The Game of Death

Manual and Documentation

by

Shardul C.

Version 0.1 (unstable)

The author of this document is Shardul C. Comments, requests, questions, errata, and any other correspondence regarding this document should be addressed to:
`shardul (DOT) chiplunkar (AT) gmail.com`.

This manual is for the Game of Death, version 0.1.
Copyright © 2014, 2016 Shardul C.
Copying and distribution of this file, with or without modification, are permitted in any medium without royalty provided the copyright notice and this notice are preserved.

Contents

1 Introduction

The Game of Death is a grid-based game, modelled somewhat after Conways' popular 'Game of Life', in which shapes on the grid follow ever-changing programmable rules to win battles versus opponent armies. It also incorporates some aspects of the game 'Cature the Flag' in the actual battle gameplay. The motivation behind the game is discussed in subsection 1.1 and a high-level overview of the gameplay, in subsection 1.2.

1.1 Motivation

The motivation for the Game of Death was to create a game that would be extremely difficult for computers to play, if not altogether impossible or out of reach of computing technology. To play the Game of Death, creativity is required as much as intelligence is; it is an Art as well as a Science, resembling pure mathematics in the manner of skill, ingenuity, and thougtful reasoning required.

However, the Game of Death is meant to be very easy to understand and play. Even without reading this document, beginners can be explained the basic concepts of the Game of Death and can start designing their own shapes for battles.

The Game of Death can also be considered an experiment by the authors in game designing. Along with creative thinking and simplicity, the Game of Death tries to be a very competitive game which encourages players to improve as humans, rather than 'upgrading' or 'modding' the 'characters' of the game. The game objects also evolve as a consequence, leading to a general evolution of the 'intelligence' of the entities in the game which is quite interesting to observe (and even more so to be a part of!).

1.2 Gameplay

Gameplay consists of two stages: design and battle. In the design stage, players design shapes which will move around on the grid and experiment with rules for the shapes, simulating battles against themselves or against other non-human agents. Shapes are 'programmed' in a sense to coordinate their actions and act as a single unit: the 'army'.

In the battle stage, the armies of two or more players are distributed around the game grid at an [initially] safe distance from each other. Each player has a 'home' on the grid which is an immobile shape containing a metaphorical 'flag'. The aim of each player is to capture the opponent's flag and bring it back to the 'home'. Once the battle starts, the shapes can move and attack other shapes (actions are specified in subsection 2.3) and try to steal opponent flags if they are within reach. When an opponent flag is brought back to the 'home', the game ends. This is not a very easy goal, and to achieve success, shapes on the grid have to communicate efficiently and work together as a team, using evolving strategies and techniques to win.

Many variations on the basic gameplay are possible, including battles with many players, very long design stages, rules for 'capturing' opponent shapes along with the set of rules that they are governed by, and more. The endgame condition can also be modified with respect to the number of shapes killed or time taken to win. However, the basic structure of the game should remain mostly unchanged.

2 Complete Specifications

These are the complete specifications for the Game of Death. All client and server programs are required to follow these specifications while implementing the Game of Death. The specifications may change to a major extent in subsequent versions of the game until the label of 'unstable' is removed; thereafter, large changes will be indicated by a change in the integer part of the version number and will be declared 'stable'. Changes in the non-integer part of the version number will indicate relatively minor changes and may receive the label 'unstable'. Implementations are advised to not implement 'unstable' changes.

2.1 Definitions

We will define the following terms:

game grid The game grid is the ground for gameplay. It is composed of lit cells and dark cells arranged in a rectangular grid, where lit cells represent occupied space and dark cells represent free space.

shape A shape is a collection of lit cells on the grid which act as a sinlge entity. All the component lit cells must have at least one common edge with another component lit cell.

army An army is all the shapes belonging to a player in the game.

The lit and dark cells can be marked in any way which clearly differentiates them, such as differing colors, differing brightnesses, or (for text-based implementations) different characters. It is also recommended to have a distinct method of differentiation for the players' armies.

2.2 Shape Attributes

Each shape has the following attributes (in sans-serif font):

size The size of a shape is the number of cells it comprises.

speed The speed of a shape is equal to $\lceil \mathsf{size}/3 \rceil$ turns per move. A move is when the shape shifts as a whole by one cell in an orthogonal direction, or when the shape changes its orientation by a one-fourth turn.

max HP and HP The max HP, or maximum health points, of a shape is equal to $2 \times \mathsf{size}$ points. The attribute HP refers to the current health points a shape has.

attack The attack strength of a shape is $\lceil \text{average thickness} \times \sqrt{\mathsf{HP}/\mathsf{max\ HP}} \rceil$ health points.

attack range The attack range of a shape is $2 \times \mathsf{size}$ cells.

defense The defense strength of a shape is $\lceil \text{average thickness}/3 \times \mathsf{HP}/\mathsf{max\ HP} \rceil$ health points.

The 'average thickness' in the attack and defense attributes is calculated as follows. From the side of the shape performing or receiving an attack, the number of cells between that side and the other side of the shape (i.e. where the shape ends into free space) for every cell in that side is obtained. The average of these 'thicknesses' is the average thickness of that side.

