Abstract—This document’s main target is the scientific field of Ad Hoc teamwork, in which an intelligent agent secures a position in a previously assembled team of agents. It must be able to coordinate with that team, without formerly acquired knowledge about their team strategies or communication protocols with the purpose of achieving a common goal (in our case, winning a RoboCup 2D Simulation League match). The difficulty of developing a solution that performs in complex domains poses a great challenge for us to solve.

We assume an approach that consists in identifying the task, teammates and planning towards achieving that goal. By applying machine learning techniques, we deployed our agent in a dynamic multi-agent environment (RoboCup 2D Simulation League), whose complexity is far bigger than the conventional settings used in Ad Hoc teamwork problems.

To achieve this goal, we modeled a Markov Decision Process (MDP) where the reward is initially unknown and is later obtained using Inverse Reinforcement Learning (IRL). The other agents and the formation are classified using supervised learning techniques. Finally, the planning consists in solving the MDP via a Policy Iteration algorithm. With the results obtained with this approach, we prove that it is possible to successfully use Ad Hoc teamwork in complex multi-agent domains.

Keywords—Intelligent Agents; Multi-agent Systems; Ad Hoc Teamwork; Markov Decision Processes; Inverse Reinforcement Learning

I. INTRODUCTION

Autonomous agents usage is increasing in recent years, with applications in areas such as health [Nealon and Moreno 2003] and robotics [Jones et al. 2006; Abbeel 2008]. The need for cooperative agents and agent teams arises, which is supported by the notion that a team of agents that coordinate their efforts to reach a common goal has a better performance, in general.

Even though attaining cooperation is a complicated goal, there is a considerable amount of literature on successful techniques for putting together a team of agents that performs well. A good example of these techniques is the early work of Tambe [1997] on flexible teamwork in complex, dynamic scenarios, such as the RoboCup synthetic soccer domain, where the author advances a model of teamwork that enables a team of agents, with inconsistent views of the world, to act in a coherent manner. The premise is that providing the agents in a team with the same teamwork model will enable them to overcome the conflicts created by the complexity of the environment and incoherent world models.

We often face situations where cooperation amongst agents that do not have prior knowledge about one another, do not communicate in the same way and may even have different world models, is needed in order to achieve certain goal or fulfill a given task. Some newly developed agents may need to be deployed in an environment where other agents are already operating. In some of these cases, the agents in place may no longer be altered, either due to financial reasons (the cost of upgrading them is not worth the effort) or their developer being out of business, for instance. These agents are known as legacy agents. To illustrate the utility of an ad hoc agent in a similar scenario, consider Example 1.

Example 1. Consider an industrial scenario, where several agents (robots) perform different tasks in an assembly line, from the manufacturing of the components to the assembly of the final product. If one of the agents malfunctions or has a performance below standard, it jeopardizes the entire business and costs the company money and time. In order to achieve maximum efficiency, that agent must be replaced, but that is a costly effort and the new agent may not know what its function is, causing the entire team to underperform. Therefore, it is worthwhile to have an agent that observes the behaviour of its teammates and can quickly use that knowledge to fill in the void.

This concept of collaboration without explicit coordination and knowledge about the teammates has been discussed under the name of ad hoc teamwork. The main challenge in ad hoc teamwork, as described by Stone et al. [2010] is “to create an autonomous agent that is able to efficiently and robustly collaborate with previously unknown teammates on tasks to which they are all individually capable of contributing as team members”.

More recent work, by Melo and Sardinha [2016], goes even further than the aforementioned by considering not only that the teammates are unknown but also the task to be performed, which introduces a new requirement to ad hoc teamwork, task identification. Despite these improvements, as the computational weight of the techniques applied increases dramatically with the number of actions available to the agent at a given time and the overall complexity of the task to be performed, the set of domains where these methods can be applied successfully is still somewhat limited. This is the main issue addressed in this paper.

A. Problem Description

Melo and Sardinha [2016] described ad hoc teamwork as a 3-step problem which includes task identification, teammate identification and planning. Although these have already been thoroughly scrutinized, there is a scalability issue: to the best of our knowledge, there still is no solution that successfully applies this 3-step approach to a complex domain where there are many agents in a team, many possible tasks to be performed and many possible actions available to each agent at a given time (with each action impacting the performance of the team as a whole in a different manner).

The techniques that have been studied so far, and combinations thereof, are still unable to provide an answer to the ad hoc teamwork problem which allows its application in a more complex domain, namely RoboCup simulation league.

RoboCup 2D Simulation League is a branch of the RoboCup tournament which consists in a 2D soccer match between two teams of eleven agents each, of them having several sensory inputs and several possible actions at a given time. The main mid-term research issues involved in this area of the RoboCup challenge, as described by Kitano et al. [1998] involve:

1) Machine learning in a multi-agent, collaborative and adversarial environment
2) Multi-agent architectures, enabling real-time multi-agent planning and plan execution in service of teamwork
3) Opponent modelling

Ensuing the above stated, we can conclude that the RoboCup 2D Simulation League provides us with a good domain to investigate the scalability of our Ad Hoc Teamwork approach.

We face a situation in which our ad hoc agent is deployed in a complex, adversarial environment and will try to coordinate with its team, to the best of its abilities, aiming towards an initially unknown goal. Moreover, this environment is unusually convoluted when compared to the classic ad hoc teamwork settings. Such environments include the pursuit domain, where a set of 4 agents tries to surround a prey (which moves randomly) in a grid world [Stone and Veloso 2000] and half field offense, which is a succession of episodic tasks in which an offense team of n players has to outsmart the defense team of n players to score a goal [Kalyanakrishnan et al. 2006]. These domains are somewhat limited in their complexity and number of agents. Thus, the problem we have in hands can be epitomized by the following question: “How can ad hoc teamwork be scaled to complex environments containing a large number of agents?”
B. Contributions

When deploying our agent in the Robocup 2D Simulation League environment, we want to create an autonomous agent that is able to:

1) Replace one of the agents in a previously assembled team
2) Figure out what the task at hand is
3) Identify its teammates
4) Become a part of the team
5) Improve the team’s performance

With the successful application of our agent in a testing environment such as the Robocup 2D Simulation League, we distance ourselves from the conventional ad hoc teamwork environments, towards a more complex one, thus addressing the scalability issue frequently associated to ad hoc teamwork solutions, which is the main contribution of our work.

Furthermore, we reach beyond the traditional view of ad hoc teamwork by implementing a solution that takes into account the fact that planning is not the only issue in play, despite its importance. Following on the work of Melo and Sarimha (2010), we adopt a perspective on ad hoc teamwork which regards task identification, team formation or behaviour and planning as the three key steps involved in this problem.

Unlike the aforementioned approach, we address these particular issues in a slightly different order. First, we tackle teammate identification using a supervised learning technique based on observations collected from the teammates. This provides us with the teammates’ models. Then, we address task identification, where we find a suitable reward policy for a Markov Decision Process (MDP), that represents our domain, using Inverse Reinforcement Learning (IRL) based on observations from a specific teammate, which is regarded as an expert in the domain. Finally, planning consists in solving the MDP via a Policy Iteration algorithm.

II. RELATED WORK

A. Impromptu Teams

Bowling and McCracken (2005) introduce a new concept of impromptu teams, in which a team is formed by a set of agents, each unknown to each other and each with its own skills and strategies, particularly on the case where only one of the members of the team is replaced by an independent one (the pickup player). Except for the pickup player, all the team members can communicate and coordinate normally, as they did before.

In the domain of robot soccer, the coordination, among the five robots that constitute the team, is achieved through the definition of a team strategy that specifies tactics for each team member, which are individual goals each robot has. Tactics are defined based on the robots’ individual skills, using a motion control algorithm and a path planning technique. Coordination can not be achieved through explicit communication because the agents may not be using the same communication protocols, thus making the exchange of information with the pickup player impossible.

