The TRIPÉ III robot: a *Colombus' egg* idea

João Carlos Capucho, José António Parente

***Abstract* - This paper presents a brief description of the TRIPE III robot which won the 2000 edition of the Micro-Rato Contest. It is a reactive robot that combines several interesting features. In particular, it has a high level of H/W integration resulting in a low chip count (μC, motor drives and voltage regulator), it is capable of achieving a speed 70% higher than most similar robots and it efficiently combines beacon-following, obstacle avoidance and wall-following behaviours allowing it to profit from its speed.**

***Resumot* – Este artigo apresenta uma breve descrição do robot TRIPÉ III, vencedor da edição 2000 do Concurso Micro-Rato. É um robot reactivo que combina várias características interessantes. Em particular, apresenta um elevado nível de integração do H/W que resulta num número muito reduzido de componentes (μC, motor drives e regulador de tensão), é capaz de atingir velocidades cerca de 70% superiores às atingidas por robots congéneres e combina de forma eficiente os comportamentos de busca de farol, evitar obstáculos e seguimento de paredes permitindo-lhe tirar vantagem da velocidade mais elevada.**

I. INTRODUCTION

The *Micro-Rato* Contest organized at the University of Aveiro has just came through its fifth edition. The main technical challenge associated to the contest is to develop complete autonomous agents that can find their way within a closed maze, in the shortest time, from a starting point to a goal area without any human intervention. The robots compete 3 at a time and thus, each of them has to cope with static obstacles (maze) as well as dynamic obstacles (other robots). Any contact with the obstacles results in an extra penalty time. The maze is limited to a square area of 5 by 5 meters and the goal is highlighted by an omnidirectional infra-red beacon. The timer for each robot stops counting when the robot enters the goal area, stops and lights up a small LED. These are, briefly, the main contest rules. In the following sections we will describe the TRIPE III robots and we will discuss the main options taken in its development.

II. TRIPE III'S DESIGN: PROBLEMS AND SOLUTIONS

This was our third participation. In the previous two, 98 and 99, we gained experience and took the robot to a point where the main limitation was the robot speed. It was too slow... However, when we tried to increase the robot speed we had to face several new problems.

Firstly, the motors: after trying several different motors we found out that cheap ones, like those we can find in inexpensive toys, are not suitable when we want to have fast control on rotation speed and direction inversions. They are particularly noisy (electrical noise!) and it is difficult to find appropriate gears to decrease their speed and increase their torque. The effect is that they react slowlier to the variations in the control voltage, in other words, the robot inertia has a bigger impact on the motion of the robot. From an electrical point of view, this means larger peaks of high current consumption during direction invertions.

On the other hand, the hacked servomotors provided by the organization of the contest have an embedded gear box with a very large reduction ratio. This allows us to get a reasonably high torque out of very low-power motors. The consequence is that the rotation speed is also very much reduced (that is why our previous robots were slow as were all the others anyway!). Thus, the desired solution would be to find a better balance between torque and speed, somewhere between both types of motors referred above. The difficulty is how to do it without changing the gear set.

Well, after all the solution was not difficult at all (a *columbus' egg* idea!), bigger wheels! If we consider the robot as a whole, instead of considering the motors separately, then we must take into account the wheels effect in the balance driving force / speed. By changing the diameter of the wheels we can adjust with fine resolution the right balance we want. Thus, we started with the high torque hacked servomotors and we increased the wheels diameter up to the point where we could have a sufficiently higher speed and yet a sufficient driving force to facilitate the speed control (i.e. fast variations in the wheels speed). The result was an increase of about 70% in the robot top speed when compared to the previous versions.

Solved the low speed problem another one came up. This time it had to do with the analog to digital converter (ADC) and the infra-red (IR) light sensors. The hardware provided by the organization to control the robots is based on the Intel 80C188 processor and on the old ADC0809. This ADC is mainly used to convert the analog outputs of hacked digital IR detectors. These are based on the well

known Sharp GP1U58 used in TV sets and VCRs in the IR remote control receiver unit. The robot uses 3 of them to detect obstacles and 2 other (actually with a different demodulation frequency, GP1U583) to detect the beacon. The obstacle detection is active in the sense that the robot emits IR light, using IR LEDs, and looks at the reflection received by the detectors. The analog hack that is carried out on these detectors allows to have a gross measure of the distance a given obstacle is from the robot. The output voltage increases proportionally to the intensity of the modulated IR light (around 40KHz) received by the detector. On the other hand, the voltage is inversely proportional to the distance between the robot and the obstacle. The relationship between voltage variation and obstacle distance is aproximately quadratic.

