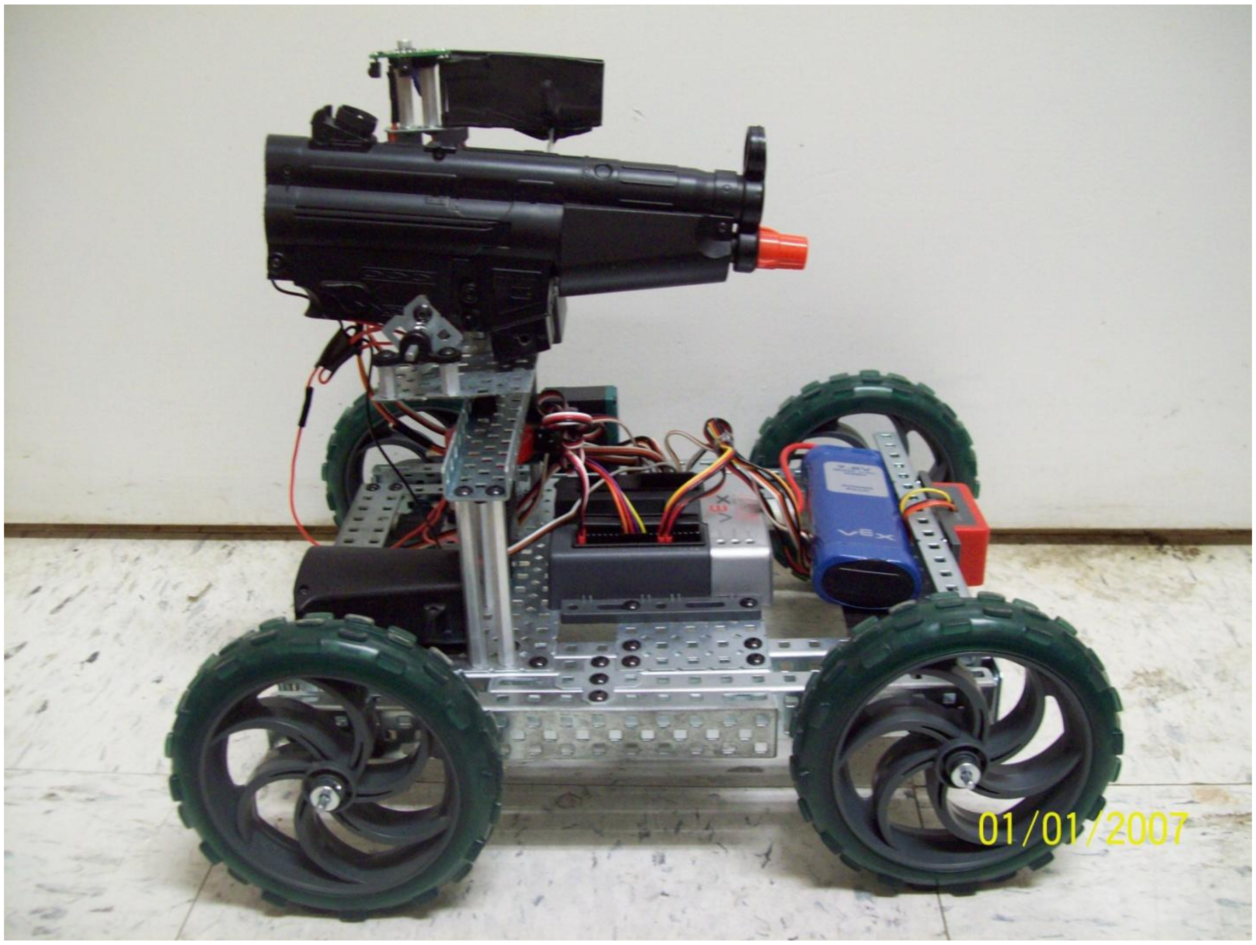
ARES User's Guide

A User's Guide for: Mobile mode, Stationary mode and Remote Controlled mode



ARES Designed, Developed and Tested By: Nick Handelman

User's Guide By: Nick Handelman

Senior Project: Fall 2010 – Spring 2011

Millsaps College – Computer Science Department

Table of Contents

1. Introduction and Getting Started……………………………………………………………Page 2

2. Mobile Mode……………………………………………………………………………………………..Page 3

3. Stationary Mode……………………………………………………………………………………Page 4

4. Remote Controlled Mode…………………………………………………………………….Page 5

5. The Enemy………………………………………………………………………………………………Page 8

6. Appendix A – Connections……………………………………………………………………..Page 9

7. Appendix B – Hardware…………………………………………………………………………Page 11

8. Appendix C – Software and Support………………………………………………………Page 13

1. Introduction and Getting Started

ARES (Autonomous Robotic Engagement Sentry) is a robot of the land vehicle variety. Constructed with parts from multiple companies (See Appendix B), ARES features three distinct operating modes: Mobile mode, Stationary mode and Remote Controlled Mode. The former two operate autonomously. The latter mode is controlled by a human operator. This guide describes each of these modes in addition to a description on the operation of the target.

1.1. Initializing ARES

Before any of the modes can be loaded, the Master Firmware, VexNet Joystick Firmware and ROBOTC firmware must be loaded onto the Cortex Microcontroller. First, follow the directions in Appendix A to physically connect ARES to a computer. Once the connection is established, click on the “Robot” tab in the menu, go down to “Download Firmware”, go right to “Master CPU Firmware” and click “Standard File”. Once the file has loaded to the Cortex, physically connect the remote controller to a computer (See Appendix A). Once the connection is established, click on the “Robot” tab in the menu, go down to “Download Firmware”, go right to “VEXnet Joystick Firmware” and click “Standard File” Once the file has loaded to the remote controller, physically connect ARES again to a computer. Proceed to the same menu as before, but select “ROBOTC Firmware” and then click “Standard File”.

Note: This step must be performed if the Cortex is ever reset.

2. Mobile Mode

Mobile mode is one of two primary features of ARES. Through the use of a pair of Infrared (IR) Sensors, ARES is capable of tracking any target on which the paired IR sensor is located (See Section 5). In this mode, ARES uses its four-wheeled drive system to chase down its target and attack it using the Airsoft gun mounted on the turret. ARES is programmed to shut down after it has fired a certain number of times.