The authors define two stages in the process of the pickup player’s integration in the already existing team, play selection and role selection. In the former, the pickup player decides which play it wants to execute and in the latter it assigns itself and its teammates a role in said play. The 1st stage is tested in two different variations: the adaptive variation where the agent uses learning to figure out which plays work best for its team based on which plays worked best in past situations, using Bowling et al. (2004)’s weight update play selection algorithm, and the predictive version, in which the agent selects the play based on the current position, trajectory of the teammates and the team’s current play style (defensive, regular or offensive). Both approaches performed equally well, with no significant differences.

Note that, unlike the work we propose, in this scenario, the team strategies and task are known a priori. Obtaining a coordinated behavior between the existing team and the new agent is an even bigger challenge for us.

B. Ad hoc agent teams

Stone et al. (2010) bring forward the notion of Ad Hoc Autonomous Agent Teams: heterogeneous teams of agents that may have different acting and sensory abilities and world models, are not even necessarily programmed by the same people and may not share communication protocols, thus shuttering any possibility of deploying team strategies a priori. The authors stress the importance of this topic with an example where a team of agents, unknown to each other and programmed by different people, is deployed in an unknown scenario to perform a rescue mission. The authors also use the concept of ad hoc human teams to exemplify the situation: in a medical emergency scenario, where a biker has an accident and is unconscious, a group of people rush to the scene with the common goal of helping said biker and, although they all know which steps need to be taken in order to help him (check if he is still breathing, call the ambulance, find a nearby policeman), they do not speak the same language and cannot coordinate explicitly.

A good ad-hoc agent would be able to efficiently examine the group and gauge each member’s abilities: for instance, if someone is a doctor, he/she should be the one to provide the victim with first aid. Note that, unlike Bowling and McCracken (2005)’s impromptu teams robot soccer scenario, there is no previously assembled team where only one member was replaced and the rest are still able to coordinate as before, neither is there a previously designed strategy. Their main challenge was to create a single autonomous agent without knowledge of its teammates and without explicit coordination protocols, as we have seen in Grosz and Sidner (1990)’s SharedPlans, that is robust and reliable in the long term, and whose performance was analysed in two different approaches.

The first of which is the theoretical approach, illustrated by the authors using an instance of the k-armed bandit problem (Robbins, 1952) known as the teacher and the learner, where 2 agents, a teacher and a learner who select arms alternately from a set of 3 with different payoffs, beginning with the teacher. The teacher’s goal is to maximize the expected sum of the payoffs received by the 2 agents.

The learner is immutable, unable to communicate with the teacher and no previous coordination strategies are in place, so the teacher must decide whether to always pick the arm with the highest payoff or allow the learner to gather some more information by picking a different one in the first round.

The second, and most pertinent to our work, is the empirical approach to the robot soccer scenario. But this is not the typical Robocup setting described before, in which the agent teams are built as a single unit with communication protocols and player decision algorithms to achieve the desired soccer formation. In this scenario (a pick-up game), the players are not able to coordinate prior to the game and the ad hoc agent should be able to identify the team’s vulnerabilities and act accordingly, either by filling in a gap in the formation or behaving in a manner that lessens those frailties, regardless of who its teammates’ are. This is a very interesting approach that already encompasses the task and teammate identification steps, which will be used in our work.

C. Apprenticeship learning

A great way of learning is to watch someone else perform the task you aim to learn. That is also true as far as autonomous agents are concerned. Abbeel (2008) contributed with a ground breaking approach to the apprenticeship learning field, where an agent has access to demonstrations of a task being performed by an expert agent and learns how to do it, that way. The author uses the expert demonstrations to get a description of the task at hand in the form of a reward function: \( R(s, a) \) represents the reward an agent gets when moving from a given state to another state by performing action \( a \).

The author models its problem in a Markov Decision Process where the reward function is not explicit and instead an expert is observed while performing the task. The expert is assumed to be trying to maximize a reward function that is a linear combination
of its features. This is tackled using an algorithm that retrieves an unknown reward function from the observed behavior. This problem is commonly referred to as inverse reinforcement learning.

Remember that solving an MDP means finding an optimal policy, a function $\pi(s)$ that returns the action, $a$, that an agent should perform when in state $s$. The size of the MDP grows exponentially: $|A|^{|S|}$.

This algorithm is proven reliable and the author demonstrates the learning agent achieves a comparable performance to the expert agent it observes, which, by mitigating the difficulty that is inherent to the specification of a reward function, eases the learning process.

This approach is particularly useful to us, since we need to determine the task, and this is a good way of achieving that purpose.

1) Agents’ modelling: Modelling both your teammates and your opponents plays a crucial role in the RoboCup Simulation League (a particular ad hoc teamwork setting). Pourmehr and Dadkhah (2012) performed an analysis on the existing approaches to this problem.

The authors identify opponent modelling as one of the major aspects for generating a competitive team and list a set of possible approaches to it.

Firstly, one can classify the current opponents into one of several models (Riley and Veloso 2000). This approach consists in:

1) Feature identification - identifying the other agent’s attributes from observation.
2) Model construction - building a model for the agent with the data gathered in the first step.
3) Classification to the predefined models - matching the model obtained in the second step with one of a list of models previously established.

Secondly, instead of classifying the agent itself, Riley and Veloso (2003) propose an approach that maps the opponent’s behavior to a set of known, pre-designed behaviors, using decision trees. This information is then used to select adaptive plays based on the opponent’s recent behaviors, online.

Lastly, the authors mention another work, from Lattner et al. (2006) who use association rule mining to predict other agents’ behavior by applying a sequential pattern mining algorithm that extracts patterns in data that describes the agent’s behavior. Then, these patterns are used to create rules that are enforced when trying to predict what will happen in the future. Although this is a powerful technique, the high complexity of its learning algorithm makes it impractical.

Although there is no correct answer for the modelling of other agents, the one that makes more sense to use, for its simplicity and adequacy to the problem, is the first mentioned in this section: a classifier that maps the model built from observing the other agents’ behaviour to one of several pre-defined behaviors.

D. The Ad Hoc teamwork problem in RoboCup

Shifting to the RoboCup domain, as ad hoc teamwork becomes an impacting research theme, a new RoboCup league, which focuses on teamwork without pre-coordination, was created in 2013 as an optional technical challenge (MacAlpine et al., 2014). In the following year it was upgraded to an official league; SPL, Drop-in Player Challenge, and the number of participants rose significantly (Agmon et al., 2015) performed an analysis on this competition at RoboCup 2014. It is important that we address this matter since our work will have a similar scope.

SPL (Standard Platform League) is one of the leagues within the Robot Soccer division at RoboCup. In SPL all teams must use a uniform robotic platform, NAO robots, to compete in soccer matches.

By analyzing the drop-in competition’s score schemes, namely the judge score, which is the score given to a team, by a human, based on several performance criteria, the authors conclude that the key indicators of the team’s proper cooperation are the pass performance and how often they push/bump into their own teammates.

The above is not easy to ensure, since the decision to play the ball depends on the teammates current positions and intentions and some robots can not communicate with their teammates, making it impossible for them to determine their intentions, at a given time. However, this is a simple but effective strategy, since the teams’ performance was quite good, as can be perceived by the close result of a match between the overall Standard Platform League champion and a team formed by the best 5 ad hoc teammates from the Drop-in Player Challenge (4-2 in favour of the previously assembled team). This analysis of the SPL Drop-in Player Challenge yields a valuable insight regarding the key indicators of a team’s teamwork performance, which we’ll use when asserting the validity of our work.

III. Multi-layer approach to Ad Hoc Teamwork

First of all, we start by defining the global strategy we adopted, on a macro level, when facing the ad hoc teamwork problem. Melo and Sardinha (2016) presented a 3-step approach to ad hoc teamwork which encompasses task identification, teammate identification and planning. As Melo and Sardinha (2016) pertinently pointed out, most of the currently existing work on the ad hoc teamwork problem focuses mostly on the planning step (Stone et al., 2013) Agmon and Stone, 2012, however planning alone is not enough to solve our problem, since both teammates and task are unknown and that information is essential for planning, so we need to retrieve it first.

