CATRAPILAS – A Simple Robotic Platform

Nuno Cerqueira

FEUP - Faculdade de Engenharia da Universidade do Porto,
Rua Roberto Frias, s/n, 4200-465 Porto, Portugal
cerqueira@fe.up.pt

Abstract. This paper describes Catrapilas, a small robotic platform, designed to be capable of solving some well known robot problems. Among these are some of the most popular robotic contests, like Micro Mouse, Fire Fighting and Autonomous Driving. It describes the major decisions and details of the physical architecture of the robot, but emphasizes on the high level approach used to control the robotic agent. This approach is based on the creation of a 2D map of the agent's environment, which should contain all the information needed in order to solve the current problem. There is also a description of the implementation used for the Autonomous Driving Competition, from the 2005 Portuguese National Robotics Festival, and the results that were obtained. There is a focus on the robot's ability to accomplish the objectives of the contest, and how this proved that the concept and ideas behind Catrapilas are correct.

1 Introduction

The major goal of robotic contests has always been to promote the growth and advance of all the sciences involved in the construction of inteligent robots for many tasks that until now were only performed by humans. Despite the controversial question of whether the competition side of these events brings more benefit than not, it is undeniable that many advances were achieved due to the sharing of ideas and the imitation of some ot the most useful features of the other robots. Among these contests are Micro Mouse [2] [3], Fire Fighting [4] [5] and Autonomous Driving [6].

But this sharing has some setbacks. In the early ages of small robot competitions the only possible approach was based on a completely custom made solution. Now, as then, the most natural approach to some parts is still based on a custom made solution. This is particularly true in mechanic components (motors, wheels...) and in hardware (sensor configuration mostly), since these are the physical components that could make a difference between equally "smart" robots.

The common practice in some of these contests is to use simple microcontrollers on the control part of the hardware. Maybe because the study area of the majority of the contestants is electronics, there is a normal tendency to use this kind of technology. However, computer technology has suffered so many breakthroughs that it's possible to fit one PDA or PC, with a processing power several

C. Bento, A. Cardoso, and G. Dias (Eds.): EPIA 2005, LNAI 3808, pp. 474–484, 2005.

dozen times bigger, in the space occupied by one of these simple microcontrollers. Processing power alone is a good justification for choosing this kind of technologies, allowing the use of Computer Vision and more complex techniques and algorithms. But this is not the only advantage. Computer technologies also simplify a great deal of tasks like using a camera or any other standard connection device (using USB, IEEE1394[1]). It's also very probable that some parts of the system (image processing libraries [7], data structures, etc...) were already been done by somebody else and might even be free to use. And in the future it will be possible to buy a new, cheaper and more powerful PC and easily use the same software previously implemented (maybe with minor changes).

All these advantages assert this approach as a good one. This idea is also shared by Sony, the company that is investing more in robotics for personal use worldwide, with its Aibo.

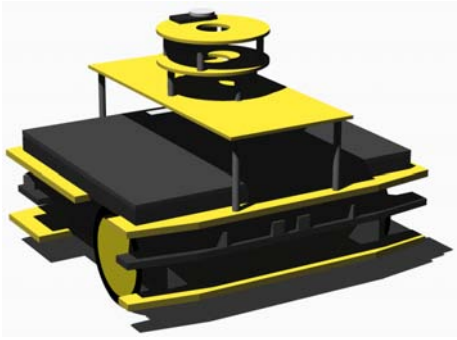

Fig. 1. Catrapilas 3D design

2 Catrapilas Architecture

2.1 Mechanics

Chassis. The robot chassis is made of expanded PVC plastic, due to it's extremely low weight, good structural resistance and easy crafting. It is composed by several decks that support all the electronics and mechanics needed and that also provide a good weight distribution and an increase in the structural resistance. This structure can be seen in Fig. 2.

All the robot's design was made keeping in mind the final weight of the whole. The weight is very important for a mobile robot, because if the robot is heavy its motors must be more powerful. More powerful motors are not only more costly but are also heavier, and require more powerful batteries that are also heavier. These heavier components make the need for a more resistant chassis structure, which must also be heavier. This is a vicious cycle, that only stops on a much

[1] Commonly FireWire.

