GLITCHED BACONS

FlapANN

Self-winning Flappy Bird

Silesian University of Technology

Faculty of Automatic Control, Electronics and Computer Science

Authors:	Dawid Grobert Julia Boczkowska
Created:	March 25, 2022
instructor	dr inż. Grzegorz Baron
year	2021/2022

Introduction

Flappy Bird is one of the most popular games of the year 2013/2014. Its simplicity is a perfect example for a short project, although it gives many possibilities for implementation.

Gameplay

Flappy bird is a so-called *side-scroller* (see info), in which the player plays the role of a flying bird. The main goal is to avoid pipes protruding from the bottom and top of the screen by "jumping" the bird. The more pipes we jump through, the better the result.

> ℹ **Side-scroller** is usually a 2D game with a character moving to the left or right of the screen. Action is viewed from a side-view camera angle, that follows the player. Some side-scrollers allows to move only one side of the screen (such as Flappy Bird). Most often this genre is associated with arcade games.

The bird must not fall or hit any of the pipes. The difficulty is that our bird does not maintain a stable movement and constantly falls and the impulsive "jumping" suddenly and with great power tosses it up, which requires the appropriate precision in execution in order not to hit any of the above mentioned pipes.

1 About the project

`FlapANN` is a self-winning game. This time, it's an artificial intelligence that will take on the challenge of playing the game that has prompted tens of thousands of people to destroy their smartphones. Additionally, from within the app the user will be able to make life difficult for the AI by dynamically changing the spacing of the pipes, or even their speed of movement. It is possible that we will implement some fun difficulties such as moving pipes up and down.

If possible, the network data will be displayed on the screen like though a graph of evolving neural network, or perhaps current generation data.

2 Overview

In this project, we will use one of the evolutionary algorithms that is inspired by the process of natural selection. Namely, the genetic algorithm. At this stage, we do not yet know whether we will use a classical neural network coupled with a GA, or whether we will decide to use NEAT (NeuroEvolution of Augmenting Topologies) to generate an evolving artificial neural network.

2.1 Classic ANN coupled with genetic algorithm

It seems to us that the gradient descent is the most widely used algorithm in training artificial neural networks. However, we want to experiment and replace it with a genetic algorithm to see how it will behave. We hope that the results we produce will be relatively good.

2.2 NEAT

Using NeuroEvolution technique, we are not limited to one fixed neural network structure. It is evolved over time using a genetic algorithm, which can lead to finding an even better network structure than the one we originally chose. Overall, the assumptions of this approach suit us very well - it seems to be a good fit for a self-learning bird.

Both approaches are of interest to us, so it takes time to decide explicitly which one we will opt for.

3 Implementation

We would like to create the project in C++, and we are aware that this is not always the main choice for this domain where python seems to be rather dominant. We have no problem with either language, although C++ seems to be more of a challenge for us. In this language there are also less projects of this type, which we hope to stand out a little. On the other hand, although python dominates here, there seems to be no shortage of available libraries for machine learning.

As for other libraries that will allow us to prepare the implementation of the game, we are happy to use the following:

- `SFML` – for creating a window and displaying the game
- `ImGui` – to display "debug" windows inside the game containing any kind of checkboxes or sliders.
- `ImPlot` – used in combination with `ImGui` to draw graphs and visualize any data

4 Sources

At this stage, we don't want to cross off any of the existing solutions until work on our project has begun. There are quite a few libraries available for C++, such as:

- TensorFlow
- mlpack
- DyNet
- Shogun
- FANN
- SHARK

[1] Laurent Gomila and many others. Sfml. `https://github.com/SFML/SFML`.

[2] Omar Cornut. Dear imgui. `https://github.com/ocornut/imgui`.

[3] Evan Pezent. Implot. `https://github.com/epezent/implot`.

[4] Tensorflow. `https://github.com/tensorflow`.

[5] Ryan R. Curtin, Marcus Edel, Mikhail Lozhnikov, Yannis Mentekidis, Sumedh Ghaisas, and Shangtong Zhang. mlpack 3: a fast, flexible machine learning library. *Journal of Open Source Software*, 3:726, 2018.

[6] Artidoro Pagnoni. Dynet. `https://github.com/clab/dynet`.

[7] Heiko Strathmann, Viktor Gal, Sergey Lisitsyn, Soeren Sonnenburg, Gunnar Raetsch, and Fernando Iglesias Garcia. Shogun. `https://github.com/shogun-toolbox/shogun`.

[8] Jakub Zelenka. Fann. `https://github.com/libfann/fann`.

[9] Oswin Krause. Shark. `https://github.com/Shark-ML/Shark`.