We propose a new approach to ad hoc teamwork, based on Melo and Sardinha (2016)’s 3-step approach. A very simple view of our approach is depicted in Figure 1.

![Figure 1: The three steps of our approach to ad hoc teamwork.](image)

The first challenge we face is teammate identification, which consists in building a model of the ad hoc agent’s teammates based on stored observations of their past behavior, in order to predict their future behavior. As far as the implementation goes, we use a teammate classifier that maps the model created using the observed agents’ behavior to the model that best fits its description in a set of previously established standard behaviors, as we have seen in Section II-C. In this stage, the first thing we do is feature extraction (where we gather the necessary features from the data that results from the observation of the other agents), then model construction (where we build a model for those agents based on those features that describes the agents’ behavior) and classification (where we match the generated model to one of those that we have created previously and stored in our behavior models’ library). Note that if we classify the remaining agents properly, the task identification process becomes simpler.

The second challenge is task identification. It consists in determining and recognizing the task at hand which is not usually considered in this kind of problem, but we believe that it contributes to a more robust agent capable of adaptation to different task-oriented domains.

For the purpose of this task we modeled an MDP where the reward function, $R : S \times A \times S \rightarrow \mathbb{R}$, is unknown, as we have seen in Section II-C. The other agents are regarded as expert agents in the domain and the reward function is obtained by applying inverse reinforcement learning, using the observations of the experts’ behavior. In order to increase the efficiency of this process and obtain the best reward function possible we use the results from the first step (teammate identification) to help us identify the best agents to use as expert(s). We then use a task classifier that matches the observed task to one of the tasks in a library of pre-established tasks.
The third, and final challenge is planning. At this stage, once we have a reward function for our MDP, the planning step consists in solving the MDP. It may seem straightforward enough, but the MDP grows exponentially: $|A|^{|S|}$ and our domain is a complex one. To solve this, as we are dealing with a large set of possible policies, we implemented a Policy Iteration algorithm (and corresponding Policy Evaluation algorithm), a dynamic programming algorithm that iterates over a policy, generating a sequence of monotonically improving policies ($\pi^*(s)$) and value functions ($V^*$) iteration after iteration, until the optimal policy is reached.

A simplistic diagram of our approach is depicted in Figure 2:

![Figure 2: A graphical representation of the architecture of our approach. The ad hoc agent is on top, the main data structures we will be using are represented in the tables on the bottom. The 4 circles in the middle represent the 3 steps of the implementation (each color matches one of the steps) of ad hoc teamwork. The arrow indicates information exchange between the components of our approach.](image)

**Figure 3: A graphical representation of the architecture of our approach, applied to the RoboCup 2D Simulation League domain. The ad hoc agent is on top, the main data structures we will be using are represented in the tables on the bottom and the domain elements are on the left hand side. The 4 circles in the middle represent the 3 steps of the implementation (each color matches one of the steps) of ad hoc teamwork. The arrow indicates information exchange between the components of our approach.**

A. Domain Specific Architecture for RoboCup 2D Simulation League

In the environment in which we are going to be testing this particular approach, the teammate identification process consists in building and classifying the models of the ten remaining players of our team. If we can properly identify each of the ten remaining teammates’ roles, we can feed that information to the task classifier, that will build a task model based on in and try to match it to one of the known task models in its task library.

In our case, the task is the team formation in which our ad hoc team is going to play, as well as the position of our ad hoc team end up chasing the ball at the same time (causing the agents to stay back to avoid *swarming* the ball in neither far nor near, but it is within its reach). Otherwise, it should run to it.

Swarming is an undesirable collective behavior in all Robocup Soccer competitions, in which a large group of agents from the same team end up chasing the ball at the same time (causing the agents to clash together in a kind of swarm), instead of maintaining the formation positions assigned to them and offering passing alternatives to the agent that actually holds the ball. The *swarming* avoidance behavior was implemented in all of our agents and it was a large contributor to the organized team behavior that our teams display. It is manifested in the form of the action *Stay Back* when the ball is in range but a teammate is closer to it.

In our Robocup 2D Simulation League domain, the agent (player) may fit in one of four predetermined roles in the field: goalkeeper, defender, midfielder or forward.

The goalkeeper is a special kind of agent whose job is simply to prevent the ball from entering its goal. Therefore its behavior consists in kicking the ball away from its goal when the ball is near and keep its position when the ball is far away.

The defenders job is to clear the ball from the surroundings of their teammate that is far from them, if there is a visible one, otherwise from stealing the ball and scoring a goal.

The forwards are an exception, because their behavior is highly dependent on the roles assigned to them. With that in mind we created a set of different formations ("Playbook"). A formation is a set of agents’ descriptions that contain the agent’s initial position on the field ($X, Y$) and its role (Goalkeeper, Defender, Midfielder or Forward). The agent description is passed on to the agent when it is created and the formation emerges from all of the agents in each team acting according to their description.

These individual agent behaviors were defined as subsumption architectures [Brooks, 1986] based on some previous knowledge about real football players’ behaviors on a number of frequent situations encountered on a real football match. To achieve these architectures first we had to define a number of states in which the agents may find themselves and a number of compound actions available to them when in a given state. Each state is defined as a set of conditions that describe a common situation on the field. The states never overlap.

All agents share similar behaviors when the ball is far away or within reach, but not near its feet. If the ball is not visible or very far from the agent, it should turn at an angle and observe its surroundings until the situation changes (the forwards are an exception, because they run towards the ball even if it is very far from them). When the ball in neither far nor near, but it is within its reach, the agent must check if a teammate is closer to it. If that is the case, the agent should stay back to avoid *swarming*. Otherwise, it should run to it.

Swarming is an undesirable collective behavior in all Robocup Soccer competitions, in which a large group of agents from the same team end up chasing the ball at the same time (causing the agents to clash together in a kind of swarm), instead of maintaining the formation positions assigned to them and offering passing alternatives to the agent that actually holds the ball. The *swarming* avoidance behavior was implemented in all of our agents and it was a large contributor to the organized team behavior that our teams display. It is manifested in the form of the action *Stay Back* when the ball is in range but a teammate is closer to it.

In our Robocup 2D Simulation League domain, the agent (player) may fit in one of four predetermined roles in the field: goalkeeper, defender, midfielder or forward.

The goalkeeper is a special kind of agent whose job is simply to prevent the ball from entering its goal. Therefore its behavior consists in kicking the ball away from its goal when the ball is near and keep its position when the ball is far away.

The defenders job is to clear the ball from the surroundings of their goal, when they are near it. Their preference is to pass the ball to a teammate that is far from them, if there is a visible one, otherwise from stealing the ball and scoring a goal.

Visibility of an object, in this context, means being able to perceive its location using the sensors available to it.

The agent’s reach is a constant that defines the maximum distance at which the ball should be pursued by the agent.
The midfielders have a mixed behavior with some defensive and some offensive traits. Similarly to the defenders, the midfielders prefer to pass the ball uphill when they have a teammate in sight. However, unlike the defenders (and much like the forwards), the midfielders dribble the ball uphill if they do not have a free teammate to pass the ball to and shoot at goal once they are in shooting range.

The forwards are the agents responsible for most of the goal scoring chances in the team. They do not have defensive-minded behavior which translates to a disregard in keeping their position and a frequent chase of the ball. Whenever a forward has the ball and has the opponent goal in sight, it dribbles towards it until it is close enough to shoot and then shoots.

C. Teammate identification

1) Player classifier: The feature extraction stage of the classification consists in obtaining the offline observations of our ad hoc teammates. An observation, \( o \), from agent \( \text{ag} \) is defined as a pair \( o_{\text{ag}} = (s, a) \), which represents that agent \( \text{ag} \) performed action \( a \) when it was on state \( s \).