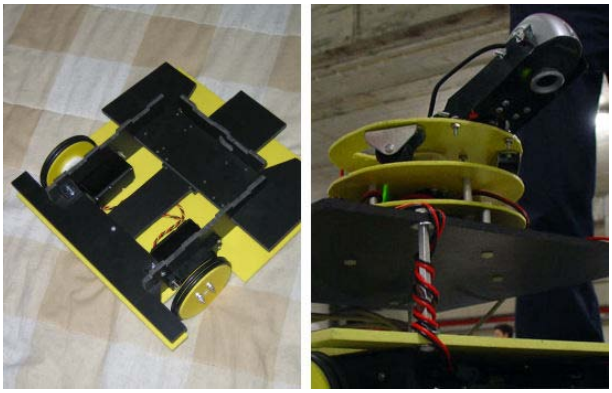

Fig. 2. Catrapilas' lower deck structure and sensor rotating tower

higher weight than needed, sacrificing the budget, the efficiency and the agility of the robot.

Our robot's upper deck is made from a thinner kind of PVC because of the low weight it has to support (just a few switches, servo motors and sensors). It contains a rotating tower in which different kinds of sensors can be installed, providing a more flexible perception of the environment. This rotating turret can also be seen in Fig. 2.

Traction. The traction is guaranteed by two wheels in the front part of the robot. They are made of the same material used in the chassis and are covered by two rubber o-rings each. These "tires" ensure good traction in carpet floors (tipically used in robotic competitions) and in most smooth surfaces (concrete, tiled and wood surfaces,...).

There is also a third wheel which serves as a caster. This wheel is omnidireccional, offering almost no resistance to frontal and lateral movement, introducing negligible distortion in the robot's motion.

Specific Changes for the Autonomous Driving Application. No changes were made to the original mechanics.

2.2 Hardware

Personal Computer. All the control is centralized in a small PC light enough to be carried inside the robot. The PC used is the Asus S5200N, with an Intel Centrino 1,6GHz processor and 512MB of RAM. The bigger side of the PC has about 28 centimeters, which is smaller than the common dimension limit of some of the Portuguese robotic contests (a bounding box of 30 centimeters). The great advantage of this option is that all the programming needed can be made using the x86 platform, undoubtedly one of the most popular, flexible and powerful

platforms of its kind. Other advantages that the Centrino architecture provides are: very low weight (less than 2 kg in this case) and long battery duration (up to 7 hours).

It features an 802.11b wireless interface, which provides valuable help in the test and debug process. There are 3 USB 2.0 ports which allow an easy connection of several input / output devices (very common nowadays). There is also an IEEE1394 port (less common than USB). Both these technologies supply their devices with power and allow the connection to hubs that increase the number of simultaneously used devices.

Sensors. The only sensor in the current configuration of the robot is a webcam. However a camera is a very powerful sensor because it is really composed of thousands of very precise and rich sensors. The data provided by the camera is also of very intuitive analysis and use. The selected webcam is the Creative NX Pro, with 320 by 240 RGB[2] pixels and USB 1.1 interface.

Actuators. All the actuators used in this configuration are of the same kind: servo motors [8]. Two regular servo motors are used to control the pan and tilt of the webcam. Two quarter scale servo motors (bigger and with more torque than regular servo motors) were hacked for continuous rotation [9] and are used to implement the robot's differential drive.

All these servos are controlled using PWM[3] signals sent by a servo control board [10] which is connected to the PC via USB.

Power. The PC has it's own power source (it's battery), and provides the power to the webcam and the servo control board via USB. The rest of the robot's electronics is powered by regular rechargeable NiMH AA batteries.

Specific Changes for the Autonomous Driving Application. Some lamps were inserted to light the tunnel, as can be seen in Fig. 3.

2.3 Software

Architecture. The robot's software is based on layers, as can be seen in Fig. 4, on a call and return architectural style. This style's major advantage is that changes inside a specific layer do not imply changes in other layers. This feature allows solving problems in the most adequate layer on a way that is invisible for the other layers. OOP[4] paradigm is used in order to simplify code reuse. There are 3 major layers in the software agent: World State, Meditator and Actuator.

World State receives data from the sensors and stores their current state as well as the actuators'. It also stores some higher level information, created from sensorial fusion, that can be of great use for the robot.

[2] Red, Green, Blue.
[3] Pulse Width Modulated.
[4] Object Oriented Programming.