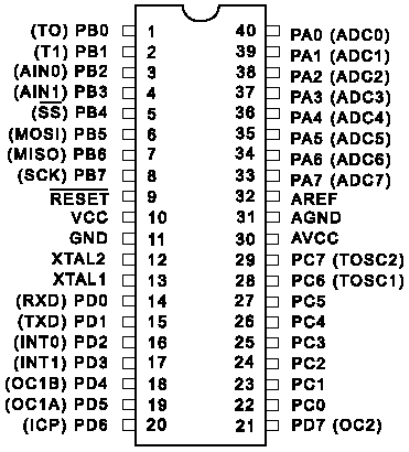
Now, the problem of using the old ADC0809 is that it requires a certain power from the detectors whenever it takes a sample. Since the detectors have a very limited drive capability, their output voltages vary when several samples are taken consecutively in a short interval. So, the ADC returns nearly correct values only if the sampling rate is relatively low (less than 50 samples/s). Otherwise, the detectors saturate. However, even if we take few samples per second, the total variation of the measured value is smaller and the reaction time is longer than the ones we expected to obtain.

Since our robot was already moving faster, we wanted to increase the sampling rate of the obstacle detectors. The idea was to travel the same distance between two consecutive samples and thus, to maintain the robot obstacle awareness. As we have just explained, increasing the sampling rate of those hacked detectors was not possible with that ADC. Thus we tried to find in the market an ADC that could improve our acquisition system. However, after a few searches, we came up with the idea of using a complete new controlling system based on a microcontroller with not only an integrated ADC but also the required memory, timers and digital ports.

We chose one AVR (AT90S series) from Atmel Company. This series is based on a high performance RISC architecture in which almost every instruction is carried out in a single clock cycle. These microcontrollers have very low power consumption, as low as a few miliamperes. We decided to rebuild all the control unit based on the AVR AT90S8535 (fig. 1), a small wonder of the modern technology. It is the most complete of it series with a very impressive set of characteristics in a single chip that we will present next.

III. TRIPE III'S HARDWARE

Like many microcontrollers the AT90S8535 operates with a Harvard architecture, opposed to Von Neuman's. Harvard machines have two separate areas of memory, one for the machine instructions and another for the data, while the Von Neuman architecture has only one area for both data and instructions. Both architectures have a set of special memory locations call registers, which hold temporary data inside the processor. Most of the program instructions operate on data kept in such internal registers.



AT90S8535 – 8PC

- 32 general use registers of 8 bits
- 8 Kbytes of flash memory, the equivalent to approximately 4000 instructions
- 512 Bytes of ram
- 512 Bytes of eeprom
- Serial interface for in system programming
- 8-channel, 10-bit ADC
- Programmable UART
- Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
- One 16-bit Timer/Counter with Separate Prescaler, Compare and Capture Modes, and dual 8-, 9-, or 10-bit PWM

Figure 1. Layout and main characteristics of the AT90S8535 μC

In this case, the microcontroller has 32 general purpose registers which is very unusual in the microcontrollers realm and highly valuable for the programmers. The assembly programmer gains allot because any of the 32 registers can be used with almost any instruction, it means that most of the significant operations can be performed between registers. For those interested in using the *C* programming language, there are several compilers available. However, there is an open software compiler, from GNU, that takes advantage of the 32 registers to the maximum, optimizing some instructions in surprising way.

As we said before, since the controller has a Harvard architecture there is an instruction area separated from a data area. On the AVR, the first one is FLASH memory while the second is SRAM (Static Random Access Memory). The major advantage of using FLASH memory is that the data is not lost when power is removed or fails while the SRAM is a lot faster when writing but loses its contents if power fails. Well, the FLASH memory does not hold the data indefinitely with power down. But it is

guaranteed to hold it for about 40 years, which we think is enough!!!