We obtain a set of observations, \( O \), from each agent (teammate) \( \text{ag} \). \( O_{\text{ag}} = \{(s_1, a_1), (s_2, a_2), \ldots (s_k, a_k)\} \) and, based on those, we build a model for each of them that describes their behavior in terms of which action they are expected to perform in each of the states \( s \in S \). Afterward, a set of predictors, one for each role, generates a score for that agent model in that role. The higher the score, the most likely it is that the agent is, in fact, performing that role.

Then, an algorithm uses those scores to produce a classification of the agent that translates into a role. The process bares some resemblance with the Weight Majority Algorithm described by [Littlestone and Warmuth 1994], but simpler.

We used two types of teammate models, best predictor model and mixed predictor model. In a best predictor model, an agent is classified as either a defender, midfielder, or forward, as opposed to the mixed predictor model where an agent may be classified as 60% defender and 40% midfielder, for instance.

The classification can be achieved either by using the best predictor approach or the mixed predictor approach. We tested both variants. In the best predictor approach, the classifier simply selects the role with the highest predictor’s score. In the mixed predictor approach, the classifier returns a membership degree, the value returned by the membership function, \( m \), for each of the roles. The membership degree for each role is computed dividing the predictor’s score for that role by the sum of the predictor scores. For example, computing the membership of the defender role consists in the following:

\[
m_{\text{Defender}} = \frac{\text{score}_{\text{Defender}}}{\sum_{\text{role}} \text{score}_{\text{role}}} \tag{1}
\]

The mixed predictor classification result is a tuple:

\[
\mathcal{C}_{\text{mixed-predictor}} = (m(\text{Defender}), m(\text{Midfielder}), m(\text{Forward})) \tag{2}
\]

D. Task identification

1) MDP modelling: In order to identify the task at hand, we regard our ad hoc agent as a decision maker in a dynamic environment that can be described using a Markov Decision Process. For that purpose, we need to define the elements of the tuple \((S, A, T, \gamma, R)\) that make up the MDP, as defined by [Ramachandran and Amir 2007]. The set of states, \( S \), and the set of actions, \( A \), match the states and actions described on Section III-B.

As we are dealing with a stochastic environment, the transition function, \( T : S \times A \times S \rightarrow [0, 1] \) needs to be defined, since the probability of transition from one state to another given an action is not always 1. It is a map \(< Key, Value >\) whose keys are

\[\text{state}_{\text{current}}, \text{action}_{\text{a}}, \text{state}_{\text{next}}\] and the value is the probability that \( \text{action}_{\text{a}} \) performed on \( \text{state}_{\text{current}} \) leads the agent to \( \text{state}_{\text{next}} \). The discount factor \( \gamma \), which represents the difference in importance between future rewards and present rewards is bounded \((0 \leq \gamma < 1)\). Under this criterion, future rewards are worth less than the current reward. If \( \gamma = 1 \), this would be the same as the total reward. When \( \gamma = 0 \), the agent ignores all future rewards. Having \( 0 \leq \gamma < 1 \) guarantees that, whenever the rewards are finite, the total discounted value of a policy \( \pi, V^\pi \), will also be finite. We experimented with several values for \( \gamma \) but it ended up being set to \( \gamma = 0.9 \).

As we described the reward function, \( R : S \times A \times S \rightarrow \mathbb{R} \), is initially unknown. To obtain it we use inverse reinforcement learning, which is further described in Section IIID2.

2) Formation classifier: An important step in the process of identifying the task is identifying the formation in which it is playing. This is achieved through a classification mechanism that matches the observed ad hoc formation to a set of formation models in our formation library.

Our formation library is a set of formation models. Each formation model is an object that describes a formation, namely the number of players that constitute it and the number of players in each of the roles (defender, midfielder or forward; the goalkeeper is present and unique in each formation and thus irrelevant to the classification mechanism).

The formation library includes a model for each of the formations in our playbook.

To be able to classify our ad hoc formation, first we need to model it, the same way we modelled the formations in our library. In order to obtain the number of players assigned to each role, we need to classify our ad hoc agent’s 9 teammates that play a field role (any role besides goalkeeper). This is achieved by the teammate classification process described in Section III-C, and corresponds to the feature extraction and model construction steps of the classification.

In the best predictor approach to teammate classification (Section III-C), the formation classifier is given as input (beside the formation library) the teammates’ best predictor models. In the mixed predictor approach it is given as input the teammates’ mixed predictor models, instead.

Independently of the approach used for teammate modelling, we compute a ratio for each player role \(d_{\text{Ratio}}, m_{\text{Ratio}}, f_{\text{Ratio}} \in \mathbb{R}^+\), for defenders, midfielders and forwards, respectively) between the number of observed players in that role in the ad hoc formation and the real number of players in each of the roles in each formation, as described below.

Given a formation \(F\) and an ad hoc formation \(F_{\text{AdHoc}}\):

\[
d_{\text{Ratio}}(F, F_{\text{AdHoc}}) = \min \left( \frac{\# \text{defenders}(F)}{\# \text{defenders}(F_{\text{AdHoc}})} \right)
\] \( \min \left( \frac{\# \text{midfielders}(F)}{\# \text{midfielders}(F_{\text{AdHoc}})} \right) \)
\[
m_{\text{Ratio}}(F, F_{\text{AdHoc}}) = \min \left( \frac{\# \text{midfielders}(F)}{\# \text{midfielders}(F_{\text{AdHoc}})} \right)
\]
\[
f_{\text{Ratio}}(F, F_{\text{AdHoc}}) = \min \left( \frac{\# \text{forwards}(F)}{\# \text{forwards}(F_{\text{AdHoc}})} \right)
\]

The similarity, \(\text{Sim}(F, F')\), is then, computed as a product of those ratios, for every formation, \(F\), in the ad hoc agent’s playbook.

The formation with the highest similarity measure, \(F^\star\), is then returned. To find the missing role in the selected formation, we simply subtract the defenders of the ad hoc formation, \(F_{\text{AdHoc}}\), to \(F^\star\) and repeat the process for midfielders and forwards. The remainder of one of those operations is going to be 1 (the missing role) and the other two are going to be 0.

3) Obtaining the reward function via IRL:

a) Selecting an expert: The other players are regarded as expert agents in the domain and the reward function is obtained using the observations of the experts’ behavior. To improve our chance of obtaining a suitable reward function for our agent, we simplified the
issue by selecting one particular agent to be regarded as the expert agent whose unknown reward function we are trying to discover.

The process of selecting the expert agent to use depends upon a successful classification of the formation in which the ad hoc team is playing (this process is defined in Section [II-D2]), which allows us to deduce which position is missing in said formation by subtracting the observed agent roles, from the remaining team members, to those that are listed on the formation description. To identify these players’ roles, we use the process described in Section [III-C]. Once we identify the position that is missing and which role the agent in that position is supposed to perform, we select one of the agents that performs that role in our ad hoc team and use it as an expert. For instance, if we identify our ad hoc formation as a 4-3-3 (4 defenders, 3 midfielders, 3 forwards) and we can observe 4 defenders, 3 midfielders but only 2 forwards in our team, our agent should probably perform the missing forward role and we are going to select a teammate that has been classified as a forward as our expert.

This selection process is an optimization to the learning process we used, we could instead average the behaviors of all observed agents, but then our agent would have an all round behavior, not “fitting” into a particular role description. In some other domains, this selection process might be more difficult or may even be unnecessary.

b) Bayesian IRL: IRL is currently viewed as a problem of inferring a single reward function that explains an agent’s behavior. However, there is too little information in a typical IRL problem to get only one answer. A probability distribution is needed in order to represent the uncertainty [Ramachandran and Amir, 2007]. Our approach’s solution is based on Ramachandran and Amir’s Bayesian IRL (BIRL), a model for IRL from a Bayesian perspective.

