



Autonomous & Dynamic Robotics

Herbinator Design Report 2010

Presented to the 7th Annual Robotic Lawnmower Competition

Club Capra
École de technologie supérieure
1100 Notre-Dame Ouest
Montréal, Québec (Canada)
H3C 1K3
(514) 396-8800 #7779
capra@ens.etsmtl.ca

I certify that the engineering design in the vehicle by the current student team has been significant and equivalent to what might be awarded credit in a senior design course.

Prof. François Coallier, Eng. Ph. D.

Chairman, Department of Software and IT Engineering

Faculty Advisor, Capra

École de technologie supérieure (ETS)

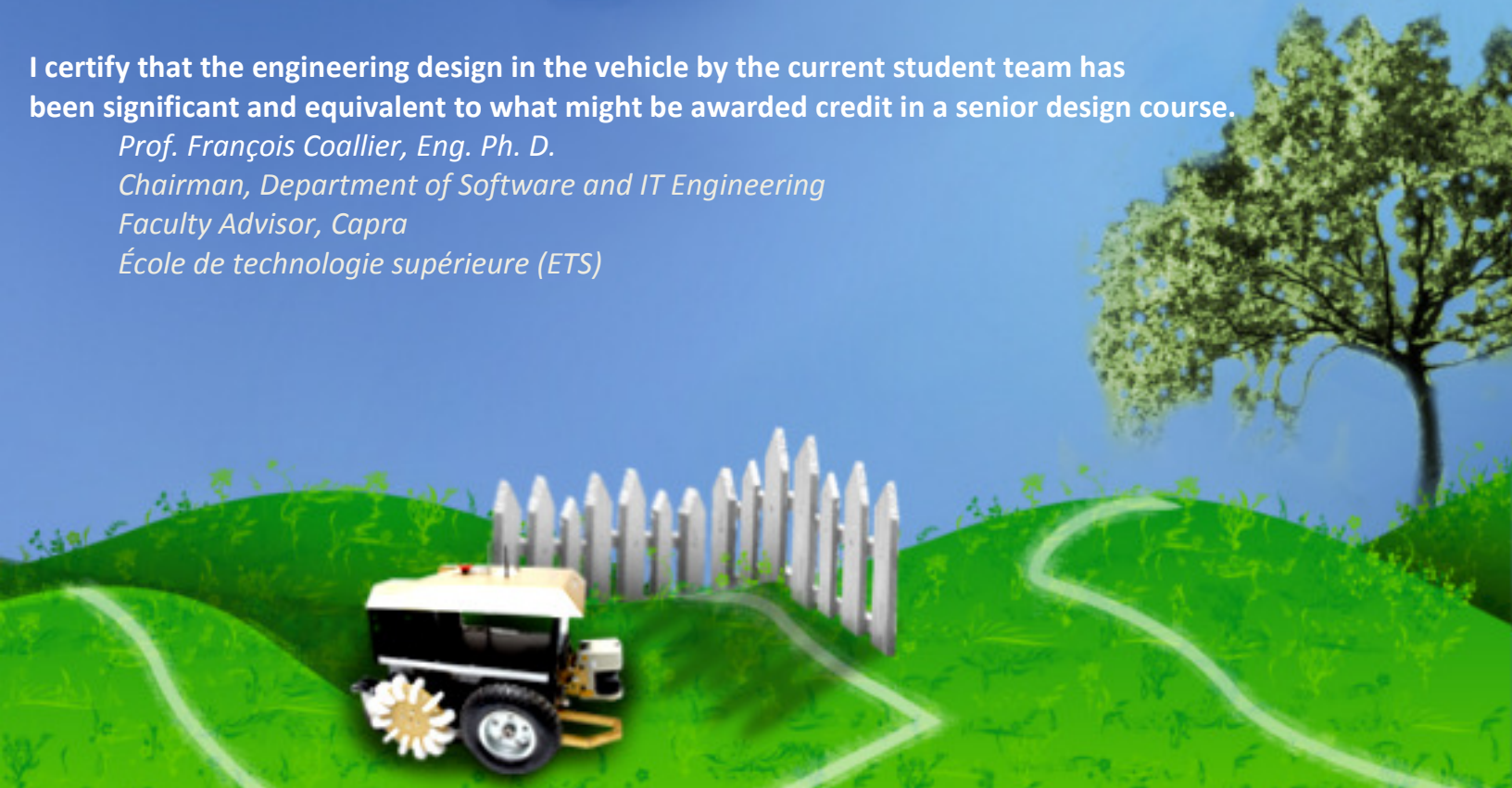


Table of content

Introduction	3
Team overview	3
Recruiting	4
Projects	4
Vehicle overview.....	5
Mechanical design	6
Electronics	9
Software	13
Conclusion	17

