Fall 2012 COMS 4115

George Brink
Shuo Qiu
Xiaotong Chen
Xiang Yao



A stack-based Imperative language Applied to designed game

- Simple
- Interesting
- Powerful

simple is beautiful.

Motivation

- Simple enough to be understood by users who know nothing about programming
- Efficient Al programs to be applied to simultaneous games (one tick per operation)

Project Overview

- The Drone War stack-based language
- The GUI Drone War programming game
- Al of drones in which the language is applied to the game

Introduction to Drone Language

Keywords

Dup, drop, dropall, swap, over, rot, read, store, jump, jumpif, sub, endsub, if, else, endif, begin, while, again, move, stop, shoot, look, wait, gethealth, random, isfoe, isally, iswall, isend, mod, and, or, not

Function

```
sub add_one
1 +
endsub
1 add_one
// 1 + 1
```

Types

Integer, boolean, flag (wall, foe, ally, end)

Language Tutorial

Variables

```
2 a store
a read 2 +
// 2 + 2
```

Operators

```
+, -, *, /, mod, ^
And, Or, Not, =, <, >
```

Game specific functions

Move, stop, shoot, look, isfoe, isally, iswall, isend, wait, gethealth, random

Language Tutorial

- Condition branches
 - Branches

```
// if
condition if
actions end_if
```

//if else condition if actions else actions end_if

Language Tutorial

- Loops
 - Endless loop

```
begin
    actions
again
// repeat actions again and again until dead
```

Conditional loop

```
begin
condition
while
actions
again
// repeat actions if condition is true
```

Game Introduction

- Fighting each other in square arena size of 1000*1000
- Each drone is controlled by AI written in Drone Language, and automatically moves, searches and shoots
- Drone freeze once died or illegal command is detected

Arena Introduction

- Control all drone and bullet objects inside the arena area
- Implement tick operations and update status of all objects
- Interact with GUI and visualize positions and status of objects

Drone Introduction

- Each drone contains info of positions, directions, health, status and etc.
- After each tick operation, variables updated by arena based on byte code compiled from AI
- Store user defined variables and subs, and helper functions are also available

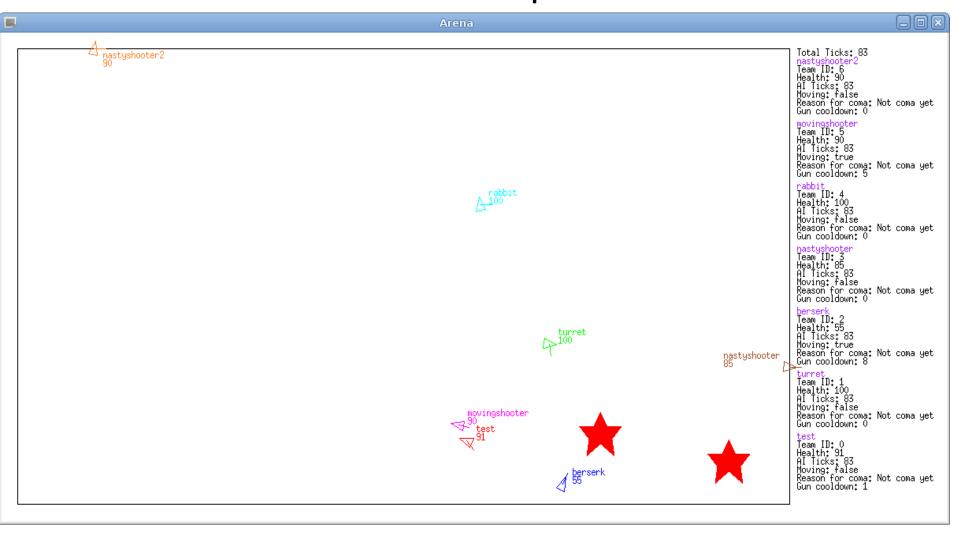
Al examples

```
100 health store
                          // set variable health to 100
main loop:
     10 wait
                          // wait for 0.1 of the second
     // read the stored value of health
     // read the current health and compare it with the old value
     health read getHealth =
                                                                       LABEL & JUMP
 // repeat indefinetely if no one harmed the drone
     main loop jumpif
     // what to do if drone recieved some damage
     0 359 random
                           // get a random value in the range 1-360
                  // move in the random direction
     move
                  // wait for 0.1 seconds
     10 wait
                 // stop
     stop
     main loop jump // and go back to the beginning
```

Al examples

```
0 direction store
                                                                              again
// keep moving to different walls, seek foe to shoot
                                                               // stop moving
       begin
                                                                              stop
              dropall
                                                                              begin
              0 shoottime store
                                                               // lay on the wall and look backward
              0 360 random
                                                                                     direction read 180 + look
              direction store
                                                                                     begin
// moving
                                                                                            dup
              direction read move
                                                                                            isfoe not
// move to the wall stop before hitting
                                                                                            swap
                                                   LOOPS
                                                                                            iswall not
              begin
                                                                                            and
                     direction read look
                                                                                     while
                                                                                            drop drop
                     begin
                             iswall not
                                                                                     again
                                                               // shoot foe
                     while
                             drop drop
                                                                                     shoot
                                                                                     shoottime read 1 + shoottime store
                     again
                                                                                     shoottime read 10 <
                     drop
                     20 >
                                                                              while
              while
                                                                              again
                                                               // repeat it until die
                                                                       again
```

GUI examples



Summary

- It is useful to apply class materials such as stack based operation to our language
- Stack-based language is always compact, efficient and easy to understand
- A good design and a good team leader always make good progress