

Bringing Home the World of Robotics

Abraham L. Howell

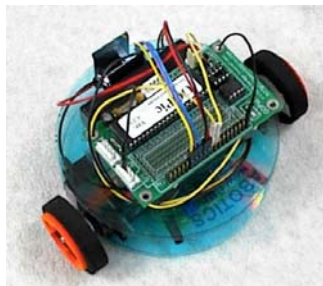
Abe Howell's Robotics

Abstract

Robotics programming is a topic found, for the most part, only at the college level. Students in the K-12 grade levels rarely have the opportunity for direct interaction with robots or robotics programming for that matter. It has been shown that simple remote controlled robots can easily intrigue a student at the middle school level and thereby facilitate the learning process¹. In this paper we would like to take this idea one step further and create a framework for the direct integration of low-cost programmable autonomous robots into a middle school math environment. Learning to program even the simplest robot can be a daunting task if undertaken by someone not familiar with programming or robotics. On the other hand, this experience can be a source of tremendous knowledge and academic confidence for the student. A solution to this problem has been researched and is described within this paper. More specifically the topics to be covered will be the low cost robot kit, software package, methodology for classroom implementation, and the current/ongoing work that is taking place in the Math department at a local Middle School in the Binghamton, New York area.

1. The Low Cost Robot Kit

Past work has identified the need for a low cost robot that could easily and effectively be utilized in an educational environment. The “NewCDBot”, a fully autonomous educational robot that is capable of data collection, was developed to satisfy this deficit in low-cost educational robots. The NewCDBot was born from its predecessor the Original CDBot, a low cost PDA-controlled robot. The Original CDBot was first introduced to 7th grade math students and was used to collect scientific data. After receiving positive feedback from the students and teacher, further development to redesign the Original CDBot was underway and finally yielded the current NewCDBot.



The NewCDBot™ by Abe Howell's Robotics and Original CDBot by Abraham Howell

One main feature of the NewCDBot kit is its low cost, which is approximately \$100, and this price includes its infrared obstacle detection sensors, OOPic™ micro-controller, modified hobby servos, wheels, battery holders, body, and programming cable. Another key feature of the NewCDBot is the fact that it can be completely

disassembled and then reassembled multiple times. This is very important when used in an educational environment. A low cost robot that can't be easily disassembled and then reassembled loses some of its teaching value. Since the NewCDBot can be taken apart, students will be able to investigate and learn about the different components that are used in building a fully functioning autonomous mobile robot. The NewCDBot is also environmentally friendly since it puts to use recycled CDs.

The NewCDBot has been designed to be low cost and it therefore makes sense that the body of the NewCDBot be designed from a readily available low cost material, the compact disc. What holds the body and other components together is a remarkable product from 3M™, low profile dual lock™ reclosable fastener. The reclosable tape allows for the easy assembly and disassembly of the NewCDBot robot. Reclosable tape is also quite inexpensive when compared to other fastening methods such as machined mounts or brackets that would accept bolts or screws. Not only would these methods be more costly than the reclosable tape, but would not allow for quick and easy assembly and disassembly of the robot. These are the reasons why reclosable tape was chosen over other methods of fastening.

Two modified hobby servos with attached foam wheels and a unique rear roller wheel form the NewCDBot's differential drive train. Hobby servos are without a doubt one of the cheapest providers of locomotion available for building a robot today. All standard hobby servos encapsulate a miniature gearbox, which enables it to provide enough usable torque to move a small robot. However, there is one small problem with standard hobby servos; they only allow for approximately 180 degrees of rotational freedom. This is not sufficient for use as a robot drive motor, since a robot's drive motor must be capable of continuous rotation. To remedy this issue a simple modification can be performed on the standard servo, which will convert it into a continuous rotation servo. Hobby servos were chosen due to their low cost, significant output torque, speed control, and modular design. The servos on the NewCDBot also provide good structural support for the body of the robot and its other components.