Table of figures

Figure 1 - Team layout	4
Figure 2 – Cost analysis.....	6
Figure 3 - Herbinator frame.....	6
Figure 4 - Herbinator propulsion system.....	7
Figure 5 - Herbinator anodizing color	7
Figure 6 - Herbinator drawer	8
Figure 7 - Herbinator bumper	8
Figure 8 - Communications with the controller board	10
Figure 9 - Communications between the sensors and the computer	12
Figure 10 – 2D Simulator	17

Introduction

Today, we don't always have the time to do housework. With the progression of technology, we have more and more ways to facilitate our daily tasks to take advantage of freeing up our time effectively. That is why our society is trying to develop ways to make life easier, using technologies that are still growing. With competitions such as ION auto-mowing, the know-how of many students from everywhere is challenged to develop some innovative ideas for the welfare of the community.

For the third consecutive year, École de Technologie Supérieure is present with the robot Herbinator, which has been improved since last year. With our third place in the last edition, we were still disappointed given the circumstances: an electrical problem occurred on our second run which pushed us out of the competition. This year, we are confident we can offer a better performance than last year with all of the improvements made during the past year.

This document will include information on our team, and how every department (electrical, mechanics and software) worked hard on making it possible

Team overview

École de Technologie Supérieure (ÉTS) is an engineering university situated in Montreal, Quebec in Canada. This engineering school is filled by students from a technical college degree which allows them to get involve early on in different projects. At ÉTS, there are about twenty student clubs working on different technological projects in all fields of engineering.

Our club is separated in four main fields of competences. For the technical job, there are the electrical, mechanical and software teams. We also have a part of the team engaged in the management of the student club. Having a management team helps the project to grow efficiently. Each of the electrical, mechanical and software teams having a leader helps the tasks to be managed well also.

The students of the club are from six different engineering domains such as electrical, mechanical, software, information technologies, automated production and civil engineering. Each member is in the club on a voluntary basis, studying at the same time. Some people are contributing to the project more than others depending on their availability. Here is a figure of our team organization. Leaders worked on average 200hrs in the last years, and members on average 40-100hrs.

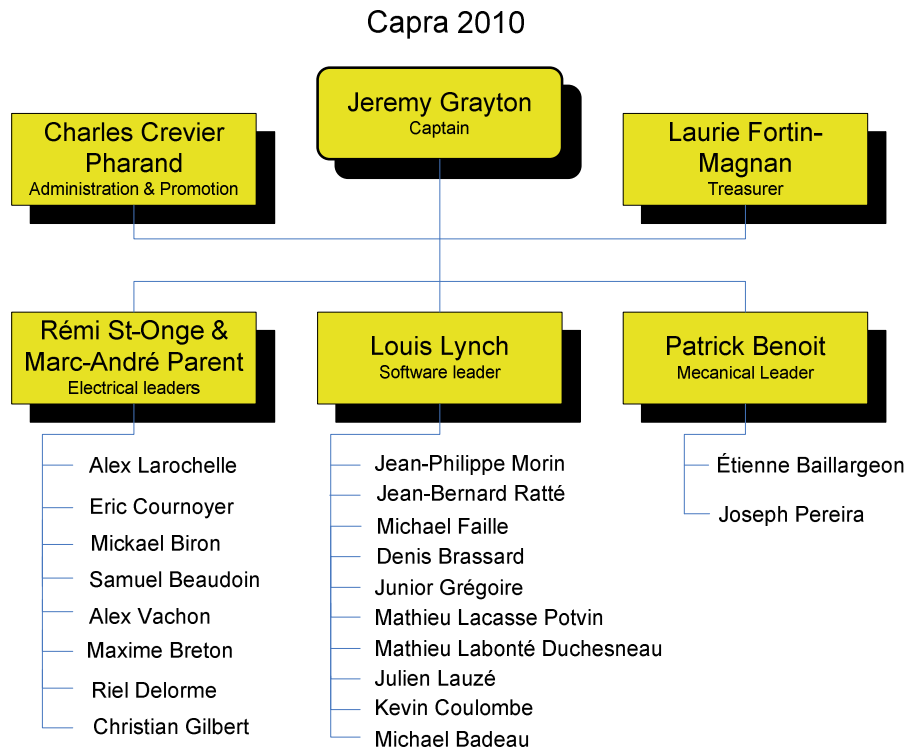


Figure 1 - Team layout

Recruiting

Although most students of the club spend an average of two to three years getting involved in our projects, it is very important to recruit new members each year to compensate for the departure of former members completing their studies. It is also absolutely necessary to transfer knowledge from the veterans of the club to the recruits before the end of their studies.

