figure 8, there are four regions on the user interface to show the image after few image processing stages. The first region, placed on the top left corner of the user interface is used to display the images captured by the webcam. The second region is used to display the binary image after color segmentation. The third region is used to display the image after sub-sampling, dilation, region labeling, and dandelions marking. The fourth region is used to map current frame and the previous frame for the counting purpose, which is to avoid counting the same dandelion multiple times.

Color segmentation will be applied first to the newly captured frame. Color segmentation was done by considering each pixel $I_{(x,y)}$ in the image I with threshold T. We then convert the image to binary image. Figure 9 is the algorithm to convert the image to binary image.

```
IF (I_R(x,y) >= (R_ave - T) AND I_R(x,y) =< (R_ave + T))
   (AND I_G(x,y) >= (G_ave - T) AND I_G(x,y) =< (G_ave + T))
   (AND I_B(x,y) >= (B_ave - T) AND I_B(x,y) =< (B_ave + T)) THEN
      I(x,y) = BLACK
ELSE
      I(x,y) = WHITE
```

Figure 9: Algorithm to convert the image to binary image.

To reduce the computational burden of the image processing process, we sub-sample the binary image to a quarter of its original size, from 320 x 240 to 160 x 120. This can be illustrated in Figure 10.

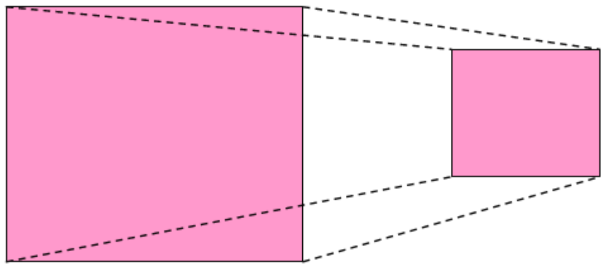


Figure 10: Sub-sampling from 320 x 240 to 160 x 120.

After the sub-sampling process, not all pixels in the sub-sampled image will be considered. The sub-sampled image will be further divided into four regions, where only the pixels in the bottom region will be considered for further processing. This can be illustrated in Figure 11.

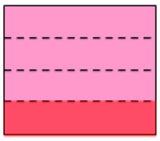


Figure 11: Sub-sampled image divided into four regions.

One of the reasons we did this process was to reduce the computational burden. The second reason was, as the robot moved the dandelions that fall within the field of view of the robot will appear from top of the frame and moved towards the bottom of the consecutive frames. The dandelions counting will start whenever the dandelions reached the bottom region of the sub-sampled image. Example of this can be illustrated in Figure 12.

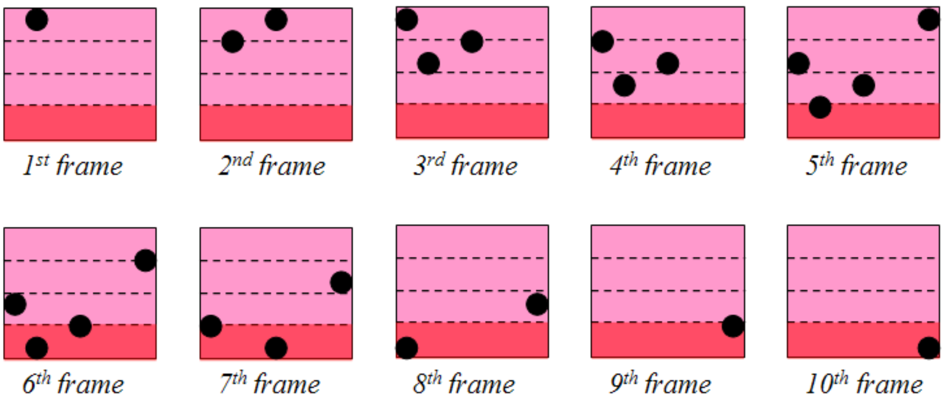


Figure 12: The figure shows how the dandelions are viewed as the robot moves forward.

The binary image after sub-sampling might appear smaller than the original size, and it might experience lost of data (pixels). To enhance the image for further processing, the sub-sampled image will be dilated. After the dilation process, we do region labeling using region growing technique to the sub-sampled binary image. During the region labeling process, each unconnected region will be labeled with different colors.

The potential dandelions will be marked by considering a region whose number of pixels falls within a specified threshold. Before the marking process, the binary image of the dandelions might appear as shown in Figure 13.

Figure 13: Before dandelions marking process.

The previous binary image containing the dandelions will be marked using *Circle* method available in Visual Basic. This will make the regions larger than the original binary image. This can be seen in Figure 14.

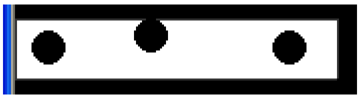


Figure 14: After dandelions marking process.

The purpose of enlarging the dandelions regions was to compare by mapping the current frame and the previous frame so that the same dandelion will not be counted multiple times.

5. Flag Detection

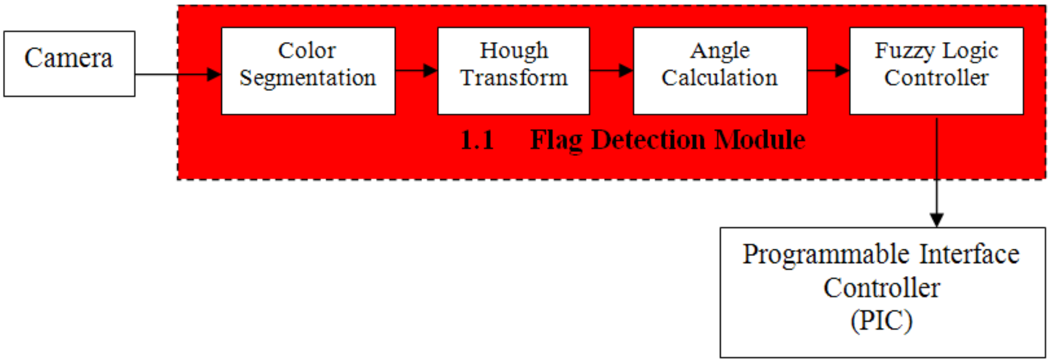


Figure 15: Schematic diagram of Flag Detection Module.

5.1. Image Segmentation

Color segmentation was used to segment the flag and the pole. This process is the same as the one used for dandelions counting.

5.2. Hough Transform

The Hough transform, first introduced by Hough [1] in 1962, is a powerful method for parameter extraction of any analytic pattern in images. The most common patterns are straight lines and circles. By mapping features in an image space into sets of points in a parameter space, the Hough transform converts a difficult global detection problem to a more easily solved local peak detection problem [2]. The main advantages of the Hough transform are its robustness to noise in the image and discontinuities in the pattern to be detected.

For the purpose of this competition, the Hough transform was applied to detect the pole of the corner flag. Figure 16 shows how the Hough transform can successfully locate the pole of the artificial corner flag.

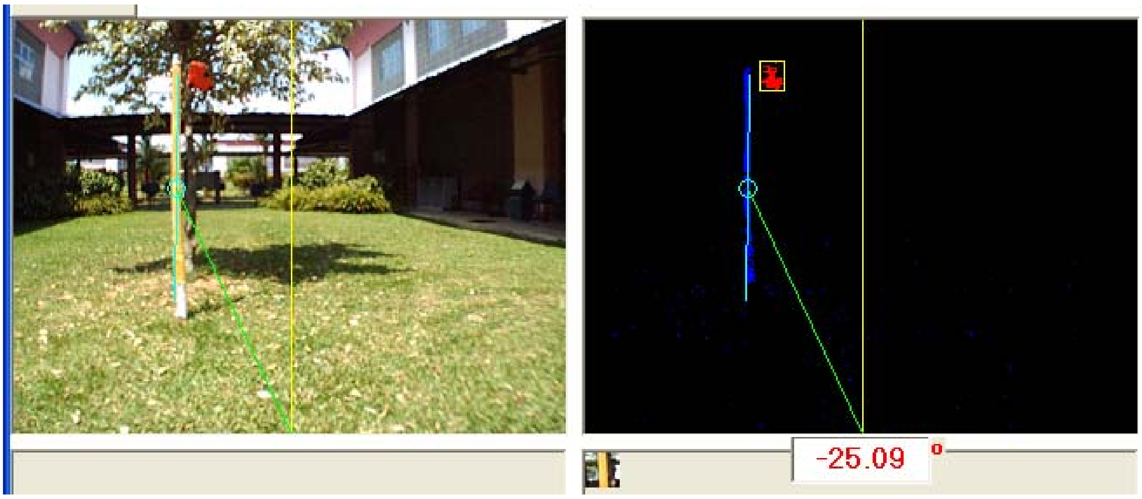


Figure 16: Corner flag detection using the Hough transform.

5.3. Fuzzy Logic Controller

In order to navigate in an unknown environment, a mobile robot needs to deal with the environment in a timely manner. This results in real-time demands on the navigation system. Due to its simplicity and capability for real-time implementation, fuzzy logic is an excellent candidate for such applications.

Fuzzy logic has been utilized in navigation systems for mobile robots for over a decade. Early in 1991, Yen and Pfluger [3] proposed a method of path planning and execution using fuzzy logic for mobile robot control. From then on, the efficiency of using fuzzy logic in mobile robot navigation systems has been demonstrated, see e.g., [4]-[7]. A comprehensive study of fuzzy logic-based autonomous mobile robot navigation systems is given in [8]. Recently, several new solutions to the mobile robot navigation problem in unknown environments based on fuzzy logic have been proposed [9]-[11].

In the development of Putrabot, fuzzy logic controller was used as a platform for controlling servomotor. There are two inputs and one output in this system. The first input is the error (angle from the center of the vision system towards the target). The second input is the change of error (angle difference between the current angle and the previous angle). The output is the PWM that need to be generated. Figure 17,Figure 18, and Figure 19 show the input and output membership functions for the fuzzy logic controller.

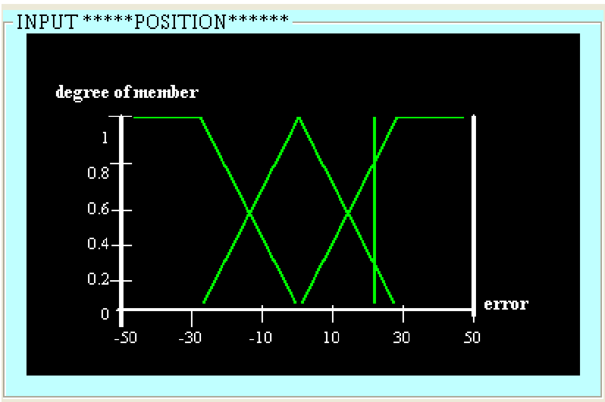


Figure 17: Membership function for Error.

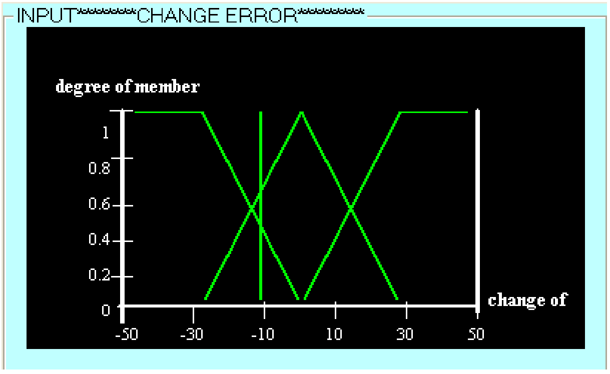


Figure 18: Membership function for Change of Error.

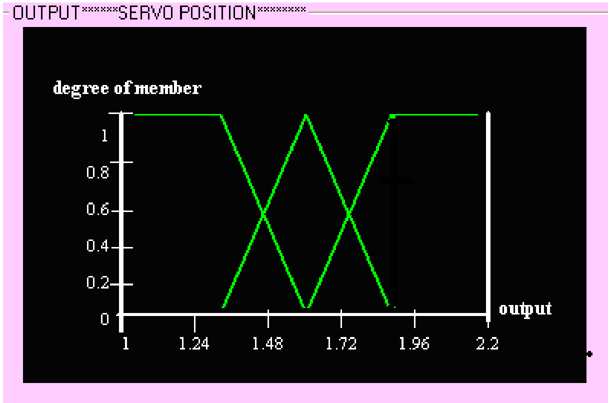


Figure 19: Membership function for output.

5.4. Drawing the Line

For the line drawing task, we used pressurized water tank to spray the white paint on the competition field. The water tank must be filled with white paint and must be pressurized prior to the competition. To control the on and off of paint spraying, a solenoid valve was installed on the water tank. This solenoid valve will be controlled by PIC to trigger the on and off paint spraying. In this competition, the paint will be sprayed automatically if the image of the corner flag felt within the center view of the vision system.

6. Hole Detection

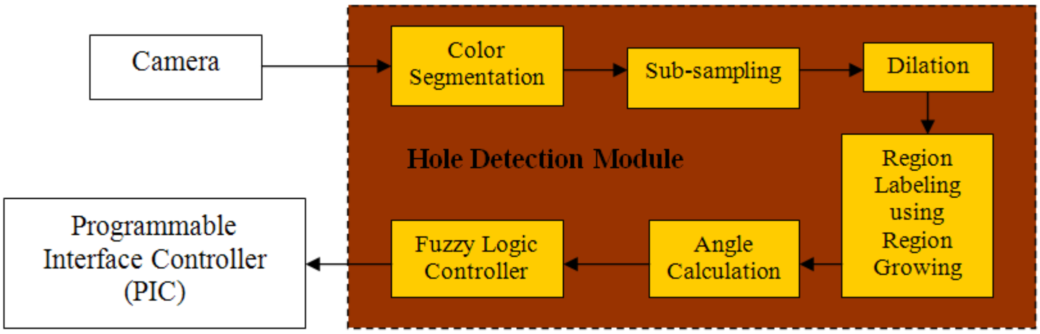


Figure 20: Schematic diagram of Hole Detection Module.

