The Nomad 200 and the Nomad SuperScout: Reverse engineered and resurrected

Conference Paper · July 2006

DOI: 10.1109/CRV.2006.76 · Source: IEEE Xplore

3 authors, including:

Michael Jenkin
York University
341 PUBLICATIONS **5,877** CITATIONS

SEE PROFILE

The Nomad 200 and the Nomad SuperScout: Reverse engineered and resurrected

Arjun Chopra, Mark Obsniuk, Michael R. Jenkin
{chopra, obsniuk, jenkin}@cs.yorku.ca
Department of Computer Science and Engineering and Centre for Vision Research
York University, Toronto, Ontario, Canada.

February 6, 2006

Abstract The Nomad 200 and the Nomad SuperScouts are among the most popular platforms used, for research in robotics. Built in the early 1990's they were the base of choice for many mobile robotics researchers. Unfortunately, lack of support and proper documentation to enhance the computing power on these robots has meant that they have slowly faded away into oblivion. We at York University have four robots from Nomadic Technologies Inc., and rather than allowing our old robots to just rust away, we decided to breathe new life back into them. In this paper we present the techniques we used to resurrect our old Nomads.

Keywords: Nomad 200; Nomad SuperScouts; Nomad upgrade;

1 Introduction

Visit any reputable robotics lab anywhere in the world and there is a good chance that you will likely find a robot from Nomadic Technologies – a SuperScout, a Nomad 200 or a XR4000 – in their inventory. Sadly, however there is an even greater chance that it will be sitting in the farthest corner of the laboratory gathering dust. What caused this phenomenon, whereby, robots like the Nomad 200 which have significant mention in the research literature(see for example [1], [10] and [12]), have been relegated to the sidelines? The original Nomadic Technologies Inc. was bought by 3Com Inc. in November 2000. Although 3Com released source code to all drivers, libraries, and applications which ran on the Nomad series of robots [5], these only addressed issues related to software failures. For troubleshooting, repairing or upgrading the robot hardware, data sheets to the various proprietary boards were required. Unfortunately these are not available. Inevitably as components failed the robot became immobile. The robots at York University were no exception. In the VGR lab at York University there are four robots from the Nomad family, three SuperScouts - Gluttony, Lust, Greed and one Nomad 200 - Sloth. Failures in the various computational, communication and sensing modules coupled with the lack of external support was quickly rendering these devices obsolete, and the situation was quickly approaching the fate that Marvin (the paranoid android of the Hitch Hiker's Guide to the Galaxy) lamented – "do you want me to fall apart where I'm standing or shall I just sit in a corner and rust?" These robots contain a wide variety of equipment like the Proxim network cards which are not compatible with today's wireless infrastructure. Even the CPUs that came with these robots are generations behind what is available today. Worse still, todays Linux no longer supports the motherboard and video options provided with the Nomads. These factors limit the capability of the Nomad to be used for research even though the basic design and capabilities are similar to contemporary robots.

How then can these robots be brought back to full operation and integrated within modern network and operating systems? Here we describe our experiences in resurrecting the Nomad 200 and the Nomad SuperScouts. It is hoped that these results will encourage others to breath new life into their old robots, and let these effective machines live again.

2 The Nomad 200

2.1 Overview

The Nomad 200 [5] is an integrated mobile robot system with a rich suite of sensors including tactile, infrared, sonar and basic vision systems. It consists of a three wheel synchronous drive nonholonomic base, on top of which is mounted an independently rotat-

Figure 1: Sloth: The refurbished Nomad 200.

ing turret, housing sensors and an Intel 80486 based computer. Although many Nomad 200s were sold very few are currently active. The computer hardware on the Nomad 200 is almost a decade old. Replacement components are difficult to acquire, the computer hardware is no longer supported by current Linux drivers and the Nomads use a non-standard wireless protocol. These factors place severe constraints on using the robot. Given the proven success of the basic design of the robot, it seems prudent to upgrade and enhance the computing power on the Nomad 200 while retaining its original functionality. We are of course not the only ones to make this observation. For example [6] and [7] describe ongoing efforts to breathe new life into the old Nomad 200s. Our goal in upgrading the Nomad 200 was to resurrect it and in the process replace some of its more antiquated components. The Intellysis 100 [4] microcontroller from Nomadic Technologies was replaced by the Adapt9S12DP256EVP [11] microcontroller from Technological Arts (see figure 2). While, the onboard computer was replaced by a computer with a 1 Ghz Intel Pentium 3 processor, equipped with standard 802.11g networking.