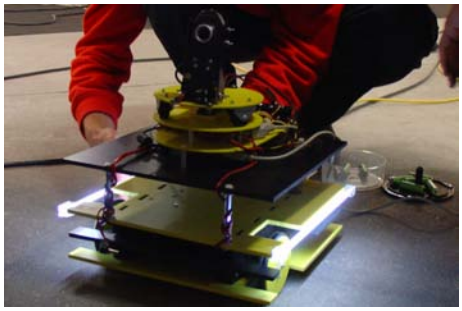

Fig. 3. Catrapilas being placed inside the tunnel for light testing

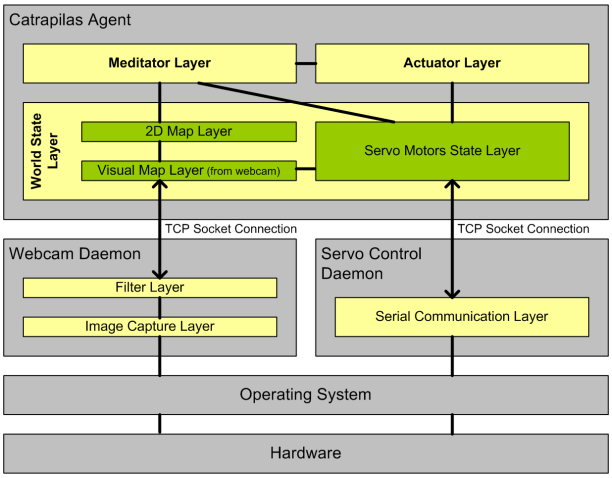

Fig. 4. Catrapilas Architecture

Meditator uses the information provided by the World State to select the best action[5] to perform. This action is then sent to the Actuator that ensures it's execution and corresponding change in the World State.

Image Processing. Most of the processing of the aquired image takes place in the Filter Layer (Webcam daemon). This allows the Visual Map Layer (World State Layer) to ignore this process as it receives an almost "perfect" image. The Filter Layer can offer different filters that can be changed by an explicit order via the socket.

[5] An action can be composed of many actuators' orders, as long as they aren't conflicting.

Neighbourhood Map Drawing. The agent bases its actions entirely on its current believed location and on a 2D map of the neighbourhood of the robot. This map is maintained in the World State's 2D Map Layer and it is created by the Visual Map Layer, using the Servo Layer (which provides the camera direction) and the filtered image got from the webcam. The map drawn this way could then be used to calculate the actions that are to be executed.

The image received is in fact a 3D perspective of the world, which does not provide an easy to use information. The process to make this information useful, in a 2D map, bases itself on a warping transformation of the image: a new image is created, in which all the pixels (corresponding to the 2D coordinates) are filled with the contents of the related pixels in the 3D perspective image. Obviously this relation depends on the pan and tilt of the camera, which are obtained from the values of the respective servo motors, available from the Servo Layer. An example of this warping process is depicted in Fig. 5, being the first the 3D filtered perspective image, and the second the 2D warped image.

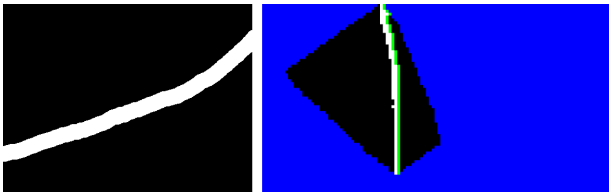

Fig. 5. Filtered 3D perspective image and 2D warped map

There are several possible methods to calculate the related coordinates on both images. The currently used method is based on a previous calibration. In this calibration some snapshots of a big pattern are taken, in different angles. This pattern contains stripes of a certain width, that allow the program to determine the coordinates of all the pixels of the image. Examples of this calibration can be seen in Fig. 6.

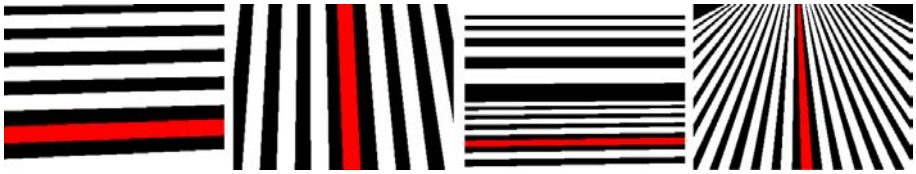

Fig. 6. Horizontal and vertical coordinate calibration images, for 2 different tilt values

Environment and Programming Languages Used. The system is implemented in Microsoft Windows XP due to some still limiting issues in hardware interface of the Linux operating system (like wireless lan).

The agent and daemons are programmed in C++ because it supports OOP, it is very eficient and it is very widely used. Java is also used on the viewer which connects to the daemons using TCP sockets.

Specific Changes for the Autonomous Driving Application. This was obviously the part that needed more changes. Firstly, there were used 2 filters in the Filter Layer: a Track Filter and a Signal Filter. The Track Filter allows to clearly "see" the limits of the track, the crossing, and the elements of the working zone. It uses binarization as well as eroding for "cleaning" the image. The Signal Filter emphasizes the color and form of the signal that the robot has to identify and obey to. It also uses binarization as well as some simple pattern recognition techniques. These filters allow the higher level to solve an easier problem, since a part of the original one was already taken care of.