For the hole detection task, we use similar concept as the one we use for flag detection. The color of the potential hole will be sampled prior to the hole searching process. Since the robot has to be placed at the corner of the competition field at the starting of the searching process, we have preprogrammed our robot to continuously move straight towards the border of the competition field, do a headland turn, and continue towards the opposite end of the border, and do the same process repeatedly. This process can be illustrated in Figure 21.

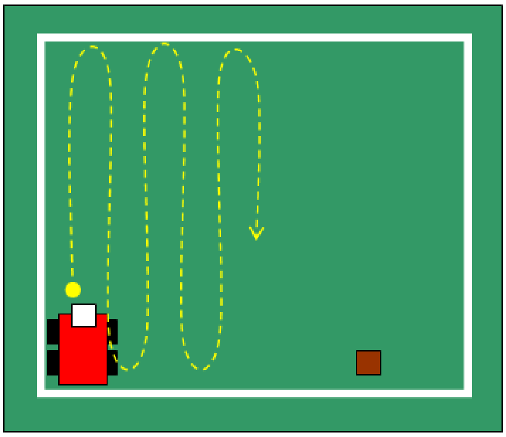


Figure 21: The pre-programmed navigation to search for potential hole.

If the potential hole falls within the field of view of the robot vision system, the robot will ignore the preprogrammed movement and move straight towards the detected hole. This can be illustrated in Figure 22. Once the hole is right in front of the robot, the robot will automatically stop and spray water onto the hole to indicate that the hole has been found.

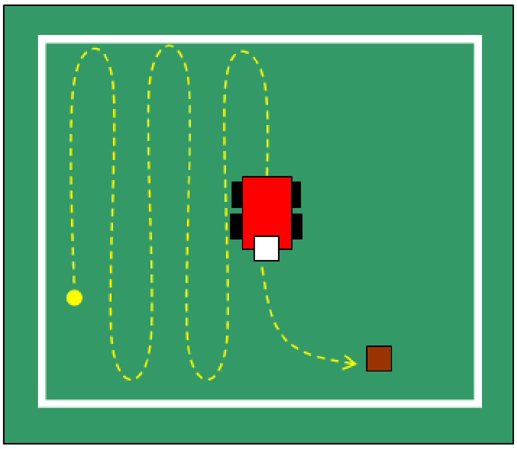


Figure 22: The robot ignores pre-programmed navigation and moves towards the hole.

7. Freestyle

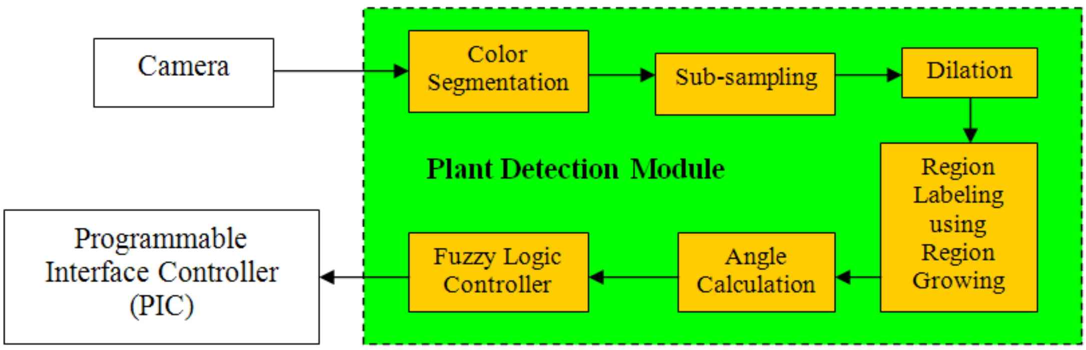


Figure 23: Schematic diagram of Plant Detection Module.

For the free style session, we have decided to demonstrate the robot capability of finding plant and spray water onto the plant. First, we sampled the color of the leaf of the plant. At the start of the searching process, the robot will move straight until it finds the potential plant within the vision system field of view. The robot will then maneuver straight towards the plant. Once the plant has come close to the robot, the robot will automatically stop and spray water onto the plant. The robot relied mainly on the vision system to control its movement. No infrared or ultrasonic sensors were involved to stop the robot.

8. Discussion and Conclusions

We have developed an autonomous robot that is capable of doing several tasks such as navigation between maize rows, calculating dandelions, flag detection, hole detection, and plant detection. For the most part of these tasks, vision system and fuzzy logic controller played an important role for the successful completion of a given task. Infrared and ultrasonic sensors were used mainly for the purpose of row navigation.The main contribution this paper is a joint modeling of vision system and fuzzy logic controller. This modeling is a generic concept and can find its application in many additional detection problems. The system has been tested in artificial environment as well as in the real environment, and we conclude that the results are encouraging. This has been proven to be successful where our team won the 1st place for the free style session. The same model was also used for the hole detection competition where we won the 3rd place in that competition.

Our first experience in the Field Robot Event 2006 allowed us to discover many potential research areas. In future, we want to tackle a number of challenges for further improvement especially in the computational speed and the recognition accuracy. In this respect, we are planning to address the following issues: the incorporation of color classification for the color segmentation and shape analysis for better recognition accuracy.

References

[1] P.V.C.Hough, Method and means for recognizing complex patterns. U.S. Patent 3069654, 1962

[2] J.Illingworth and J.Kittler, A survey of the Hough transform. Computer Vision, Graphics, and Image Processing 44, 87-116, 1988.

[3] J. Yen and N. Pfluger, “Path planning and execution using fuzzy logic,” AIM Guidance, Navigation and Control Conference, vol. 3, pp. 1691-1698, New Orleans LA, Aug. 1991.

[4] P. Reignier, “Fuzzy logic techniques for mobile robot obstacle avoidance,” Robotics and Autonomous Systems, vol. 12, pp. 143-153, 1994.

[5] P. S. Lee and L. L. Wang, “Collision avoidance by fuzzy logic control for automated guided vehicle navigation,” Journal of Robotics Systems, vol. 11, pp. 743-760, 1994.

[6] B. Beaufrere, “A mobile robot navigation method using a fuzzy logic approach,” Robotica, vol. 13, pp. 4371148, 1995.

[7] J. Yen and N. Pfluger, “A fuzzy logic based extension to Payton and Rosenblatt’s command fusion method for mobile robot navigation,” IEEE Trans. Syst. Man, Cybern., vol. 25, no. 6, pp. 971-978, 1995.

[8] A. Saffioni, “The uses of fuzzy logic in autonomous robot navigation,” Soft Computing, vol. 1, pp. 180-197, 1997.

[9] M. B. Montana and A. R. Serrano, "Fuzzy knowledge-based controller design for autonomous robot navigation," Expert Systems with Application, vol. 14, pp. 179-186, 1998.

[10] H. Maaref and C. Barret, "Sensor-based fuzzy navigation of an autonomous mobile robot in an indoor environment:' Control Engineering Practice, vol. 8, pp. 747-768, 2000.

[11] L. Doitsodis, K. P. Valavanis, and N. C. Tsourveloudis, "Fuzzy logic based autonomous skid steering vehicle navigation," IEEE Int. Conf. on Robotics and Automation, vol. 2, pp. 2171-2177, Washington, DC, May 2002.

Relaxsystems

N. van Gerrevink, J. van Goethem, E. Haak, J. van Kessel

Hogeschool van Amsterdam

Abstract

Students from Hogeschool van Amsterdam build a robot for Field Robot Event 2006. The name of the robot was Relaxsystems. In this article development process of the robot is presented and the construction of the robot is described in details. The robot utilizes camera technology and distance sensors and the artificial intelligence is programmed into ATMega microcontrollers. The robot participated in Field Robot Event 2006 and the overall ranking was 5th, despite of all unexpected problems.

Keywords

mobile robots, field robot event, mechanics, electronics design, PCB, infrared distance sensors, microcontrollers, DC motors

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 corn field in Wageningen. The cars had to drive through straight and curved corn rows while counting the plants they passed. At the end of a row, the car should enter the over next row. This year the Field Robot Event was organized by the University of Hohenheim. Hohenheim is situated in Germany, near Stuttgart.

The rules of this year's event were a bit more complicated then the years before. Not only did the robot’s had to drive through the corn field, the also needed to count yellow golf balls. There was also a speed race where the competitors had to compete against each other in a race between the corn plants. The fastest robot wins this part of the event.

Finally the teams got 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

This year only one team from the Hogeschool van Amsterdam joined the Field Robot Event. The name of this team is Relaxsystems. Most of the competing robots had names that had something to do with corn or the fieldrobot event. The name Relaxsystems was formed in the pub beneath the Hogeschool van Amsterdam. The name of the pub is LAX. Since we are a group of friends which are most of the time pretty relaxed, we decided to name ourselves Relaxsystems.

The team consists of four members, Nivard van Gerrevink, Jeroen van Goethem, Emiel Haak and Joris van Kessel. We are all third year students E-technology.

We all built a robot boat last year. This robot had to sail around two buoys in the Amstel river. This made us enthusiastic with robots, therefore we decided to compete in the Field Robot Event 2006.

We divided the group into two little sub-groups. One had to design and build the car and make the car driving. So they needed to design the hardware and software for the driving. The other two team members had to write the software and design the PCB's for the image processing.

1. Materials and methods

1.1. Mechanics

1.1.1. Properties

We used an existing chassis from remote controlled car, with the following specifications:

- Wheels with a diameter of 8 cm
- Four wheel drive
- 6 cm ground clearance
- Small chassis: 34x50 (WxL)
- Steering front wheels
- Variable sensor height
- High torque motor
- Good suspension so all the wheels will be in contact with the ground all the time.

With these specifications it is possible to drive at various kinds of soil like sand, clay, wet conditions or with big clods. It doesn't matter if the corn is small or big because we can adjust the height of the sensors.

1.1.2. Design

We started our design with the chassis we already had. We made a aluminium frame on top of the chassis where we mounted all of our PCB's. Because the robot needs to drive outside we also made a hood from Plexiglas where we mounted the display and the buttons.

The car is approximately 380mm in length and is 50mm wide. The ground clearance is 60mm.

Suspension

When we started to build the car we thought that the original springs of the suspension of the chassis was good enough. But, when the first parts and the battery where mounted, they appeared to be to soft. We decided to buy stiff suspensions. The new suspension is precisely stiff enough.

The robot can still take bumps and humps of clay, but now without getting stuck on the ground.

Wheels

The original tires of the robot where a bit to soft. This causes the car to steer heavy, and as a result of that to use more energy. To make them harder we stuffed the tires with foam. After stuffing we noticed that the car steered a lot lighter and the energy consumption decreased a lot.

Sensors

The car has five infrared sensors (figure 2). They are all placed at the front and four of them are used to navigate through the corn plants. The fifth is used to detect holes in the ground. The whole construction of the sensors is easy to adjust in height. We made them height-adjustable to make sure that we can detect the corn under all conditions. The sensors can be placed at heights between 5 and 35 cm’s. To protect the sensors against rain and sunlight we have mounted plates above and beneath it.

In our first design we only had two infrared sensors (figure 1) , but already by our first test it came out that 2 sensors are to sensitive. This is because of the small angle of sight. When there is a hole in the corn rows, the robot needs a larger angle of sight.

Therefore we designed a new system with 2 infrared sensors mounted in a slightly different angle on each side. The average value of these two infrared sensors is used to navigate through the corn plants. The working will also be explained in the software section (chapter 1.7)

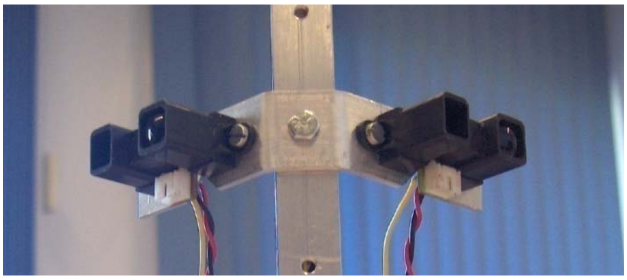


Figure 1: Two sensors

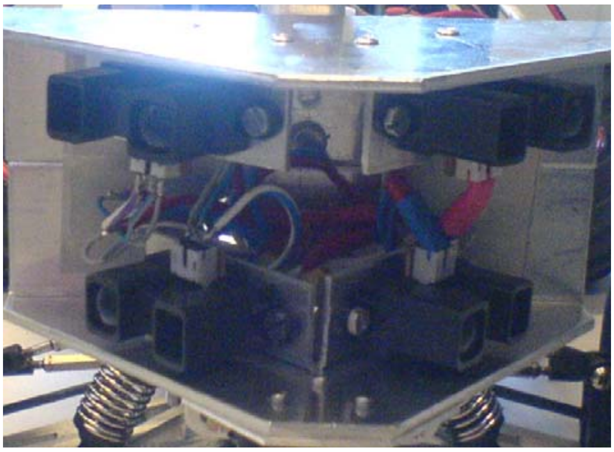


Figure 2: Four sensors

Motor

The car is driven by one 12Vdc motor, which will use 1 Ampere at maximum. The motor is 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.

On top of the motor is a gearbox mounted (figure 3). We bought the same gearbox that last year's team used. The reduction ratio of gear was 1:50. When we started testing we found out that the gearbox had a reduction that was to large for us (we wanted a fast robot). So we decided to rebuild the gearbox. After removing some of the gearings the reduction ratio was about 1:12, which was much better for our purpose.

Steering is done by a servo, a motor that can turn only a view degrees, depending on a PWM signal.

