To appear in *AAMAS Autonomous Robots and Multirobot Systems Workshop* (ARMS 2014), Paris, France, May 2014.

The RoboCup 2013 Drop-In Player Challenges: A Testbed for Ad Hoc Teamwork

Patrick MacAlpine, Katie Genter, Samuel Barrett, and Peter Stone

Department of Computer Science, The University of Texas at Austin
{patmac,katie,sbarrett,pstone}@cs.utexas.edu

Abstract. Ad hoc teamwork has recently been introduced as a general challenge for AI and especially multiagent systems [15]. The goal is to enable autonomous agents to band together with previously unknown teammates towards a common goal: collaboration without pre-coordination. While research to this point has focused mainly on theoretical treatments and empirical studies in relatively simple domains, the long-term vision has always been to enable robots or other autonomous agents to exhibit the sort of flexibility and adaptability on complex tasks that people do, for example when they play games of "pick-up" basketball or soccer. This paper chronicles the first evaluation of autonomous robots doing just that: playing pick-up soccer. Specifically, in June 2013, the authors helped organize a "drop-in player challenge" in three different leagues at the international RoboCup competition. In all cases, the agents were put on teams with no pre-coordination. This paper documents the structure of the challenge, describes some of the strategies used by participants, and analyzes the results.

1 Introduction

The increasing capabilities of robots and their decreasing costs is leading to increased numbers of robots acting in the world. As the number of robots grows, so will their need to cooperate with each other to accomplish shared tasks. Therefore, a significant amount of research has focused on multiagent teams. However, most existing techniques are inapplicable when the robots do not share a coordination protocol, a case that becomes more likely as the number of companies and research labs producing these robots grows. To deal with this variety of previously unseen teammates, robots can reason about *ad hoc teamwork* [15]. When participating as part of an ad hoc team, agents need to cooperate with previously unknown teammates in order to accomplish a shared goal. Reasoning about these settings allows robots to be robust to the teammates they may encounter.

In [15], Stone et al. argue that ad hoc teamwork is "ultimately an empirical challenge." Therefore, we helped organize a series of "drop-in player challenges" at the RoboCup 2013 competition,[1] a well established multi-robot competition. These challenges bring together real and simulated robots from teams

[1] http://www.robocup.org/

from around the world to investigate the current ability of robots to cooperate with a variety of unknown teammates.

In each game of the challenges, a number of robots were drawn from the participating teams and combined to form a new team. These robots were not informed of the identities of any of their teammates, but they were able to share a small amount of information using a limited standard communication protocol that was published in advance. These robots then had to quickly adapt to their teammates over the course of a single game and discover how to intelligently share the ball and select which roles to play. Teams from all around the world submitted robots for this challenge, without any attempt to pre-coordinate the robots that participated.

The rest of the paper is structured as follows. Descriptions of the ad hoc team problem and the different RoboCup domains used for this research are provided in Section 2. Section 3 explains the drop-in player challenges. Section 4 details some strategies used during the challenges, and the results are given in Section 5. Further analysis of the challenges is presented in Section 6. Section 7 situates this work in literature, and Section 8 concludes.

2 Problem Description

Robot soccer[1] has served as an excellent research domain for autonomous agents and multiagent systems over the past decade and a half. In this domain, teams of autonomous robots compete with each other in a complex, real-time, noisy and dynamic environment, in a setting that is both collaborative and adversarial. RoboCup includes several different leagues, each emphasizing different research challenges. For example, the humanoid robot league emphasizes hardware development and low-level skills, while the 2D simulation league emphasizes more high-level team strategy. In all cases, the agents are all fully autonomous.

2.1 Ad Hoc Teamwork

During the more than 15 years of annual RoboCup soccer competitions, participants have always created full teams of agents to compete against other teams. As a result, they have been able to build in complex, finely-tuned coordination protocols that allow them to agree upon which player should go to the ball, which player(s) should play defense, etc. To the best of our knowledge, there has never been any organized effort to evaluate the ability of a player to coordinate with players from other teams in an ad hoc teamwork setting.

