**OBJECT-S AI AGENT DESCRIPTION**

**-------------------------------------------------------------------------------------------------------------------------------------------------------------**

OBJECT-S

**Leader** - KyoungJun Ahn **Adviser** - KyungJoong Kim

**Members**  - DongHyun Jang

 - JunSu Cho

- HyunJin Kim

- TaeSan Eom

**-------------------------------------------------------------------------------------------------------------------------------------------------------------**

Strategies

**Classification of actual points**

Actual point means points that birds hit in processes to hit targets. Type of actual Points is not important in cases that actual point equals target. But in the case that actual point doesn't equal target, a effective bird differs according to type of actual Point. So this agent classify actual point base on type.

**Priority ranking of target points in each array list.**

In many case, considering targets that closer to the sling is helpful to clear levels, because it usually reduces number of obstacles when the agent hit a target that farther to the sling. And the agent stores each target points in each array lists according to type of actual point. And then decide the ranking of target points in each array lists according to distance from sling.

**Priority ranking of array lists**

As mentioned earlier effective type of birds differ according to type of obstacles. So agent decides the ranking of array lists.

**Decision about two trajectories**

In many case, the agent has two trajectories to hit a target. In this situation, the agent calculate searched trajectory’s score and select a trajectory that have lower score.

Of course, the agent considers effectiveness of a bird on the sling to obstacles in the calculation.

**Tap decision**

Each type of birds has a special ability without red bird. So, assign different tap time according to type of birds.

|  |  |
| --- | --- |
| Red bird | Red birds don’t have a special ability. |
| Yellow Bird | Yellow birds have a ability to acceleration. And this ability can help to destroy obstacles. But it makes hard to calculate the trajectories. So yellow birds tap in front of obstacles or targets. |
| Blue Bird | Blue birds have a ability to dispersion. When launch blue birds in low angle, too earlier tap decrease effectiveness so when launch blue birds in low angle, blue birds tap in front of obstacles or targets. And when launch blue birds in high angle, blue bird tap at adequate high after pass the highest point to make damage to large area. |
| Black Bird | Black birds have a ability to explosion. Too early tap decrease effectiveness because they don’t touch targets of obstacles, therefore black birds tap immediately after touch. |
| White Bird | White birds have a ability to dropping a egg that explode, therefore white birds tap when white bird right above targets. |

**-------------------------------------------------------------------------------------------------------------------------------------------------------------**

Vision

**Improvements**

**Recognizing a Bird Type**

A sample agent can't distinguish blue and black birds. So we modify this problem to make new function. First, we make arrangement named 'type'. When the function finds the colour of birds, this automatically add 1 to type of each index. Next examine type of each index, get a biggest value and save it into variation named 'state' and return it.

**Ref) VIsion.java ->public int ActiveState1(Rectangle p)**

**Saving Score Each Point of Obstacle Colour**

We think that sample agent isn’t suitable for our idea. Thus, we find arrangement(480\*840) named ‘\_scene’ having an information of colours saved in each pixel at Vision.java file. We make new arrangement which is same size of \_scene and examine each colour of point and give score to ice colour in 1point, wood colour in 2point, TNT colour 3point, stone colour 4point, land colour 100point. Use this arrangement saving information of score when function named ‘t\_score’ in TrajectoryPlanner.java adds the score on trajectory.

**Ref) Vision.java -> public void Divco(int [][] A,Rectangle s)**

**TrajectoryPlanner.java-> public int t\_score (int[][] arr,Rectangle s,Point o,Point release,int birdType)**



 **Poached Eggs 1 – 2 Map Score Information Arrangement printing result**

**-------------------------------------------------------------------------------------------------------------------------------------------------------------**

Custom Map

**Improved Agent by Custom Maps(Stages)**

**Change An Approach of Test Agent’s Performance**

Our first approach is testing an agent at Poached Eggs 21 level. But we think that existing map of Poached Eggs has a limit to testing. So we create test maps which was made by Angry Birds Map Editor. Then we use that map to test when a new modified agent releases. Then we could find many problems and fix them. One example problem is throwing a bird to land when a targeting unreached pig. That is, the pig is existed where can’t reach to bird. So we made a limit distance and area when targeting a pig. So we can fix that problem.

**What Kind of Approach to Make Test Maps(Stages)**

First, making almost 30 maps which are based on the degree of Poached Eggs 1 - 1 to 1 - 21. Poached Eggs 2 - 1 to 3 – 21 maps are difficult to play sample agent and our agent. So the map lowered difficulty. Test maps are based on overall top shape. Offering a number of birds to lower the difficulty. **Example of Custom Maps**