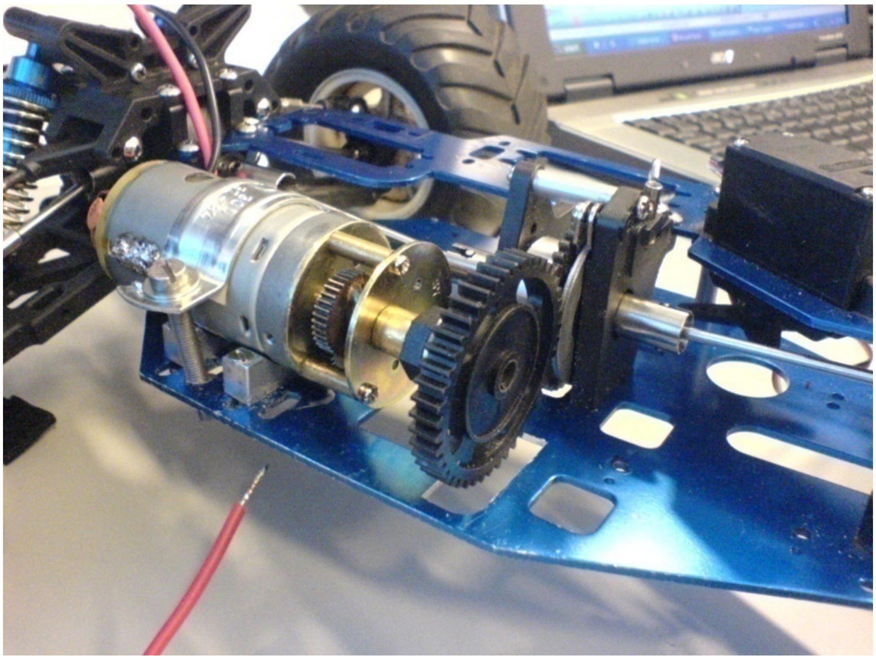


Figure 3: Gear box & motor

1.2. Electronics

All the electronics in our car where developed by ourselves. In figure 4 the block diagram of our electronics is shown.

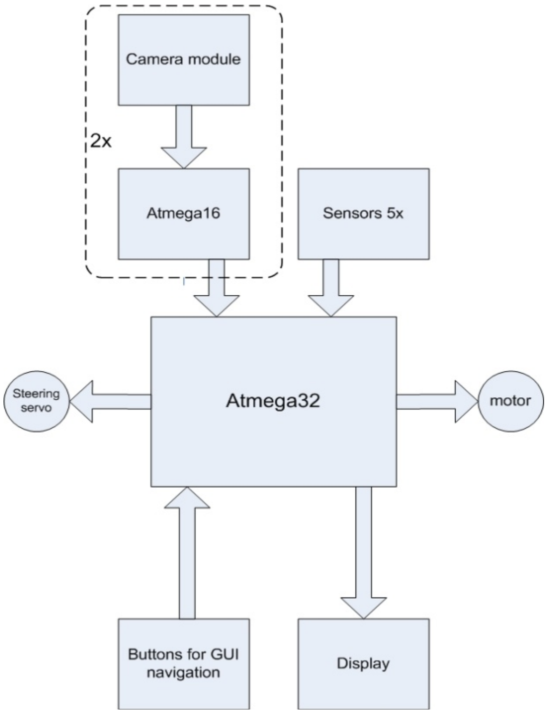


Figure 4: Block diagram

1.2.1. Sensors

We have chosen to use five 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. This signal is sent right into the ATMega32 where an internal Analogue/digital converter converts the signal into an 8 bits hexadecimal value. With these sensors (figure 5) it's possible to detect distances from approximately 10 to 150 cm.

Figure 5: Infrared sensor

1.2.2. Microcontroller

We have chosen to use an ATMega32, because of the number of I/O ports, the speed and the low costs of the chip.

The microcontroller is the main part for the intelligence of the car. To control the car, the microcontroller communicates with the sensors, the buttons and the two ATMega16's from the camera modules. It also sets the speed of the motor, the steering and it writes the text on the display.

During the race the ATMega32 uses a clock source of 8 MHz which is provided by an external crystal.

1.2.3. Main print

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

Also mounted on the main print is the user interface. The user interface consists of 6 buttons and a display. The buttons are used to switch between our software programs and to give a reset to the ATMega32.

The display has 4 lines with 20 characters, so we can easily navigate through the menu, which contains 4 options.

1.2.4. H-bridge

We were planning to use a H-bridge, for the motor control, because with a H-bridge you can drive forward as well as backwards. But a week before the Field Robot Event our H-bridge fried up, so we used a MOSFET for the motor control. This because our turning circle is small enough, so we don't need to drive backwards

The H-bridge PCB converts the low-power signal from the ATMega32 to a high-power PWM signal for the driving motors.

1.2.5. Intelligence

Before we started to build the car we decided how we wanted to drive. We had 4 infrared sensors as inputs. As output we had one motor and one steer servo. A schematic picture of the car with the sensor placements is shown in figure 6 The infrared sensors at the front are placed at an angle of approximately 30^o and 45^o.

There were three different competitions. We had to drive as fast as possible in a straight row and we had to drive in a curved row while counting yellow golf balls. We also had to detect a corner flag in an open field, and draw a straight line to it. For all of them we had to make different programs. The two corn row competitions are a bit similar to each other so they will be discussed first.

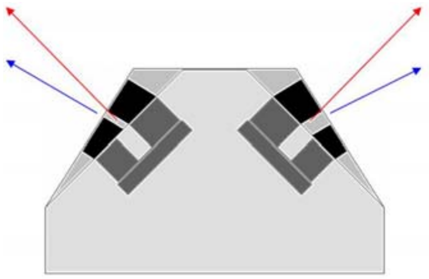


Figure 6: Schematic diagram of sensors

1.3. Driving through the rows

While driving through the corn rows the car checks the four infrared sensors on the front.

We found out that the best way to drive is to stay as much in the middle of the row as possible. When you want to stay in the middle you mustn't steer a lot. So we told our robot that it shouldn't steer a lot unless it's really close to the plants.

To make sure the robot stays in the middle of the row, we check both sides of the robots sensors and divide them from each other. left - right = steering. When the robot drives in the middle of a row the value of both sensors will be equal. In that case the algorithm will be : 30 - 30 = 0, so the robot won't steer (and stays in the middle) When the robot drives a bit more to the right the algorithm will be: 30 - 20 = 10, and the robot will steer to the left. It's the same when it's driving on the left side of a row. Because we're checking the sensors every 20 ms, this gives a very fast response time, and makes sure that the robot stays in the middle of a row, whether the row is curved or straight.

If we don't see objects for a certain time the robot 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 for a certain time. This angle and time is tested to skip one row. If we're turned 180o the car is straight in front of the over-next row. At that moment the normal routine of checking the sensors starts again.

The only difference between the navigation in straight rows and the curved rows is, that in the curved rows the robot has to drive into the over-next row. In the straight row routine, the robot stops at the end of the first row.

While driving through the curved rows the robot also has to detect and count golf balls.

1.4. Flag detection

We tried to detect the flag with image processing but that was considering the amount of time we had, not possible. So we decided to use an infrared transmitter in the field (next to the flag) and three infrared sensors on the robot. The software for the detection is quite simple. The robot checks the sensors all the time and when the left sensor detects something, the robot steers to the right till the middle sensor sees the infrared beam. It's the same for the right sensor. When the middle sensor detects the beam, the robot drives simply forward without steering at all. When the robot is close to the transmitter it stops driving.

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 corn plants, without touching them. At the end of the project we eventually seemed to have to little time to think of a freestyle device. Therefore we didn't compete in the freestyle competition.

1.6. Advanced competition

In the advanced competition we had to detect a hole in the ground. We decided to detect the hole with an infrared sensor which is aimed to the ground. When we detect de hole the robot stops driving.

Around the field is a white line, which is detected using our camera modules. For this part of the competition the camera modules are pointed directly to the ground so they can only see the white line. When a white line is detected the car drives in the opposite direction of where the line is seen. This is a very easy to program, and low-cost solution for a difficult problem.

1.7. Software

We used three microcontrollers in the robot. One ATMega32 and two ATMega16's.

The ATMega32 is the main microcontroller which controls the two ATMega16's and the rest of the robot.

1.7.1. Main microcontroller (ATMega32)

The software for the microcontrollers in our robot is written in the language C. The final software has a size of approximately 18kB.

The software consists of different parts. It consists of some 'hardware drivers' to control the IO. Furthermore, there is a menu part and an intelligence part.

The software checks in which program it is (race, golf ball detection or driving to the flag) then it checks the relevant sensors (1 or 4 infrared sensors, and the camera modules).

The software then checks the measured inputs and makes decisions on these values.

The microcontroller determines the speed and steering and sends the required PWM signals to the motor and the servo. In all programs the speed of the robot is different, and also the angle of steering is different. Therefore the microcontroller first checks in which program it is and after that it checks the sensors.

1.7.2. Camera module microcontrollers (ATMega16)

We decided to detect golf balls with a camera module and some image processing software. Because we want to learn from this project we made all the software and hardware by ourselves. Figure 7 displays the block diagram of the camera software module.

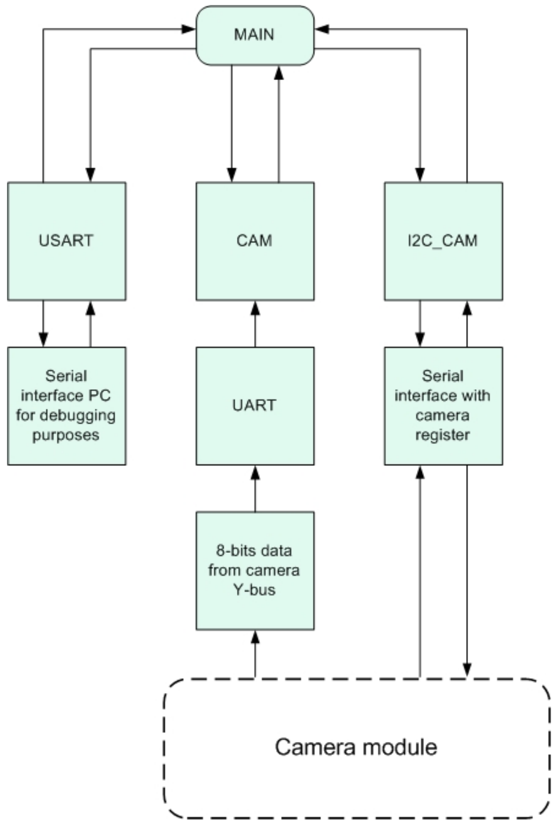


Figure 7: Block diagram of the software

There are different programs combined to make the image processing work. The program which controls them all is called “main”.

USART is used for debugging and direct serial communication with the PC. While writing and testing the software we communicated whit the laptop using HyperTerminal. In that way we could see what happened inside the chip. That’s very important because we decided not to put a laptop on top of the robot, or to communicate with a laptop during the race. We thought that the robot must be capable of driving and image processing without a laptop, so with just a microcontroller to do the calculating.

CAM is the program where the processing is done. When we start the program for the golf ball detection the camera module immediately starts to make pictures and send them through the Y bus to the microcontroller. Every pixel of the image is send as 1 byte of data. (8 bits parallel) Every clock cycle a pixel is send to the microcontroller and the next clock cycle is used to transform the 8 bit value into a hexadecimal value. That’s because it’s much more easy to handle a hexadecimal value then a 8 bit binary code.

the hexadecimal value is compared with a preset value and if they match, that means that a yellow (or white) pixel is seen. When there are more than a certain number of yellow pixels counted a signal is given to the ATMega32, which then writes on the display that there is a golf ball detected.

I^2C_CAM is only used to set and read the register values of the camera modules. The register values where one of the most difficult parts of the entire project. We bought a camera module with the OV6630 chip from OmniVision on it. This chip has 92 registers built in. All the registers present one or more settings of the camera. It was hard to find the right registers and set the good values so we could make the best pictures.

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 corn plants with paper. With this test field we were able to see the robot’s behaviour.

Already in the first tests it appears that our robot responded really good in the rows.

2.1.2. Outdoor

We had a lot of problems during this project, most of them were hardware problems, but also a lot of strange errors in the software. Due to al these problems we had no time to test outdoor until the night before the race. But the robot did what it had to do and drove nicely between the corn rows. After an hour of testing, our self-made gearbox broke down. And we didn’t even think that we could compete in the race.

But luckily Steffen Walther helped us out on Saturday morning (8.00 h). Due to our hard work we were able to test 2 hours before the race. We programmed the row ends and the turns. While testing we were the only robot that can drive without failures through 7 rows of corn.

Of course that pleases us a lot, because the other teams all had a much bigger budget.

2.2. Race results

The day of the race actually came a bit to soon for us, but we discovered that our robot drove very good, so we didn’t worry too much. 15 minutes before the first part of the competition, we found out that our infrared transmitter wasn’t working as we would like it to work. We couldn’t detect it at all. So we decided not to compete in this part.

The second part of the competition was the golf ball detection. The rows were in very bad condition, and our robot lost its way after a good first turn and a very fast first row. Half way the row, it thought it was at the end and made a turn - and immediately another turn after that. Then he was completely lost and didn’t want to drive through the rows at all.

It surprised us a bit since the testing went so well, but we had already seen that the rows of the speed race were in much better conditions. And we’d also seen that our robot was one of the faster robots competing in the event.

The last part of the competition was the speed race. We won our first heat despite of a stop halfway when one of the camera’s got stuck in the corn plants. For the finals we decided to remove the camera’s so that wouldn’t happen again. In the final we had to compete against the three other heat-winners. After a quick start the robot stopped again, this time we didn’t know the reason. We think that it was overheated in the full sunlight (30°C). But after a quick reset it raced again, passed all the other competitors and won with over 5 meters on the rest!

Due to this great result we ended on the 5th overall place.

This place is not what we came for (we came to win), but concerning all the problems we had and the little time/budged we had compared to other teams we are satisfied with this result.

3. Conclusion

On Friday evening we never thought that we would even participate in the Field Robot Event. This was the situation because the gearbox broke down on Friday evening.

Compared to other teams we had a small budget and not so much time to work on the project, because we also had to design and build all the hardware. That was something we didn't see with the other teams.