While the majority of multiagent research in general focuses on creating coordinated teams that complete shared tasks, ad hoc teamwork research focuses on creating agents that can cooperate with unknown teammates without prior coordination. Rather than creating a whole team of agents that share a coordination protocol, we assume that each developer can only create a single agent or a small subset of agents on the team; the other agents are under the control of other developers. The objective of each agent is to be capable of adapting to

any teammates it may encounter while working on a shared task, in this case winning robot soccer games. In order to compare the abilities of different agents in the ad hoc teamwork setting, we use Algorithm 1, based on the evaluation framework proposed by Stone et al. [15].

2.2 RoboCup Standard Platform League (SPL)

In the Standard Platform League (SPL),[2] teams compete with identical Aldebaran Nao humanoid robots,[3] as shown in Figure 1. Since teams compete with identical hardware, the SPL is essentially a software competition.

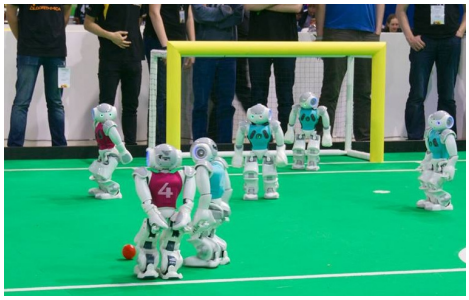

Fig. 1: UTAustinVilla's Aldebaran Nao robots (in pink) competing at RoboCup 2013.

In the SPL, games are played with 5 robots per side on a 6m by 9m carpeted surface and last for two, 10 minute halves. Robots play completely autonomously and are able to communicate with each other via a wireless network. As can be seen in Figure 1, the objects used in the games are color-coded, such as the ball being orange. In the SPL, robots may be penalized for various behaviors, such as pushing opponents, entering their own penalty box (unless they are goalie), and attempting to leave the field.

2.3 RoboCup 2D Simulation League

As one of the oldest RoboCup leagues, 2D simulation soccer has been well explored, both in competition and in research. The domain consists of autonomous agents playing soccer against each other on a simulated 2D soccer field. The agents communicate with a central server that controls the game. The agents receive abstract sensory information about the game, including the position of the ball and other agents, from the central server. After processing this information, the agents then communicate to the server what abstract actions they want to take in the world such as dashing, kicking, and turning. 2D soccer abstracts away many of the low-level behaviors required for humanoid robot soccer, including walking and computer vision, and thus affords the chance to focus on higher-level aspects of playing soccer such as multiagent coordination and strategic play.

[2] http://www.tzi.de/spl/
[3] http://www.aldebaran.com

Games consist of 11 versus 11 agents playing two 5 minute halves (6,000 simulation steps each lasting 100 ms). Additionally, each team is allowed a coach agent which observes the action on the field and is periodically allowed to broadcast messages to members of its team. Figure 2 shows the soccer field during a game.

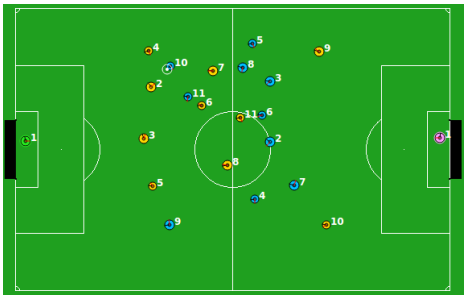

Fig. 2: A screenshot of a 2D soccer simulation league game.

2.4 RoboCup 3D Simulation League

The RoboCup 3D simulation environment is based on SimSpark,[4] a generic physical multiagent systems simulator. SimSpark uses the Open Dynamics Engine[5] (ODE) library for its realistic simulation of rigid body dynamics with collision detection and friction. ODE also provides support for the modeling of advanced motorized hinge joints used in the humanoid agents.