2.1. Loading Mobile Mode to ARES

Open RobotC and choose "File" in the menu, then select "Open and Compile" and select the file named "mobile mode.c". After this file opens, make a connection to ARES (See Appendix A), then go to "Robot" in the menu and select "Compile and Download Program". The program will load a small screen and will present options such as "Start". The file is now loaded to the Cortex and the connection can be removed.

2.2. Starting ARES

Now, ARES should be powered off but the battery should be connected. First, load the BB's into the Airsoft gun (this is optional). Next, connect power to ARES's IR sensor using the 9-volt battery placed closest to it. Ensure the batteries in the Airsoft gun are locked into the battery compartment. Finally, prepare the target for battle. To begin, flip the switch on the Cortex to "ON" and the program will load in about 7-10 seconds.

3. Stationary Mode

Stationary mode is the second primary feature of ARES. Through the use of a pair of Infrared (IR) Sensors, ARES is capable of tracking any target on which the paired IR sensor is located (See Section 5). In this mode, ARES acts a stationary turret with its back against a wall since the rotating turret can turn only approximately 220 degrees. It uses this turret to face its target and attack it using the Airsoft gun mounted on the turret. ARES is programmed to shut down after it has fired a certain number of times.

3.1. Loading Stationary Mode to ARES

Open RobotC and choose "File" in the menu, then select "Open and Compile" and select the file named "stationary mode.c". After this file opens, make a connection to ARES (See Appendix A), then go to "Robot" in the menu and select "Compile and Download Program". The program will load and a small screen will present options and ask you to start. The file is now loaded to the Cortex and the connection can be removed.

3.2. Starting ARES

Now, ARES should be powered off but the battery should be connected. First, load the BB's into the Airsoft gun (this is optional). Next, connect power to ARES's IR sensor using the 9-volt battery placed closest to it. Ensure the batteries in the Airsoft gun are locked into the battery compartment. Finally, prepare the target for battle. To begin, flip the switch on the Cortex to "ON" and the program will load in about 7-10 seconds.