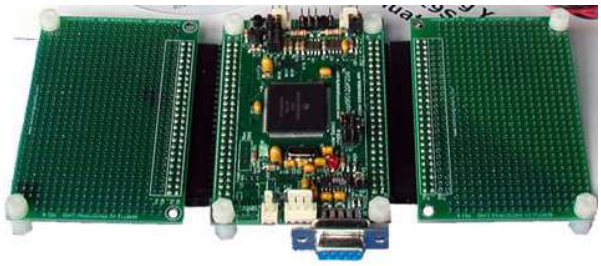


Figure 2: The Adapt9S12DP256EVP microcontroller.

2.2 Microcontroller Integration

From a control point of view the Nomad 200 essentially consists of an onboard computer with a motorcontroller card, along with a microcontroller that controls the onboard sensors. Integrating the microcontroller was the most challenging part of the project. We replaced the Motorola MC68HC11 [3] based microcontroller with a microcontroller module based on the current product from the same vendor – the 16-bit 9S12DP256 [8]. Although, we managed to find some documentation on the Nomad such as [5] and [4], replacing the microcontroller requires much more detailed documentation regarding the operation of the sensors and motor control than is publicly available. At the very least the pin-out of the interface to the original microcontroller is required. Unfortunately, none of this information was available. Instead we had to resort to considerable reverse engineering – using a continuity tester, data sheets for the ICs and our intuition – to identify connections from the microcontroller to the individual components. The Adapt9S12DP256 microcontroller communicates with the onboard computer through the serial port and was programmed using the GNU Toolchain [2] . The detailed results of our efforts are provided below.

2.2.1 The Sensus 200 Sonar Ranging System

The sonar ranging system (figure 3) on the Nomad is a 16 channel system, consisting of 16 polaroid transducers driven by a Polaroid 6500 ranging board [9]. The Polaroid 6500, (see figure 4), is a popular board and considerable literature on it can be found on the internet. However, it is usually used with a single transducer whereas in the Nomad 200 a single polaroid board controls 16 transducers. On examining the circuitry it was found that the board was multiplexed to the transducers through electronic relay switches. A multimeter, intuition and informed guessing were used to interface the sonar ranging

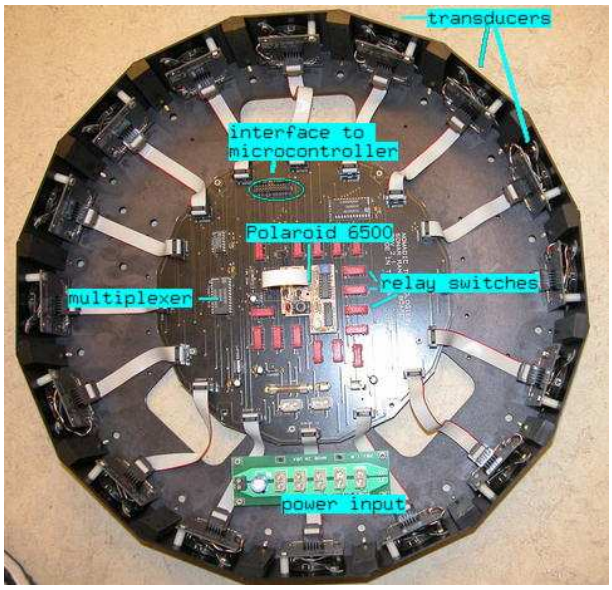


Figure 3: The Sensus 200.