The robot agents in the simulation are homogeneous and are modeled after the Aldebaran Nao robot.[3] The agents interact with the simulator by sending torque commands and receiving perceptual information. Each robot has 22 degrees of freedom, each equipped with a perceptor and an effector. Joint perceptors provide the agent with noise-free angular measurements every simulation cycle (20 ms), while joint effectors allow the agent to specify the torque and direction in which to move a joint. Although there is no intentional noise in actuation, there is slight actuation noise that results from approximations in the physics engine and the need to constrain computations to be performed in real-time. Abstract visual information about the environment is given to an agent every third simulation cycle (60 ms) through noisy measurements of the distance and angle to objects within a restricted vision cone (120°). Agents are also outfitted with noisy accelerometer and gyroscope perceptors, as well as force resistance perceptors on the sole of each foot. Additionally, agents can communicate with each other every other simulation cycle (40 ms) by sending 20 byte messages.

Games consist of two 5 minute halves of 11 versus 11 agents on a field size of 20 meters in width by 30 meters in length. Figure 3 shows a visualization of the simulated robot and the soccer field during a game.

[4] `http://simspark.sourceforge.net/`
[5] `http://www.ode.org/`

Fig. 3: A screenshot of the Nao-based humanoid robot (left), and a view of the soccer field during a 11 versus 11 game (right).

3 Challenge Descriptions

At RoboCup 2013, the authors led the organization of a series of "drop-in player challenges" in the SPL, 2D simulation, and 3D simulation leagues. In this section we describe the set-up and rules of the challenges held in each of the three leagues.

3.1 SPL Challenge

The SPL challenge required teams to commit to participating in the challenge approximately four weeks prior to the competition. This decision was made for two main reasons: (1) to ensure that enough teams would participate to make the challenge worthwhile and (2) to encourage teams to work on their ad hoc agent behaviors (on their own) before arriving at the competition. Though nothing prevented teams from coordinating with one another prior to the competition, doing so was clearly against the spirit of the rules, and we have no reason to believe that anyone did so.

Six teams participated in the challenge. Since games consist of ten robots, each team contributed one or two drop-in players for each game. In games in which a team contributed two drop-in players, these players always played on the same team. Participating teams were aware of the possibility of having a teammate of their own design to collaborate with.

Each SPL drop-in game was 5 minutes, a quarter the length of a normal SPL game. This length was chosen to allow for more games with different teams to be played during the limited time allotted for the challenge. For each drop-in game, both teams were composed of randomly selected drop-in players. Each team participating in the drop-in challenge participated in four drop-in games. Two teams contributed 6 drop in players over the four games while four teams contributed 7 drop-in players over the four games.

All of the normal SPL rules[6] applied during the challenge except that instead of having a predefined goalie (usually in the SPL, the goalie is the player with "1" on its jersey), any player was able to claim the goalie position. When a defensive player first entered the penalty box in a game, that player became the

[6] http://www.tzi.de/spl/pub/Website/Downloads/Rules2013.pdf

goalie for the remainder of the game and any additional defensive players to enter the penalty box were penalized as "Illegal Defenders."

In the SPL drop-in challenge, each drop-in player was able to communicate with its teammates using a simple protocol, but players were not required to utilize this protocol. Although most of the teams chose to use the protocol, at least one team did not use it. This protocol allows for communication of the player's position on the field, the variance (uncertainty) of this position, the ball's position, position variance, velocity, and the time since this robot has seen the ball, as well as boolean variables representing whether the robot believes it has fallen or has been penalized. Note that any of this information may be incorrect if the robot is mis-localized or has mis-identified the ball.

The SPL challenge was scored using two metrics: average goal difference (AGD) and average human-judged score from three judges. The human judges were used in the SPL challenge to help identify good teamwork abilities in agents to ameliorate the problems of random variance in the games and the quality of teammates affecting the goal differences given the small number of games played. To reduce any bias in the judges, teams were asked to remove or cover any distinguishing markers on their robots. In addition, the judges were recruited from teams not participating in the challenge, and care was taken to select judges that were not as familiar with the walking gaits and other low level skills that could identify particular teams.

For each game, each judge was asked to award each drop-in player a teamwork score ranging from 0 to 10, where 10 means the drop-in player was an excellent teammate and 0 means that the drop-in player ignored its teammates. The judges were instructed to focus on teamwork capabilities, rather than individual skills, such that a less-skilled robot could still be given top marks for good teamwork.