The two foam wheels and plastic roller wheel allow the NewCDBot to freely roll across a relatively smooth surface. The NewCDBot is not able to traverse rough terrain, but does well on tile, linoleum, and low nap carpet. The plastic rear roller wheel allows the NewCDBot to spin freely and make full use of its differential drive system. There are many robot designs that do not use a roller wheel, but instead make use of Teflon skids, which will definitely degrade the robot's maneuverability. For these reasons alone it was decided that a roller wheel would be used in the design. The focus of the NewCDBot's design was to minimize cost without sacrificing the overall functionality of the robot.

The brain of the NewCDBot is an OOPic I micro-controller. The OOPic is a great micro-controller for introductory and advanced applications. The OOPic can be programmed in C, basic, or java. This multi-language capability makes the OOPic attractive, since it can be used to teach three different programming languages. Savage Innovations Inc. offers the OOPic Multi-Language Compiler as a free download on their OOPic website: <http://www.oopic.com> This is a nice feature of the OOPic, since some

micro-controllers require the user to buy expensive high-level compilers before they can even begin programming their robot.

An object-orientated approach is used when it comes to programming the OOPic. Programming the OOPic using Basic syntax is much like writing code in Visual Basic. Therefore, many Visual Basic programmers will be able to begin programming the OOPic almost immediately. A PIC16F877 Microchip is at the heart of the OOPic and also provides the OOPic with a vast array of hardware tools. The OOPic only provides access to 4 out of the possible 8 analog to digital (A/D) converters that are on the PIC16F877. The OOPic does however maintain a large quantity of predefined objects, which can aid the user in building a complicated program that has tremendous capability. For example, there is an oServo object that allows the user to connect a hobby servo to the OOPic and then control its motion. To use this oServo object the user only needs to declare an instance of the object then configure the object's properties and it is ready for use in the program.

Two on-board memory slots, which are upgradeable, allow the OOPic to be flexible when it comes to the memory department. One of the memory slots is used for storing the user's program code, while the second slot can be used for storing data or whatever the user desires. The OOPic comes equipped with a 4K Bit EEPROM in the first slot. Either EEPROM can be upgraded all the way up to a 64K Bit EEPROM. This would allow the OOPic to run either a larger program or store large amounts of data, depending upon which memory slot was upgraded.

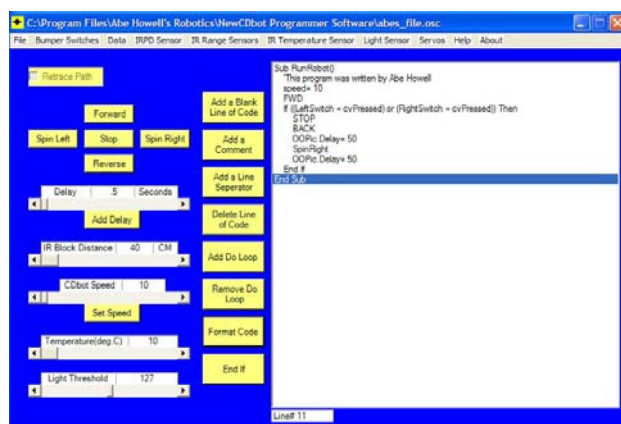
Access to an I²C™ bus is also available on the OOPic. This can allow for the connection of multiple I²C compatible devices. Such devices can be in the form of sonar sensors, infrared temperature sensors, speech synthesizers, digital compasses, accelerometers, or even additional micro-controllers other than the OOPic. Multiple OOPics can also be connected together through the I2C bus. All in total up to 128 I2C devices can be connected to one OOPic controller. There is a pre-defined OOPic object, the oI2C, which allows the user to connect, configure, and use any I²C compatible device in a matter of minutes.