More importantly, the total available instruction memory is 8 Kbytes used in 16 bit chunks, or words, which means that each instruction is 2 bytes long. Therefore we have space for about 4000 instructions. Well you could say that in today's gigabytes world 4000 instructions is nothing but when you are programming simple reactive robots you will see that it is enough for many applications.

The data memory is smaller, 512 bytes. Once again it looks short but it proved to be more than enough to hold all the variables and stack that we needed.

The 512 bytes of EEPROM (Electrically Erasable Programmable Read Only Memory) are quite different from the rest of the data memory. It works like the FLASH memory but endures a lot more rewrites, about 1000 for the FLASH against 100000 for the EEPROM. Besides, it can be rewritten while the system is running via the I/O ports. This is very handy when, at run-time, we want to save some dynamic parameters and keep them even after power down.

The access to the I/O ports as well as to the peripheral units (ADC, UART, EEPROM, Timers, Interrupt controller…), is very easy and very well documented. Besides, all peripheral units have the possibility to work with interrupts which can be time-saving.

The communication with the outside world can be carried out serially by two standard means, the UART (Universal Asynchronous Receiver and Transmitter) and the SPI (Serial Peripheral Interface) which allows high-speed synchronous data transfer. In particular, the controller can be programmed in-circuit by downloading the program via the SPI port.

In what concerns the ADC (the initial cause to build a new robot controller) it has 8 multiplexed input channels with sample and hold amplifier. This ensures that the voltage at each ADC input is held at a constant level during conversion. It can work in free run at 200 kHz with 10 bits resolution, resulting in 15000 samples per second. However, if the 10 bits are not needed, you can increase the clock to 1 MHz and with 8 bits resolution achieve 77000 samples per second!

In our case, we used a sampling rate of 600 samples per second and per channel without any negative effect on the IR detectors. Remember that with the ADC0809 at 50 samples per second the sensors did not work, they completely saturated.

The AT90S8535 also provides three general purpose Timer/Counters, two 8-bit T/Cs and one 16-bit T/C, which are important to generate the square waves required to modulate the IR LEDs (T/C0) and to control the motors using a PWM scheme (T/C1). In the first case, the Timer/Counter 0 was programmed as an overflow counter to generate the required 40KHz for the IR LEDs modulation. In the second case, the Timer/Counter 1 supports two output compare functions which were used to generate two independent PWM signals and thus control both motors independently.

The control of each motor via PWM, pulse width modulation (fig. 2), has two main advantages over other forms of linear control: it allows a very simple interface to the control logic, no analog signals are used; and it is highly efficient in terms of dissipated power. Notice that the switching transistor that is normally used in series with each motor is always either in saturation, when switched on, or cut off. In both situations the power dissipated in the switching transistor is minimal since when it is switched on it has a near zero voltage and when it is cut off it passes no current.

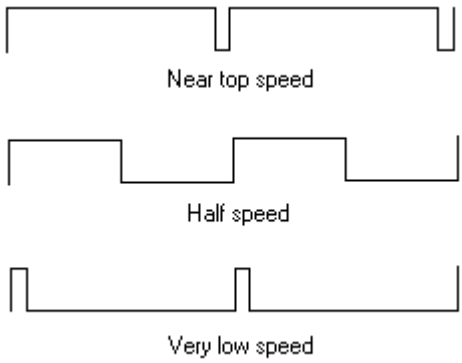


Figure 2. PWM control of motor speed

Finally, the effective motors drive was carried out by means of an integrated dual H-bridge, the L293, which allowed an independent bidirectional control of each motor. The circuit has the capability to drive the two motors and does not require a heat sink.

Figure 3 shows the layout of the robot control system where the low chip count is clear. Notice that the overall control system has just 3 integrated circuits, the microcontroller, the motor drive and the voltage regulator. This aspect makes the system very easy to assemble, decreases the probability of hardware failure and facilitates any required fixing.