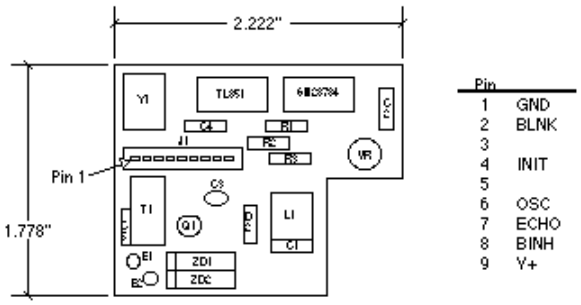


Figure 4: A schematic of the Polaroid 6500 module. (http://www.acroname.com/).

module to the Adapt9S12DP256 microcontroller. In our design, four selection lines for the multiplexer M0 –M3 and the BLNK, INIT, ECHO and BINH pins (see figure 8), were connected to PORT A of the microcontroller. The four multiplexer select lines are used to fire the 16 transducers in succession and then listen for an echo on each of them. Figure 6 shows the timing diagram for firing the sonars. The INIT pin is the trigger for "pinging" the transducer. Pulsing this pin high causes the transducer selected by the multiplexer, to emit an ultrasonic ping. After emitting the ping the sonar goes into listen mode where it waits for the ECHO pin to go high. The sonar determines distance by measuring the time elapsed between setting the INIT pin high and receiving an ECHO (.i.e the instance when the ECHO pin goes high). There is an initial default blanking period of 2.8 ms after INIT goes high, where the sonar is

```
/*fires all sonars and writes data back to serial port SCI1*/
#include <libhc1x.h>
#define PORTH PIM_PORTA
#define PORT SCI1
static char hex[] = "0123456789abcdef";
static char hex1[8];
static volatile dword_t elapsed_time;
static volatile dword_t current_time;
static volatile dword_t start_time;
static volatile byte_t echo_state;
static volatile byte_t prev_sonar;

int fire(byte_t sonar_id)
{
        dword_t i,j;
        byte_t mask;
        _io_ports[BASE_PIM+PORTA+PIM_R_PT] &=0x00;
        mask =0x01;
        /*select sonar*/
        _io_ports[BASE_PIM+PORTA+PIM_R_PT] |=sonar_id;
        /*debounce switch for 0.5 msec*/
        start_time = clock_msecs();
        elapsed_time = start_time;
        while((elapsed_time -start_time)<5)
        {
                elapsed_time = clock_msecs();
        }
        /* fire.....*/
        start_time = clock_msecs();
        elapsed_time = start_time;
        /*ping sonar*/
        pim_set(PORTA, 6,1);
        start_time = clock_msecs();
        pim_set(PORTA, 4,1);
        i=0;
        current_time = clock_msecs();
        elapsed_time = current_time -start_time;
        /*blank for 2.8 msec*/
        while(elapsed_time <28)//2.8
        {
                current_time = clock_msecs();
                elapsed_time = current_time -start_time;
        }
        /*should be zero as echo was  blanked*/
        echo_state = pim_get(PORTH,5);
        /*set blnk low to receive echo*/
```

Figure 5: Snippet of code written for controlling the Sensus 200.

forced to ignore an echo. The BINH pin can be used to force the sonar into the listen mode. Setting the BLNK pin high after receiving the first ECHO high is detected allows the sonar to listen to echoes from a more distant object.

2.2.2 The Sensus 100 Tactile System

The tactile sensor on the Nomad consists of 20 push buttons arranged in two rings of 10. The bumper was not wired to the motor controller in the original design. To sense a collision the onboard computer continuously monitored the bumper's state through the microcontroller and instructed the microcontroller appropriately. The microcontroller would then disable the AMPENABLE on the servo-amps, to stop all motion. We have replicated the same design in our system. On sensing a collision the BINT pin of the bumper sensor, triggers an interrupt in the microcontroller. The microcontroller then uses PORT K to read into the Bumper's data latch through a multiplexer, and determines which of the push buttons is actually depressed. This sets a byte in the mi-

Figure 6: The Nomad 200 with its servo amp exposed.

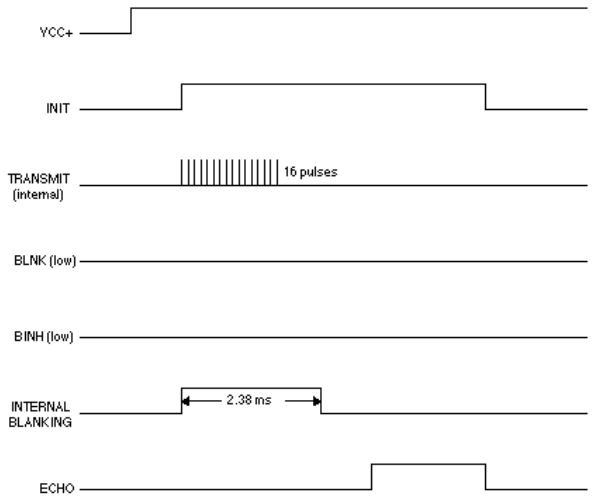


Figure 7: Sonar timing diagram. (http://www.acroname.com/).

crocontroller and the onboard computer which keeps monitoring the state of this byte instructs the microcontroller through the serial port, to stop all motion by disabling the AMPENABLE on the servo-amps.