4. Remote Controlled Mode

Remote Controlled Mode is the secondary feature of ARES. In this mode, ARES is at the user's disposal. Using the remote controller, she can control all of ARES's functions. In this mode, the IR sensor is disabled. However, the sonar sensor continues to function normally and presents the one case where ARES will be taken from user control. If the sonar sensor detects an obstacle with a set amount of inches, it will not allow the user to move forward. This prevents it from running into objects head on and causing damage. A wireless camera is installed for covert operations.

4.1. Loading Remote Controlled Mode to ARES

Open RobotC and choose "File" in the menu, then select "Open and Compile" and select the file named "remote controlled mode.c". After this file opens, make the proper connection to ARES for wireless operation (See Appendix A), then go to "Robot" in the menu and select "Compile and Download Program". The program will load and a small screen will present options and ask you to start. Note that the Cortex must be on for VEXnet to be active. At this point, click start and ARES is now under user control. The orange serial wire can be disconnected from the remote controller.

4.2. Remote Controlled Operation

Refer to Appendix B for a picture of the controller to identify the joysticks and buttons with their channel or button number.

4.2.1. Right Analog Joystick

The x-axis of this joystick is mapped to Channel 1. The y-axis is mapped to Channel 2. This joystick has no function. The four wheel drive operations are discrete, meaning only one can occur at a time. For this reason, it would be very difficult and tedious to control ARES using an analog joystick.

4.2.2. Left Analog Joystick

The x-axis of this joystick is mapped to Channel 4 and the y-axis is mapped to Channel 3. It provides one of two options for controlling turret rotation and panning operations. The joystick is useful for these two operations since they are independent of each other and may occur simultaneously. Pushing up or down will pan the gun up or down. Pushing left or right will rotate the turret counterclockwise or clockwise.

4.2.3. Buttons 5U and 6U (Upper Shoulder Buttons)

Both of these buttons perform the same function since some may have a preference over which hand they use to fire the mounted Airsoft gun. To fire, simply hold either button down. Release that button to stop firing.

4.2.4. Buttons 5D and 6D (Lower Shoulder Buttons)

When both of these buttons are pressed, they issue a shut down command to ARES which ends the program. Individually, these buttons have no function.

4.2.5. Button Pad 7

These buttons control the driving operations. Button 7U issues a move forward order. Button 7D issues a reverse order. Button 7L issues a turn left and Button 7R issues a turn right. **Note: the four wheel drive motors on ARES all operate at one uniform speed.**

4.2.6. Button Pad 8

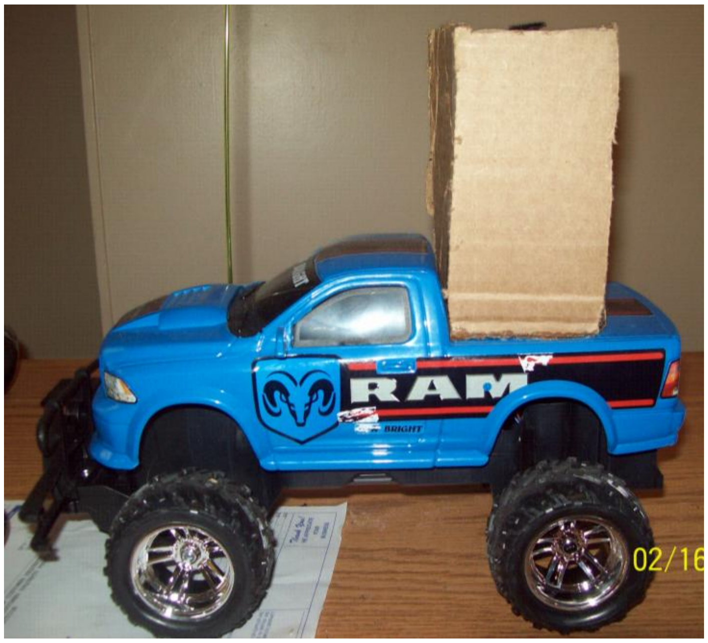
These buttons provide the second option for controlling turret rotation and panning operations. Button 8U issues a pan up order. Button 8D issues a pan down order. Button 8L issues a rotate counterclockwise and Button 8D issues a rotate clockwise.