Therefore we're pretty satisfied with our fifth place. Of course we aimed for a top three ranking and we were very close, but just missed a few days of testing experience.

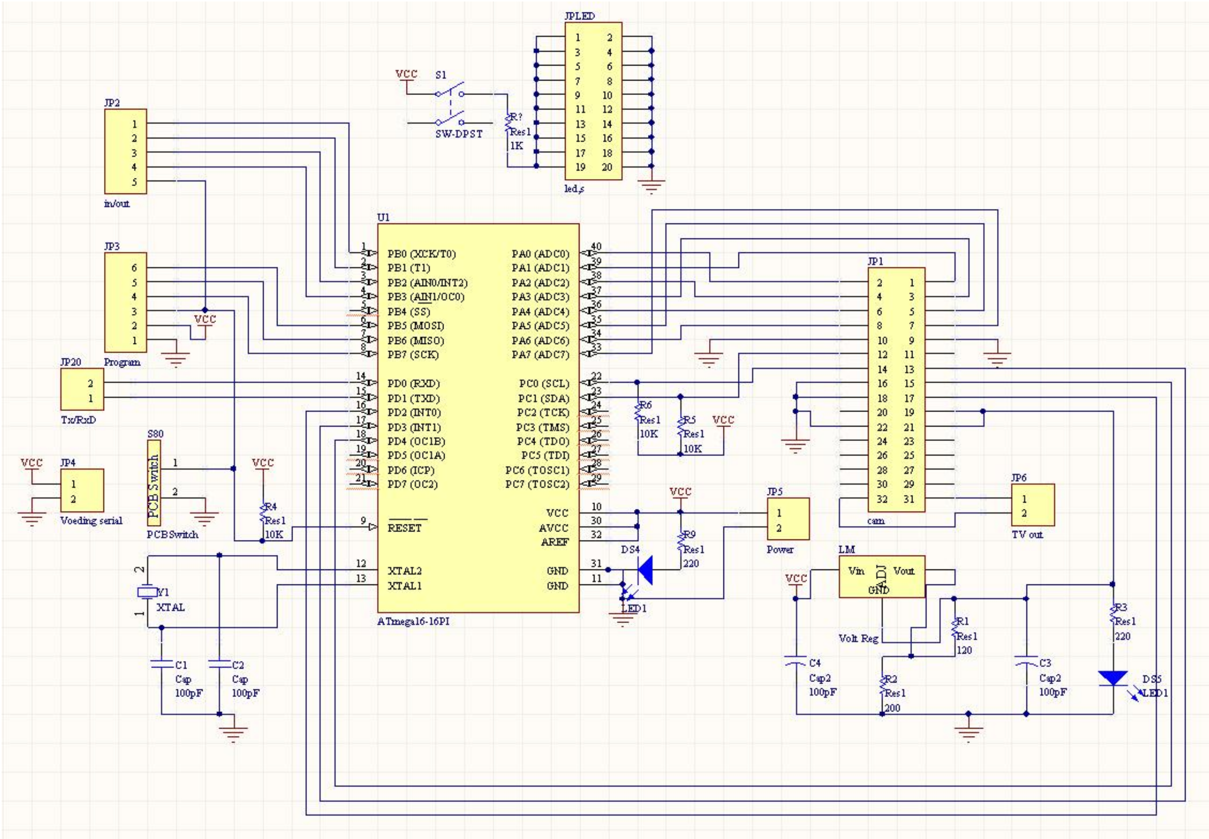
The main advantage of our robot was the technique of our 4 infrared sensors. That made a great driver of the robot. Also because we only used infrared sensors for navigation we didn't need to write complicated image processing software for the navigation.

In our school career we learned not to think to difficult. So we tried to keep the car as simple as possible to keep the costs low. Simplicity avoids to much complex systems which are also complex to build (much time) and harder to debug.

The weakness of our robot was that we didn't have time to test outside. If we had tested we would have known our robot a bit better. And we could have written a better software code. Time was a great pressure for us because we only had three months to build a robot out of nothing - including all hard and software.

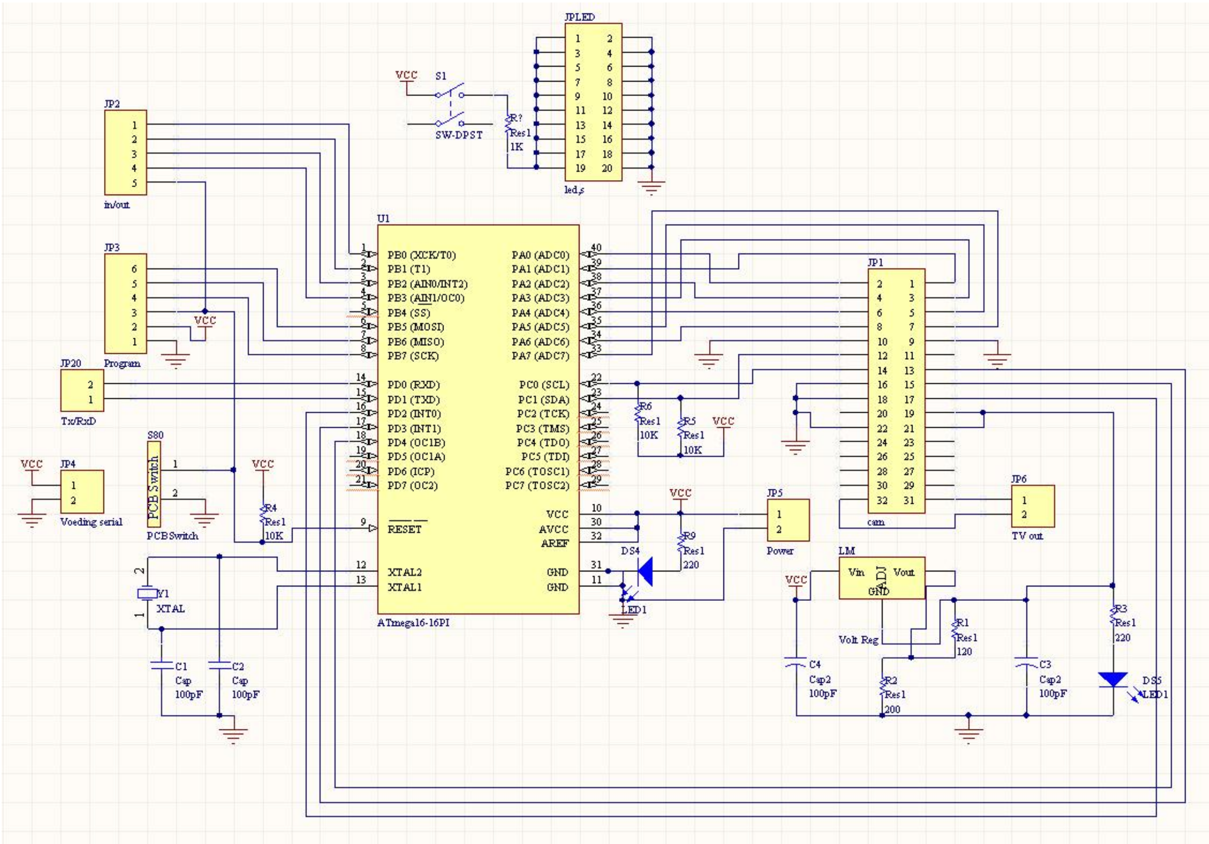
Overall we can conclude that we didn't work on needless difficult solutions, but we used our time to build a simple car, which is easy to adjust to new requirements. We are very satisfied with the performance of our car.

Appendix A: Hardware schematics (AVR cam)



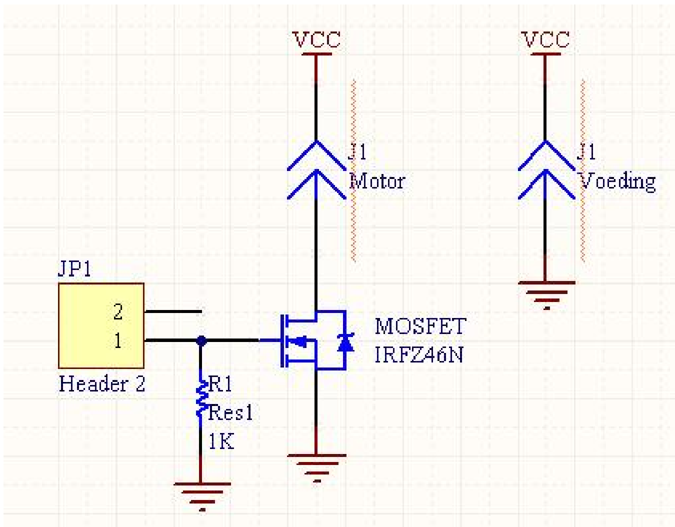
Figure

Appendix B: Hardware schematics (Main print)



Figure

Appendix C: Hardware schematics (MOSFET)



ROBOKYB

J. Friedrich, S. Husser, T. Ruppel

Universität Hohenheim
Institut für Agrartechnik 440
Garbenstrasse 9, 70599 Stuttgart, Germany
email: fieldrobot@uni-hohenheim.de

Abstract

Field robots are the cutting edge of science in Precision Farming - and fascinating hands on learning objects for the upcoming e-generation. Hence University of Hohenheim invited high-school and university teams to participate in an international open-air field robot contest. These teams have to compete with self-constructed robots, navigating and operating autonomously in a maize field.

Keywords

field robot, agriculture, autonomous, student competition, infrared sensors, dandelion detection, object detection, image processing

1. Introduction

Robokyb is created for the fourth Field Robot Event in Hohenheim/Germany and is conceived as an executable operative and less interference-prone robotics. Robokyb is a consistent enhancement of ROBOKYB1, ROBOKYB2 and Turtle. Autonomous navigation in our vehicle is realized by optical sensors which are based on infrared sampling, assisted by an electronic compass-module. At the beginning a tracked vehicle was planed as the basic module, where four infrared sensors and the compass module have been affixed. For the contest we switch to a wheel-based vehicle, because of problems with the tracks. The vehicle is operating on the field autonomously without remote control.

2. Material and Methods

Robokyb is constructed as an autonomous crawler. In its first conception Robokyb was intended to use the following components: Heading Signal from a digital compass module and distance signals from four infrared sensors. Robokyb should process this data in one computer module, running with 16MHz and using a flash-memory of 32kB. This module is able to be in-system-programmed via ISP-Port.

The module is able to run two PWM engine drivers, several LED's for man machine communication and an RS232-interface for external computer communication/data logging. Concerning the sensor- and processor-techniques we started as planned.

Robokyb navigates inside the rows using two infrared sensors in front. It recognises the end of a row by missing reflection from the side mounted IR-sensors. These side mounted sensors are activated after a predefined distance (approximately 90% of row-distance).

Turning on head land bases on a relatively turn based on the signal from the compass module. Robokyb measures the actual value at the end of the row and turns, until the value reached plus 90 degrees. Robokyb drives forward, acquired by the last two IR-sensors, measuring row numbers. The second turn stop is defined as a difference of 180 degrees to the stored value at the beginning, in order to avoid inaccuracies from twist while driving at the head land. We achieve more accuracy by using of the front IR-Sensors for detecting the middle of the row. Slip of the tracks while driving forward can be ignored under all conditions. Driving performance is improved by a control cycle with IR-Sensors and pulse duration modulation by dint of a PID controller. The controller-cycle had the following characteristic and was easily been integrated into the software, Figure 1.

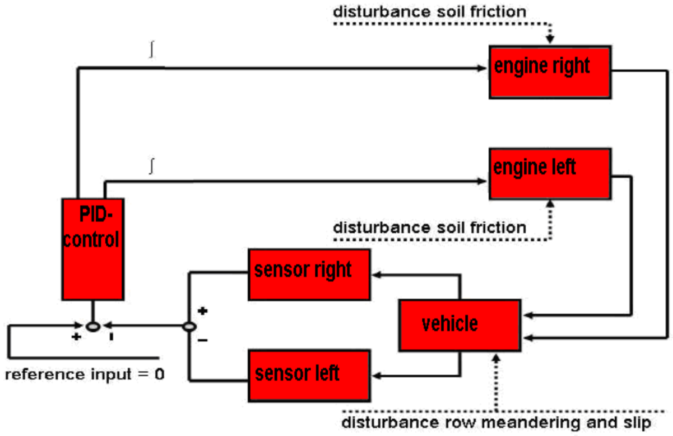


Figure 1: Schematic diagram of control cycle

This control-cycle prevents a buildup of the oscillation between the rows. The course becomes better, as closer Robokyb is to the middle lane because the needed course correction is smaller. Every correction results in breaking and loosing speed. The two signals from the distance sensors are summarized and being approached towards zero.

During the freestyle session, after changing the basic vehicle, we got the possibility to use an old grass-cutter. So we improvised an autonomous full harvester.

3. Ball detection and Image Processing

To master the ball detection task we equipped Robokyb with a camera device connected to a stand-alone IBM laptop. As an image capture device we used a digital radio-camera (CMOS Total Pixel 628*582). The signals (PAL B standard) were picked up by a Video USB converter, interpolated to a 640*480 pixel resolution and processed on the IBM laptop in Matlab R14.