2.2.3 Motor control

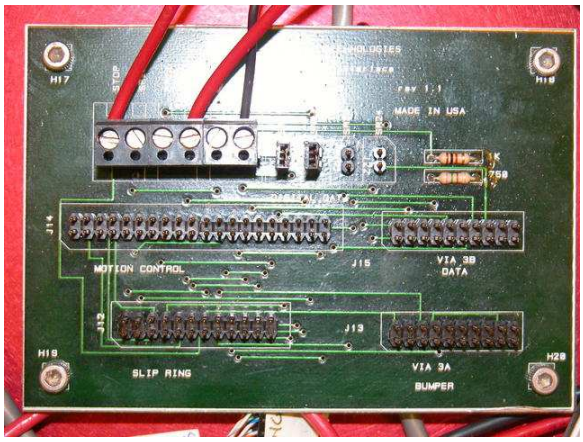
Mobility on the Nomad 200 is provided by three Pittman[1] DC servo motors. The synchronous drive robot has one motor for translation and another for rotation. The motion of the individual wheels are linked together. The third motor controls the angular position of the turret. The robot has a zero gyro radius. In the original design, the GMC-630 motor controller card from Galil Motion Control [2] was used as a dedicated card for motor control. This card has long since been phased out and no documentation regarding its use is available. We contacted Galil Motion Control, to obtain a pin-out of the GMC–630 motor controller card. Using this information we were able to diagnose how to interface the proprietary servo amplifiers from Nomadic Technologies with our new microcontroller. We determined that each servo amplifier in the robot requires a +24 V, a +5V DC and a GND input from the batteries for power. Motor control is accomplished through SIGN, PWM and AMP-ENABLE inputs. The SIGN input determines the direction of rotation of the DC servos, while, varying the PWM duty cycle varies the rate of rotation. We found that a PWM input supplied at a frequency of 23.4 kHz to the amplifiers worked rather well. The microcontroller (see figure 8) provides PWM, SIGN and AMP-ENABLE control to the amplifiers through the slip ring. The DC servo motors also have built-in dual channel encoders which are used for odometry. The turret and wheels have home position sensors. The rotation of the turret and wheels is measured relative to the home position.

2.3 Onboard computer

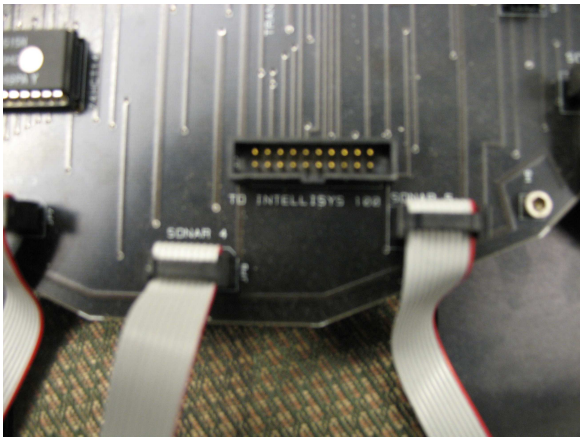
In selecting a new more powerful onboard computer for the Nomad 200 the major constraint was that its power requirements be similar to the original, in order to use the existing power supply. The Evalue ECM-5612 a socket 370 embedded board was chosen. With a Intel Pentium 3 processor installed, the result was an onboard computer with similar power requirements, but with a quantum increase in processing power. The board also has Firewire and USB ports for additional sensors and a serial port interface for communication with the microcontroller. An 802.11g wireless card sits in the PCI slot of the board for communication with a base station. Figure 9 shows the new hard drive computer and the microcontroller sitting in the original chassis. As can be seen the new boards occupy a fraction of the place occupied by the original components. In fact we could mount the hard drive which was earlier mounted on top of the nomad inside the robot, next to the new motherboard board.