Projects

The student club has two main projects going on all year long. We have our robot RS3 which is participating to the IGVC competition in Michigan and our robot Herbinator participating to both ION and IGVC for the first time this year. Since our older robot, RS3, needed almost no more improvements mechanically, we will start building a new frame for a completely new robot after our competitions wishing to have this new project participating to the next year's editions.

We separated our job on our two main projects in tasks attributed to members of the team. Some tasks are accomplished by more than one student, others are done by only one member but we manage to get everyone involved with their time availability to a project to each member.

Developpement process

Tools

The task management is very important, as well as every team member participating in it. We need a tool that helps us manage our projects efficiently and get rid of complicated procedures. The team members are in constant rotation so it is important to have low learning curve because the whole project itself is complicated.

We experimented using Edgewall Trac, a forum, a FTP and SVN during the past year, however we want to centralize all our current tools to one point of access to improve maintainability and clarity. We therefore decided to change our project management web application to use Redmine. While the installation is much easier than Trac, it even supports a cross-project management which we believe is one of the key of our success.

This new tool brings also these several major advantages:

- Centralized authentication
- Display of differences between the revisions of the SVN
- Master projects and sub-projects managing
- Feeds, email notifications and news
- GANTT chart and calendar
- Easy customization of every fields and options for the whole application
- Simple and flexible permission access
- Wiki, Forum and file system management

There is also another feature we are looking forward to use, the Time tracking system. The problem is, we don't want it to bring conflicts and become unhealthy during competition. However, if well integrated, we believe it could help us estimate more precisely the time frame for a task, improve our planification on each project and to focus the whole team on the same goals.

Vehicle overview

Cost

Component	Sponsor by	Detail cost	Price paid
Batteries	Batterie Experts	\$400	\$160
Electronics	Labo Circuit	\$800	\$800
Encoders		\$340	\$275
Electric motors		\$1,400	\$100
Grass Trimmer		\$80	\$80
GPS	NovAtel	\$19,000	\$4,600
IMU	MicroStrain	\$1,500	\$0
Camera		\$500	\$500
Lawn mower		\$250	\$175
Mechanical		\$200	\$200
LCD screen		\$300	\$300
Omnistar service	Omnistar	\$2,500	\$0
Computer	Kontron	\$1,450	\$450
Range Finder		\$6,500	\$2,550
Structure	Palardy	\$600	\$200
Wheels		\$60	\$60
Total		\$35 880	\$10 450

Figure 2 – Cost analysis

Mechanical design

Herbinator is a four-wheel vehicle robot, with two pneumatic motor wheels at the front and two free swivel wheels at the back. The chassis is made of welded 1 inch aluminum 6061-T6 square tubing, and the aluminum 6061-T6 shell protect all internal components. The total weight of the robot is about 200 pounds, and overall the robot is 43 inches long, 30.5 inches wide and 24 inches high without the pole. The pole is almost 4 feet long which contain the computer screen, the GPS antenna the IMU and the camera with a cutting diameter of 18 inches.

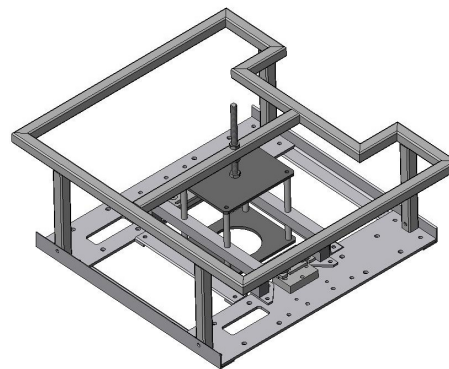


Figure 3 - Herbinator frame

Propulsion

Herbinator's propulsion system is composed of two wheelchair electrical motor. Each motor are connected to a sprocket. These sprockets are connected on a driveshaft of the front wheel. We choose chain and sprocket because this system is reliable and robust for outside environment.

