

The Drone War

Fall 2012

COMS 4115

George Brink

Shuo Qiu

Xiaotong Chen

Xiang Yao



The Drone War

A stack-based Imperative language

Applied to designed game

- Simple
- Interesting
- Powerful



**simple is
beautiful.**

The Drone War

Motivation

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

The Drone War

Project Overview

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

The Drone War

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)

The Drone War

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
```

The Drone War

Language Tutorial

- Condition branches
 - Branches

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

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

The Drone War

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
```


The Drone War

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

The Drone War

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

The Drone War

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

The Drone War

AI 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 =

// 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                      // move in the random direction
10 wait                   // wait for 0.1 seconds
stop                      // stop

main_loop jump           // and go back to the beginning
```

LABEL & JUMP

The Drone War

AI examples

```
0 direction store
// keep moving to different walls, seek foe to shoot
begin
```

```
dropall
0 shoottime store
0 360 random
direction store
```

```
// moving
```

```
direction read move
// move to the wall stop before hitting
begin
```

```
direction read look
begin
```

```
iswall not
```

```
while
```

```
drop drop
```

```
again
```

```
drop
```

```
20 >
```

```
while
```

```
again
// stop moving
stop
```

```
begin
// lay on the wall and look backward
direction read 180 + look
begin
```

```
dup
```

```
isfoe not
```

```
swap
```

```
iswall not
```

```
and
```

```
while
```

```
drop drop
```

```
again
```

```
shoot
```

```
shoottime read 1 + shoottime store
```

```
shoottime read 10 <
```

```
while
```

```
again
```

```
// repeat it until die
```

```
again
```

LOOPS

The Drone War

GUI examples

The screenshot shows a window titled "Arena" with a 2D battlefield and a console log on the right. The battlefield contains several entities represented by colored triangles with labels and health values:

- nastyshooter2** (orange triangle, health 90) in the top-left corner.
- rabbit** (cyan triangle, health 100) in the center.
- turret** (green triangle, health 100) in the lower-center.
- nastyshooter** (orange triangle, health 85) on the right side.
- movingshooter** (magenta triangle, health 90) and **test** (magenta triangle, health 91) near the bottom center.
- berserk** (blue triangle, health 55) at the bottom center.

Two red stars are located in the lower-right area of the battlefield. The console log on the right displays the following information:

```
Total Ticks: 83
nastyshooter2
Team ID: 6
Health: 90
AI Ticks: 83
Moving: false
Reason for coma: Not coma yet
Gun cooldown: 0
movingshooter
Team ID: 5
Health: 90
AI Ticks: 83
Moving: true
Reason for coma: Not coma yet
Gun cooldown: 5
rabbit
Team ID: 4
Health: 100
AI Ticks: 83
Moving: false
Reason for coma: Not coma yet
Gun cooldown: 0
nastyshooter
Team ID: 3
Health: 85
AI Ticks: 83
Moving: false
Reason for coma: Not coma yet
Gun cooldown: 0
berserk
Team ID: 2
Health: 55
AI Ticks: 83
Moving: true
Reason for coma: Not coma yet
Gun cooldown: 8
turret
Team ID: 1
Health: 100
AI Ticks: 83
Moving: false
Reason for coma: Not coma yet
Gun cooldown: 0
test
Team ID: 0
Health: 91
AI Ticks: 83
Moving: false
Reason for coma: Not coma yet
Gun cooldown: 1
```

The Drone War

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