[1] http://www.pennmotion.com

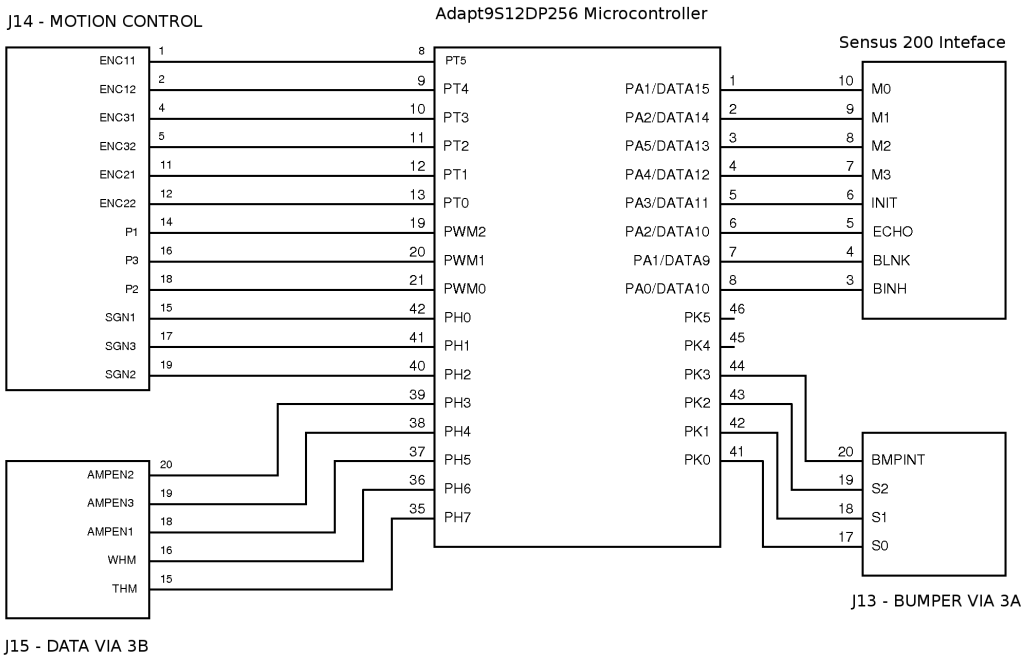
[2] http://www.galilmc.com/



(a) Bumper and Motion Control interface



(b) Interface to the Sensus 200



(c) Shematic describing how to interface the microcontroller with the individual components of the Nomad 200

Figure 8: Integrating the new microcontroller into the Nomad 200.

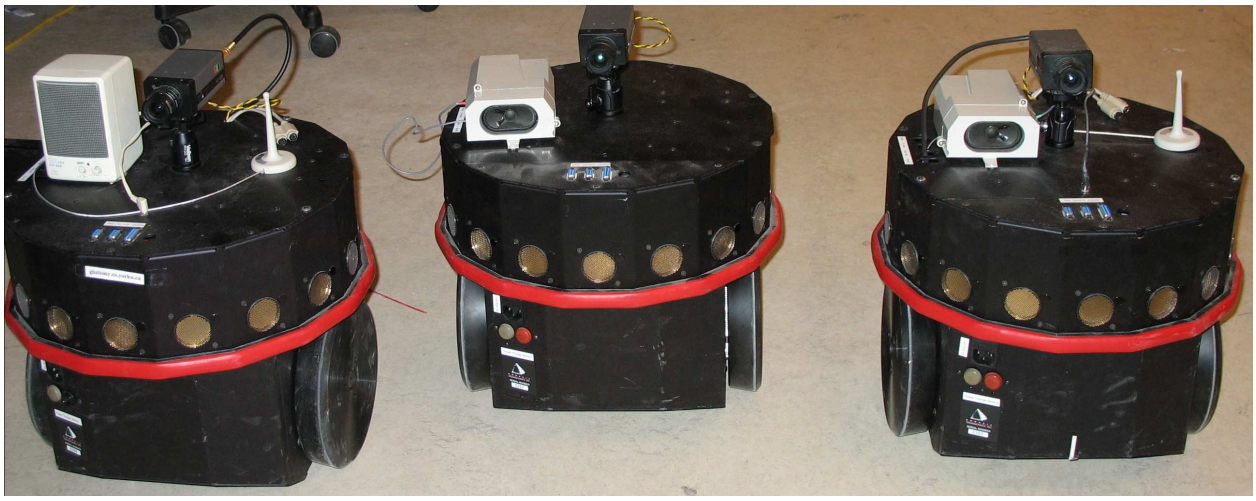


Figure 10: The refurbished Nomad SuperScouts - Gluttony, Lust and Envy.