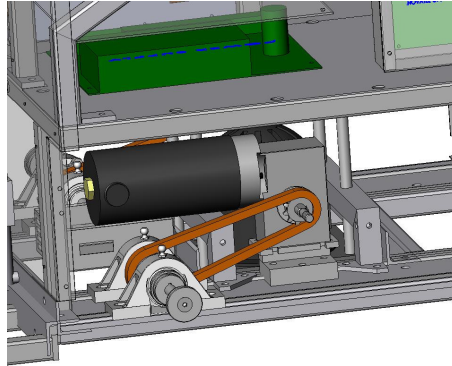


Figure 4 - Herbinator propulsion system

Lawn Mower

The mower is positioned on the lower parts of the robot and at the center. This is why most of the weight is located in lower parts of the robot. A 1hp motor is used for the mower. A safety was placed around the blade to ensure the security of people around.

Innovations

Anodization

All the 1/16 aluminum shell have been anodized to avoid short-circuits and protect it against scratches and marks. We have chosen two colors, black and gold for esthetic consideration.

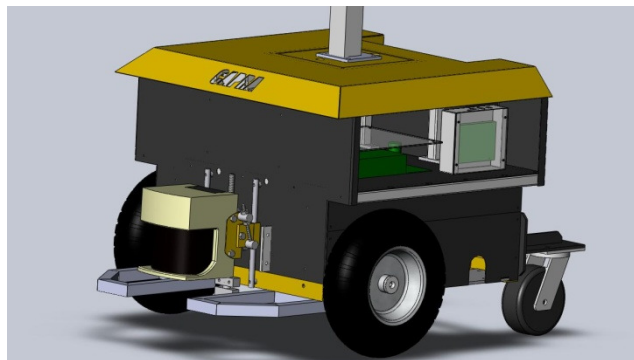


Figure 5 - Herbinator anodizing color

Pole

Because we decided to use new electrical components, we had built a new pole to support all these components (GPS, computer screen, camera and IMU). The 1½ inches square pole is placed in the middle of the robot and fixed with four bolts.

Drawer

Out of last year's experience, members found that the laser range finder needed the capacity to be adjustable in order to face different ground conditions. This is why the range finder holder allows us to adjust the height and the angle of the range finder. This holder lets us adjust the range finder to face different ground conditions. To make the best use of space, we built a drawer behind the range finder. This drawer was built to put a battery in order to supply the lawnmower motor. This way, a full battery is used for the rotation of the blade.

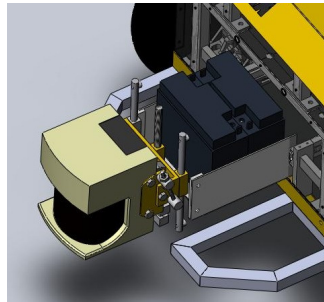


Figure 6 - Herbinator drawer

Power Supply case

The electrical team has decided to buy a brand new case to put the power supply. We made some modifications for the Weidmuller connector and fans. We had to mill some rectangular holes for these connector. Finally, we drilled holes in a pattern for the fan to let cool air enter in the power supply case.

Bumper

A bumper made of aluminum tubes has been created to protect the range finder which is a critical component and also protect the rest of the robot. A rubber strap has been screwed on the perimeter of the bumper to reduce collision shocks.

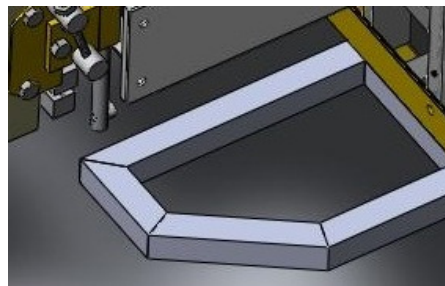


Figure 7 - Herbinator bumper

Electronics

The electrical groups have the task to connect the artificial intelligence with the mechanical parts of the robot. The main task of this team is to create a system that controls the behavior of the robot according to the commands sent by the artificial intelligence. This year, the main goal was to make a robot as reliable as possible on the electrical side. We concentrated our efforts on the control system and on the GPS system.

Computer

The computer is used to make all the processing needed to control the robot autonomously. In other words, if this computer stops working during the competition, the robot will not go anywhere. We also need a computer that uses the smallest amount of energy possible without compromising the speed. For these reasons, we absolutely needed a solution that is durable, reliable and efficient.

We are using a Kontron EXT-CD Computer-On-Module embedded computer in the robot and it proved to be a really good choice for our application since this computer is fast, reliable and stable, regardless of what it encounters (vibrations, movements, bumps, etc).