The higher level, after translating this image into a 2D map, can use it to: calculate the distance and direction of the side line that limits the track, determine if the crossing is in front of the robot and align with the crossing, among others. The results of the refered calculations can then be used to decide which are the best actions to take.

3 Autonomous Driving Application

3.1 Introduction

In order to test the platform constructed, it was developed an agent for the Autonomous Driving Competition [6], from the 2005 Portuguese National Robotics Festival. This choice might seem weird, since from all the three major Portuguese robotic competitions, this is the less oriented to small robots and where PCs are used profusely (in oposition to the problem stated in 2.2). However it was chosen, due to the short amount of time available to implement the solution, and to the fact that it is the easiest to solve using only the camera as a sensor.

3.2 Contest Description

The Autonomous Driving Competition is a contest where robots must be capable of driving correctly through a eight-shaped track like the one that can be seen in Fig. 7. The contest is composed of three rounds:

First Round. The robot starts from the oposite side of the parking lot, just before the crossing line. It has to accomplish 2 complete laps (passing over the crossing line 4 times). When it is completing the second lap, it must stop over the crossing line. This is a speed run and no other rules apply. The score is based only on the elapsed time.

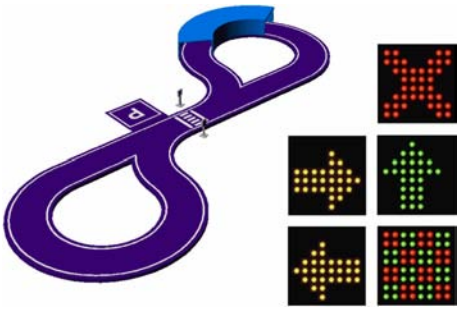

Fig. 7. Track layout and signals used during the contest

Second Round. The basic structure of the round is very similar to the first one. However there are introduced some new rules: after stoping at the crossing line, the robot must follow the orders imposed by the signals depicted in Fig. 7; and at the end of the round the robot must park in the according zone. Failure to accomplish these rules imply a time penalty that is added to the time score.

Third Round. The basic structure of the round is the same as the previous round. There are also introduced some new obstacles: a tunnel where visual guidance is not possible under normal conditions; and an optional working zone. This zone is a detour from the original track and is delimited by small yellow cylindrical landmarks. Scoring is similar to the last round, with a penalty to the robots that decide not to run with the working zone setup.

Final Score. The final score is the sum of the scores of all the three rounds. The winner is the robot with the smaller score.

3.3 Results

This is a short description of Catrapilas' participation in the competition.

First Round. This round was the first real test to Catrapilas as an autonomous driving robot. Despite the easiness of the round, the robot was only able to complete one lap, going off the track on the second pass through the crossing. This was due to a hardware bug that ocurred when the batteries were weaker, causing the robot to oversteer when cruising at low speeds (in the approach to the crossing) and was only found during the real contest, since it was the first time it was tested for a longer period of time. It's worth mentioning that the track is very hard to reproduce because of it's size, and the first full track test was only a few hours before. Fortunately all the ideas that could not be tested before, were proven right in these tests. Catrapilas scored 520 points and was ranked 10th out of 17 robots in this round.

Second Round. This round surely was the most successful of all. The bug observed in the previous round was partially solved by a program around: it was verified if the robot's last order contributed to solving the current sub-goal, and if this case wasn't true a correcting order was applied. The robot was able to recognize correctly all the signals despite some bugs discovered in the test stage. Catrapilas scored 380 points and was ranked 11th out of 17 robots in this round.

Final Round. This round was a disappointment compared to the previous one. The lamps used were found too weak to correctly light the tunnel. So another program around was made: when Catrapilas detected the tunnel (when captured images became very dark) it would apply a fixed value to the motors. This value was calculated by many tests and trials, and was proven very acceptable during the tests. However it failed during the round and because of this the robot was unable to even complete a single lap. The choice was made to go into this run without the working zone of the track, because although a strategy for the working zone was fully implemented, it was not very tested at the time. Catrapilas scored 528 points and was ranked 8th out of 17 robots in this round.

End of Contest. Catrapilas ended the contest with 1428 points and ranked 12th out of 17 robots. However this result is promising since this robot's top speed was not even closer to the majority of the other contestants. However, there were two slower robots that were better ranked than Catrapilas, due to the fact that their performances were more regular.

From the point of view of the robot's concept this participation was a success, since it proved that the basic idea of operation is correct. A good example of that is the fact that the winner robot followed a very similar strategy, only more tuned and with a better hardware.

