Creating Human-like Fighting Game AI through Planning

Roger Liu
Carnegie Mellon University
rogerliu@andrew.cmu.edu

Abstract

Games are a major testing ground for Artificial Intelligence. Though AI has become proficient at playing games such as Space Invaders, we have yet to see AI which emulates a human-like playstyle. This human element is important, as a large part of the enjoyment in competitive multiplayer games comes from outwitting other human strategies. To address this issue, we investigate a novel AI technique that leverages planning and human demonstrations to create an opponent that exhibits desirable qualities of human play. We introduce the idea of "action effects", which the AI uses in order to figure out the causal relationship of actions and the game state. These action effects are learned from human demonstrations and are used to help the AI plan out a strategy for hitting the opponent. We implement a simple fighting game called *FG* for the AI to compete in and provide it a human demonstration to learn from. Lastly, we evaluate the effectiveness of our AI by comparing its similarity score against other algorithms' and other demonstrations by the same human player.

1 Introduction

Fighting games are unique among competitive games in that they are real-time, 1-on-1 games where small mistakes lead to huge consequences. The best way to get better at these kinds of games is to practice against other humans, but that is not always an option. While online play exists, is not ideal because of network latency. In addition, the AI in these games are generally considered a poor substitute. They often exploit natural advantages such as perfect reaction time and perfect game state information, but even disregarding that they still only have fixed spectrum of behavior patterns which players can learn to exploit and consistently defeat. Worse still is that these behavior patterns might not even be representative of the human opponents that players encounter in competition.

That said, there are avenues to improve the AI in fighting games to make them useful for players. One approach is to make an optimal AI which is able to adapt its strategy based on its performance. This would provide players a challenge by removing the ability to exploit the AI, but it still doesn't necessarily capture the strategies and techniques used by other human players. Another approach is to make a *human-like* AI, one that plays like another specific human opponent. This task seems feasible, as long-time fighting game players can identify the differences between the playstyles of different players, meaning that there is some quality that differentiates one behavior from another.

In this research, we tackle the problem of creating this human-like AI. To do this, we will first explore previous approaches taken to create human-like AI and discuss their merits and limitations. We will also describe other attempts at creating AI for fighting games to contextualize our efforts compared to theirs. We will then introduce the environment we created to test our approach and define concepts and terminology used by our algorithm. Lastly, we will describe our algorithm, where we plan on the actions provided by human demonstrations to reach a desired outcome.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

2 Background

2.1 Racing and Imitation

One of the few documented instances of Imitation AI came from the racing game *Forza Motorsport*. In this game, players could train "drivatars" to race just like themselves. This was implemented by having the AI learn how the player behaves on different segments of track and try to replicate that behavior when it encounters those sections. However, this imposed a restriction on the types of tracks that could be present in the game, as they had to be formed from the basic segment building blocks.

Researchers then expanded on this approach through the use of genetic algorithms [DrivingPlayerModeler]. In these kinds of algorithms, candidate solutions are continually evolved toward better solutions. These candidates have a set of parameters which are altered between each iteration of the process, and are then evaluated according to a fitness function. For the particular case of creating robust human-imitating racers, a fitness function made up of 3 different components was used. 1 focused on matching the players progress on the track. Another focused on it matching the player's steering and a final one had it match the player's speed. The resulting AI did an alright job of mimicking some aspects of the repsective players, as the AI imitating a slow, careful driver behaved in a significantly different way from the faster, reckless driver. However, closer inspection of the driving AI showed that the resulting behavior was not conceivably human. Later attempts which had the fitness function that incoporated a focus on having the AI drive optimally also did not obtain convincing results, but it showed a clear trade-off between driving optimally and improving driver similarity. [MultiObjectiveMimic]

2.2 Super Mario Bros.

Researchers also developed several methods to mimic human behavior is in the space of 2D platformers, specifically a modified version of Super Mario Bros. [MarioImitation]. Novel methods tested in this space were Inverse Reinforcement Learning and Neuroevolution.

The results of the Inverse Reinforcement Learning approach were discouraging, as the agent wasn't able to consistently handle situations that weren't often seen in the demonstration and was unable to match a human's ability to predict things not in the immediate detection area. In addition, the optimal policy obtained by IRL is deterministic, further reducing the human-like appearance of the AI.[MarioImitiation2]

Neuroevolution produced much better results. In this method, a neural network was first trained to play Super Mario Bros. The state of the game was encoded into various genre specific variables that denoted the state of Mario and the distance of Mario to various obstacles. This was handed as input to the neural network, which was then expected to output the buttons that should be pressed in that situation. The resulting weights were then evolved and evaluated using a fitness function. The fitness function in this case was the distance between the AI and player's traces through the level. A key improvement made to suit this genre was to reset the AI's position once the distance exceeded some threshold and apply a flat penalty. This is because an AI can easily get stuck in a 2D platformer, leading to a very bad final fitness score. The result was that the AI did the best job of mimicking human playstyles compared to many other algorithms. However, the agent achieved a lower score in the game compared to human players, showing that the agent had not really achieved a truly human-level of performance in the game.

