Cooperative Agents for Rocket League

Concept proposal

Diogo Silva*
diogo.goncalves.silva@tecnico.ulisboa.pt
Instituto Superior Técnico
Lisbon, PT

Jorge Brown†
jorge.brown@tecnico.ulisboa.pt
Instituto Superior Técnico
Lisbon, PT

Tiago Melo‡
tiago.melo@tecnico.ulisboa.pt
Instituto Superior Técnico
Lisbon, PT

ABSTRACT

Team-Based Multiplayer Video Games are games which force the cooperation of players in order to achieve a common goal, commonly by defeating an opposing team of players. These types of games represent a big chunk of the current industry market and as such a lot of effort has been dedicated into their development. A problem they face, however, is the fact that a lot of times players may not be able to fill an entire team, and as such, most games employ the usage of bots to sub in for the missing human players. Unfortunately, many of these agents are very limited and are often unable to coordinate and accomplish impromptu team tactics. One such game where this happens is Rocket League - a vehicular soccer game. By developing a team of agents capable of seamlessly coordinating with each other and doing tactical squad plays we aim to fix this issue and give human players more capable artificially intelligent teammates capable of fulfilling the void left by the lack of real human colleagues.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Multi-agent planning**; **Multi-agent systems**.

KEYWORDS

Multi-Agent System, Artificial Intelligence, Video Games, Team Coordination

ACM Reference Format:
Diogo Silva, Jorge Brown, and Tiago Melo. 2018. Cooperative Agents for Rocket League: Concept proposal. In *Proceedings of* . ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

The Video Game industry is in constant expansion. With booming hardware developments, increasingly cheaper processing and graphical power is available to most consumers which, in turn, allows for more complex games to be launched to the market. Furthermore, the Internet broadened the horizons even further by making it possible for geographically spread out individuals to play together, drastically changing the competitive and cooperative video game landscape altogether. Multiplayer team-based games now occupy a significant chunk of the industry.

This evolution in the industry has fomented the creation of a diverse array of real-time cooperative challenges which force players to coordinate towards a common goal. Although these multiplayer games are designed for Human-Human cooperation, a lot of the times players may require the assistance of Artificial Intelligent teammates in order to have a "full team" and be able to play. This may occur due to the lack of ability for the players to find enough colleagues to fill a team, or, more notably, due to players leaving mid-matches, for example, due to connectivity issues.

A good representative of such games, where team play and coordination is paramount to success is **Rocket League** - a 3D, third-person perspective, soccer video game in which each player controls a car in a team environment with the goal of scoring goals by bumping the ball into the opposing team's net, while protecting their own. Each match of *Rocket League* lasts for 5 minutes, at the end of which, the team with the most goals wins. Although the game offers several game modes, including a *1v1* mode, our focus will be directed towards the *2v2* and *3v3* environments, since these are the ones which require the most coordination and team tactics. Rocket League quickly soared to one of the most popular competitive online sports games and has sparked the interest of many intelligent agents developers, fomenting a healthy community of not only players but also avid programmers who pit their own bots against each other in organized tournaments, not unlike those seen in *Robot Soccer*.

Naturally, *Rocket League* is a game that lends itself to diverse team strategies and technical prowess. By developing a team of intelligent agents in this multi-agent system, capable of coordinating and fulfilling team roles, we can then use our agents to play with human players, seamlessly filling the void that the lack of a human teammate may create. Ultimately, our goal is to develop a set of bots capable of implementing effective team strategies and working together to achieve victory.

In terms of implementation, we will be building our bots and using them directly on the game via a framework called RLBot. This framework allows, not only the development of bots using an array of languages (such as Java, C++ and Python), but also provides a graphical interface that allows us to load in our bot scripts directly into the game, enabling us to play matches utilizing our bots.

*IST-ID: 98776
†IST-ID: 82416
‡IST-ID: 98773

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

, ,
© 2018 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

2 APPROACH

2.1 Player Actions

In *Rocket League*, players control their car from a third-person perspective (i.e each player follows their car from a camera placed behind their car) and are able to *accelerate forwards*, *backwards*, *turn left* or *right*, *jump*, *double jump* and use a consumable *turbo boost*. Whilst the first actions are self explanatory, the rest may seem odd and as such warrant some further explanation. *Jumping* simply lifts the car off the ground and, while on the air, the player may freely rotate their car or activate a *second jump* (which can only be done once before landing). The *Boost* is a consumable that the player may activate in order to give a speed boost to their car. While boosting, if they run into another car, they destroy it (and the player whose car was destroyed must wait a brief moment before being spawned into the world again). Activating the boost while midair, and combining it with the ability to rotate the car whilst airborne, allows the player to "fly" their car.