5. The Enemy

The targeted enemy is a small remote controlled truck with a cardboard tower mounted on the bed. The IR sensor rests on top of the tower with its emitters slightly over the edge. This helps prevent the IR sensor from getting hit by a BB from the Airsoft Gun. Within the tower is located the power source for the IR sensor: a 9V battery. Finally, a simple remote controller is included that allows for forward/reverse and turning movements.

5.1. Initializing the Enemy

This step is simple and requires only three parts. First, ensure that both the RC truck and remote controller have batteries. Next, slip the cord connected to the 9V battery within the tower through the small slit in the top of the tower. Connect it to the IR sensor to power it on (See Appendix A). Finally, turn the RC truck on. The switch is located on the bottom of the truck.

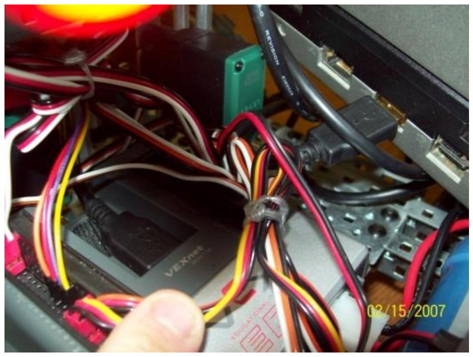


6. Appendix A – Connections

There are three physical connections for which the user is responsible. One involves connecting the Cortex or remote controller to a computer via a male-male USB cable. The second involves connecting one VEXnet USB stick to the Cortex and the other to the remote controller, in addition to connecting the orange two-part serial-USB from the remote controller into a computer. Finally, the user must connect the power source to the IR sensors before they will be functional.

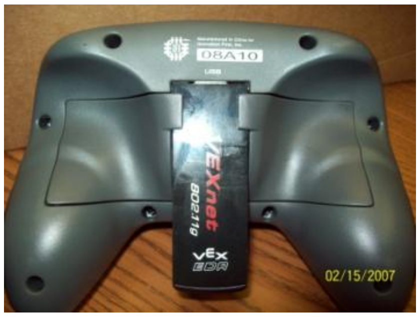
6.1. USB Cable connection

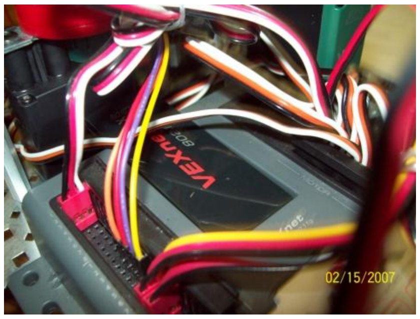
Plug one end of the USB cable into a computer and plug the other end of the USB cable into the Cortex microcontroller, as pictured to the right.

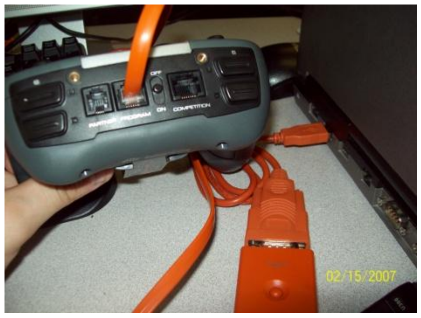


6.2. VEXnet Connection

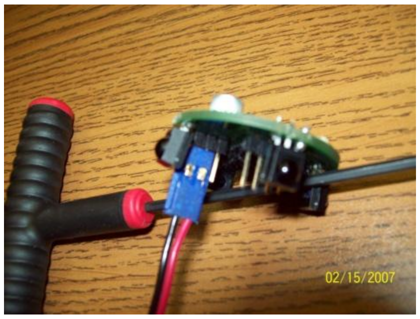
Connect one VEXnet USB stick to the remote controller. Connect the other to the Cortex Microcontroller. Finally, plug the serial cable into the controller in the slot labeled "Program" and plug the USB end into a computer. Turn both the remote controller and the Cortex on. This is for wireless connectivity and is necessary for the Remote Controlled Mode to function.