This computer is powered by a Intel Core Duo processor running at 1,66 GHz. This processing power is more than enough for our needs, and its power consumption is very reasonable. We chose a processor more powerful than what we needed for longevity purpose. We don't want to change it every year because the AI is more complicated and takes more processing power.

Another interesting feature of this computer is the compact flash card. This technology is very well suited for our application because it doesn't contain fragile mobile parts like a conventional hard drive. These compact flash cards make the computer a lot more reliable in hostile environments.

Finally, we have a LCD touch screen installed on the robot to control or debug it more easily. This feature is crucial when we want to do software changes on-the-go.

Actuators and control system

The robot is moved by two 1/3 HP brushed DC motors (one on each side of the robot). These motors are usually used to move an electric wheelchair so they have a lot of usable power to deliver. We use a RoboteQ AX1500 drive to feed these motors with the right amount of power. This drive has a current output of 30A per motor, more than enough for our needs.

To use these motors, we had to analyze its behavior very precisely to be able to create a reliable control system. To do this, we did a lot of tests on it to find all the constants we need to control it efficiently. Once we had these results, we took the old control system from last year and we modified it to react more accurately to the controls sent by the computer. This should improve the robot reaction to the AI commands and give us a better cutting precision.

Electrical system

There are two major sections in the electrical system, each connected to a battery: the electronic section (computer, sensors and controller) and the power section (motors and drive). This allows us to change the motor battery without having to shut down the computer. Each battery is made of two 12V batteries connected in series to get 24V, connected to the power supply through an anti-spark relay on the battery door to keep accidental sparks from happening

Electrical Wiring

Also, our homemade power supply is compact and energy-efficient, since it was made to fulfill our particular needs. In addition, for ease of debug, it will switch from battery power to AC power when it's plugged in, without shutting down the electrical system, and will switch back to battery power when the AC is removed. The power wiring uses an easy to understand color code, preventing misunderstanding and allowing anyone to work on the robot with a minimal risk to break parts.

Finally, as requested, we installed a simple safety system to stop the robot as needed. There are two relays that have to be activated to give power to the drive and motor, allowing the robot to start. The first relay is toggled by a big red button at the top of the robot, and the second one by a wireless remote control. Those buttons are both visible and easy to activate, and the robot will stop moving immediately if one of them is pressed. Finally, if those relays are activated, the mower can be started and stopped with the second button of the wireless remote control.

Electronics

The controller board, powered by a CAN-capable Atmel AVR microcontroller, interacts with many of the robot's peripherals. Here is an overview of those communications:

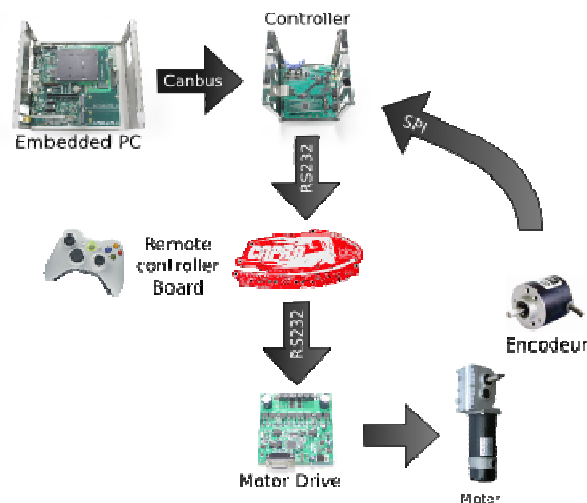


Figure 8 - Communications with the controller board

The controller and the computer communicate with each others with an Ixxat USB-to-CAN Compact interface, allowing the AI to send commands to the controller. It then processes the information, along with the feedback from the encoders processed by the quadrature counter chips (LS7366) and available on an SPI port. Finally, the appropriate commands are sent to the drive, a RoboteQ AX1500, via a RS-232 connection.