Given an expert, X, operating on our MDP, we assume that a reward function, R, is chosen for expert X from a prior distribution. P(R). We obtain a set of observations from X, O_X = \{(s_1, a_1), (s_2, a_2),...,(s_i, a_i)\} which means that X was in state s_i and took action a_i at time step i. α_X represents the degree of confidence we have that X will choose actions with high value. We make the following assumptions:

1) X is attempting to maximize the total accumulated reward according to R. For example, X is not using an epsilon greedy policy to explore his environment.
2) X executes a stationary policy, i.e. it is invariant with respect to time and does not change depending on the actions and observations made in previous time steps.

Since X’s policy is assumed to be stationary, we can assume that:

$$Pr_X(O_X|R) = Pr_X((s_1, a_1)|R)Pr_X((s_2, a_2)|R)...Pr_X((s_i, a_i)|R)$$

We define the optimal Q-function, Q^*(s,a), as the Q-function of the optimal policy, π^*, for reward function R. The expert’s goal is to maximize accumulated reward. This can be posed as finding the action for which Q^* value at each state is at its peak. The larger the Q^*(s,a), the more likely the expert would choose to perform action a when in state s. Ramachandran and Amir [2007] model this as an exponential distribution for the likelihood of (s_i, a_i) with Q^* as a potential function, as follows:

$$Pr_X((s_i, a_i)|R) = \frac{1}{Z_i}e^{Q^*(s_i, a_i)}$$

where Z is a normalizing constant. Thus, extrapolating for the universe of expert observations, the likelihood is:

$$Pr_X(O_X|R) = \frac{1}{Z}e^{E(O_X, R)}$$

where \(E(O_X, R) = \sum_s Q^*(s, a)\) and \(O_X = \{(s_1, a_1), (s_2, a_2),...,(s_i, a_i)\}\) is a set of observations from X. If we apply Bayes’ theorem, we can compute the posterior probability of the reward function, R, as follows:

$$Pr_R(O_X|R) = \frac{Pr_X(O_X|R)Pr_R(R)}{Pr_O(O_X)} = \frac{1}{Z}e^{E(O_X, R)}Pr_R(R)$$

where Z’ is a normalizing constant.

Next, as Ramachandran and Amir [2007] explain, it is required to compute the mean of the posterior distribution. Since computing the mean of the posterior distribution is a computationally hard process, we instead generate samples from the distributions and compute their mean, returning it as an estimate of the real mean. The sampling algorithm we use for that purpose is Policy Walk. An auxiliary algorithm, Policy Iteration, is required for this task. Policy Walk, is portrayed in Figure [4].

**Algorithm PolicyWalk(Distribution P, MDP M, Step Size δ )**

1. Pick a random reward vector R ∈ R^|S|/δ.
2. π := PolicyIteration(M, R)
3. Repeat
   a) Pick a reward vector R uniformly at random from the neighbours of R in R^|S|/δ.
   b) Compute Q^*(s, a, R̃) for all (s, a) ∈ S, A.
   c) If ∃(s, a) ∈ (S, A), Q^*(s, π, R̃) < Q^*(s, a, R̃)
      i. π := PolicyIteration(M, R̃, π)
      ii. Set R := R̃ and π := π with probability min{1, P(R | π)}
      Else
      i. Set R := R̃ with probability min{1, P(R | π)}
4. Return R

![Figure 4: The policy walk algorithm.](image)

The Policy Walk algorithm is a modified version of another Markov Chain Monte Carlo algorithm, Grid Walk [Vempala 2005], that generates a Markov Chain. However, Policy Walk makes use of the auxiliary Q-function and is more efficient [Ramachandran and Amir 2007], so it suits our purpose.

Having done this, we now have a reward function, R : S × A × S → R, for our MDP, and the task identification process is concluded.

**E. Planning**

The final challenge of successfully implementing ad hoc teamwork is planning. Once we have a reward function for our Markov Decision Process, the planning step consists in solving the MDP (finding an optimal policy π(S) for each state returns an action for the agent to perform). As we are dealing with a large set of possible policies, we implemented a Policy Iteration algorithm (and corresponding Policy Evaluation algorithm), which consists in continuously improving a policy, every iteration.

With this explanation, we conclude our solution’s description and will now present the results we obtained, on the following section.

IV. RESULTS

Since our approach can be subdivided into three steps, it is only logical that we evaluate each of those steps separately, first, and then as a whole.

As far as the teammate identification step goes, its performance can be measured in terms of our agent’s success in identifying which roles its teammates are committed to. As described in Section [III-C], the task identification performance depends hugely upon the success (or failure) of this task.

Regarding the task identification step, its performance can be evaluated by determining our agent’s success rate when identifying the formation in which its team is playing as well as figuring out which role it should be playing in said formation.

When it comes to evaluating our planning step, due to difficulty of evaluating the obtained policy in an intelligible way, our approach
was to evaluate the agent performance as a virtual soccer player in the team. Alongside that we also evaluated the performance of the ad hoc team, which the ad hoc agent influences.

First off, we needed to establish some kind of baseline, so we could compare that with the performance of our ad hoc team. Thus, we used a set of 2 RoboCup 2D Simulation League teams (created by us), selected one of them to be our opponent team and the other to be our host team (the team which would later have one of its agents replaced by ours). Then, we ran a set of matches between the host team and the other team, using all the team’s different formations, which, as described in Section II-A, may have a great deal of influence in the team’s performance.

After having ascertained that, we are ready to replace a randomly selected agent from our host team with our own ad hoc agent. Having done so, we test the ad hoc team against the same opponent team as before, in the same combinations of team formations.

As [Genter et al., 2015] showed, we must analyze the match scores. However, that is insufficient, because the team can have a drop in other performance indicators and still score goals. As such, we also studied our team’s passing accuracy and passing frequency. Furthermore, we analyzed how often the ad hoc agent bumps with its teammates, bumping frequency. If it occurs very frequently, it’s an indicator of bad coordination.

We can also analyze the ad hoc agent’s influence in the team by comparing its individual stats with the team’s, so we can determine if our agent is a good, bad or neutral influence to the team’s performance. The agent’s ability to dribble the ball successfully towards the opponent’s goal is also a good performance indicator. We consider a dribble successful if it is not intercepted by a rival player or if the ball does not cross the field’s limits after the player tries to dribble. A “smarter” agent is able to know when to dribble, to prevent the loss of the ball’s possession.

We ran two sets of trials for each individual step and for the in-game tests of our approach. The first set was run with clear data, straight from the observations of the expert agent(s). The second set was run with noisy data, in which we randomly replaced ~10% of the observations fed to the agent by changing the action performed in that state to a randomly selected action from the set of actions. By doing that we intend to prove the robustness and tolerance to perturbations of our approach.

### A. Teammate Identification Tests

The results from the teammate identification process (identifying the teammates’ roles) are shown on Table II. On the 2nd, 3rd and 4th column, we list the average normalized score attributed by the defender predictor, the midfielder predictor and the forward predictor, respectively, based on the offline observations of the agent being classified. On the 5th column we can observe the best predictor model’s average classification and on the 6th (last) column, the mixed predictor model’s average classification (a triplet that consists of the average defender, midfielder and forward normalized scores). Considering the impossibility of gathering data and performing this task while the game is underway (online), these scores were achieved by running tests over the offline observations we gathered from previous matches.

On Table III we display the results from the teammate identification tests, performed with noisy data. Although the results obtained in these tests were not as good as those obtained with clear data, as expected, the agent is still able to perform a correct teammate identification, in average.

Overall, the results obtained were very positive, as can be deduced from the previously mentioned tables, thus contributing to better task identification results and, therefore, a better agent performance.

### B. Task Identification Tests

As explained before, we used two different approaches to tackle this problem, best predictor model and mixed predictor model. We tested them separately so we could determine which performs best in this task. To properly test this, we placed the agent in every possible scenario, by removing 1 agent from each role from each of the formations in our playbook.

The results from the best predictor model when identifying the formation are shown on Table III. As can be observed, the agent correctly identified the formation in 87% of the scenarios. In the only 2 cases in which it missed, the correct formation was the 245 with the highest score, which means it was not far off. Since the teammate classifications are absolute in the best predictor models, and the agent got all of them right in these tests, the results of the trials performed with noisy data were the same. So there is no need to elaborate further.