There is one minor problem with the OOPic that can be easily and quickly resolved. A 100ma on-board voltage regulator comes standard on the OOPic I. This on-board regulator is capable of supplying power to a few connected devices provided that they don't draw a substantial amount of current. For example, if using Sharp GP2D12 infrared sensors, it is best to limit your total number of sensors to only one or two. The GP2D12 draws approximately 33ma, so two of them will draw about 66ma and this leaves 34ma for the processor, which is sufficient. However, once your battery supply begins to drop a little, the processor may not be supplied with enough current and will therefore reset itself. This low power reset can be a real problem to troubleshoot when trying to debug your code, since you may think that it is your code causing this unpredictable behavior, but it is actually the low power and insufficient current supply. Detailed instructions for replacing the 100ma regulator with a 1000ma regulator are available on the OOPic website. This modification will allow multiple devices to be

connected to the OOPic without any loss in performance or troublesome processor resets. Even after the voltage regulator upgrade one must still be careful not to go over the new 1000ma current limit.

2. The Software Package

The “NewCDBot Programmer Software” (NCPS) is a free GUI based software package that is specifically designed for use with the NewCDBot robot or any other robot that utilizes the OOPic and a differential drive system consisting of two modified hobby servos. The NCPS was designed and developed by Abraham Howell and works in conjunction with the OOPic Multi-Language Compiler². Savage Innovations Inc is responsible for the development of the OOPic Multi-Language Compiler and provides the software free of charge. The OOPic Multi-Language Compiler is responsible for compiling and downloading the code created by the NCPS. The OOPic Multi-Language Compiler is easy to use if you are already somewhat experienced with programming in C, Basic, or Java. However, if you have never programmed in any of the above listed languages, writing code for the OOPic can quickly become an arduous and unrewarding task. The NCPS was developed to help overcome this temporary obstacle in robotics programming. Students can create fairly complicated code using the NCPS and without having to worry about all the tedious details related to proper coding syntax. The NCPS generates the code using Basic syntax. Using a desktop or laptop computer, students can create code for their NewCDBot using the NCPS GUI and thereby give it some form of simple behavior in only a matter of minutes.



Screenshot of the NCPS.

The NCPS has built on the idea of creating predefined functions so that students can easily learn to program their robots. Previous work at the University of Missouri-Columbia focused on creating pre-defined functions so that students could program their robots during an outreach activity named, Saturday Science³. This work proved beneficial since the students were able to create their programs using easy to understand functions. We have taken this one step further with the NCPS by creating a complete GUI interface, which encapsulates all the predefined functionality of the robot.

Students primarily interact with the NCPS through an easy to use graphical user interface (GUI) and write code by simply clicking buttons, moving scrollbars, or

selecting items from the program's drop-down menu. After clicking a specific button the student will immediately see their code generated in the Code Window. This is a great way for the student to learn how to write code using Basic syntax. The software only displays code from within the "RunRobot()" sub routine, which is the only code that the student is allowed to interact with and does not yield access to the pre-existing code template that is the backbone to their program. The code template declares, initializes, and configures all the objects and variables that are used throughout the generated program. The template also sets up various sub routines that enable the NewCDBot to collect data, save data to the second EEPROM chip, transmit/receive collected data through its serial port, check a specific sensor's status, and control the NewCDBot to move with a specified motion. The amount of code required to perform all the above tasks is immense and would easily discourage any student interested in learning to program robots.

Once a student has created their program and saved it to disk they can compile and download it directly to a connected NewCDBot by simply selecting Compile and Download and then pressing the F5 key. At this point the OOPic Multi-Language Compiler will load and begin to compile the student's program. If compilation is successful the program will then be downloaded to the connected NewCDBot. Many thanks go to Savage Innovations Inc. for making their OOPic Multi-Language Compiler readily available as a free download. Once the code has been compiled and downloaded the student may browse over their entire program and view the code template that was generated by the NCPS. This feature allows the student to explore the behind the scenes code that they weren't able to see before when they were constructing their program. By exploring the code, students can begin to learn how to write code on their own. After students become familiar enough with writing code they can begin to develop programs using the free OOPic Multi-Language Compiler software from Savage Innovations or they may even venture off and investigate a completely new micro-controller and programming language.