Devices

We are using exactly the same sensors in the robot as last year. Here is a list of the sensors used:

- **Renco optical encoders:** Keeps track of the motor movement with 360 counts per turn to acquire the rotation feedback of our wheels.
- **SICK LMS 291 laser scanner:** Used to detect obstacles in a 180 degrees radium in front of the robot.
- **OEMV-3 GPS receiver coupled with a GPS-532-C L1/L2 aircraft antenna:** These are used to receive a GPS signal to pinpoint the location of the robot. An innovation in relation with the GPS this year is that we use the Omnistar HP differential GPS service.
- **8" touch screen LCD:** We installed an 8" touch screen LCD on the robot. This feature allows us to interact easily with the robot computer without having to access it via remote connection. This gives us a big advantage if we have to debug something or if we have to do some last minute setups on the robot. The other big advantage is that the robot becomes totally independent of any external peripherals.
- **A test and diagnostic program:** A test and diagnostic program was created to help control the robot during tests and collect data by sending and receiving CAN-Bus messages directly from and to the controller, without interferences from the AI. We already had a basic tool that could send a few messages, but it couldn't do everything that was needed to help setup the control system with the new wheels. The program is made of two parts: a dynamic-link library (DLL) handling the sending and reception of the messages, and a Visual Basic GUI to select the messages to send and display and/or save the received messages. This design will allow anyone to reuse the DLL in other programs and scripts, without needing to know the implementation details about the CAN-Bus protocol and the Ixxat USB-to-CAN Compact interface.
- **Status Panel:** This board is use to control the power to all the device used on the robot. The power of all devices pass through switches so we can choose which device needs to have power. This board also provides fast diagnostic of all the devices.
- **Microstrain 3DM-GX1 inertial measurement unit :** Used to feed our movement information back to the computer for more precision and to detect changes in the terrain angle.

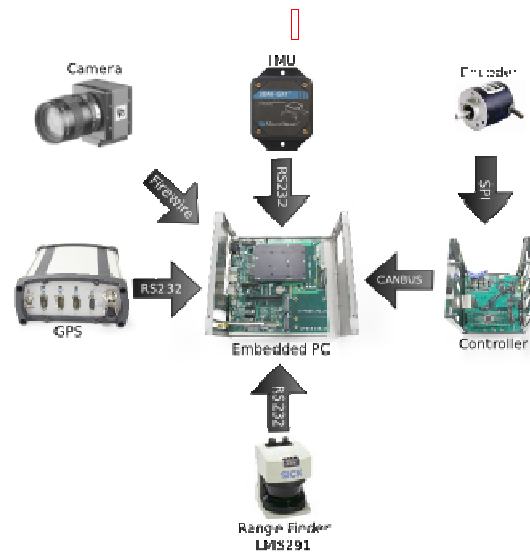


Figure 9 - Communications between the sensors and the computer

Electrical innovations

- All the wiring system has been rebuilt to optimize the performance of the robot. It is now more safe and also easier for troubleshooting.
- This year, we decided to change our remote controller for our robot. We made a board to communicate to the drive from an Xbox's controller. It is useful when we move our robot for testing or for a demonstration.
- We have a new board to control the power to the entire device used on the robot. The power of all devices pass through switches so we can choose which device needs to have power. This board also provides fast diagnostic of all the devices.
- We also add a router wifi on our robot to make easier to modify AI, troubleshooting, check for settings, etc.
- Our homemade power supply is compact and energy-efficient, since it was made to fulfill our particular needs. The power wiring uses an easy to understand color code, preventing misunderstanding and allowing anyone to work on the robot with a minimal risk to break parts. All the outputs of our power supply are wired to a terminal block mounted on a DIN rail. All 12V supplies and higher pass through a fused terminal to prevent high current and to protect the electrical circuit. Each of them is on dedicated fuses. The main board is screwed for safety reason in a heavy duty industrial enclosure which is equipped with a 80mm fan, for air circulation.
- A camera has been added to our robot to get images of the course and feed them to the vision system so it can detect the lines.

Software

Linux

Last year the robot was running under the Windows XP Professional operating system. With this system we had several problems; we then considered the possibility to change the OS.

Motivation for change

First, software can be expensive and requires a cd key for installing it. Then, you are responsible of managing your keys and you have to activate it every time you install it.

Our window XP was not very good at dealing with hard shutdown (and it sometimes happened). When that occurred, Windows was most of the time unable to reboot in less than a couple of minutes or was not able to reboot at all. We were then forced to reinstall the OS and all the programs required to run our AIs. We also had to reinstall the whole system when we were changing the computer. We could have only swapped the flash card but windows would not boot from it because its protection mechanism checked for hardware signature.

Since our AI is mostly implemented in Java (except for vision and can communication with the controller) we knew that porting the whole system to another operating system was achievable in less than a year.