The ball detection mainly consists of four parts:

- preprocessing and segmentation
- image separation and threshold function
- ball finding
- ball detection and tracking

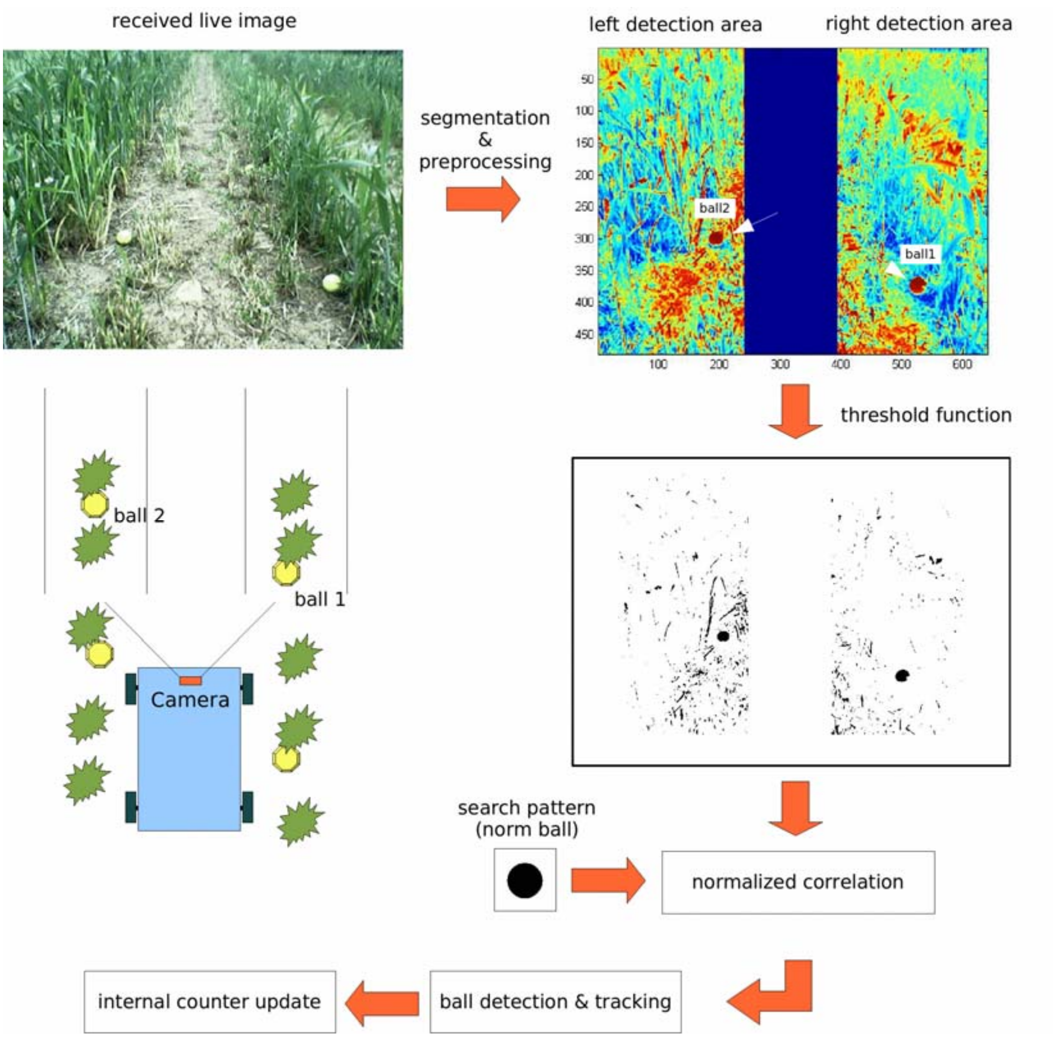


Figure 2:

In a first step the received color image is converted to a grayscale image and reduced to the left and right detection areas. The following threshold function and low pass filtering is supposed to suppress noise and small object processing. In a next step potential balls will be found by correlation algorithms based on norm ball images. The ball size, color, and speed (estimated by tracking algorithms) will be determined and compared to a predefined ball model. If a probability function of these values is sufficiently large, the ball will be counted and tracked.

The complete algorithm works with 5 Hz and allowed robust and satisfying ball detection.

4. Our sensor and electronic techniques

As infrared distance sensors SHARP GP2Y0A02YK were used, Figure 3. The distance range with this sensor is 20-150cm. The sensor gives the measurement as voltage (0-5V), and the corresponding distance can be get using the curve in Figure 4.

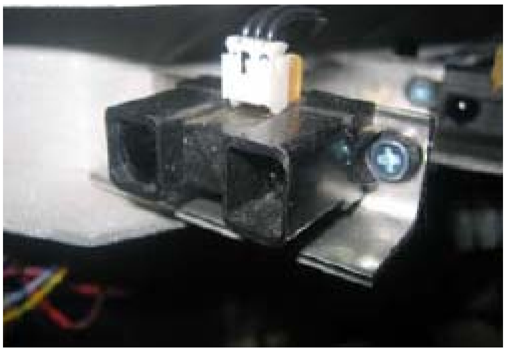


Figure 3: Infrared sensor devices (SHARP GP2Y0A02YK)

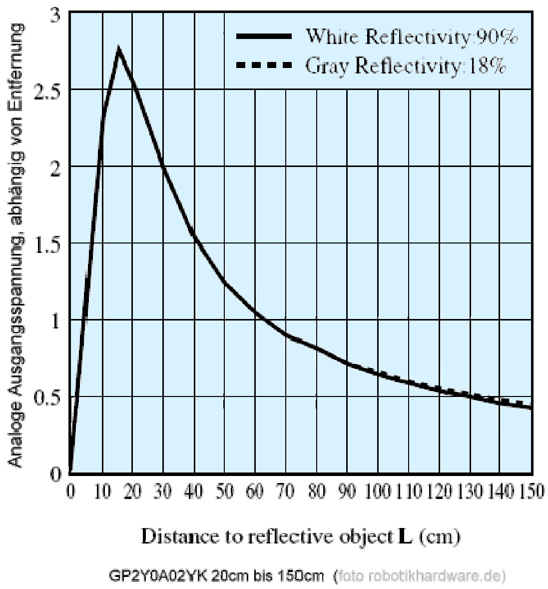


Figure 4: Voltage to distance curve of GP2Y0A02YK sensor

As a electronic compass, a compass module CMPS03 was used, Figure 5. The module gives a I²C interface signal of orientation. The accuracy is reported to be exactly horizontal 0.1 degrees. In our experiments it was found that accuracy on Robokyb was better than 4 degrees.

Figure 5: Compass module (CMPS03)

As a Main Board it was used a board shown in Figure 6. It has 5V-regulator (7805), ISP-interface, processing unit, RS232-interface, MAX3222 interface driver. For motor control dual channel motor driver was used, shown in Figure 7.

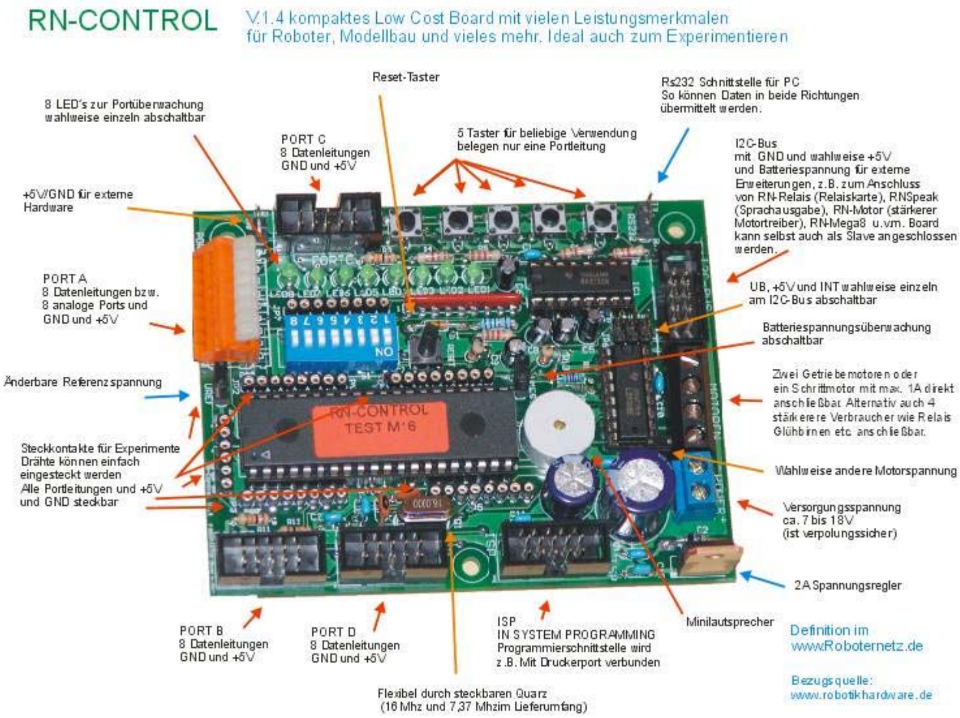


Figure 6: Main board

Figure 7: Dual channel motor driver,16V/10A per channel

5. Problems

In the last test one day before the contest the tracks and wheels crashed. The chassis had to be changed completely in the night.

At the contest we had problems with the heat and dust. The electronic crashed before the hole detection task.

6. Conclusions

It is worth mentioning, that we were a group containing exclusively students. Most of our knowledge had to be acquired; this knowledge was just in small fractions part of our course of study.

Within this Project we collected experiences with:

Electronic design
Programming
Controlling with different control cycles
Manual skills concerning Electronic, Electric, chassis building

Within this project we used the following software:

Mind Manager: developing the conception
AutoCAD 2000: design of the top cover and light barrier
Matlab: image processing
Bascom AVR: programming

In consideration of our problems, we are very glad with the result (3rd place). Until the evening before the contest Robokyb did not move an inch. After this we had a broken axle.

However, just thereby this project fulfilled its intended purpose. We had to be creative until the last second and finally solved this task in our way. Afterwards it can be said that next time we would work differently.

Thus, the participants learned additional skills in electronics, programming and organizing such a project. The field robot event should stay a student contest.

Sietse III, or: Robot recognizes *Rumex*

Frits K. van Evert[1], Arjan Lamaker[2], Arjan de Jong[3], Eltje Groendijk[1], Gerie van der Heijden[1], Lambertus A.P. Lotz[1], Susan L. Peterson, Gerrit Polder[1], Pieter Polder, Ary Polder, Ton van der Zalm[1]

[1]Plant Research International, PO Box 16, 6700 AA Wageningen, The Netherlands
[2]WUR Division ICT, PO Box 59, 6700 AB Wageningen, The Netherlands
[3]Alterra, PO Box 6700 AA Wageningen, The Netherlands

Abstract

In previous Field Robot Events our robot has used a visual, color-based method to detect crop rows. The main objective of the work presented in this paper was to develop a visual, texture-based method to detect crop rows. We also describe how the same method can be used to detect weeds in grassland. The texture-based detection starts with the transformation of the color image to a grey-scale image. The grey-scale image is divided into tiles of 8x8 pixels and a two-dimensional Fast Fourier Transform (FFT) is performed on each tile. Then, a new image is generated where each pixel represents one tile from the original image, and where the value of each pixel is related to the total power of the Fourier spectrum of the corresponding tile. Application of a threshold yields a binary image in which white pixels indicate crop material when detecting rows or weed material when detecting weeds. Detection of crop rows proceeds from here as with the color-based method; detection of a weed takes place when a sufficient number of adjacent tiles are classified as containing weed material. Both texture-based crop row detection and texture-based weed detection performed satisfactorily.

Keywords

Textural analysis, weed detection, broad-leaved dock

1. Introduction

The goal of precision farming is to maximize profitability and minimize environmental damage by managing each part of an agricultural field at just the right time, using just the right amount of fertilizer and/or pesticide. This is at odds with the trend to minimize the cost of labour by using ever larger machines. It has been suggested that in the future small, autonomous machines ("robots") will make it possible to precision-farm large areas without incurring large labor costs.

Weed control is often mentioned as a likely application area of agricultural robots in precision agriculture. One of the earliest references is the robot of (Tillett et al., 1998) in cauliflower. In some recent literature, the focus was on weeds in sugarbeet (Åstrand and Baerveldt,

2003) and on volunteer potato in maize (Van Evert et al., 2006). While these authors address weed control in arable fields, robotic weed control may have application in grassland as well. Broad-leaved dock (*Rumex obtusifolius* L.). is a troublesome grassland that is best controlled by manual removal of the plants, possibly combined with grassland renewal and rotation with a grain crop (Van Middelkoop et al., 2005). A motorized tool exists to shred dock plants (Anonymous, 2006a) but operating this tool is physically demanding. A robot that detects dock plants and destroys them using this tool would be a logical development. (Gebhardt and Kühbauch, 2006) describe a vision system to detect dock plants but it seems that the system is at present too slow for application in a mobile robot.

Robust navigation under all conditions is a sine qua non for agricultural robots. In the context of the Field Robot Event, the navigation problem is reduced to driving between regular rows of maize plants and making accurate headland turns. For FRE 2006 the intention was to increase the difficulty of the navigation problem by sowing the maize into an existing sward of grass.

The detection of maize rows and of broad-leaved dock against a background of grass have in common that discrimination on the basis of color doesn't work because maize, docks, and grass are all green. Thus, for FRE2006, the objectives of our work were to use texture-based method to detect maize rows and weeds.

2. Materials and methods

We modified the robot that was used in the 2005 Field Robot Event (Van Evert et al., 2006). This robot is based on a radio-controlled toy truck (TXT-1, Tamiya, Shizuoka City, Japan; Fig. 1). The radio-control parts were removed and a microcontroller board (LPC2129, Olimex, Plovdiv, Bulgaria) was installed to control speed and steering. Sensors for navigation included an ordinary webcam (NX Ultra, Creative, Singapore) mounted atop a mast at a height of 1.5 m, a home-built rotary encoder affixed to the main axle, and a solid-state gyroscope (ADXRS150, Analog Devices, Norwood, MA, USA). Images captured by the webcam were processed by an on-board mini-ITX PC board (G5M100-N, DFI Inc., Taiwan) equipped with a 1.6 GHz Pentium-M processor (Intel, Santa Clara, CA, USA). Power to the motors, the microcontroller board and the PC is provided by four 7.2 V NiMh batteries.

2.1. Improvements to the hardware

The home-built encoder used to measure rotational speed of the main drive axle was a source of trouble in 2004 and 2005. First, its resolution was not sufficient to enable fast and accurate regulation of the driving speed. Second, it tended to get stuck, resulting in the robot running away at high speed. This year, we replaced it with an optical incremental rotary encoder (model 3720, Kübler, Germany) which was just small enough to fit in the limited space around the drive shaft. The installed version provided 100 pulses per rotation, or about one pulse every 2.5 mm.