2.3 Fighting Games and Neural Nets

Neuroevolutionary techniques have also been applied to fighting games. On a simple fighting game with 1 axis of movement, researchers found that evolutionary neural networks were able to quickly converge to an optimal playstyle [FightingAIComparison]. Additionally, Deep Reinforcement Learning has been able to create AI agents that can compete with top players in the popular commercial game Super Smash Bros. [SuperSmashBros]. However, optimal AI are not ideal substitutes for playing against human opponents. In the case of the Deep RL AI, it was specifically trained against only one kind of opponent, meaning that it was limited in the range of matchups it could perform well in. In addition, it exhibits obviously artificial traits such as impossible to perform back and forth movements.

2.4 Ghost AI

With regards to creating AI that was specifically human-like, the most notable attempt was something called Ghost AI, which was implemented on a version of the commercial game Street Fighter. This AI essentially initializes a table with the frequencies that the target player did a move and then used those moves at the same frequency[Ghost AI]. In the adaptive version, the AI would update the frequencies based on the reward gained from performing those moves [Ghost AI2].

To evaluate this AI, they recorded player sessions and sessions of the corresponding mimicked AI. The players were then asked to watch these recored sessions and perform inputs as if they were in the same situation. The recordings were then scored by the similarity of the recordings inputs to the "fake" test subject inputs. By this evaluation metric, this method showed promising results, as it was able match around 75% of the real recordings accuracy. Players also expressed high qualitative satisfaction with their recordings, and the adaptive component allowed the AI to adjust itself to the strategies of the opponent. This approach has some notable pitfalls, as it does not account for specific player strategies that include varying timing. It also fails to account for contextual information, such as position on the screen, which factors into the decision making process for a real human player.

2.5 Data-driven Finite State Machines

One final approach that to this problem was to create Data-Driven Finite State Machines. In this method, a multi-layered finite state machine is formed from the a log of a human demonstration. Specifically, the moves performed during the demonstration are annotated and used as designations for the AI states that it transitions between.

This approach has some clear limitations. For one, the annotation of moves is cumbersome and not well suited for a general purpose algorithm. Furthermore, the strategy that a player uses could be determined by an arbitrary number of in-game and out of game variables, which makes reducing player behavior to an FSM an daunting task. Lastly, this method was implemented on a 1D fighting game, which puts a huge limitation on the types of techniques that can be expressed by players.

2.6 High Level Overview

In this section we discussed several different existing methods for creating AI that mimic human-behavior. In domains where players progress on a path to an objective, such as racing games and 2D platformers, neuroevolution proves to be a strong strategy. However, there is a clear tradeoff between improving similarity and improving performance in these games, and even then these AI's have a hard time recovering from getting stuck.

When looking specifically at fighting games, there is currently a lack of new developments. Though neural methods have proven effective at creating optimal agents, they exhibit traits that prevent them from being suitable substitutes for human players. Other techniques such as Ghost AI have demonstrated an ability to express traits of human play, but lack the capability to respond to an opponents action in a human-like manner.

3 System Overview

We created our own fighting game called *FG* as a testbed. We did this because this allowed up to control and modify the dynamics of the game. This also gave us access to internal game data which would have been considerably more difficult to access had we instead opted to modify an existing fighting game.

The game is structured as a traditional fighting game. Players move back and forth on in a 2D space, trying to land blows on one another to reduce the opponents health to zero. Unlike other fighting games, the core

The game removes things like requiring complex inputs for special attacks, simplifying most fighting games to their core mechanics. The specific actions that players can take are described in the Additional Details section.

4 Algorithm Design and Implementation

5 Results

6 Discussion

7 Additional Details

Table 1: AI Situation description

Field Name	Description
x Position	Insert description here
y Position	Insert description here
x Velocity	Insert description here
y Velocity	Insert description here
opponents x Position	Insert description here
opponents y Position	Insert description here
opponents x Velocity	Insert description here
opponents y Velocity	Insert description here
status	Insert description here
opponents status	Insert description here

Table 2: Player Status descriptions

Status	Description
Stand	TEMP
Crouch	TEMP
Air	TEMP
Highblock	TEMP
Lowblock	TEMP
Hit	TEMP
KnockdownHit	TEMP
Tech	TEMP
Moving	TEMP
Dashing	TEMP
AirDashing	TEMP
StandAttack	TEMP When the opponent has the stand hitbox out
LowAttack	TEMP When the opponent has the low hitbox out
OverheadAttack	TEMP When the opponent has the overhead hitbox out
AirAttack	TEMP When the opponent has the AirAttack hitbox out
DP	TEMP When the opponent has the Dp hitbox out
Recovery	The recovery period after an attack

Table 3: Player Action descriptions

Action	Description
Stand	TEMP
Crouch	TEMP
WalkLeft	TEMP
WalkRight	TEMP
JumpNeutral	TEMP
JumpLeft	TEMP
JumpRight	TEMP
Attack	TEMP
Overhead	TEMP
LowAttack	TEMP
AirAttack	TEMP
StandBlock	TEMP
CrouchBlock	TEMP
DashLeft	TEMP
DashRight	TEMP
AirdashLeft	TEMP
AirdashRight	TEMP
DP	TEMP
TechNeutral	TEMP
TechLeft	TEMP
TechRight	TEMP