TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY

HELSINGIN YLIOPISTO

The Development of an Autonomous Robot for Outdoor Conditions

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

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

[2] Department of Agrotechnology
Helsinki University

Abstract

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

Keywords

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

Introduction

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

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

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

Hardware construction

Platform

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

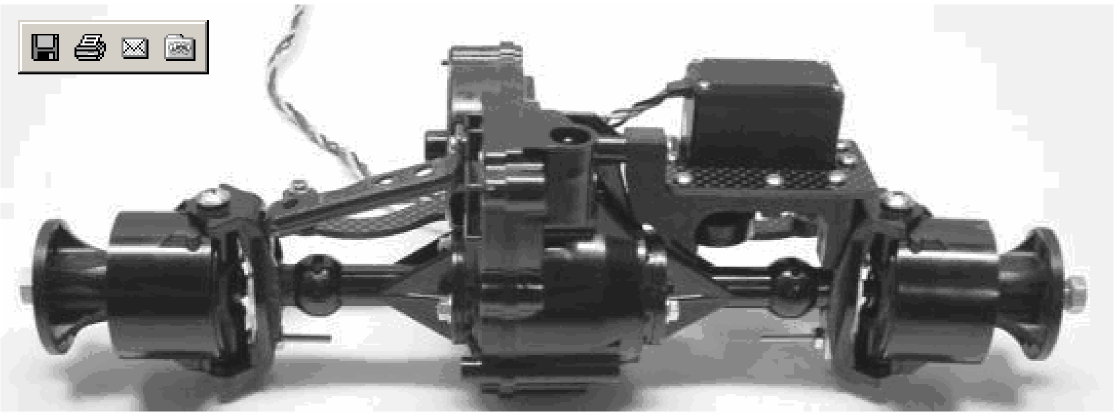


Figure 1 Power transmission

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

Motors and gearbox

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

Steering servos and motor power controllers

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

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

Instrumentation and electronics

Web-cam

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

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

Figure 2 Logitech QuickCam Pro 4000.

Ultrasonic sensors

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

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

Infrared sensors

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

Electronic compass

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

Hall sensors

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

Power supply

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

Processing equipment

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

Laptop

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

Microcontroller

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

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

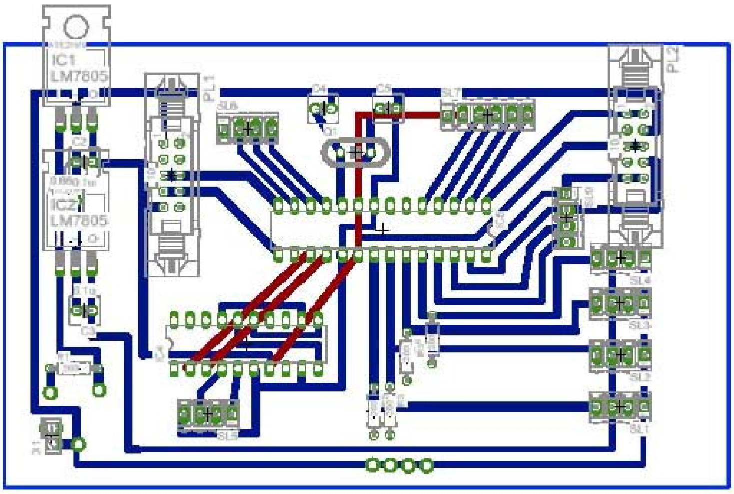


Figure 3 Layout model of the microcontroller board

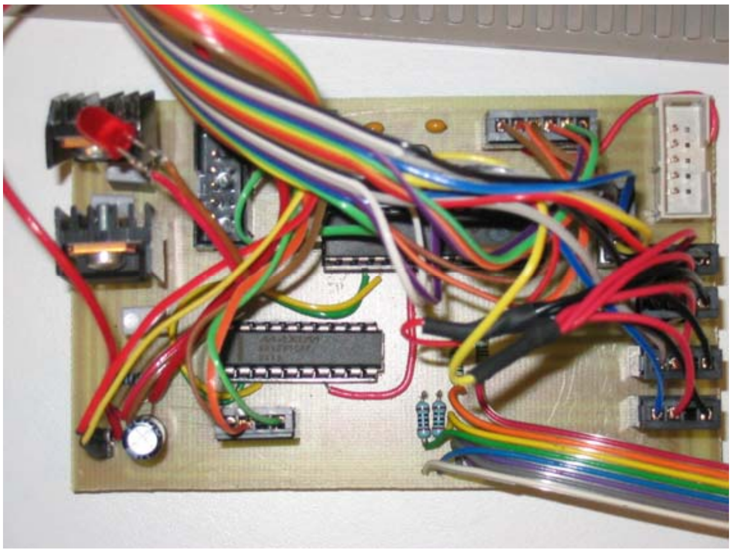


Figure 4 Picture of the actual constructed board

Additional hardware (Freestyle)