### Table I: Teammate identification test results (average scores and classifications) - clear data.

<table>
<thead>
<tr>
<th>Role</th>
<th>Defender Normalized Score</th>
<th>Midfielder Normalized Score</th>
<th>Forward Normalized Score</th>
<th>BP Model Classification</th>
<th>MP Model Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defender</td>
<td>0.878</td>
<td>0.060</td>
<td>0.060</td>
<td>[1, 0, 0]</td>
<td>[0.878, 0.060, 0.060]</td>
</tr>
<tr>
<td>Midfielder</td>
<td>0.091</td>
<td>0.818</td>
<td>0.090</td>
<td>[0, 1, 0]</td>
<td>[0.091, 0.818, 0.090]</td>
</tr>
<tr>
<td>Forward</td>
<td>0.034</td>
<td>0.254</td>
<td>0.712</td>
<td>[0, 0, 1]</td>
<td>[0.034, 0.254, 0.712]</td>
</tr>
</tbody>
</table>

### Table II: Teammate identification test results (average scores and classifications) - noisy data.

<table>
<thead>
<tr>
<th>Role</th>
<th>Defender Normalized Score</th>
<th>Midfielder Normalized Score</th>
<th>Forward Normalized Score</th>
<th>BP Model Classification</th>
<th>MP Model Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defender</td>
<td>0.792</td>
<td>0.102</td>
<td>0.106</td>
<td>[1, 0, 0]</td>
<td>[0.792, 0.102, 0.106]</td>
</tr>
<tr>
<td>Midfielder</td>
<td>0.129</td>
<td>0.758</td>
<td>0.113</td>
<td>[0, 1, 0]</td>
<td>[0.129, 0.758, 0.111]</td>
</tr>
<tr>
<td>Forward</td>
<td>0.093</td>
<td>0.293</td>
<td>0.654</td>
<td>[0, 0, 1]</td>
<td>[0.093, 0.254, 0.654]</td>
</tr>
</tbody>
</table>

### Table III: Task identification test results - best predictor model - formation selection - clear data.

<table>
<thead>
<tr>
<th>Formation</th>
<th>Missing Role</th>
<th>631 Score</th>
<th>641 Score</th>
<th>642 Score</th>
<th>631 Score</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>631</td>
<td>Defender</td>
<td>0.833</td>
<td>0.750</td>
<td>0.149</td>
<td>0.750</td>
<td>0.100</td>
</tr>
<tr>
<td>631</td>
<td>Midfielder</td>
<td>0.666</td>
<td>0.146</td>
<td>0.083</td>
<td>0.146</td>
<td>0.055</td>
</tr>
<tr>
<td>631</td>
<td>Forward</td>
<td>1.000</td>
<td>0.625</td>
<td>0.125</td>
<td>0.625</td>
<td>0.083</td>
</tr>
<tr>
<td>541</td>
<td>Defender</td>
<td>0.500</td>
<td>0.800</td>
<td>0.250</td>
<td>0.500</td>
<td>0.166</td>
</tr>
<tr>
<td>541</td>
<td>Midfielder</td>
<td>0.833</td>
<td>0.750</td>
<td>0.149</td>
<td>0.750</td>
<td>0.100</td>
</tr>
<tr>
<td>541</td>
<td>Forward</td>
<td>0.750</td>
<td>1.000</td>
<td>0.133</td>
<td>0.399</td>
<td>0.083</td>
</tr>
<tr>
<td>343</td>
<td>Defender</td>
<td>0.083</td>
<td>0.133</td>
<td>0.666</td>
<td>0.133</td>
<td>0.444</td>
</tr>
<tr>
<td>343</td>
<td>Midfielder</td>
<td>0.166</td>
<td>0.149</td>
<td>0.750</td>
<td>0.149</td>
<td>0.222</td>
</tr>
<tr>
<td>343</td>
<td>Forward</td>
<td>0.188</td>
<td>0.300</td>
<td>0.666</td>
<td>0.300</td>
<td>0.444</td>
</tr>
<tr>
<td>361</td>
<td>Defender</td>
<td>0.166</td>
<td>0.266</td>
<td>0.149</td>
<td>0.666</td>
<td>0.500</td>
</tr>
<tr>
<td>361</td>
<td>Midfielder</td>
<td>0.300</td>
<td>0.500</td>
<td>0.266</td>
<td>0.480</td>
<td>0.277</td>
</tr>
<tr>
<td>361</td>
<td>Forward</td>
<td>0.250</td>
<td>0.599</td>
<td>0.222</td>
<td>1.000</td>
<td>0.333</td>
</tr>
<tr>
<td>262</td>
<td>Defender</td>
<td>0.091</td>
<td>0.266</td>
<td>0.148</td>
<td>0.166</td>
<td>0.500</td>
</tr>
<tr>
<td>262</td>
<td>Midfielder</td>
<td>0.099</td>
<td>0.160</td>
<td>0.355</td>
<td>0.277</td>
<td>0.833</td>
</tr>
<tr>
<td>262</td>
<td>Forward</td>
<td>0.166</td>
<td>0.266</td>
<td>0.148</td>
<td>0.666</td>
<td>0.500</td>
</tr>
</tbody>
</table>

The results from the best predictor model when identifying the formation are shown on Table III. As can be observed, the agent correctly identified the formation in 87% of the scenarios. In the only 2 cases in which it missed, the correct formation was the 245 with the highest score, which means it was not far off. Since the teammate classifications are absolute in the best predictor models, and the agent got all of them right in these tests, the results of the trials performed with noisy data were the same. So there is no need to elaborate further.
Table IV: Task identification test results - best predictor model - role selection - clear data.

<table>
<thead>
<tr>
<th>Real Formation</th>
<th>Selected Formation</th>
<th>Real Role</th>
<th>Defender Probability</th>
<th>Midfielder Probability</th>
<th>Forward Probability</th>
<th>Score 1</th>
<th>Score 2</th>
<th>Score 3</th>
<th>Score 4</th>
<th>Score 5</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>631</td>
<td>631</td>
<td>Defender</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Defender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>631</td>
<td>631</td>
<td>Midfielder</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Midfielder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>631</td>
<td>631</td>
<td>Forward</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Forward</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>541</td>
<td>541</td>
<td>Defender</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Defender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>541</td>
<td>541</td>
<td>Midfielder</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Defender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>541</td>
<td>541</td>
<td>Forward</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Forward</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>343</td>
<td>343</td>
<td>Defender</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Defender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>343</td>
<td>343</td>
<td>Midfielder</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Midfielder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>343</td>
<td>343</td>
<td>Forward</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Forward</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>361</td>
<td>361</td>
<td>Defender</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Defender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>361</td>
<td>361</td>
<td>Midfielder</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Midfielder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>361</td>
<td>361</td>
<td>Forward</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Forward</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>262</td>
<td>262</td>
<td>Defender</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Defender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>262</td>
<td>262</td>
<td>Midfielder</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Midfielder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>262</td>
<td>262</td>
<td>Forward</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Defender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Although it may appear a coincidence, this can be explained by the mechanism used by our agent (in the best predictor approach) to deduce the missing role from the identified formation. The teammate classification is absolute, in this case, which means that if the agent identifies a 361 formation (last row in Table IV) it is expecting exactly 3 defenders, 6 midfielders and 1 forward. When it observes 2 defenders, 6 midfielders and 1 forward, the missing role is identified as defender. From this we can extrapolate that whenever the agent fails to identify the formation it will fail in identifying the missing role.

We identified that as the main weakness of this approach, however, note that it only happens when the observed ad hoc formation is very similar to 2 or more formations in our playbook. This is why the agent scores the correct formation with a high score as well (2nd best). To sum up, this approach proves to be very reliable. As we had discussed with formation selection, the role selection process with noisy data performed equally to the one with clear data.

