Volume : 1 | Issue : 1 | January 2012 ISSN : 2250 - 1991

Research Paper Engineering

ROBOCODE
(A robotic java based gaming platform)

* Abhishek Tripathi ** Manish Anand

* 09BIT063

** 09BIT093

ABSTRACT

Robocode is a robotic battle simulator where robots in the form of tanks compete to survive in the battleship. All the robots are operated with the help of JAVA code embedded inside it. We consider this game a good choice for testing agent programming strategies and an excellent learning opportunity for those interested in JAVA. Besides providing a healthy environment to develop competitive skills, this game also has a lot to explore in the field of learning that altogether makes it a wonderful and enjoyable experience.

Keywords : Robocode, Java, Robotic

Introduction

Robocode is a Java based programming game platform developed by IBM Alpha works. The main objective of this game is to program a robot battle tank to fight with other robot battle tanks. The programs controlling the tanks are written in Java and are executed as threads in the main program. During the game the tank must navigate around its environment, avoiding the walls, bullets fired by the other tanks. At the same time it must locate the other tank, anticipate its likely position and try to hit it or ram it. To get started for novices some sample robots are already been given. In addition there are multiple resources on the internet illustrating advanced strategies such as pattern matching, gravity movement and wave surfing that makes your robot more competitive and skillful to face any challenge. There is a Robocode league as well, where different robots from all across the globe compete against each other with the view of taking league table honors.

Fig.1. Robots on battlefield

Prior Works

Robocode is an easy-to-use framework that was originally created to help teach object-oriented programming in Java. The project was started by M. Nelson in late 2000 and was available through IBM's Alpha works in July 2001. Now there is a strong open-source community around it with competitions running all over the world.

Aim of this paper

Although the writing of the Java programs to control such robots is in itself an interesting task, the question of how to use such robots in practical, gains much importance. Can such robots be implemented in the defense systems of any country, if yes then to what extent this technology can be exploited so that it doesn't poses any threat to the peace and serenity of the nation. Also, in what ways it can be protected from falling into the hands of misusers; such questions need to be answered before bringing this idea into real time use.

Brief Overview of a Robocode Tank

Each robot is made up from three parts: the vehicle, the radar and the gun. They can be moved both together and independent of each other. The vehicle can be moved ahead and back, in order to reposition the robot's body. Any of the parts can be turned left and right. Turning the vehicle changes the direction of further moves, turning the scanner changes the area that the robot “senses” and turning the gun sets the direction where the bullet will be fired on.

A tank consists of the tank body, a turret containing the gun, which can rotate with or separately from the body, and a scanner used to detect enemies that can rotate with or separately from the gun. A tank can move forward and/or turn. While turning its maximum forward speed is reduced. Each tank starts with a certain amount of energy. It loses energy each time it crashes (either into a tank or a wall) and each time it is hit (either by a bullet or another tank). It gains energy if one of its bullets hits an opposing tank. When it fires its gun, it can vary the amount of energy used to fire the bullet, the more energy put into the bullet the slower it travels but the more damage it does and the higher the energy gain to the firing tank (if it hits). Guns also need to cool down before they can be fired, the higher the energy of the bullet the longer the gun takes to cool. Tanks can detect other tanks, their energy level, speed and direction. They cannot detect bullets nor can they directly detect the fact that a bullet has been fired, although the energy loss in an opposing tank is noticeable and provides an indication that the tank may have fired (or suffered an energy loss by other means). The tanks fight in an empty arena of a fixed size, surrounded by walls. Typically the start position and orientation of the tanks is randomized for each round. A typical battle consists of 10 rounds.

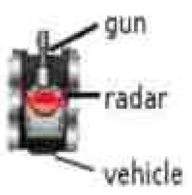


Fig. 2. Anatomy of a Robocode robot

A

The program to control a Robocode tank (a robot) is usually written in Java. It is executed as a separate thread that is activated from the main Robocode program. The structure of typical robot therefore looks like this.

1. Import statements

Here various classes are imported inside the program from pre-defined libraries. The functions and events are mainly present in these classes without importing which the required behavior of the robot could not be produced. This is not similar to including header files in C or C++ where all the functions and classes are included inside the program. It's just a way to give the path to reach to the destination class or package name that is already present in JAVA API. For eg: import robocode.*; Here robocode is the package name and * signified that all the classes present inside robocode package should get imported.

2. Class name extends interface

This marks the beginning of your program. Here you start all the declaration and initialization of your variables. Interface as the name itself suggests provides an interface between your program and imported classes and java libraries. Extends is a keyword that associates your class file to the interface. For eg: public class NewRobot extends Robot.