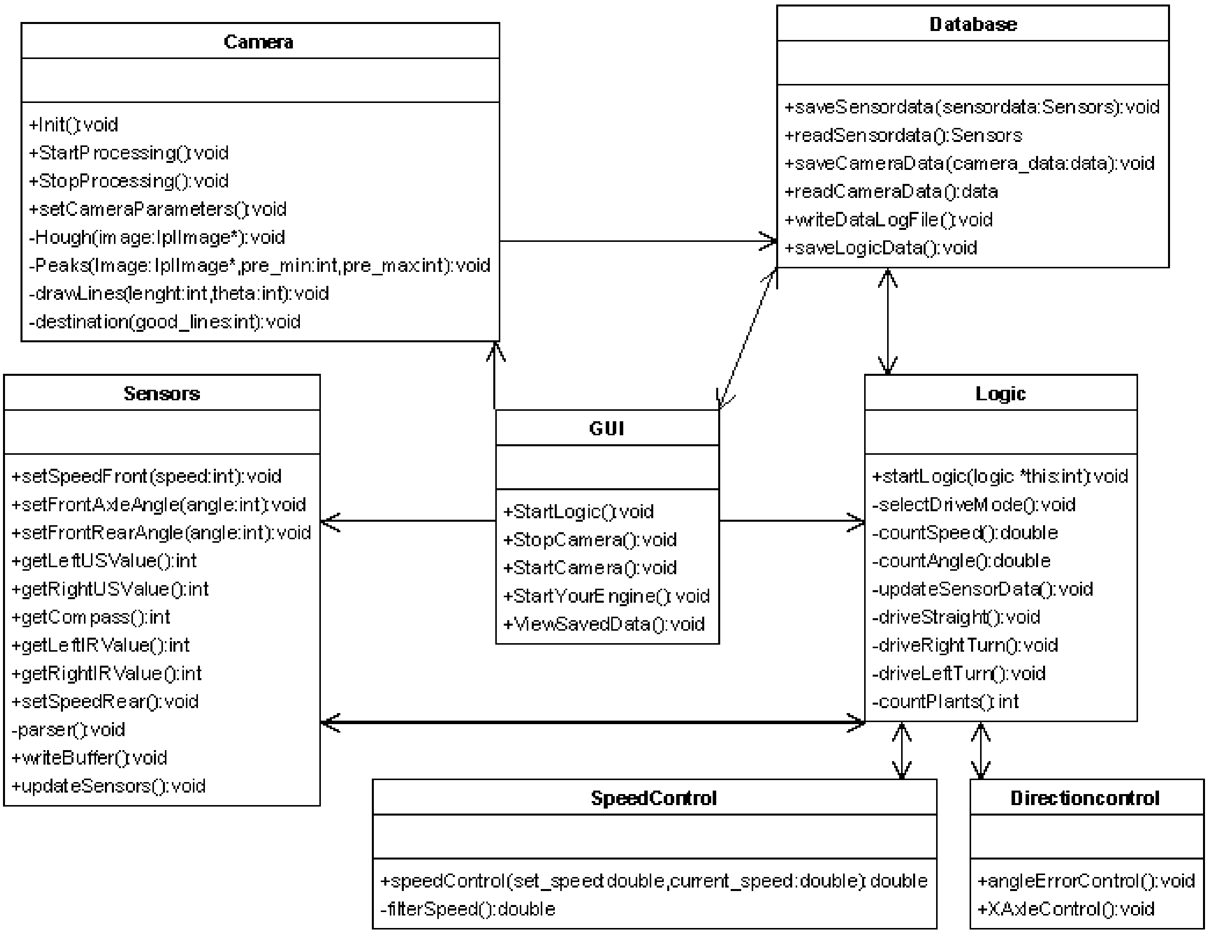
The primary idea was to develop a system for measuring the soil density (cone-index) value but in absence of time the calibration and testing remained undone. The principle was to penetrate the surface with a probe and to measure the force.