The NCPS currently supports a total of five different types of sensors and they are as follows: Sharp GP2D12 Infrared Range Sensor, Lynxmotion Infrared Proximity Sensor, Melexis MLX90601 Infrared Thermometer Module, light sensing photocell sensor, and single pole double throw (S.P.D.T.) miniature long lever snap action switches. All of the above sensors are used for obstacle detection except for the MLX90601, which is used to collect temperature measurements and the photocell, which is used to detect changes in light intensity. These sensors need to be properly connected to the OOPic micro-controller, but once connected the individual sensors can easily be referenced and utilized in a program.

The Sharp GP2D12 is an infrared sensor capable of detecting an obstacle that is within a 10-80cm distance from the sensor's face. This is a great feature, since some detectors will only provide a digital signal notifying whether or not the sensor is blocked, but with the GP2D12 a fairly precise distance to the detected obstacle can be determined. This sensor only requires three connections to the OOPic and it is ready for use. The analog signal from the GP2D12 must be converted to a corresponding distance

measurement so that the NewCDBot can determine exactly how far away a detected obstacle is and whether or not to take action to avoid collision. Another benefit of the GP2D12 is that it is small and can be reconfigured on the NewCDBot to “look” in different directions; this is not possible with some of the other obstacle detection sensors. A disadvantage with this sensor is that there must be an 8-bit Analog to Digital (A/D) port on the corresponding micro-controller to be used with this sensor. This sensor also requires more coding on the end of the programmer. The analog output from this sensor is non-linear and code must be written to approximate the distance of a detected obstacle using the 8-bit conversion result from the analog signal. This can be as simple as pre-determining several points and then using some form of interpolation to determine the distance from the 8-bit value. The method of distance calculation to be used depends upon the required accuracy of the result. High accuracy is not a concern with the NewCDBot and therefore a simple form of interpolation can be used to calculate the detection distance.

The Lynxmotion IRPD-01 is also an infrared sensor capable of detecting an obstacle, but only has a digital output signal that corresponds to whether or not one of its sensors is blocked and if so which side is currently blocked. The IRPD-01 has an adjustable detection of range of 4-26”. Adjusting the on-board potentiometer sets the detection distance. The IRPD-01 requires a total of five connections to the OOPic, two of which are used to power the sensor. One advantage of the IRPD-01 is that it comes furnished with all the necessary connectors for attaching it to a standard micro-controller. This allows for an almost “Plug and Play” setup of the sensor. The IRPD-01 is small at only 2.3”L x 0.75” W. The IRPD-01 is also easy to configure in the code. A clear disadvantage is that the sensor only provides status as to whether the sensor is blocked and which side is blocked. The IRPD-01 also uses two additional I/O lines when compared to the GP2D12. This is not an issue with the NewCDBot, but can become an issue when multiple sensors are used. The two extra I/O lines need only be regular I/O pins and not an A/D capable pin as was the case with the GP2D12 sensor. The IRPD-01 only consumes approximately 8mA, while each GP2D12 will consume about 33mA.

The easiest to use sensor for obstacle detection is the SPDT Miniature Long Lever Snap Action Switch. This sensor is the easiest to use, but must make contact with the obstacle before a detection can be made. The snap switch also uses the fewest number of I/O lines, so multiple snap switches can be used while still conserving micro-controller I/O. A single snap switch will require one I/O line and a connection to ground. This type of configuration is called active low and would normally require the connection of a pull-up resistor between +5 volts and the I/O line, but the OOPic provides access to internal pull-up resistors that can be used instead. One final advantage to the snap switch is that approximately ten snap switches can be purchased for the price of two GP2D12’s or one IRPD-01.

The Melexis MLX90601 Infrared Thermometer Module is capable of measuring the temperature of an object without touching the object. This non-contact temperature measurement is possible since the MLX90601 utilizes infrared technology to measure the temperature of an incident object. The MLX90601 only requires one I/O line and power

to function. The I/O line must be A/D capable since the signal from the MLX90601 requires an 8-Bit A/D conversion. The analog signal from the MLX90601 is linearly proportional to the measured temperature of the incident object.