2.2 Environment

Rocket League takes place in a simplified version of a soccer field, with fewer players and reduced dimensions. The field is closed off by walls in order to prevent cars from leaving the map. Additionally, there are also power-ups scattered around the map that grant the player a given amount of the consumable turbo boost upon pick-up as seen in 1. At the start of each match, and after a goal is scored, each player is placed on a predetermined starting position in their team's side of the field and a super-sized soccer ball is placed in the middle.

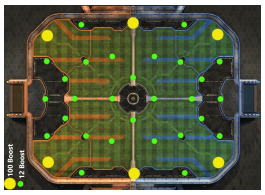

Figure 1: Overview of the field where Rocket League matches take place.

For a better categorization of *Rocket League*'s environment, we can say that it is **Multi-Agent**, **Stochastic**, **Sequential**, **Dynamic**, **Continuous** and **Partially Observable**. *Multi-Agent* because the game is played with several players, separated into opposing teams. *Stochastic* rather than Deterministic, because of the complex physics that the game employs - One cannot predict the exact speed and direction the ball will be sent off after a bump (only an approximated guess) due to all the factor that influence it, for example. *Sequential* rather than Episodic because, while the game does reset the ball and player's position after each goal, this is the only time where there is some respite. During actual game play, the players' decisions constantly impact the game and their future decisions, hence the game cannot be divided into independent episodes. *Dynamic* rather than Static because the state of the world (e.g position of the ball and players) is in constant change, even while the player deliberates. *Continuous* rather than Discrete because the speed and location of the ball and players varies in a range of continuous values smoothly over time. And finally, *Partially-Observable* rather than Fully due to the fact that players don't have access to the complete state of the world. While they can observe the overall position of boosts, the ball and other players, they can't accurately predict the ball's speed, the enemy team's strategy, velocity of other players, and so on.

2.3 Multi-agent System

The system comprises of teams of collaborative agents that work together towards the same goal: winning the game. As such, there ought to be an underlying component that allows the communication between agents in order for the aforementioned tactics to develop and take place. Although it will be clearer upon implementation, some instances of communications that can be expected revolve around strategic calls such as coordinating an attack or go on the defensive. Besides this, the system is also a competitive environment due to the fact that there are two teams of agents pitting themselves against each other to achieve opposing goals.

2.4 Architecture

In terms of architecture, we know our bots should implement a decentralized brain but should, nevertheless, be able to communicate with each other to arrange the aforementioned tactics. Due to the chaotic nature of the game, we also know our bots should be able to make decisions on the fly and quickly adapt to changes on the world. Still on that topic, we believe our bots should keep the state of the world and they should update it as appropriate - Their decisions should not only be based on what they see at the moment of deliberation, but on both past and present observations in order to allow for a more sophisticated reasoning . All in all, we will be implementing Rational Agents through a hybrid approach between Reinforcement Learning and certain hard coded meta rules, but as we delve deeper into implementation this is subject to change, and we will continuously employ some research and RLBot's active community feedback to foment any changes or improvements.

3 EMPIRICAL EVALUATION

Whilst individually we can evaluate each agent by metrics such as *number of goals scored*, *number of saves*, *number of assists*, and so on - all of which are metrics the game itself presents the players and hence can be extrapolated - what we are aiming to do is creating bots able to perform team plays and to play off of each other. As such, besides the aforementioned metrics we will also look at the overall *coordination* of the team, whether they're chasing the ball attempting individual plays or making passes to their teammates and setting up for shots and whether they're strategically communicating, employing layouts (such as having an agent play more defensively while the others advance on the field, for example), and whether they're able to dynamically adapt as the game goes on, adjusting their strategies and swapping between more defensive or aggressive plays, depending on the score and time left on the match. Furthermore, for comparisons sake there is also an active community of *Rocket League* bot makers, to whom we can compare and pit our bots against to evaluate their performance, most notably in community-organized bot tournaments that take place weekly.