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

PC Software architecture

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

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



Created with Poseidon for UML Community Edition. Not for Commercial Use.

Figure 5 Class diagram

User interface

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

Camera

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

Database

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

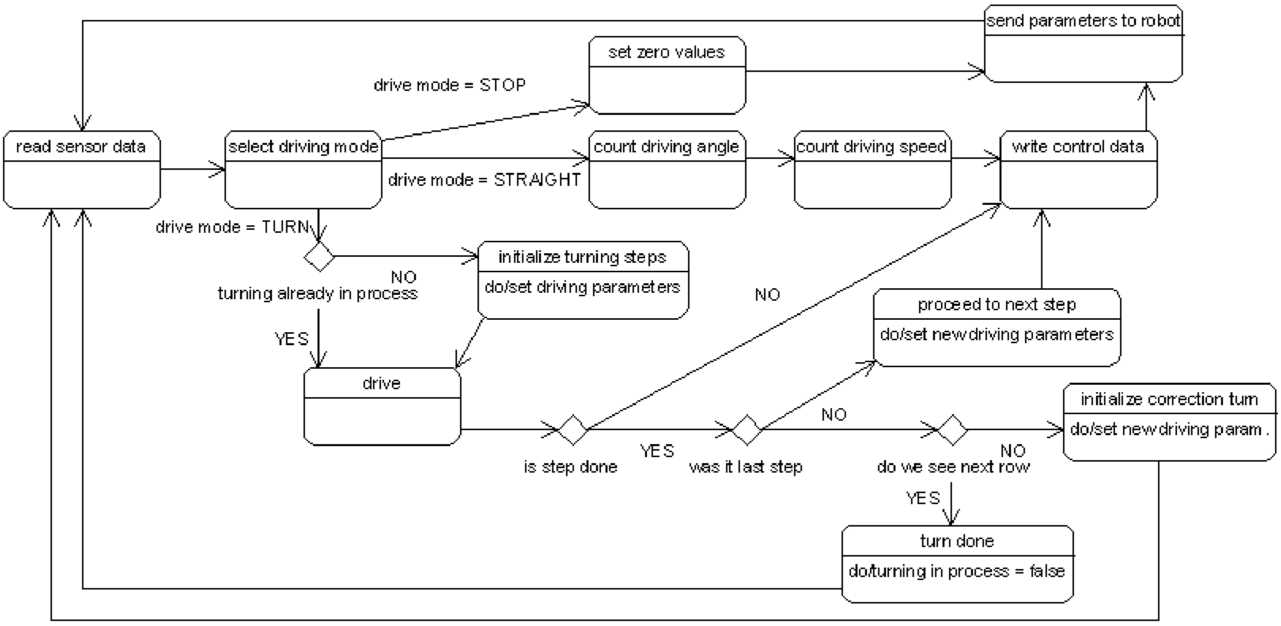
Interface to sensors

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

Program logic

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

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



Created with Poseidon for UML Community Edition. Not for Commercial Use.