For academic and personal purposes it was also a success since there was a lot of idea sharing and an introduction to the competition environment.

Just one more fact worth of notice: Catrapilas was the favorite robot of the competition. Not only to the crowd, that cheered when the successful run was accomplished and also when the robot camera moved (this was the robot that more clearly showed what and how it was "thinking"); but also to the organization, that described the robot as the "visually more well achieved robot".

4 Future Work

4.1 Hardware

Camera. There were detected some faults in the camera used. Although having a good frame rate the camera introduces a lag in image capture. This lag is around 500 ms to 1000 ms and has no relevance at its intended use (webcam). However, this lag is very bad for robotic applications were several decisions must be made each second. Another problem was discovered in images where there are fast color changes. The autoregulation of the camera blends these colors and makes impossible to detect the flashing checkered signal, for instance. So, a replacement must be found.

Encoders. The implementation of wheel encoders is already underway, and will allow better positioning, adding the capability of merging with older maps. An encoder is a mechanism that allows the robot to know how much it has moved since the beggining of its execution. The encoders that are being implemented are quadrature encoders [11] and give more information than regular ones.

Sensors. Inclusion and connection of new kinds of sensors is being planned. Some examples are: IR[6] distance, IR ground color, IR beacon detector and UV[7] flame detector. These sensors will allow more precise information gathering, and will also help in the reduction of the global uncertainty factor.

Portable Remote Access. Debugging, controling and testing the robot is very difficult without another machine with GUI[8] capabilities. The use of a PDA with wireless connection will simplify the entire process.

4.2 Software

Image Filters. The used image filters were a bit slow in their operation. The implementation of faster filters can improve the performance of the robot. Investigation will be made in order to determine if there is an already implemented visual library (like CMVision [7]) that offer all the needed features and others that could increase the robot's capabilities.

Other Applications. Implementations of the robotic agent are intended for the Micro Mouse contest [2] and the Firefighting [4] contest.

5 Conclusions

We can say that the project and the competition have had a very positive outcome. The major accomplishment was to prove that the concept of the high level approach based on a 2D map to solve a specific problem was not only correct, since the robot was clearly able to accomplish the goals it was programmed for; but also that it is very easy to implement.

We can see that the needed changes in the entire platform were of little difficulty and in very low quantity. The changes that are planned in order to solve other kinds of problems are also of this degree of difficulty.

The participation in the contest also reinforced the idea that robotic competitions of this kind are very useful events, since it is a very inspirational and motivating environment that promotes the sharing of ideas and approaches.

[6] Infra Red.
[7] Ultra Violet.
[8] Graphical User Interface.

References

1. Reis, Luis Paulo et al.: *Proceedings of the Scientific Meeting of the Portuguese Robotics Open*, FEUP Edicoes, Coleccao Colectaneas, Vol. 14, ISBN 972-752-066-9, April 23-24, 2004
2. *Micro Mouse Contest*, University of Aveiro, [online] available at http://microrato.ua.pt [consulted on May 2005]
3. *Micromouse UK*, University of London, [online] available at http://micromouse.cs.rhul.ac.uk/ [consulted on April 2005]
4. *Fireman Robot Contest*, Informatics Department, E.S.T.G, I.P. Guarda, [online] available at http://www.ipg.pt/estg/robobombeiro/ [consulted on April 2005]
5. *Fire Fighting Home Robot Contest*, Trinity College, [online] available at http://www. trincoll.edu/events/robot/ [consulted on April 2005]
6. *Competition, Autonomous Driving class - Rules and technical specifications*, National Robotics Festival 2004, [online] available at http://www.robotica2004.org/regulamentos/competicao_ca_040124.pdf [consulted on June 2004]
7. *CMVision*, CORAL Group's Color Machine Vision Project, [online] available at http://www-2.cs.cmu.edu/~jbruce/cmvision/ [consulted on June 2005]
8. *What's a Servo?*, Seattle Robotics Society, [online] available at http://www.seattlerobotics.org/guide/servos.html [consulted on April 2005]
9. *Hacking a Servo*, Seattle Robotics Society, [online] available at http://www.seattlerobotics.org/guide/servohack.html [consulted on April 2005]
10. *USB 16-Servo Controller*, Pololu, [online] available at http://www.pololu.com/products/pololu/0390/ [consulted on February 2005]
11. McManis, Chuck: *Quadrature Encoder*, Pololu, [online] available at http://www.mcmanis.com/chuck/robotics/projects/encoders/enc_quad.htm [consulted on June 2005]