The gyroscope (ADXRS150, Analog Devices Inc.) was another component that did not work well in 2005. We identified and resolved problems related to grounding and problems related to the power supply. First, re-examination of the electrical circuit revealed the presence of ground loops. We modified the grounding scheme and removed all ground loops. Second, when we examined the gyroscope's power supply with an oscilloscope we found that it showed significant ripples of about 10 mV (nominal 5 V) with a frequency of a few Hz. We also discovered that activating the steering servos led to a large drop in voltage. The ripple problem was solved by feeding the gyroscope from an additional voltage regulator with a low pass filter to suppress the ripple. The voltage drop problem was solved by feeding the servo's from a separate battery. The final power schematic is shown in Fig. 2.

2.2. Improvements to the control software

We modified the speed control software in order to control the driving speed more aggressively. The high-resolution signals of the drive shaft rotation encoder made this possible.

Figure 1. Picture showing Sietse III while taking part in FRE2006.

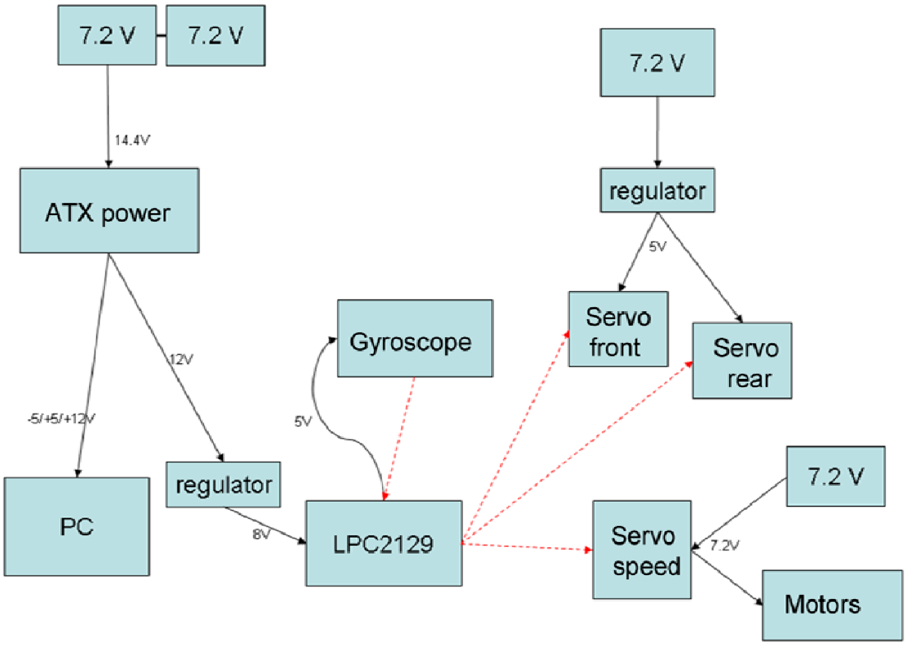


Figure 2. Schematic showing the electrical components of the robot. Black (solid) arrows indicate power connections; red (dashed) arrows indicate electrical signals.

We replaced the poll-loop acquisition of images that was used in the 2005 version of the robot with an interrupt-driven loop.

We modified the steering algorithm to make full use of the two signals that are generated by the row recognition algorithm. These signals are, first, the distance between center-of-row and center-of-robot ("lateral deviation"), and second, the angle between robot direction and row direction ("heading deviation"). When the robot is positioned exactly in the middle of two rows of plants, but it is not parallel to the rows, only the front wheels are used to steer. When the robot is positioned parallel to the rows, but not in the middle of the rows, both front and rear wheels are used to "slide" to the middle of the rows without changing the direction of travel.

2.2.1. Flag

A forward-looking camera (Intel, Santa Clara, CA, USA) was added to locate the flag in the first element of the competition. The flag recognition algorithm searches for contiguous groups of red (flag) and yellow (pole) pixels (red and yellow "blobs"). Given the resolution of the camera, the size of the flag and the pole, and the starting distance, we expected to be able to see the flag at all times; and to see the pole only after the robot had approached the flag quite closely. Therefore, if one or more red and one or more yellow blobs were found, the flag was assumed to be at the location of the red and the yellow blob that were closest together. If one or more red blobs were found, but no yellow ones, the flag was assumed to be at the largest red blob. If no red blobs were found, it was assumed that no flag was visible. We used the VXL image processing library (VXL, 2004) and some custom code to implement the algorithm.

The control algorithm for this element lets the robot drive a left-hand turn until the flag was detected, and then moves into seeking mode by generating a steering signal that is proportional to the distance between the flag and the middle of image. The vehicle is stopped when the red blob representing the flag reaches a threshold size.

2.2.2. Row following while counting golf balls

This task consists of two independent tasks and was tackled by implementing two separate, independently running programs.

Row-following: This program uses a camera to determine robot position with respect to the rows of plants. The camera is mounted atop a mast at a height of 1.5 m and looks straight down. A typical image captured by the webcam is shown in Fig. 3a. This image has a resolution of 320*240 pixels. Plants are separated from the background after which plant rows are recognized with an algorithm inspired by the Hough transform (Hough, 1962): imaginary lines are drawn over the segmented image and scored by the number of plant pixels that they cover. These scores become pixel values in a new image where the vertical coordinate of each pixel corresponds to the slope of the imaginary line, and where the horizontal coordinate corresponds to the intercept of that line with the vertical axis. Because a plant row typically contributes pixels to several imaginary lines, plant rows show up in the new image as areas with bright pixels (Fig. 3b). Thresholding (Fig. 3c) and dilation on the resulting binary image are used to merge areas of bright pixels into contiguous areas (Fig. 3d), after which the center of gravity of each contiguous area is taken to represent a plant row. Fig. 3e shows recognized rows superimposed on the thresholded image. Depending on the size of crop plants, the level of weed density, lighting conditions and algorithm parameters, three or more plant rows may be detected in this way. In that case, rules are used to filter out unlikely rows. Two examples of such rules are "plant rows must be on either side of the robot" and "plant rows must be reasonably parallel". After filtering, at most two rows are left and the robot's heading is obtained as the average of the headings with respect to these two rows.

For the segmentation between plants and background we have implemented two different algorithms. The first is based on color and was used already in 2004 and 2005 (Van Evert et al., 2004). For the 2006 Event, the organizers had indicated that the maize would be undersown with grass, making differentiation on the basis of color infeasible. We developed an alternative segmentation algorithm based on structure (see Fig. 4). Using the same camera and resolution, a grayscale image is made by a linear combination of the red, green and blue channels. This image is divided in square tiles of 8x8 pixels. A two-dimensional Fast Fourier Transform (FFT) is performed on each tile. Then, a new image is generated where each pixel represents one tile, and where the value of pixel is related to the total power of that tile's FFT spectrum. Application of a threshold yields a binary image in which plant rows can be found with the row finding algorithm described above.

Finally, the end of the plant rows needs to be detected. The method used in 2005 did not always perform well, so we reverted to the method used in 2004. With this method, it is assumed that the end of rows has been reached when, over a certain distance, no lines are detected.

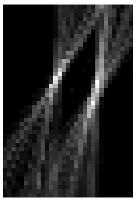
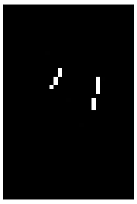
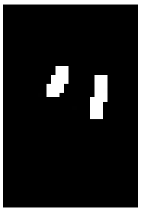
A		Original colour image, 320*240 pixels.
B		Hough space obtained after the original image was segmented. Each pixel represents a line in the original image; the vertical coordinate corresponds to the slope of the line and the horizontal coordinate corresponds to the intercept of that line with the vertical axis. Pixel intensity is related to the number of green elements covered by the line and is thus an estimate of the probability that the line represents a crop row.
C		Thresholded Hough space.
D		Dilated Hough space, clearly showing that two crop rows are detected
E		Detected crop rows superimposed on the thresholded image.

Figure 3. Steps in image processing to detect rows.

Rows simulated by red-white construction tape.	Rows simulated by weed leaves.	Rows simulated by reeds planted in a lawn.

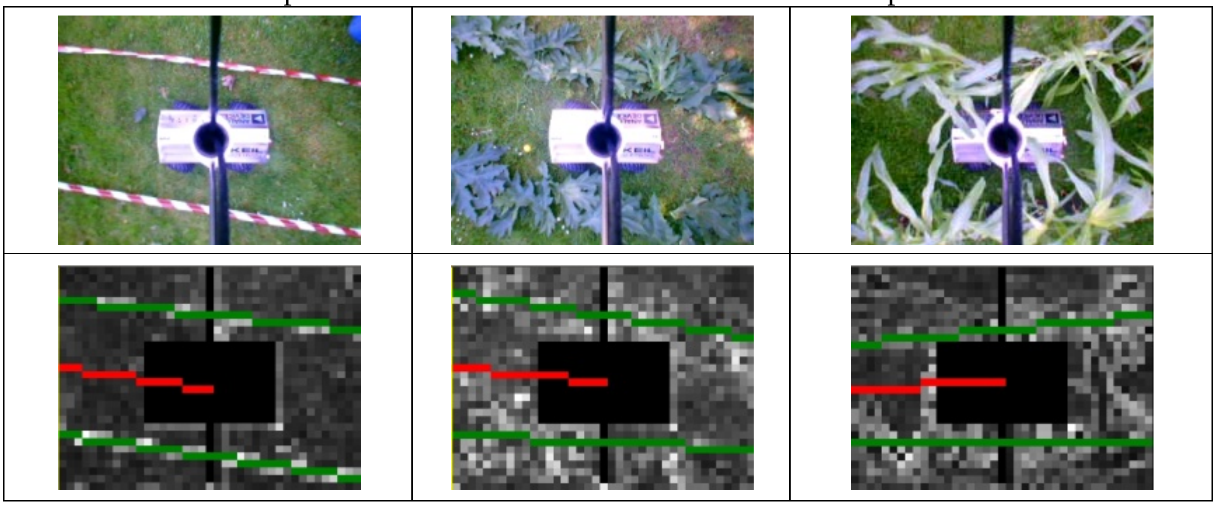


Figure 4. Texture-based detection of crop rows, demonstrated here for three types of artificial crop rows. Top row: colour images, showing simulated crop rows and the robot. Bottom row: images in which each pixel represents an 8x8 pixel tile of the original image and where the brightness of the pixel represents the the total power of the Fourier spectrum of that tile; green lines indicate detected rows and red lines indicate the orientation of the robot.

When the robot reaches the end of a row, it executes a turn by making two 90° turns separated by a straight section. The turning is measured by the gyroscope. At the end of this maneuver, the robot has executed a 180° turn and has moved sideways by exactly one inter-row distance.

Golf balls: This program uses images from the same camera that is used for row following. Golf balls were found by a formula of the form rR + gG + bB, where R, G and B are the intensity of pixels in the red, green and blue bands; and r, g and b are parameters. Optimal parameters were found by using a number of training images for which the segmentation had been performed by hand. After segmentation, golf balls were found by thresholding, erosion and dilation, after which each blob was taken to represent a golf ball. The driving speed was 0.4 m/s and images were processed at a rate of 2 Hz, meaning that one picture was processed every 0.2 m. Each golf ball was typically visible in more than one image, because successive pictures overlapped by about 50%. The counting algorithm deals with this by looking only at increases in the number of golf balls.

2.2.3. Speed race

For the speed race, the driving speed set point was larger than during the other contest elements; otherwise, no special software or special parameters were used.

2.2.4. Hole detection

For the hole detection element, we programmed the robot to drive straight for 3 m, make a 180-degree turn, and repeat this pattern. The golf ball recognition program was used to detect the holes, parameterized this time to detect non-green blobs of a certain size.

2.2.5. Freestyle: detection of *Rumex obtusifolius* in grassland[1]

(Ahmad and Kondo, 1997) used uniformity analysis to detect the presence of broad-leaved weeds in lawns. A similar algorithm was used by (Gebhardt and Kühbauch, 2006). We implemented the algorithm of Ahmad and Kondo and found that it performed reasonably well for docks in grass, but at several seconds per image it was too slow to be usable for real-time detection.

We developed a method for detection of broad-leaved weeds in grassland that is based on textural discrimination (see Fig. 5). We obtained almost 100 color images containing grass as well as dock plants. The images were taken with an ordinary digital camera (Cybershot DSC-60, Sony, Tokyo, Japan), which was hand-held while aiming straight down. Image parts containing only grass were characterized by high spatial frequencies, whereas in the parts containing one or more weed leaves, lower spatial frequencies are more important. On the basis of this observation, we divided our images into square tiles and performed a 2-D FFT analysis on each tile. The total power of the spectrum of a tile was found to be a measure of the presence of dock leaves. Application of a threshold made it possible to detect tiles containing weed leaves. A weed was assumed to be detected when a sufficient number of adjacent tiles were classified as containing weed material. We determined optimal values for the parameters of the above algorithm on four of the images. When these parameters were used on the remaining pictures, the absence or presence of dock plants was correctly determined.