Choosing the OS

We chose Linux because it would get rid of the scan disk with ext4 and would fail gracefully on hard shutdown. It is also possible to copy the flash at any time and boot it on any other computer. The flexibility of Linux also allows us to completely script our installation process in case someone wants to install our software on his computer.

Using Linux also simplifies a lot of things when comes the time to work remotely. With SSH, it is really easy to send or retrieve files on a distant computer and gives the possibility to execute commands remotely and forward X windows. This allows us to run Eclipse and our simulator on the robot through a local connection using our display. The system also boots very fast and is responsive; windows had some period where it would hang completely for a couple of seconds. Our custom Linux distribution is based under Ubuntu Minimal and run LXDE desktop, a very light but complete and attractive desktop.

The advantage of Linux is that it is a fairly modern OS and is customizable so we can save disk space (we try to limit the OS to 8Gb) and have the fewest and reduces the processes running. We would difficultly have been able to achieve this goal with Windows 7 or Mac OS. The biggest advantage with Linux, a free OS, is probably that there is absolutely no restriction on what you are able to do with this system (in contradiction with a proprietary OS).

Position Manager

The Position Manager's problem is caused by the interpretation we made from all the sensors streams. The GPS data, wheel encoders and the Inertial Measuring unit (IMU) must be managed effectively to accurately evaluate the errors in measurements. It is important to have a correct approximation of the position at any time, however it is also required that this position translate smoothly in harmony with the robot orientation and speed.

Last year we did not put the emphasis on the second requirement: consistency in position. So it was trying to approximate the most accurate position possible at any cost. This technique was implying that the robot would be all the time repositioned to what position the GPS was pointing at neglecting the IMU. This approach was causing the robot to constantly recalculate its orientation making it oscillate between two directions. The robot was not going in straight line so it was taking more time than necessary to cover the same distance and an extra CPU load was added because of the constant recalculation of the trajectory.

This year, we'll use the position given by the wheel encoders to evaluate our position until we reach a certain distance from the value estimated by the GPS and then correct our position. Since the encoders' data offers more stability than the GPS, correcting the encoders with the GPS is safer. Working this way may be less accurate than before in term of absolute position but the benefit gained by a straight line is compensated.

Also, for the robot's orientation on its virtual map, we now calculate an error between our Inertial Measurement Unit (IMU) and the encoders' orientation so we can observe the drifting of the robot live while it progress.

Changes made to the position manager improved our system foundations.

Tools

This year we mainly focused on developing new tools to support our debug session and help configure the robot faster. By using the Linux operating system, we not only had to learn using new libraries, but we had to implement those debug tools for the devices. The software that we had before could only be used on Windows, it was more advantageous to program them ourselves rather than use another Windows computer to diagnose problems with the sensors.

First we started by a scanner to automatically detect hardware (RangeFinder, IMU, GPS) speed and ports to avoid a fastidious manual configuration. We also implemented visual tools to help us interpret the information received by the different sensors. Log files are verbose but it is far more intuitive to have a panel showing an arrow pointing the direction the IMU indicate per example.

We used Java language to develop those tools because it's portable and is the same language than we use to develop the robot's AI. This allows us to use our tools on a standalone mode or on embedded with our simulator's GUI.

The tools we have built:

- **Imu calibrator and viewer:** a tool to calibrate our Inertial Measurement Unit and show an arrow to see the orientation of the robot.
- **Rangefinder viewer:** a tool to detect the port of the Rangefinder and show a graphic of the accessible area.
- **Motor control tool :** a tool to communicate with the controller sending messages to the engines functioning like a remote and showing encoders speed, angle and position.
- **Encoders theta viewer :** as the Imu viewer, the Encoders theta viewer shows the orientation of the robot according to the encoders
- **Motor drive debugger:** a tool to communicate with the drive exclusively. This tool can verify if drive is operational.
- **GPS antenna configuration:** a tool to communicate and configure GPS antenna (in development).

All those tools put together helped us in the improvement of our position manager.

Vision

Since we are working on two different robots and always thinking about reusing for the software part of the project, we built a vision system that we might use on the field this year in ION. With the non-square side of the field, having a system detecting the white lines on the ground is the only way to be fully autonomous in addition to the obstacle detection. This system would allow us to be able to mow on a field without knowing the dimensions if the field is surrounded with lines on the ground.

We used a vision system on another robot competition which didn't work well with lightness variations. It was hard to configure and was using a lot of CPU. Our new system runs as a client/server architecture communicating through sockets. This allows the vision process to be executed in a different process than the one we use for decision making.