The two scoring metrics were normalized to determine the overall winner of the SPL challenge. The normalization factor was computed to be

$$\frac{10}{\max \text{AGD of a drop-in player}}$$

Each drop-in player's average goal difference was then multiplied by this normalization factor, to get each drop-in player's normalized average goal difference. The normalized average goal difference was then added to the average human-judged score for each drop-in player to obtain its overall score.

The complete rules for the SPL Drop-In Player Challenge at RoboCup 2013 can be found on the SPL webpage.[7]

3.2 2D Challenge

For the 2D drop-in player challenge[8] each team contributed two drop-in field players to a game. Each drop-in player competed in 10 minute 7 on 7 games

[7] `http://www.tzi.de/spl/pub/Website/Downloads/Challenges2013.pdf`

[8] Full rules of the challenge can be found at `http://www.cs.utexas.edu/~AustinVilla/sim/2dsimulation/2013_dropin_challenge/2D_DropInPlayerChallenge.pdf`

(two 5 minute halves) with both teammates and opponents consisting of other drop-in field players. The decision to play 7 on 7 games, as opposed to full 11 on 11 games, was due to the majority of teams in the league using the same base code release (agent2d [1]). This fully functional code release provides default formations that if used in 11 on 11 games by all members of a team would provide implicit coordination. Each team was also assigned a common goalie agent (the latest version of agent2d). No coach agents were used during the challenge. It was suggested that drop-in players use the default agent2d communication protocol in order to communicate. However, drop-in players were not required to utilize this protocol — the use of this or any communication protocol was purely optional.

All normal game rules applied in this challenge. Each player was randomly assigned a uniform number from 2-11 at the start of a game. Players were scored exclusively by their average goal difference across all games they played in — there is no human judging in the simulation leagues. Given that average goal differential is the only metric used for performance, it was required that every team play at least one game against opponents from every other team during the challenge to ensure the possibility of each team having a different goal differential.

A total of nine teams participated in the challenge with six teams playing in five drop-in games and three teams playing in four drop-in games. Game pairings were chosen by a greedy algorithm given in Algorithm 1 that attempts to even out the number of times agents from different teams play with and against each other. In lines 6 and 7 of the algorithm agents are iteratively added to teams by `getNextAgent()` which uses the following ordered preferences to select agents that have:

1. Played fewer games.
2. Played against fewer of the opponents.
3. Played with fewer of the teammates.
4. Played a lower maximum number of games against any one opponent or with any one teammate.
5. Played a lower maximum number of games against any one opponent.
6. Played a lower maximum number of games with any one teammate.
7. Random.

Algorithm 1 terminates when all agents from teams have played at least one game against opponents from every other team.

Algorithm 1 Drop-In Team Pairings

Input: *Agents*

1: $games = \varnothing$
2: **while not** `allAgentsHavePlayedEachOther()` **do**
3: $\quad team1 := \varnothing$
4: $\quad team2 := \varnothing$
5: $\quad$ **for** $i := 1$ **to** $AGENTS\_PER\_TEAM$ **do**
6: $\quad\quad team1 \leftarrow$ `getNextAgent`$(Agents \setminus \{team1 \cup team2\})$
7: $\quad\quad team2 \leftarrow$ `getNextAgent`$(Agents \setminus \{team1 \cup team2\})$
8: $\quad games \leftarrow \{team1, team2\}$
9: **return** *games*

3.3 3D Challenge

For the 3D drop-in player challenge[9] each participating team contributed two drop-in field players to a game. Each drop-in player competed in full 10 minute games (two 5 minute halves) with both teammates and opponents consisting of other drop-in field players. No goalies were used during the challenge to increase the probability of goals being scored.

Each drop-in player could communicate with its teammates using a simple protocol, containing similar information as used in the SPL challenge — the use of the protocol was purely optional. A C++ implementation of the protocol was provided to all teams.

All normal game rules applied in this challenge. Each player was randomly assigned a uniform number from 2-11 at the start of a game. The challenge was scored by the average goal difference received by an agent across all games that an agent plays in.