When an attack is performed, the attacker does not lose any health points; instead, it inflicts damage in the range of its attack equivalent to attack health points. All attacked shapes (within the attack range) lose $\mathsf{attack} - \mathsf{defense}$ health points: note that the attack attribute is

of the attacking shape, while the defense attribute is of the attacked shape. If the quantity attack − defense is negative, then no health points are lost (or gained).

When multiple attackers attack a shape, the health points lost by the attacked shape are $\sum(\text{attack}) - \text{defense}$, where $\sum(\text{attack})$ is the sum of all the attacking shapes' attack attributes.

Every 2 turns, a shape's HP is incremented by 1 if it is not attacked in those 2 turns. This process is continued until HP = max HP.

2.3 Shape Actions

Each shape *must* define the following methods:

update The 'update' method specifies what the shape should do on each of its turns. The various actions a shape can perform are described later in this subsection.

die The 'die' method specifies the last actions a shape performs before it dies. This method will only be used once, and no actions will be performed by the shape after it dies.

Note that after a shape dies, its 'body' remains behind on the grid. This 'body' cannot perform any actions, and cannot be moved, but can sustain $10 \times \text{max HP}$ damage before it disappears completely.

A shape *may* define more methods, such as 'initiate plan 101', which can be called from the 'update' or 'die' methods and which may use any of the actions given below.

A shape may use the following actions in its methods:

rest The 'rest' action does nothing for one turn.

move(direction) The 'move' action initiates a move in the given direction (forward, backward, left, or right). The move may not occur immediately after the action is performed; the number of turns taken to move is defined as the speed of the shape. A move, once initiated, may be cancelled before it is actually performed by intiating a move in a different direction or by 'resting'.

turn(direction) The 'turn' action initiates a move in the given direction (clockwise or counterclockwise). Like the 'move' action, this is dependent on the speed of the shape.

broadcast(message) The 'broadcast' action instantaneously broadcasts the given message to all shapes within a predefined radius[1]. However, the broadcast will be receivable by the shapes only in the next turn.

receive The 'receive' action returns a list of new messages received. Once a message has been received, it will be cleared from this list.

capture The 'capture' action captures the enemy flag is the shape in in direct contact with the enemy 'home'. This action takes 3 turns to complete.

attack(direction) The 'attack' action attacks all shapes within the attack range in the given direction. Health points are lost (if at all) by the attacked shapes instantaneously. After the 'attack' action has been performed, the shape cannot perform it again until 1 turn has passed.

suicide The 'suicide' action makes the shape die; the 'die' method will be called in the next turn by the game.

location The 'location' action tells the shape its location on the game grid with respect to its 'home'.

[1] A radius of 200 cells is recommended.

By default, the orientation of all shapes is facing forwards (towards the enemy).

If a shape bearing the enemy flag is killed, the flag is returned to the enemy 'home' immediately. If the shape reaches its own 'home', the flag is said to be captured and the enemy is defeated.

2.4 Miscellaneous

Some technical details: the game grid is usually 200×300 cells for two players. In this grid, players are limited to 500 total cells to make up their shapes. These values can, of course, be changed if all players agree to the changes.

Players may watch the battle simulation at a slower pace, or may allow it to proceed at speed until the game ends. Watching the battle simulation is obviously possible only for reasonably-sized game grids.

A shape is allowed to 'see' all cells of the game grid within a 50-cell radius. The shape can know the lit/dark state of these cells, and also whether a lit cell is friendly or hostile. However, the shape itself has no concept of 'shapes' and cannot perceive groups of lit cells as such. The 50-cell radius parameter can also be changed if all players agree to the change.

It is recommended that when a player's army actively kills more than a given percentage[2] of any particular type of enemy shape, then the rules for that shape are made available to the player. This encourages evolving strategies and introduces new variations of existing strategies, making gameplay richer and more intersting.

[2] 80% is a recommended value.

3 Tips

Note: These tips are not to be considered a 'cheat-sheet' but rather as pointers in the direction of shape development.

The shape of the shapes should be made structurally sound to keep the average thickness of each side high enough to withstand damage and provide strong attacks. Because of the average thickness parameter, strong shapes are usually solid or contain internal supports (such as spokes in a wheel). The health of a shape also affects its damage resistance and attack strength.

Dead shapes can be used tactically as cover from enemy shapes and as places for shapes to safely regroup. Maneouvering around dead shapes may also present an unforeseen challenge for attackers if a shape strategically chooses to commit suicide, blocking an attacker's path.

Shapes should act together as an army, or at least as units of an army. This increases the capabilities of individual shapes multifold and promotes complex, 'intelligent' behavior. Also, if the enemy uses group tactics, then the individualist would be at a severe disadvantage!

Ideally, the design stage should be prolonged (say, two weeks per stage) with ample testing and simulations to check the capabilities and behaviors of shapes in actual battles. Players may even collaborate on design stages and have meta-design conventions.

In environments with more than two players, a system for counting points, maintaining ranks, creating tournaments with groups of players working together, etc. can be established to enhance gameplay. Players may choose to keep track of 'points' depending on various parameters, but these points should not influence gameplay.

Have fun!