6.3. IR Sensor Connection



Locate the three-pronged connector on the bottom of the IR sensor. Now, notice that the prong nearest the outside has a negative (-) label above it. The connector to the power source should be oriented in a way such that the black wire connects with the negative (-) prong.

7. Appendix B – Hardware

[1]

7.2V Battery --

Powers the Cortex Microcontroller

9V Battery ---

Powers the IR sensor

Airsoft Gun (Modified) ---------------------------------

Battery Compartment on the Left

Firing Mechanisms on the Right

[1]

Battery Charger --

Recharges the 7.2V Battery

[1]

Cortex Microcontroller --

The "brain" of ARES. The Cortex controls all motors, sensors and other inputs using the uploaded program.

[2]

Infrared (IR) Sensor---

6 Infrared emitters send out an infrared signal

4 Infrared sensors detect incoming infrared signals

4 Signals are output to Cortex indicating the direction of IR paired sensor

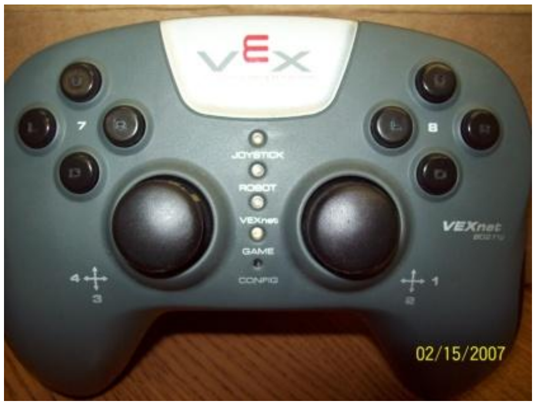
3 Pronged connector to receive a connection to a power source[1]

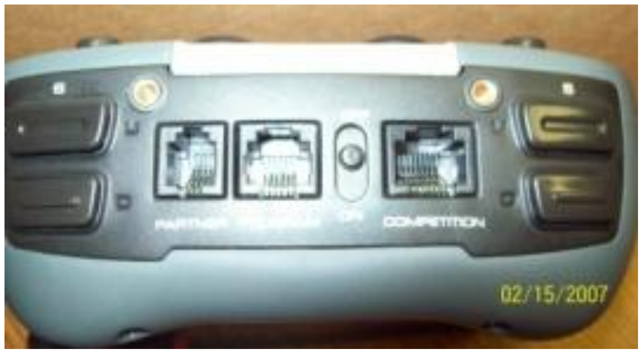
[1] Pictures from http://www.vexrobotics.com/
[2] Picture from http://www.pololu.com/

Infrared Sensor View Limiter--

Narrows the "front facing" IR sensor's field of view

Remote Controller (Top and Front View) --------------





Serial to USB Cable ---[1]

Connects a computer to the Remote Controller

Sonar Sensor ---[1]

Emits ultrasound waves to detect obstacles in ARES's path

USB to USB (Male – Male) Cable ---

Connects a computer to ARES

VEXnet USB stick --[1]

Used to communicate between ARES and the remote controller

Wireless Camera --[1]

Consists of a camera and a receiver. The camera is

mounted on ARES so a live action feed can be watched remotely.

8. Appendix C – Software and Support

The software used in the development of this project is "ROBOTC for Cortex and PIC 2.30", created by the Carnegie Mellon Robotics Academy. Included with the software are functions to interface with a Cortex or PIC microcontroller, a RobotC function library and sample programs to help get a feel for and understanding of the software.

Additionally, two support forums were quite useful for finding answers to difficult problems. The first, http://www.robotc.net/, was a great resource for learning about the RobotC language and solving software related problems.

Secondly, the Vex Robotics website, http://www.vexrobotics.com/, offered many resources to help with the development of this project. The forums were helpful for answering questions not covered by the plethora of documentation and support made available by the website.

Ultimately, learning the software and debugging the code came down to just practicing with the language to see what did and didn't work, and testing the code in a live environment, i.e. the Cortex microcontroller.