Description for Eagle's Wing, an AI Angry Bird Playing Agent

Tian Jian Wang
University of Waterloo
Zazzle Inc.

Description: a multiple strategy affordance based structural analysis that gives multiple decisions and a manually tuned utility to decide between them. Particular attention is paid to buildings, with unique strategies for different types and shapes of buildings. The utility function is learned through the machine learning method xgboost at first using thousands of shots in the first 42 levels in Poached Eggs, and then rewritten in normal programming for much simpler (i.e., tunable) and faster calculation.

The agent is based on the open sourced version of the agent from Team Datalab in the 2014 competition. However, Team Datalab literally left out most of important parts in each of their strategies, only leaving behind a general structure to follow. I went on to re-imagine the strategies, made many additions and upgrades, added another strategy and rewrote the utility function entirely.

5 strategies:

1. Pigshooter, the bread and butter strategy. It will try to find the trajectory that either targets an unprotected pig or one that includes mutliple pigs on the trajectory. It can make great shots in the beginning and make neat finishing touches at the end.
2. Destroy TNT. Sometimes there are TNT on the map that can devastate a large region. Aim for them.
3. Destroy as many blocks as possible. For bluebirds, blackbirds, whitebirds, and yellow birds, they can destroy many ice, stone, wood and wood blocks respectively in their paths.
4. Destroy objects close to high round objects(preferably large stones) to release them.
5. Building strategy All the blocks are then sorted based on their type and relative position in the building. The best block for a given bird on sling is then selected. We also differentiate between three types of buildings:

- Pyramid
- Rectangle
- Skyscraper

Agent building process:

I wrote a separate program to fire 200 shots aimed at different objects in each of the first 42 levels and collect 200+ current game state and scene features in each before shots scene. I used the features to train a gradient boosting tree xgboost model with target variable being the score difference between shots. As I looked at the feature importance graph, I begin to understand the important features and was able to formulate and fill in the details for the 5 strategies.

As I have the 5 strategies written, I first tuned the tap time as it is very important. I tried over more than 1000+ shots for each bird for each strategy on the 42 levels and used the average of the tap time of the best shots for each birds for each strategy.

Then, I tried to tune the utility using xgboost by firing thousands of shots for each strategy. The resulting model is relatively straight forward. As it turns out that the situation is almost always super clear which strategy to use in a given scenario. Looking at the decision trees, I simply coded up a very simple utility function for each strategy: if some particular conditions meet, give INT_MAX(choose for sure). If some other conditions meet, give INT_MIN(never choose). Else give a default score or a score

that takes into account some features in the current scene. I imagine that my utility scores are very very different from DataLab agent's utility score.

Finally, I noticed that a small portion of the shots just misses the pivotal points(weak points in a structure, a small pigs, a small round object) by a tiny margin but adjusting the trajectory planning code for their sake would make other shots significantly worse. Thus, I implemented a logic where for levels that have been tried once, add a small random factor to some trajectories. This is a crude way of adding some adaptation and some non determinism into the agent for improving score on retries.