The server is brilliantly engineered to allow certain properties to be changed lively, no need to restart any processes and it is completely transparent for the AI. This client server architecture is allowing the AI process to connect to that stream while at the same time one can also stream the results and change the settings to find the best possible configuration.

We can now rely on a very flexible "Vision Editor" to add and configure filters in a very effective way without having to recompile or modify manually any configuration file. The benefits are that we can test more configurations than before and tweak them at runtime. For this year in ION, we'll see if this system will help us and will certainly test it on the practice field to see its effectiveness.

Artificial Intelligence

Herbinator can be compared to an intelligent agent evolving in a particular environment. In this case, the environment is a delimited field of grass in which the agent must move to mow the grass while avoiding the static and dynamic obstacles it encounters. Several characteristics define an intelligent agent and allow it to achieve its goals:

- Autonomy
 - An agent must be able to take initiatives by itself according to its knowledge of its environment.
- Mobility
 - An agent must be able to interact within its environment by moving itself around obstacles towards its goal.
- Ability to think and react to the environment
 - An agent must be able to evolve accordingly to a changing environment where world parameters may change without notice (for example, mobile obstacles or even other agents).

The different sensors onboard help the robot to virtualized its environment and allow it to calculate a path along every corner of the area to mow. Two strategies were compared in order to find the one that would maximize the path taken by the robot most effectively. They are the following.

The first strategy involves taking the field's dimensions in order to divide the field in lanes that the robot can follow, according to the pre-calculated waypoints. The main advantage of this strategy is its simplicity of implementation as an artificial intelligence. The 2D simulator has shown that the robot could mow the entire field adequately if it does not encounter any obstacle. However, when adding obstacles across the robot's path, like a flowerbed, an eye-shaped area is left unmowed like in figure #10 below. Another disadvantage of the lanes strategy is that the robot passes close to the border lines on every round-trip. This considerably increases the risk of leaving the field with the lawnmower activated if line detection fails or the GPS lacks of precision.

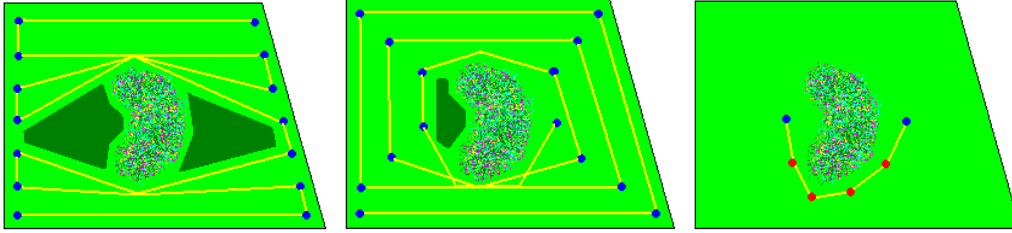


Figure 10 – 2D Simulator

On the other hand, the second strategy relies on a different way of distributing the waypoints on the field. In comparison to the other strategy, the robot goes around the field in decreasing spirals. This is an advantage considering that the eye-shaped area around the flowerbed, which was previously left un-mowed, is considerably reduced (picture #10).

Both strategies must take into account the fact that the robot may encounter not only the previously mentioned flowerbed, but also other various obstacles such as mobile entities. At any moment while reaching a waypoint, the robot must be prepared to modify its trajectory in order to bypass any type of obstacle. To be able to do this, a path finding algorithm was implemented using the A-star algorithm concepts which gives good performance in a real-time environment. To be brief, the chosen heuristic selects only the graph nodes of the virtual map with the shortest remaining distance to the goal waypoint. The picture #10 shows the robot dividing its trajectory into several small steps to bypass an obstacle.

Conclusion

Once again, Capra has high expectations for this year's edition of ION. We have worked hard all year long to prepare for this and we are now ready for action. The combination of the mechanics, software and electronics team from Capra gives a solid base to this competition. The solid foundation, along with the minor and major innovations makes Team Capra one of a kind. Capra is proud to participate in this multidisciplinary project and its members always try to surpass themselves.

Special thanks

Team Capra would like to thank all its sponsors for their monetary support on equipments and services they provided us. A special thank to the École de technologie supérieure (ÉTS) for their financing support and to the departments of mechanic, electric and software. We appreciate the time spent by the alumni members of the team for the knowledge transfer to the new "Capriots" recruits. Finally, thank to ION for offering us this great competition and the opportunity to prove ourselves year after year.