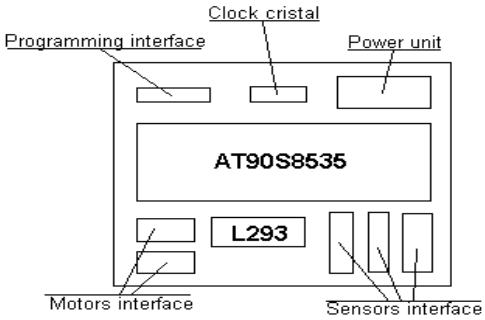


Figure 3. Layout of the robot control system

The dimensions of the PCB are 7x5cm. Since it is highly compact its placement in the robot is also facilitated. Moreover, it contributes to decrease the robot weight.

IV. SOFTWARE

And what about the software, the *brain of the beast*? We think that, the less complex the robots are, the better they work, so we tried to reduce the software complexity to the minimum possible. It has two main behaviors which alternate in priority, follow the beacon and avoid obstacles while following walls. The first one (fig. 4) simply turns the robot to the beacon and moves towards it. The second one (fig. 5) makes the robot go around walls without touching them, whenever one appears on its way to the beacon.

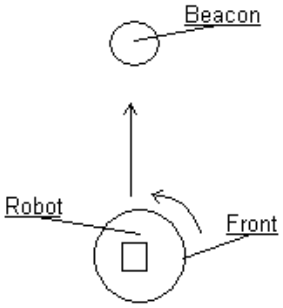


Figure 4 . Beacon following behaviour

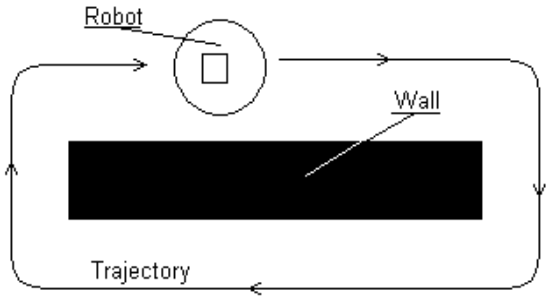


Figure 5. Wall following behaviour

The combination of these two basic behaviours is carried out in the following way: if the way to the beacon is clear, follow the beacon, if a wall (or an obstacle in general) is found in the way, go around until the way to the beacon is clear again (fig. 6).

V. CONCLUSION

In this paper we described the TRIPE III robot. It is a good example that it is possible to build small, simple inexpensive robots which can, nevertheless, perform well in many situations. In fact, it won the Micro-Rato Contest this year.

The robot's most interesting features are the optimization of the driving force / speed, which resulted in an increase of 70% on the robot speed just by using larger wheels, and the compactness and integration of the control system with a total chip count of 3. This latter aspect facilitated the placement of the control system in the robot structure and contributed to reduce the robot weight. Particularly the weight reduction also contributed to maintain a good controlability of the robot movements even with a higher speed.

The software structure was also very simple, based on 2 main behaviours, beacon following and obstacle avoidance with wall following.

Finally, in this paper we hope to have shown that by *digging* a little, instead of directly using the parts supplied by the organization, one can find very interesting and different solutions to existing problems and come up with a different robot that can even perform better as was the case.

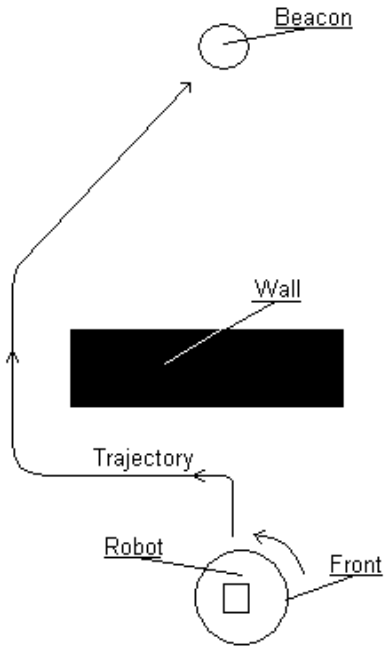


Figure 6. Combining both behaviors

VI. ACKNOWLEDGEMENTS

We would like to thank Luis Almeida by his patience in the reviewing of this paper that allowed to improve the quality of the writing as well as the structure of the contents and consequently, the paper clarity.