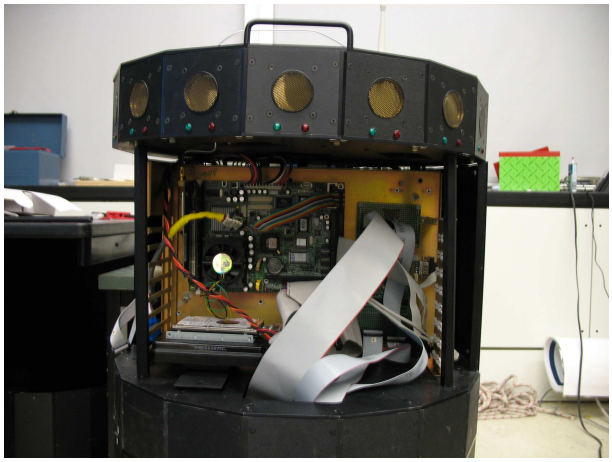
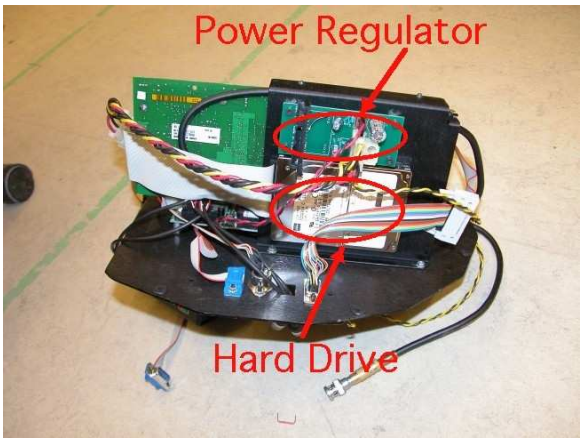


Figure 9: The Nomad 200 with the new computer and microcontroller installed into the original chassis.

3 Nomad SuperScouts

The Nomad Super Scouts are a small robot produced by the Nomad company. These robots were driven by a differential drive which made for easier robot control. The robots used industry standard lead acid batteries and could provide power for an hour (or so) of use depending upon the amount of driving and maneuvering that was required. The robots communicated using a Proxim Wireless system. This allowed the robots to roam around untethered and still be able to communicate with a base station. The Proxim wireless system is not compatible with the 802.11a/b/g networks. The data rate of a Proxim wireless network is very low by todays standards. The CPU's that were installed on the Super Scouts were 200 MHz Pentium II with maximum of 4 MB ram. Video capture is suppled by an RGB video camera and this camera uses manual focus and aperture. The video was encoded using a PCI based video encoding card. The existing mother boards required drivers that are no longer part of the Linux developer tree. There is no way to update the version of Linux on the Super Scouts due to this change in the Linux device tree. In order to run a new version of Linux new mother boards and digitizer hardware is required. Luckily there are a number of hardware options that are available and have full Linux support. The Nomad Super Scout uses a 5.25" form factor Embedded System Mother Board. The motherboard that we choose as a replacement is a low power board from Nova. We originally planned on using three Nova-7810 MB with Pentium III 800 MHz CPU but only two were obtained before this model became unavailable. A similar mother board from Nova-7810 is the Nova-7820 which utilizes a 400 MHz VIA CPU. So we currently operate two SuperScouts with the Nova-7810 board and one with the Nova-7820 board. The Nova 78x0 series of embedded mother boards contain a number of built-in features that are very desirable. The boards are all low power and have four channel video digitizers built in. There is a PCI slot and allows addition of a 802.11a/b/g wireless card. The Nomad Super Scout has the following new hardware: Nova-7810/7820 embedded motherboard; MSI PC54G2 Wireless 802.11g PCI card (based on RaLink 2500 chip-set), with an external antenna. The setup of the system hardware is straight forward. The video feed from the video camera is plugged into one of the channels on the motherboard. The PCI



(a) Original hardware

(b) Bare Top plate with new holes

(c) New hardware mounted

Figure 11: SuperScout top plate.

wireless card plugs into the PCI slot. The installation of all the new hardware requires modifications to the top plate which contains all the new hardware. The robots use the serial port to communicate with the microcontroller in their base.