The photocell empowers the NewCDBot with the ability to quantify changes in light intensity and react according to the programmer's intentions. The photocell must be arranged into a voltage divider with the appropriate size resistor so that an A/D converter on the OOPic can measure the changes in voltage, which correspond to the changes in light intensity. The photocell requires a total of three connections to the OOPic. The output signal from the voltage divider must connect to an A/D capable pin on the OOPic.

3. A Methodology for Classroom Implementation

The NewCDBot Programmer Software package and NewCDBot robot can be integrated into a classroom in an easy and cost effective manner. The NewCDBot Kit can be acquired from Abe Howell's Robotics at www.abotics.com for approximately \$100 and includes extensive documentation with regards to its assembly and classroom use. The NCPS software is available as a free download. Abe Howell's Robotics has developed several classroom lab modules, which are also available from the website. Each of these lab modules will supply the teacher with explicit instructions for how to prepare, setup, and perform the specified lab. Additionally, each module provides several example questions that may be asked of the students before they perform a lab. Students can be asked to revise their answers to these example questions after they have completed the lab. These questions can also be used to help understand what the students learn from each of the lab experiences. Mr. Howell will continue to work with local schools to revise and develop new classroom modules.

Students can work in groups of two or four to assemble their NewCDBot kit. The group sizes depend upon the number of available kits. Detailed assembly instructions are to be provided to each of the student groups. The assembly instructions contain full color pictures that aid in the assembly process. The NewCDBot Kit comes standard with all the necessary soldering and machining already completed. Students only need to follow the instructions to properly place the reclosable tape and snap together all the robot's components, properly connect the wires, install the batteries, and finally get ready to program their robot!

After the NewCDBot kits have been handed out, students should be asked to verify that their kit contains all the necessary components. This will help students to become familiar with the various parts that make up a complete robot. A brief description of each component and its function within the robot should be discussed at this time. For example, it should be explained that the two hobby servos provide a means of locomotion for the robot.

Once a brief description about each of the components has been provided, students will be ready to begin the actual assembly process. At this time students can begin assembling their NewCDBot and may require assistance or have questions with regards to the assembly process or may just have an inquisitive question about the robot.

The assembly process will progress smoothly if there are additional assistants available to help and provide answers to student's questions. Students should not be given their 9-volt battery until an assistant has verified that their wiring is correct. This step will help to prevent any possible damage due to improper wiring and will ensure a positive experience for the students. However, the four AA batteries can be given to the students at this time.

Students can be given a 9-volt battery after their NewCDBot's wiring has been verified. Once the batteries have been installed, students will be ready to test their robot. A test program should have been pre-loaded into the robot. This will allow the students to test that their NewCDBot is working properly. The test program should drive the NewCDBot forward until one of its sensors is blocked at which time it should backup and spin in a direction such that it will avoid the detected obstacle. Students can test their robot by building a simple maze, obstacle course, or just by letting it roam around the classroom.

After a group has verified the proper operation of the robot by using the test program, they can begin writing their own programs using the NCPS. At this point the students may be asked to write a specific program or can be allowed to write a program of their own design. Specific tasks for the student groups may be as simple as programming their robot to drive around while avoiding obstacles or as complicated as programming their robot to trace out a simple geometric shape like a square or triangle. One popular learning technique is to have a group write their program and then post their code on an overhead projector for the other students to examine. After examining the code students can then be asked to explain how they think the robot might behave if the displayed program were downloaded and executed. This will help to reinforce their programming skills.