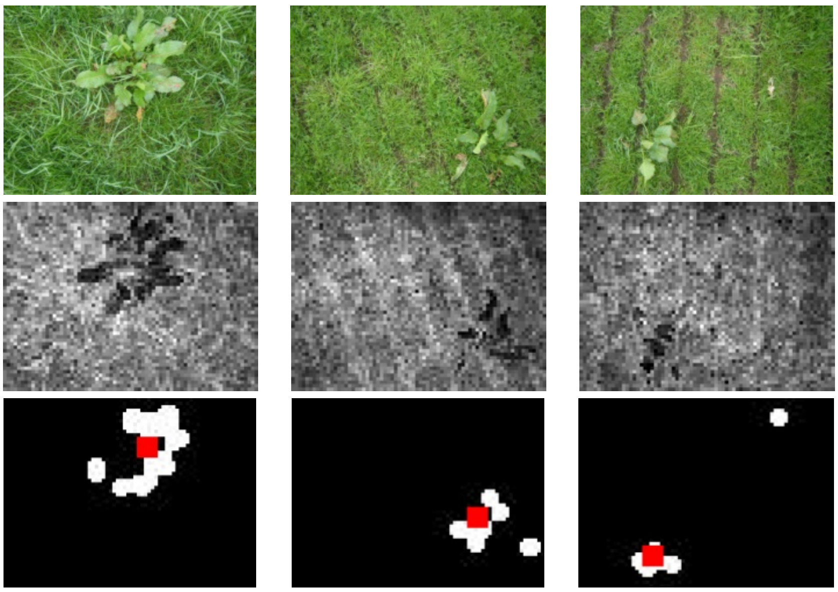


Figure 6. Detection of Rumex obtusifolius in grassland (results shown for three different weeds). Top row shows the original colour image. The area viewed is approx. 1.46 m x 1.09 m. The middle row shows images in which each pixel represents 1 cm^2 of the original image and where the brightness of the pixel represents the total power of the Fourier spectrum of the tile. After thresholding and morphological processing, the images shown in the bottom row are obtained. Here, the area covered by broad-leaved dock is shown in white; the red square indicates the center of the detected weed.

[1] This part of the work has now been expanded and published as follows: Van Evert, F.K., G. Polder, G.W.A.M. Van der Heijden, C. Kempenaar, and L.A.P. Lotz. 2009. Real-time, vision-based detection of *Rumex obtusifolius* L. in grassland. Weed Research 49(2):164-174.

We mounted a second webcam (PVC740k, Philips, The Netherlands) at a height of approx 1 m so that the pixel size correspond to the pixel size of the optimal parameter setting.

Once detected, weeds need to be controlled. To illustrate the possibility of mechanical control, we created a Rumex drill that was inspired by the WUZI (Anonymous, 2006b). The drill consisted of a kitchen mixer hook driven by a continuous rotation servo. The hook was attached to a Meccano boom that could be raised and lowered by a regular servo. We programmed the robot such that whenever a dock plant is detected with the camera (mounted in front), it continues to drive until the drill is located over the weed; stops; lowers the boom; "destroys" the plant by activating the drill; raises the boom, and continues searching weeds.

3. Results and discussion

3.1. Hardware

The encoder and the speed control software functioned well. Robot speed was always very close to set point speed, regardless of battery charge and terrain roughness.

Gyroscope readings were reliable. The drift was approx. 1 degree minute^{-1} and did not affect robot performance.

3.2. Software

Replacing the poll-loop for image acquisition with an interrupt-driven loop reduced processor usage from almost 100% to about 10%.

3.3. Flag

During practice runs toward a one-meter pole with an A4-sized piece of red paper, the robot performed flawlessly. The same was true for practice runs on the contest field on the day before the contest. Disappointingly, in the contest the robot headed toward spectators wearing red clothing. A contributing factor to this failure was the fact that the camera was placed close to the ground and thus had to look up to see the flag, increasing the chance of seeing red in the audience

3.4. Row-following

Detection of rows based on color segmentation was as robust as it had been in 2005 and 2004. Detection of rows based on texture was limited by the resolution of only 30*40 pixels of the image after tiles had been formed. This meant that the location and direction of rows could only be determined with less precision and reliability. It was possible to navigate the robot with this info, but it resulted in less smooth operation.

As it turned out, there was no maize at FRE2006. Instead, the rows consisted of stunted wheat plants in heading, while the ground between the rows was covered by very short and patchy wheat stubble. We could obtain good discrimination between the blue-ish wheat and the yellow-sand background. Thus, we decided to use color segmentation during the contest and navigation was accurate.

With the gyroscope giving accurate reliable readings, the headland turns went well. However, during some turns, two rows were skipped instead of the desired one row.

3.5. Golf-balls

The robot detected 9 of the 10 golf balls that were placed in its path.

3.6. Speed race

The robot repeatedly crashed into the rows and had to be put back by hand.

3.7. Hole detection

The search pattern was executed well enough, but the robot did not detect the hole when it drove over it.

3.8. Freestyle: detection of Rumex in grassland

We simulated a *Rumex obtusifolius* plant by placing several dandelion leaves on the same lawn that had earlier been used for hole detection. We received a solid round of applause when the robot detected the weed, lowered the boom, and simulated mechanical control of the weed.

Of note is the fact that we use the total power of the FFT spectrum, rather than the power of a specific range of frequencies. It seems that our cheap camera (a webcam with low resolution and poor optics) acts as a low-pass filter that removes all grass-related signals. Consequently, all remaining power in the spectrum is related to the presence of coarse elements.

4. Discussion and conclusions

We added the new functionality of texture-based image analysis, which was used both for recognizing rows and for detecting weeds.

In addition to the above, we were able to improve the robot: speed is regulated within narrow limits, and better electrical circuitry allowed the gyroscope to perform well.

Unfortunately, some problems remain. The majority of processing, including all image processing, is done on a Windows-based computer. Log files indicate that Windows at infrequent and random moments becomes unresponsive for up to several seconds. When this happens, the robot veers off course and crashes into the crop row. We assume that Windows becomes unresponsive when it reorganizes its virtual memory. It may be possible to address this problem by switching to a multi-threaded programming approach in which time-critical processing takes place in a high-priority thread.

Overall we must conclude that neither we nor the other teams are making satisfactory progresses in row following. It seems we're lacking some crucial element. For our robot, this might be that we are navigating solely on the basis of the most recent image. In doing so, we fail to use the information from the previous images. We assume that we will only be able to make significant progress when we find a way that will allow us to integrate information about the previous position of the robot with information from the camera, with information from the odometer, and with information from the gyroscope.

Acknowledgements

We acknowledge the help of Hans Jansen with designing the power supply circuitry.

References

Ahmad, U., and N. Kondo. 1997. Weed detection in lawn field. Available online at http://mama.agr.okayama-u.ac.jp/lase/weed.html. Verified 9 April 2008. [Online] http://mama.agr.okayama-u.ac.jp/lase/weed.html (verified 27 november 2005).

Anonymous. 2006a. Wuzi zur Ampferregulierung. [Online] http://www.bioland.de/nrw/service/wuzi.html (verified 2006-09-14).

Anonymous. 2006b. Wuzi zur Ampferregulierung. Available online at http://www.bioland.de/nrw/service/wuzi.html. Verified 23 September 2008. [Online] http://www.bioland.de/nrw/service/wuzi.html (verified 2008-07-20).

Åstrand, B., and A.J. Baerveldt. 2003. A mobile robot for mechanical weed control. International Sugar Journal 105:89-95.

Gebhardt, S., and W. Kühbauch. 2006. Automatische bildanalytische Klassifikation von *Rumex obtusifolius* in gemischten Grünlandbeständen. Journal of Plant Diseases and Protection - Zeitschrift für Pflanzenkrankheiten und Pflanzenschutz Special Issue XX:189-195.

Hough, P.V.C. 1962. Method and means for recognizing complex patterns. USA Patent 3069654 1962.

Tillett, N.D., T. Hague, and J.A. Marchant. 1998. A robotic system for plant-scale husbandry. Journal of Agricultural Engineering Research 69:169-178.

Van Evert, F., A. Lamaker, K. Van Evert, G. Van der Heijden, A. De Jong, J. Lamaker, B. Lotz, J. Neeteson, and T. Schut. 2004. Een grote toekomst voor kleine robots in open teelten. Agro-Informatica 17:21-26.

Van Evert, F.K., G.W.A.M. Van der Heijden, L.A.P. Lotz, G. Polder, A. Lamaker, A. De Jong, M.C. Kuyper, E.J.K. Groendijk, J. Neeteson, J. , and T. Van der Zalm. 2006. A mobile field robot with vision-based detection of volunteer potato plants in a corn crop. Weed Technology 20:853-861.

Van Middelkoop, J., M. De Visser, and H. Schilder. 2005. Beheersing van ridderzuring op biologisch grasland in het project Bioveem. [Control of broad-leaved dock in organic grassland in the "Bioveem" project]. Animal Sciences Group Report 14. Animal Sciences Group, Lelystad, The Netherlands.

VXL. 2004. VXL: C++ libraries for computer vision [Online] http://vxl.sourceforge.net/ (verified 21 October 2004).

WURking - An autonomous robot for agriculture applications

J.W. Hofstee[1], R. Jansen[1], A.T. Nieuwenhuizen[1], M.H.A.M. Govers[1], S.K. Blaauw[1], J.D. Stigter[2]; L.G. van Willigenburg[2], T. Bakker[2], C.J. van Asselt[2], G. van Straten[2], S.L. Speetjens[2]

[1]Farm Technology Group
Wageningen University
Bornsesteeg 59, 6708 PD Wageningen
The Netherlands

[2]Systems and Control Group,
Wageningen University
Bornsesteeg 59, 6708 PD Wageningen
The Netherlands

Abstract

Autonomous robots for agricultural practices will become reality soon. These mobile robots could take over regular task such as weeding. Other new applications can be thinking of such as plant specific spraying and the release of infochemicals for attracting predators of pests. Therefore the Farm Technology Group and the Systems and Control Group of Wageningen University decided to develop a robot for agriculture applications. This field robot has to navigate and steer autonomously on the field. This task is performed using a BasicATOM micro-controller coupled to a PC based system. Data from ultrasonic and infrared sensors, a camera, a gyroscope and a guidance rail are used to control the robot. A kinematic vehicle model is used to calculate the setpoints for the motor controller, based on the values of the different sensors.

1. Introduction

Robotic systems for agricultural applications are not new. Tillet *et al.* (1997) already discussed a robot for plant scale husbandry. Robots in agriculture can be used for several applications. Examples are robotic weed control (Astrand and Baerveldt, 2002), mapping in-field variability (Godwin and Miller, 2003) or detection of volunteer potatoes (Evert *et al*., 2006).

Wageningen University started in 2003 with the organization of the Wageningen UR Field Robot Event (Van Straten, 2004). An important reason is to stimulate the further development of robots in agriculture, The organization of the event in 2006 by the University of Hohenheim made initiated the joint development of a small robot for field applications by the System and Control Group and the Farm Technology Group. There were two objectives: (1) to have a robot to participate in the 4th Field Robot in Germany and (2) to have universal platform to be used for further research and education on small robots for agricultural applications. The robot had to be able to fulfill the following tasks: (1) drive through maize rows and count dandelions (yellow golf balls), (2) detect holes in a grass field of 10 x 10 m, (3) detect a corner flag and drive to it, (4) drive as fast as possible through maize rows, and (5) give a good performance in the free style session.

2. The field robot

2.1. Overall design

The field robot (Figure 1) is based on an aluminum frame and three independent wheel units. Two wheel units are placed in the front and one wheel unit in the rear of the frame. Each wheel unit is equipped with two DC motors, one motor for steering the unit and one motor for driving the wheel. The robot is equipped with six ultrasonic and two infrared sensors for row detection. Furthermore the robot is able to follow a row based on a guidance rail with two potentiometers. Additionally a camera is attached to the robot for detection of golf balls, lines, and the corner flag. A gyroscope is used to determine the orientation of the robot.

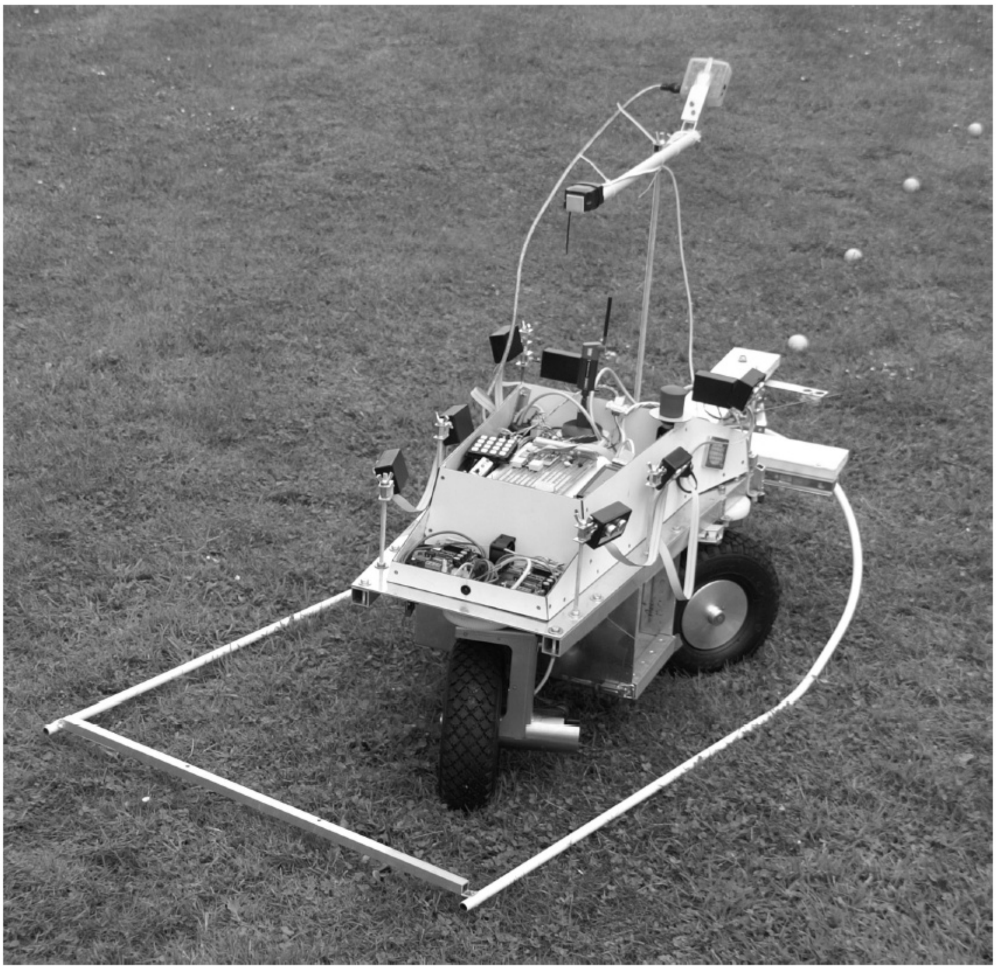


Figure 1: WURking.