The results from the mixed predictor model when identifying the missing formation, when it is fed clear observations data, are shown on Table V. As can be observed, the agent correctly identified the formation in 87% of the scenarios. In the only 2 cases in which it missed, the correct formation was the 2nd with the highest score, as had already happened with the best predictor model. In fact, as far as the formation identification process is concerned, these two approaches perform equally well. However, the best predictor model delivers predictions in which the difference between the scores of the selected formation and the next best formation is generally higher than in the mixed predictor model.

On Table VI we present the results of the trials analog to those we just mentioned, but conducted with noisy observations’ data. If we weigh the data in this analysis table against the data from the trials with clear data, we can conclude that the performance of the formation selection was very similar. The noisy data used in these trials did not impact the results obtained as much as expected. In the cases where the formation selection fails, the correct alternative is the one with the second highest score. This is another step towards proving our approach’s robustness.

The results from the mixed predictor model when identifying the missing role are shown on table VII. As we mentioned on Section IV-A, the mixed predictor model has difficulties finding out which role is missing, since this depends on accurate player classification. Since all defenders and forwards exhibit some traits that could also be found in midfielders, most defenders and forwards get a high enough midfielder score to disturb the process of finding the missing role.

For instance, consider the case in which a defender performs a pass to a teammate. Performing a short pass to a teammate is a behavior typically associated with midfielders, so the midfielder predictor will increase the agent’s midfielder score, which means that it is more likely that the agent in question is a midfielder.
type of event repeated, over time, will contribute to reaching a less than accurate conclusion regarding the missing role in the ad hoc team. This difficulty translates to a poorer role selection than the one observed in the best predictor model test results, with an accuracy of approximately 47% versus the much higher 87% we had seen before.

On Table VIII we present the results of the trials analogous to those we just mentioned, but conducted with noisy observations’ data. If we weigh the data in this analysis table against the data from the trials with clear data, we come to the conclusion that there are few differences in terms of the classification efficacy. However, in this second test of sets, the classifications have less certainty, i.e., the difference between the selected role and the next best option is smaller than in the previous tests, which denotes a decrease in the quality of this process, due to the noisy data used.

### Table VIII: Task identification test results - mixed predictor model - role selection - noisy data.

<table>
<thead>
<tr>
<th>Real Formation</th>
<th>Selected Formation</th>
<th>Real Role</th>
<th>Defender Probability</th>
<th>Midfielder Probability</th>
<th>Forward Probability</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>631</td>
<td>631</td>
<td>Defender</td>
<td>0.483</td>
<td>0.066</td>
<td>0.451</td>
<td>Defender</td>
</tr>
<tr>
<td>631</td>
<td>631</td>
<td>Midfielder</td>
<td>0.489</td>
<td>0.230</td>
<td>0.281</td>
<td>Defender</td>
</tr>
<tr>
<td>631</td>
<td>631</td>
<td>Forward</td>
<td>0.442</td>
<td>0.058</td>
<td>0.500</td>
<td>Forward</td>
</tr>
<tr>
<td>541</td>
<td>541</td>
<td>Defender</td>
<td>0.219</td>
<td>0.262</td>
<td>0.520</td>
<td>Forward</td>
</tr>
<tr>
<td>541</td>
<td>631</td>
<td>Midfielder</td>
<td>0.483</td>
<td>0.066</td>
<td>0.451</td>
<td>Forward</td>
</tr>
<tr>
<td>541</td>
<td>541</td>
<td>Forward</td>
<td>0.267</td>
<td>0.233</td>
<td>0.500</td>
<td>Forward</td>
</tr>
<tr>
<td>343</td>
<td>343</td>
<td>Defender</td>
<td>0.437</td>
<td>0.132</td>
<td>0.431</td>
<td>Defender</td>
</tr>
<tr>
<td>343</td>
<td>343</td>
<td>Midfielder</td>
<td>0.488</td>
<td>0.520</td>
<td>0.432</td>
<td>Midfielder</td>
</tr>
<tr>
<td>343</td>
<td>343</td>
<td>Forward</td>
<td>0.336</td>
<td>0.328</td>
<td>0.336</td>
<td>Forward</td>
</tr>
<tr>
<td>361</td>
<td>361</td>
<td>Defender</td>
<td>0.231</td>
<td>0.490</td>
<td>0.279</td>
<td>Midfielder</td>
</tr>
<tr>
<td>361</td>
<td>541</td>
<td>Midfielder</td>
<td>0.090</td>
<td>0.484</td>
<td>0.426</td>
<td>Midfielder</td>
</tr>
<tr>
<td>361</td>
<td>361</td>
<td>Forward</td>
<td>0.065</td>
<td>0.500</td>
<td>0.435</td>
<td>Midfielder</td>
</tr>
<tr>
<td>262</td>
<td>262</td>
<td>Defender</td>
<td>0.468</td>
<td>0.478</td>
<td>0.057</td>
<td>Midfielder</td>
</tr>
<tr>
<td>262</td>
<td>262</td>
<td>Midfielder</td>
<td>0.451</td>
<td>0.457</td>
<td>0.092</td>
<td>Midfielder</td>
</tr>
<tr>
<td>262</td>
<td>262</td>
<td>Forward</td>
<td>0.231</td>
<td>0.490</td>
<td>0.279</td>
<td>Midfielder</td>
</tr>
</tbody>
</table>

Despite adding noise to the experts’ observations’ data, the agent was able to correctly identify the formation in which it was playing and select the appropriate role most of the times. The imperviousness of our approach to the noise in the data becomes clearer when we compare the results with clear and noisy observations.

Although these test results were favourable, we must underline the superior performance of the simpler best predictor teammate models when compared to the more complex mixed predictor teammate models. This becomes evident to a greater extent if we compare the efficacy of the role selection mechanism in both approaches.

### C. In-Match Agent and Team Performance

1) Baseline Performance: One of the most important indicators that our agent is performing well (or not) is its behavior when deployed in a match. In order to determine what the introduction of the ad hoc agent changes in the match(es), we first need to assert how the teams perform without its presence.

First off, we started by conducting a set of matches among all formations (all versus all) and extracted some important data. The indicators we gathered match with those previously mentioned in the beginning of this section. Note that we discarded the matches between two teams using the same formation, as no useful information could be drawn from those matches, since the agents in each team are exactly the same.

Although we could confirm that, as expected, there is a big uncertainty associated with these matches, some conclusions can certainly be drawn from this data. The passing accuracy is generally very high, certainly higher than in real football matches, which contributes to a more pleasant match to watch but does not necessarily lead to wins, as we can deduce from the data in Table IX. Bumping into teammates is not an issue as big as expected, since there is only 1 bump per game, on average. The shooting accuracy is also very high and that contributes to a lot of goals being scored every match (an average of ~8.6 in all matches).

We sum up the statistics from these matches in Table IX which we will now focus on, trying to determine which factors contributed most to the success/failure of certain formations. The shots, goals and passes represented in this table are average values per match.

<table>
<thead>
<tr>
<th>Formation</th>
<th>Wins</th>
<th>Draws</th>
<th>Losses</th>
<th>Goals (Scored - Conceded)</th>
<th>Passes (Attended - Completed)</th>
<th>Pass Accuracy (%)</th>
<th>Shots (Fired - Allowed)</th>
<th>Shot Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>541</td>
<td>3-0</td>
<td>6</td>
<td>2-1</td>
<td>6.25 - 4.75</td>
<td>10.25 - 14.00</td>
<td>86.15</td>
<td>7.25 - 4.75</td>
<td>86.21</td>
</tr>
<tr>
<td>343</td>
<td>2-1</td>
<td>1</td>
<td>1</td>
<td>5.00 - 5.50</td>
<td>9.75 - 8.25</td>
<td>84.62</td>
<td>5.25 - 6.25</td>
<td>95.24</td>
</tr>
<tr>
<td>631</td>
<td>2-2</td>
<td>0</td>
<td>0</td>
<td>3.50 - 3.75</td>
<td>13.00 - 11.50</td>
<td>76.67</td>
<td>3.75 - 4.5</td>
<td>86.95</td>
</tr>
<tr>
<td>361</td>
<td>1-1</td>
<td>2</td>
<td>2</td>
<td>4.25 - 4.50</td>
<td>24.50 - 19.50</td>
<td>79.59</td>
<td>6.00 - 5.5</td>
<td>70.83</td>
</tr>
<tr>
<td>262</td>
<td>1-0</td>
<td>3</td>
<td>2</td>
<td>4.25 - 6.00</td>
<td>16.75 - 14.75</td>
<td>88.06</td>
<td>5.25 - 7.25</td>
<td>80.95</td>
</tr>
</tbody>
</table>

Table IX: Test matches’ statistics - regular teams - average values per match ordered from best to worst win-draw-loss ratio, disregarding matches between two equal formations.