Figure 6 Logic's state diagram

Speed controller

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

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

Direction controller

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

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

Program sequences and threads

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

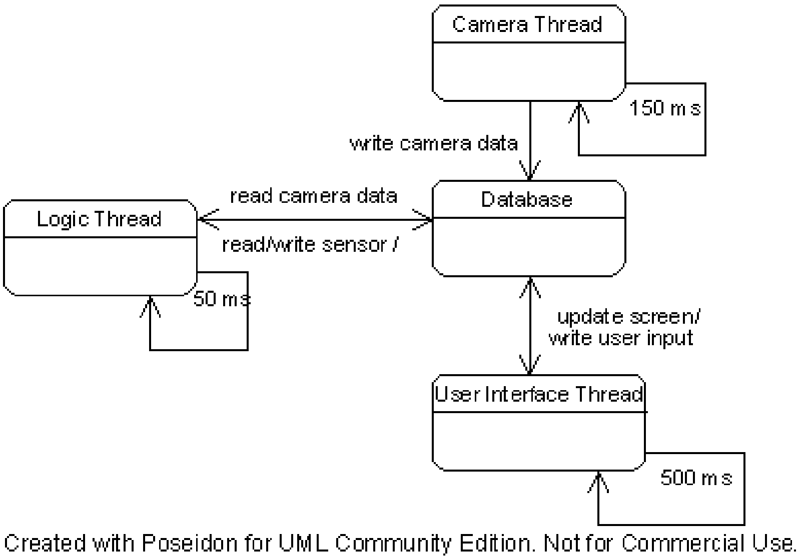


Figure 7 Thread state diagram

Plant counting

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

Motor Control with Microcontroller

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

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

Machine vision

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

Approaches

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

Preprocessing

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

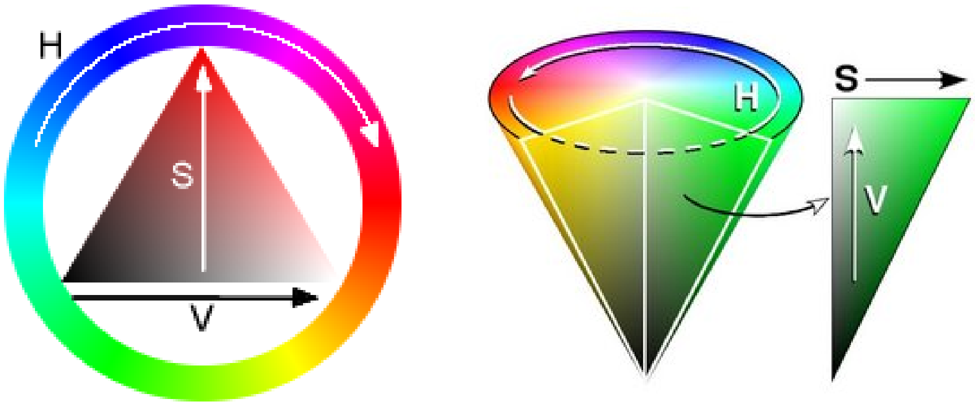


Figure 8 HSV color space

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

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

Figure 9 Color segmentation

Method 0: Moments

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

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

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

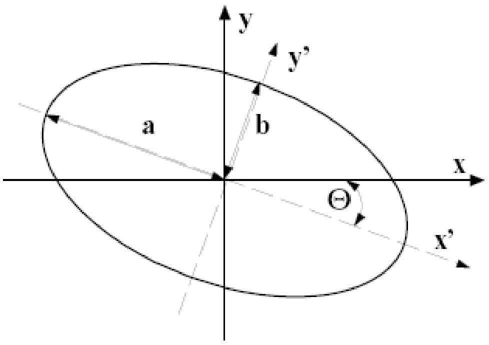


Figure 10 Orientation angle from moments.

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

Method 1: Slicing

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

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

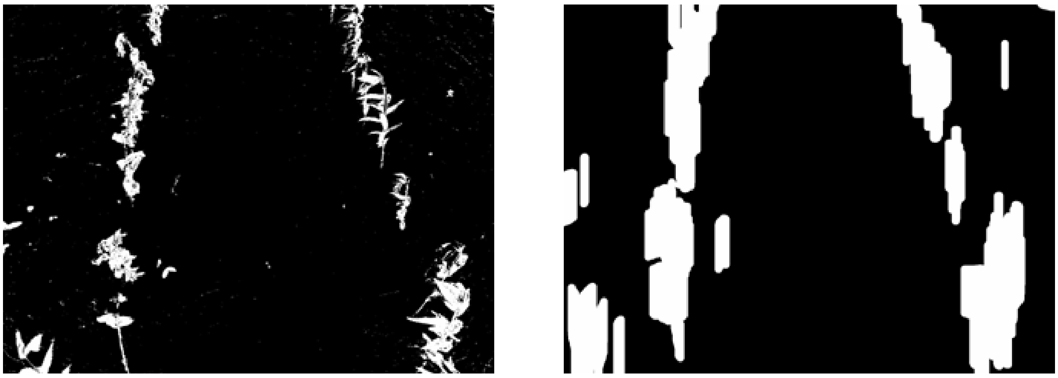


Figure 11 Morphological filtering

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

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

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

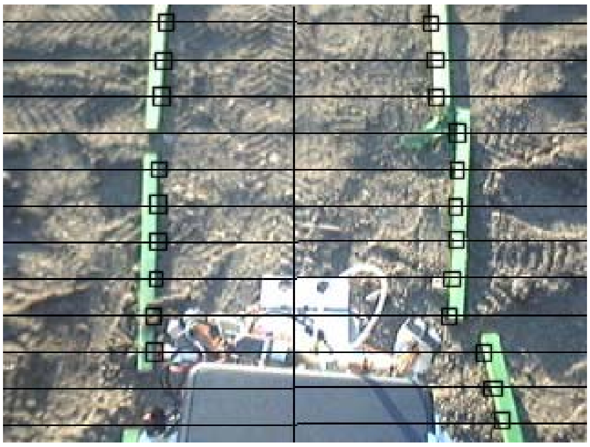


Figure 12 Slicing method principle. (mock-up)