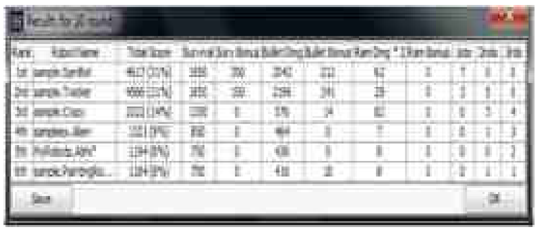
3. run ()

Here all the default operations that need to be performed by the robot always are written. The order to move ahead or get back, fire or ram is given here. This is the core part which once gets started, continues its execution until the program is halted or comes out of the rum-time.

4. FunctionName (FunctionEvent object)

Here the statements for a particular event are given, that means how the robot should react or respond to the situation when a particular event occurs. Suitable handlers are provided to handle the event and take necessary actions. The functions are called by the object created inside the function. Some functions and events are onHitRobot(), onHitByBullet(), onHitWall() etc.

Scoring



Rank	Robot Name	Total Score	Survival	Surv Bonus	Bullet Dmg	Bullet Bonus	Ram Dmg * 2	Ram Bonus	1sts	2nds	3rds
1st	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
2nd	sample.Tracker	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
3rd	sample.Crazy	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
4th	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
5th	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
6th	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

The Robocode system scores each participant in a fight. The score is based on a number of factors, there are 1) survivability points if the tank survives a round (i.e. wins); 2) points for bullet damage which related to the damage done by the tank's bullets; and 3) points for ramming which are awarded at twice the rate of points for similar damage done by a tank's bullets.

Real-Time Applications:

Robots are proving themselves to be fast, efficient, and reliable at performing dangerous missions for military personnel and first responders. Autonomous and semi-autonomous robots are being deployed for missions such as:

- Explosive ordnance disposal (EOD), including the detection and detonation of roadside bombs, car bombs, unexploded ordnance, and other hazardous explosives, while personnel remain at a safe standoff distance.
- HazMat detection, including the detection of chemical, nuclear, or biological hazards.
- Route clearance, such as removing debris or ordnance from the path of a convoy.
- Structure mapping, which involves surveying buildings, bunkers, and other structures for hazards before personnel encounter them.
- Person-tracking and person-following, either for surveillance, police actions, or troop maneuvers.
- Long-term surveillance (sometimes called "persistent stare") of a fixed location.
- All these jobs from helping first responders in urban centers to conducting surveillance in mountain passes require unwavering attention and precision, despite great physical risk. That's why they're best performed by robots.
- Robots can move along roads, searching for bombs and defusing them.
- Using sophisticated treads and flippers, robots can clamber through streets, alleys, parking lots, stairwells, and hallways, in search of accident victims, suspicious persons, or bombs.
- Robots can also manipulate objects, lifting things, setting them down, or pushing them asideskills useful for tasks as varied as delivering supplies in emergencies to removing hazardous objects.
- Robots not only do these jobs; they do them very well. The U.S. military finds that robots detect and disable roadside bombs five times faster and with more accuracy than humans. Increasingly, semi-autonomous or autonomous operation to reduce the war fighter's workload and distraction while in hazardous environments.

Conclusion and Future Scope

The Robocode game can be used both as learning as well as a joyful experience, but if properly utilized it may transform into a technology that can strengthen ones defense systems to a great extent. Also the numerous applications that we listed out can be achieved easily through robocode robots. Fully automated and programmed tanks and machines can yield greater results than the manually operated machines that involve lots of errors and wrong judgments leading to loss of lives too. But before bringing this to practical use a lot of points are needed to be considered so that it may prove to be a healthy technology which can be used more for protection rather than destruction.

REFERENCES

Robocode homepage, http://robocode.sourceforge.net/. | TYZX Case Study: Giving Robots Real-time Vision and Depth Perception, www.tyzx.com | S. Li, "Rock 'em, sock 'em Robocode!, Learning Java programming is more fun than ever with this advanced robot battle simulation engine. | J. S. Russell, P. Norvig, Artificial intelligence: A modern approach, Prentice Hall, Englewood Cliffs, NJ, 1995. | R. A. Brooks, "A robust layered control system for a mobile robot", in IEEE Journal of Robotics and Automation, 2(1), 1986, pp. 14-23. | IBMdeveloperWorks,2002, http://www.ibm.com/developerworks/java/library/j-robocode/.