Starting from the top, the most successful formation was clearly the 541. Even though this formation produced better results than all others, most of the matches it lost were the ones where the shot accuracy was the lowest (this formation averaged 86.21%).

That, combined with the higher shot accuracy of the top 3 most successful formations, brings us to the conclusion that shot accuracy is, in fact, one of the most important factors to consider. Also, shot frequency is the highest in the top formation (the 541). If we turn our attention to passing accuracy, we will find that, against all odds, the team with the highest passing accuracy was the least successful (262 formation). This is due to the fact that a crowded midfield sector (6 players in that role) translates to more short passes, which are easier to complete. Although the team in this formation passes the ball around nicely, the lack of defenders (only 2) leads to a lot of chances from their rivals and a lot of conceded goals.

We can observe that the amount of conceded goals decreases with the increase in the number of defenders in the formation, which is logical, however, not conceding as much is not enough to win (the 631 is a very defensive approach - see Section III-B - and its results were only average). On the opposite end of the spectrum, an increase in the number of forwards does not always translate to an increase in goal chances. A crowded defense and a crowded midfield sector, even with less forwards, prevails over an unbalanced attacking formation.

2) Ad Hoc Agent/Team Performance: Now that we have established a baseline performance, we will proceed to reveal the results obtained when replacing one of the agents from the home team with our ad hoc agent, in each match and then compare those with the indicators gathered above. This time around, we also had to run every possible game in which the ad hoc formation was the home formation. Earlier some games were omitted since those games had already been played with the home and away sides reversed (on the previous tests it did not make sense to play a 541 vs 631, since we had already played a 631 vs 541 with no ad hoc players).

The tests referred in this section were ran with noisy data. The in-game tests with clear data are not explicitly described in this particular section, since the results did not differ much from those conducted with noisy data. Given the imperviousness of the teammate identification and task identification steps to the noise in the observations’ data (see Sections V[B] and V[A]), it seemed pointless to detail two very identical results, so we selected the one that puts our approach to a harder test. The aforementioned tests allow us to compare the results and performance indicators prior and posterior to the insertion of the ad hoc agent, since the only factor that changed was exactly its presence. The ad hoc team is always the home team and the away
team is always a regular team (with no ad hoc agent). Remember that our agent will sometimes fail to identify the missing position due to an error in the task identification or teammate identification process.

Having analyzed both the ad hoc team, in general, and the ad hoc agent, in particular, we summed up the test matches’ statistics in Table VI. If we compare these numbers with those in Table VI we notice an increase in passing accuracy, as well as an obvious increase in the number of shots fired and goals scored by the home team (ad hoc). The shot accuracy is lower but it is compensated by the shot frequency (which is a determining factor in the team’s success, as can be perceived by the win-draw-loss column). No particular formation performed better than all of the others, however we maintain that the formations with less defenders concede the most goals and vice-versa. The 262 performed worse than all others due to our ad hoc agent’s misidentification of the missing position mentioned before.

If we analyze the remaining collective performance indicators, we reach the conclusion that there was an increase in the passing accuracy and shooting frequency (which seem to be the key drivers of a good performance). The bumping frequency remained fairly low, the passing frequency dropped very slightly and the shot accuracy also dropped due to an increase in the number of shots off target or saved.

<table>
<thead>
<tr>
<th>Ad Hoc Formations</th>
<th>Wins</th>
<th>Draws</th>
<th>Losses</th>
<th>Goals (Scored-Conceded)</th>
<th>Passes (Attempted-Completed)</th>
<th>Pass Accuracy (%)</th>
<th>Shots (Fired-Allowed)</th>
<th>Shot Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>541</td>
<td>3-2-0</td>
<td>11.20</td>
<td>9.40</td>
<td>83.93</td>
<td>6.20-4.40</td>
<td>67.74</td>
<td></td>
<td></td>
</tr>
<tr>
<td>343</td>
<td>3-1-1</td>
<td>4.40</td>
<td>2.80</td>
<td>91.58</td>
<td>6.40-3.80</td>
<td>68.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>631</td>
<td>3-1-1</td>
<td>3.80</td>
<td>3.00</td>
<td>91.93</td>
<td>6.20-4.20</td>
<td>61.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>262</td>
<td>2-2-1</td>
<td>3.00</td>
<td>3.50</td>
<td>90.12</td>
<td>6.30-4.40</td>
<td>88.24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>361</td>
<td>1-1-3</td>
<td>2.80</td>
<td>3.20</td>
<td>88.37</td>
<td>4.40-4.80</td>
<td>59.09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>2.4</td>
<td>3.6-3.1</td>
<td>15.2-13.6</td>
<td>89.2</td>
<td>5.9-4.3</td>
<td>69.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table VI: Test matches’ statistics - ad hoc teams - average values per match - noisy data.

Despite the importance of the team performance, we decided to analyze the performance of the ad hoc agent separately so we could draw some conclusions regarding its influence in the ad hoc team. We measured the ad hoc agent’s individual performance in terms of its passing frequency, passing accuracy, dribbling frequency, dribbling accuracy, shot frequency and shot accuracy.

Our agent’s passing accuracy was above average, rarely missing a pass. Its dribbling accuracy was also very good, as well as its goal scoring ability. Even considering the fact that the shot accuracy was lower than the team’s average, its frequency was high, which resulted in a lot of goals being scored, in some of the matches. With this being said, we can ascertain that the ad hoc agent was a positive influence in the team’s performance.

V. CONCLUSIONS

We can conclude by saying that the results we obtained matched our initial expectations. Our ad hoc agent became a productive member of the ad hoc team and not only did it not reduce the team’s performance but also managed to slightly increase it. Furthermore, we also reached some interesting conclusions regarding the importance of certain factors (such as shot frequency and passing accuracy) in the overall performance of a team in Robocup 2D Simulation League, as we demonstrated in Section V. With this work, we took a step in the right direction in the mission to solve, or at least mitigate, the scalability issue that limits ad hoc teamwork settings and also contributed with yet another analysis of Robocup 2D Simulation League matches.

Despite the favorable results obtained, our solution is not perfect. Although not often, sometimes our ad hoc agent still fails to identify the missing role and therefore tries to interpret a role that was not originally intended, as described in Section V. This is exacerbated by the fact that our formations’ library contains formations that are very similar to each other, which were included on purpose to test our solution as hard as possible.

Furthermore, due to the limitations imposed to us by the Robocup 2D Simulation League’s SoccerServer, namely the restrictions in the sensory data provided to the agent, we were unable to run our agent with online observations and, instead, used agent observations collected from previous matches. The agent is not able to obtain a full description of the game from the server. Instead it gets a limited description of the field and the other game components based on what direction it is looking to and its field of vision, which impacts our ability to collect accurate online observations.

With this work, we managed to lessen the scalability issues that had been associated with ad hoc teamwork settings, however this does not close the ad hoc teamwork topic. A possible future solution would be one that is able to surpass the difficulty in collecting observations and learning online, contributing to a more robust and flexible approach. It would also be interesting to apply it to other fields and bigger, even more complex domains.

REFERENCES