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

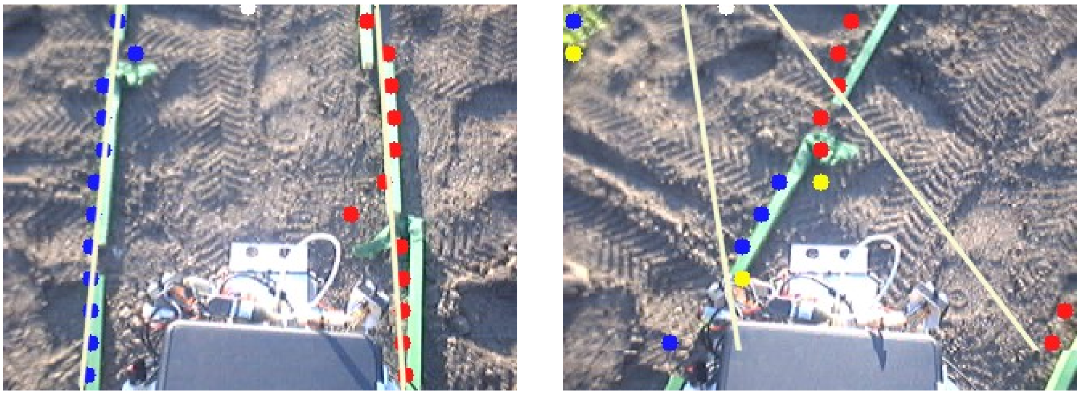


Figure 13 Good and bad cases with the slicing method. (actual algorithm)

Method 2: Hough Transform

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

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

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

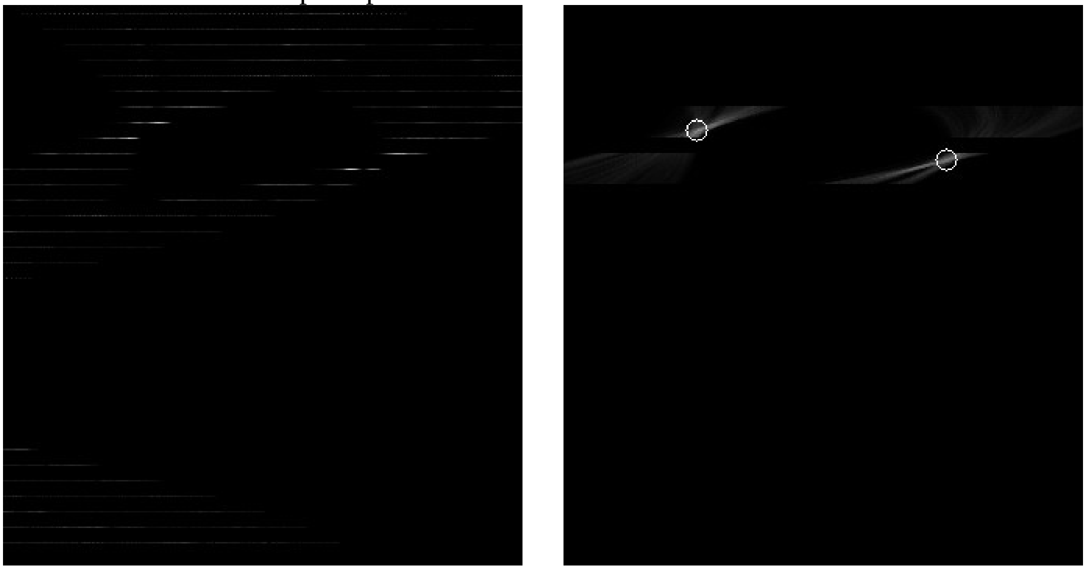


Figure 14 Two-pass Hough transform illustrated. Found peaks are marked with circles.

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

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

Figure 15 Results with the Hough method

End of Row Detection

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

Conclusions

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

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

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

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

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

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

References

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

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

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

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

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

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

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

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

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

hp®

invent

koneviesti

UNIVERSITY OF HELSINKI

MTT
Agrifood Research Finland

HELSINKI UNIVERSITY OF TECHNOLOGY
Automation and Systems

KEMIRA
GrowHow®
partnership • knowledge • solutions

VALTRA