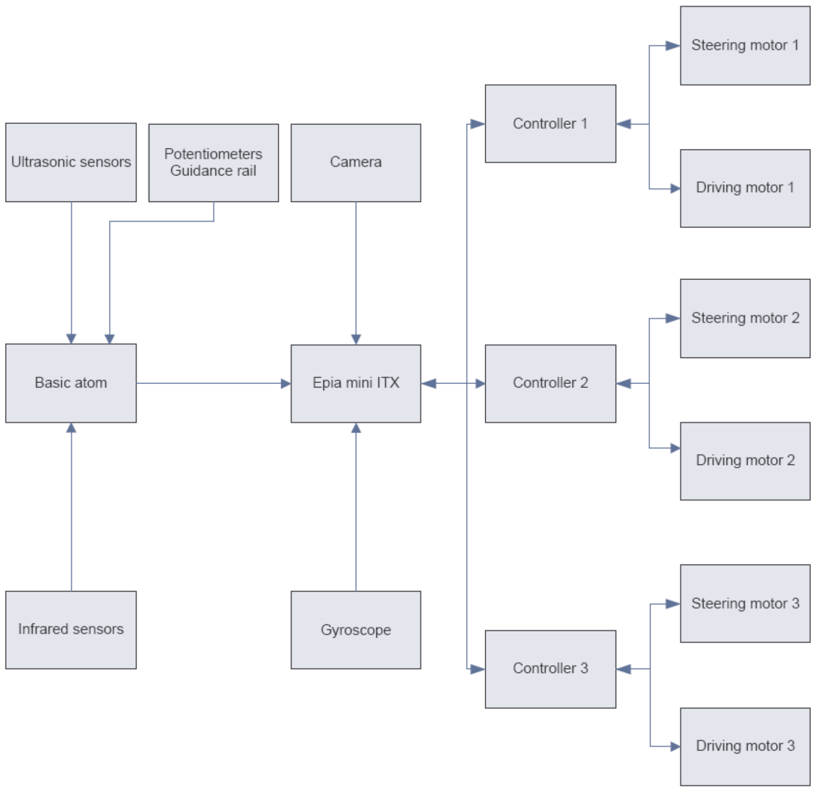


Figure 2: Overview of the different components of WURking.

The electronic signals from the potentiometers, the ultrasonic and the infrared sensors are processed by a microcontroller. These signals are, together with camera and gyroscope signals, processed by high-level software. Part of the high level software is a kinematic vehicle model. This vehicle model calculates the control signal for each motor based on the output signals of all different sensors.

An overview of components of the data acquisition and the motor control system is given in Figure 2.

2.2. Platform

The platform consists of an aluminum frame, a cover to protect the electronics, a battery pack, several sensors and three independent wheel units. The wheelbase is 500 mm and the track width between the front wheels is 320 mm. The total width of the platform is 400 mm and the total length is 700 mm. Each wheel unit is able to steer left (+135$_o$) and right (-135$_o$). The individual wheel units are independently driven by a DC motor connected to a planetary gear head. A battery pack is attached to power the motors, the personal computer, the controllers and the sensors. This battery pack is located in between the wheel units to realize a low centre of gravitation. The clearance of the platform is 120 mm. The weight of the platform including battery pack is 39 kg.

2.2.1. Wheel units

The wheel units (Figure 3) are designed in cooperation with the Mechatronics Department of the Kverneland Group in Nieuw Vennep, The Netherlands. Each wheel unit is equipped with two DC motors. The wheel is powered by a single 150 W motor at 24 volts (Maxon Precision Motors, brushed DC motor, model RE40). The maximal torque delivered is 181 mNm at 7580 rpm. This motor is connected to a planetary gearhead with a reduction of 15:1 and a maximum efficiency of 83%. (Maxon Precion Motors, model GP52C). The drive motor is equipped with an encoder to measure the speed in counts per turn (Maxon Precision Motors, encoder, model HEDS 5540). Steering is realised by a 20 W motor at 24 volts (Maxon Precision Motors, DC motor, model RE25) connected to a planetary gearhead with a reduction of 66:1 and a maximum efficiency of 70% (Maxon Precision Motors, model GP32C). The wheel units are equipped with conventional tubed tyres (Ø 250 mm, 80 mm). During the FRE 2006 the tubes were inflated to approximately 3 bar for optimal traction. The maximum steering velocity of the unit is approximately 115 deg/sec. The weight of the wheel unit including the motors is 4.1 kg.

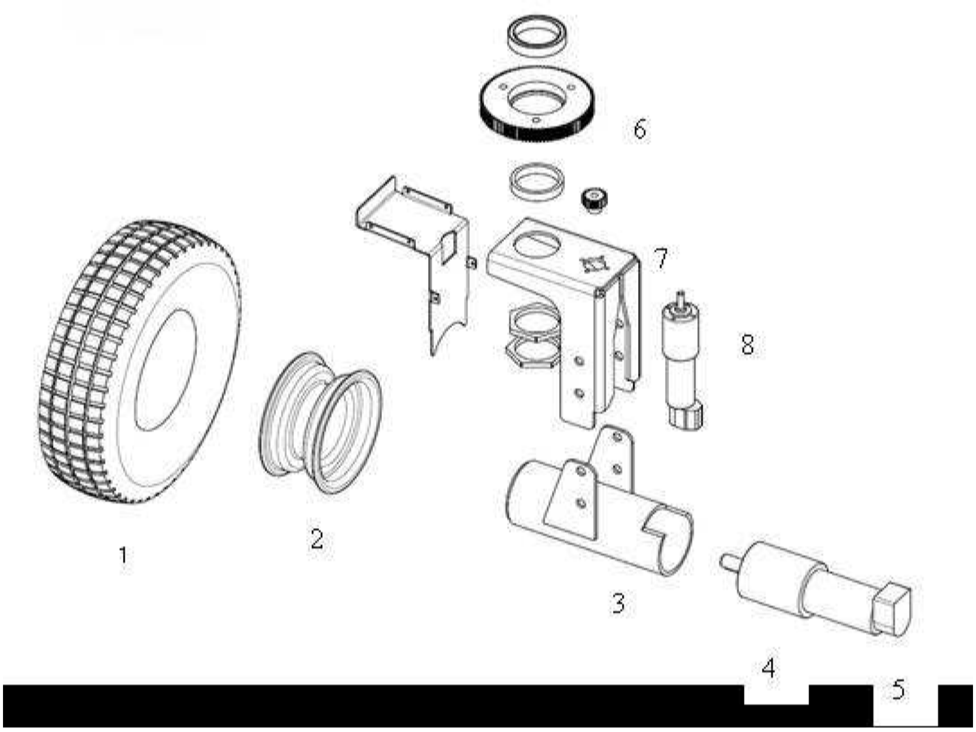


Figure 3: Details of the wheel unit: (1) Tyre, (2) wheel rim, (3) frame for drive motor, (4) drive motor, (5) encoder, (6) steer transmission, (7) frame for steering motor and (8) steer motor (Drawing: Kverneland Group).

2.2.2. Motor controllers

Each wheel unit is controlled by a Motion Mind DC motor controller. Each controller is capable of controlling two brushed motors. In this case one motor for driving and one for steering.

2.2.3. Battery pack

The platform is equipped with a battery pack containing three batteries. Two 12V, 7Ah batteries are used to power the driving and steering motors. Additionally there is one 12V, 12 Ah battery to power the PC platform, controllers and the sensors. The total weight of this battery pack is 10 kg. This battery pack is mounted to the platform in such a way that it can be easily exchanged with a spare battery pack for continuous operation.

2.3. Sensors

2.3.1. Ultrasonic

The robot has six Devantech SRF08 ultrasonic sensors, three at each side. These sensors measure the distance from the robot to the crop row. The range of the ultrasonic sensors is 3 cm to 6 m; the frequency is 40 kHz. The ultrasonic sensors are connected to the I_2C bus of the BasicATOM microcontroller.

2.3.2. Infrared

The robot has two Sharp GP2D12 infrared sensors. These are until now not used.

2.3.3. Guidance rail

The guidance rail consists of a frame around the robot. The frame is connected to the robot by two potentiometers. One potentiometer measures the rotation of the robot with respect to the guidance rail and the second potentiometer measures the lateral displacement of the robot within the frame. This frame slides over the soil and follows the crop rows. The steer signals for the robot are based on the rotation and lateral displacement of the robot with respect to the frame.

2.3.4. Camera

The camera is a Unibrain Fire-i firewire camera. The camera has a ¼" CCD (659 x 494 pixels); the pixel size is 5.6 µm in both horizontal and vertical direction. The frame rate is up to 30 frames per second (uncompressed VGA picture).

2.3.5. Gyroscope

The robot is also equipped with a XSens MT9-B gyroscope. This gyroscope has a 3D compass, 3D accelerometer and 3D gyro’s and yields by integration very precise values for yaw, roll, and pitch. The angular resolution is 0.05º, the static accuracy is <1º, and the dynamic accuracy is 3º RMS (XSens, 2006).

2.4. Controllers

2.4.1. Basic ATOM

Part of the data acquisition is realised with a BasicATOM40 microcontroller. Inputs for the microcontroller are the two potentiometers, the two infrared sensors and the six ultrasonic sensors. The microcontroller processes the raw sensor signals and creates a message with the calibrated values which is send to high level control program.

2.4.2. PC

WURking has a VIA EPIA SP13000 PC for high level control. This is a low power compact motherboard with built in CPU, graphics, audio and network. The PC has 512 MB RAM and a 40GB hard disk with a Windows XP operating system. The PC also has a WiFi connection for remote control and monitoring purposes.

2.5. Vehicle models

Navigation and control design issues were addressed in an advanced model-based design using a kinematic mathematical model of the three-wheel vehicle. The kinematic model is in state-space format, meaning that vehicle position in the x-y-plane, vehicle orientation with respect to the positive x-axis and the corresponding velocities build up the so-called state-vector of the system. Our dynamic kinematic model describes the rate of change of these variables as a function of time so that, once an strategy has been chosen (meaning that setpoints for wheel and angle velocity-controllers have been decided upon), the model can be integrated forwards in time as to arrive at a prediction of the new state of the system at the prediction time. Subsequently, an estimate of the state vector is constructed using odometry and angle measurements from all three wheels plus, in addition, a measurement of the vehicle orientation. These measurements are processed with the help of a so-called Kalman filter.

The state estimate, generated by the Kalman filter, can be used for several goals, e.g. calculation of the setpoint errors and the control actions that follow from these setpoint errors or as reliable estimate of the position and velocity of the vehicle that can be used for higher level decision making in the control software (e.g. protocol choice as to decide what to do in case the end of the row has been reached).

2.6. High level control

The high level control of the robot is realized by a LabVIEW program. The program consists of several processes that run independently from each other. Each process is represented by a VI (Virtual Instrument). and several sub VI's and realizes a specific task. There are VI's for initialization, motor control, the kinematic vehicle model, camera control, communication with the BasicATOM, communication with the gyroscope, and sensor fusion. A state machine controls the activation and de-activation of processes. The VI's exchange data with each other via global variables.

3. Field tests

It was unfortunately not possible to finish WURking in time to have it participating in the 2006 Field Robot Event.

Nevertheless, before and after the event some tests are done. Both during development and tests in the field some problems occurred with the gear heads of the wheel units. The high forces that can occur during start up can exceed the limits of the gear head, resulting in damage inside the gear heads.

4. Discussion

A major improvement of WURking compared to many other small robots is the use of a kinematic vehicle model for control of the robot. With this model it is possible to realize a very efficient and high quality control of the robot.

The high level control based on several independent processes is a good concept. A drawback is that it is rather complex and debugging is not easy because of the independency of the processes.

One of the initial requirements for the robot was that it should perform well in the speed race. This requirement resulted in a combination of wheel motor and gear head that is not able to take up the large forces and moments that arise during start of the robot when the velocity is low.

5. Conclusions

WURking has a very good underlying concept and is a solid basis for further development of a small robot for in field application. Some more design and development time is necessary to complete the robot. Some redesign of the wheel units is also necessary.

Acknowledgments

The authors would also like to acknowledge Mechatronics Department of the Kverneland in Nieuw Vennep for the cooperation in the design of the wheel units, XSens Motion Technologies for providing the gyroscope, Priva for some financial support and the colleagues of the other robots teams of Wageningen UR for the cooperation and discussions during the pre field robot event era...

References

Astrand, B. and Baerveldt, A. J. (2002). "An Agricultural Mobile Robot with Vision-Based Perception for Mechanical Weed Control." Autonomous Robots 13: 21-35.

Evert, F. van, Heijden, G. W. A. M. van der, Lotz, L. A. P., Polder, G., Lamaker, A., Jong, A. de, Kuyper, M. C., Groendijk, E. J. K., Neeteson, J. J. and Zalm, T. van der (2006). "A mobile field robot with vision-based detection of volunteer potato plants in a corn crop." Weed Technology (in press).

Godwin, R. J. and Miller, P. C. H. (2003). "A Review of the Technologies for Mapping Within-field Variability." Biosystems Engineering 84: 393-407.

Straten, G. van (2004). "Field Robot Event, Wageningen 5-6 June 2003." Computers and Electronics in Agriculture 42: 51-58.

Tillett, N. D., Hague, T. and Marchant, J. A. (1997). "A Robotic System for Plant-Scale Husbandry." Journal of Agricultural Engineering Research 69(2): 169-178.

XSens (2006) http://www.xsens.com/download/MT9_brochure.pdf (18-08-2006)