Programming the NewCDBot to collect a set number of "Velocity" and "Time" data points can be accomplished with the NCPS. However, an additional EEPROM must be installed in the OOPic's second EEPROM slot, E₁, and a serial data downloading cable is also required. Using the NCPS, students can specify the total number of data points for their NewCDBot to collect. There are two ways the NewCDBot can be programmed to collect data: (1) it will collect the specified number of data points and then stop so that the data can then be downloaded, or (2) it will collect the specified number points at which time it will then begin to retrace its path back to where it started, so that the data can be downloaded. After programming for data collection, the students can send their NewCDBot out to collect the data. When the robot is done collecting the specified number of data points, students bring it back to a desktop computer or laptop and download the data using the NCPS. The data can be downloaded using the appropriate serial data downloading cable. Once the NewCDBot is properly connected to the computer the data can be downloaded using the NewCDBot Data Downloader, which is built into the NCPS. After all the data has been downloaded, the NCPS allows the collected data to be saved to two separate files, one for velocity data and the other for time data. Students can name each of these two files with whatever name they feel is appropriate. Students will then be able to use the saved data to create spreadsheets and

graphs of “Velocity versus Time” and “Distance versus Time”. A discussion about how the graph relates directly to the motion of the robot will help to solidify these new ideas.

4. Current and Ongoing Work

The above-described method has been partially implemented into a local middle school’s after school math club, the Vestal Math Counts Math Club. The inventor of the previously described robots and owner of Abe Howell’s Robotics, Abraham Howell, has already spent time with the 7th grade students and is also working directly with their teacher and manager of the Math Counts math club, Cathy Jeremko. Students have learned to program their NewCDBots with basic forms of behavior using the NCPS. This work was initiated at the end of the 2003 spring semester and was therefore cut short due to the end of the school year. The Math Counts project will startup again in the 2003 fall semester at which time students will learn to program their robot for specific tasks. Programming their robot to trace out a specific geometric shape might be one such task that is delegated to the students. Students can also use basic physics formulas and experiments to calculate the actual speed of their robot and then use this information in their programs for solving specific distance based problems. Currently, the robots are only outfitted with infrared sensors for obstacle detection, but we are going to add photocell light sensors and EEPROM chips in the fall. Students will gain additional insight and knowledge into the physical sciences once they complete several experiments with the new photocell light sensors and EEPROM chips.

5. Conclusion

This research has helped to bring the world of robotics and robotics programming down to the level of middle school math students. Robots don’t have to cost thousands or even hundreds of dollars to be effective and efficient teaching tools. Schools with a small budget can take advantage of the many teaching benefits offered by the NewCDBot due to its low cost. Using the NewCDBot to teach math and programming has been the primary focus in this paper, but it should be noted that students in other courses could also benefit from working with the kit. The work discussed in this paper has been with middle school math students, but could easily be implemented at the high school and even college level. Freshman and sophomore level students could gain insight and prepare for various student design competitions by working with the NewCDBot kit. Future research will focus on developing additional classroom modules along with an upgrade kit for the NewCDBot. This upgrade kit will take the majority of the components from a NewCDBot kit and utilize them in a larger more capable robot platform. This upgrade kit will be designed to be low in cost, since it will primarily consist of a larger body, wheels, and other various pieces of required hardware. A larger robot platform will also provide students with the possibility of experimenting with multiple sensors such as cameras, ultra-sonic transducers, and even mechanical grippers.

6. References

- [1] Howell, Abraham L., McGrann, Roy T.R., “Using PDAs on Autonomous Robots to Promote Engineering to Middle School Students”, 2003 A.S.E.E. Annual Conference & Exposition, June 22-25, 2003, Nashville, Tennessee.
- [2] Savage, Scott. OOPic Multi-Language Compiler, 1999. Savage Innovations Inc. 10 March 2003 <http://www.oopic.com/OOPicSV501.zip>.
- [3] Skubic, Marjorie, “Saturday Science Robots for Junior High Students”, Dept. of Computer Engineering and Computer Science University of Missouri-Columbia, Columbia, Missouri.

7. Biographical Information

ABRAHAM L. HOWELL

Founder/Owner of Abe Howell’s Robotics

Area of research is robotics, especially small autonomous robots, design, prototyping, sensors, and software development. Abraham is also involved in the development and integration of low cost robots into K-12 curriculums. Abe Howell’s Robotics was founded to further facilitate the development of low-cost robots for use in education.