3.1 Hardware Modifications

Although the SuperScouts are somewhat simpler to refurbish than the Nomad 200, not all of the modifications are straight forward. In the original installation the power regulator and hard drive were mounted to a riser plate that was used to attach the video digitizer card to the top plate and motherboard. This riser plate needs to be removed and the hard drive and power regulator were mounted directly to the top plate [3].

3.2 Software Modifications

The first step in configuring the new hardware is to install Linux on the new computer. We chose to install Debian Linux with a custom 2.6.x Linux kernel. We chose to do a network install of the operating system using a "netinst" CD. The CD contains the minimal amount of software to start the installation of the operating system. The remaining packages are downloaded from the internet. This means an ethernet connection for the installation, as the drivers for wireless internet connection were not built into the installer at the time we installed Linux. We chose a Standard Workstation installation. This type of install will install the X Window System, which allows you to use GUI based applications with a monitor hooked to the computer.

4 Conclusions

In this paper, we described how to refurbish a Nomad SuperScout and a Nomad 200. With some effort we managed to restore these two well known robots, with adequate computing power in them to serve as useful platforms for research. Consider this statistic:
We recently acquired a new robot with a forward looking sonar, a tactile sensor and an onboard computer with a Pentium M processor for $12000 US. The total cost of hardware to upgrade the Nomad 200 was less than $1000 US.

Acknowledgments Many thanks to Matt Robinson for his HCS12 software libraries and invaluable help while interfacing the components with the microcontroller. We appreciate Andrew Hogue's help in troubleshooting motor control for the robot . The financial support of NSERC is gratefully acknowledged.

[3] http://www.cs.yorku.ca/ mobsniuk/robot/

References

[1] Laura Barnes, Todd Quasny, Richard Garcia, and L. D. Pyeatt. Multi-agent mapping using dynamic allocation utilizing a centralized storage system. In *Proceedings of the 12th Annual Mediterranean Conference on Control and Automation*, Kusadasi, Aydin, Turkey, June 2004.

[2] Stephane Carrez. Using the gnu development tools for 68hc11 and 68hc12. Available at http://www.gnu-m68hc11.org/, April 2003.

[3] Motorola Inc. Mc68hc11f1 technical summary. Technical Summary MC68HC11FTS/D, Freescale Semiconductor, Inc., January 1997.

[4] Nomadic Technologies Inc. *Nomad 200 Hardware Manual*, February 1997.

[5] Nomadic Technologies Inc. *User's Manual*, 2.6 edition, March 1997.

[6] Mattias Lindstrom. Nomadic 200 hardware guide. http://www.nada.kth.se/ mattiasl/nomad200/.

[7] Francesco Monica. Nomad. Available at http://rimlab.ce.unipr.it/, July 2005.

[8] MotorolaInc. Mc9s12dp256b device user guide. Technical Report 9S12DP256BDGV2/D, Freescale Semiconductor, Inc., January 2005.

[9] SensCompInc. 6500 series ranging modules specification sheet. Available at http://www.senscomp.com/, September 2004.

[10] E. P. Silva JÚnior, P. M. Engel, M. Trevisan, and M. A. P Idriart. Autonomous learning architecture for environmental mapping. *Journal of Intelligent and Robotic Systems*, 39:243–263, 2004.

[11] TechnologicalArts. Adapt9s12dp256 connector pinouts. Technical Report AD9S12DP256EVPR1a, Technological Arts, 2002.

[12] N.C. Tsourveloudis, K.P. Valavanis, and T. Hebert. Autonomous vehicle navigation utilizing electrostatic potential fields and fuzzy logic. *IEEE Transactions on Robotics and Automation*, 17(4):490–497, August 2001.