A total of four drop-in games were played during the challenge with each of the ten teams participating in the challenge playing in every game. Team pairings for the games were determined by Algorithm 1 and as such all drop-in players played in at least one game against every other drop-in player in the challenge.

4 Drop-In Player Strategies

In addition to leading the organization of the RoboCup drop-in challenges, UTAustinVilla participated in all three of them as well. This section presents how UTAustinVilla prepared to participate in the challenges.

4.1 SPL Strategies

In the main Standard Platform League competition, UTAustinVilla uses a team strategy in which it assigns players to roles based on their positions on the field. Each player calculates a bid to play as chaser (player who goes to the ball) based on its distance from the ball, its angle to the ball, and whether chasing the ball would require going downfield or upfield. This bid and the robot's location are communicated whenever the robot sends messages. The robot with the highest bid becomes chaser and the remaining players are assigned to the remaining roles, such as defender, forward, or midfielder. Each of these roles has a defined location relative to the position of the ball bounded by some maximal locations and affected by how the robot can position in order to prevent shots at its own goal. These roles are then assigned based on the priority of the role as well as how close the robot is to the desired position for the role. The positions for each

[9] Full rules of the challenge can be found at `http://www.cs.utexas.edu/~AustinVilla/sim/3dsimulation/2013_dropin_challenge/3D_DropInPlayerChallenge.pdf`

role are specified via configuration files, which allows for fast changes to be made during the competition.

For the drop-in player challenge, the robots from UTAustinVilla try to follow a similar strategy. The robots assume that the information communicated by its teammates is correct, and that if a teammate does not communicate, it is either penalized or otherwise removed from play. To identify the chaser, the robot estimates the bids of its teammates based on the positions that they communicate. The player that has the highest estimated bid is assumed to be the chaser. The remaining players are assumed to assign themselves to roles based on their locations similarly to above. Of course there is no guarantee that the other players will behave based on the same assumptions. An improvement for future competitions would be to further estimate what roles the robots appear to be playing. This improvement would reduce problems in which multiple players attempt to play the same roles or fail to fill an important role. However, this estimation problem can be quite difficult, and the current approach worked well in initial testing.

In addition to assigning roles to players, the robots also reasoned about how to pass the ball and where to move in order to receive passes. Whenever a robot was in possession of the ball, it considered a variety of kicks as described further in [4]. When selecting a kick, the robot will prefer kicks that will allow its teammates to receive the ball. In addition, several of the roles are positioned in order to allow the robot to receive passes and move the ball upfield. Furthermore, during the team's kickoff, the robot considers a variety of set plays and selects the one that it believes will best help the team. The robot prefers set plays that pass the ball to its teammates when its teammates report that they are in opportune positions. If the other players do not cooperate by moving to receiving positions, the robot executes set plays that do not require their help, such as burying the ball deep near the opponents' goal.

4.2 2D Strategies

For the 2D competition UTAustinVilla builds on the `agent2d` [1] base code release which provides a fully functional soccer playing agent team. This code release includes basic skill macros for dribbling, passing, and guarding opponents. Additionally, `agent2d` includes default formation files to specify different role positions for agents to assume on the field. Positions for agents are computed from sets of fixed ball positions and agent home positions using Delaunay triangulation [2]. In the `agent2d` base code, agents are statically assigned to role positions on the field based on their uniform numbers. We modified the code base to use our own dynamic role assignment system [12] which attempts to minimize the makespan (time for all agents to reach target home positions) while also preventing collisions among agents.

The same binary used in the main 2D competition was also used during the drop-in player challenge. An important aspect of the drop-in challenge is for an agent to be able to adapt to the behaviors of its teammates: for instance if most of an agent's teammates are assuming offensive roles, that agent might better

serve the team by taking on a defensive role. UTAustinVilla's dynamic role assignment system implicitly allows for this adaptation to occur as it naturally chooses roles that do not currently have other agents nearby.

4.3 3D Strategies

For the drop-in challenge the UTAustinVilla agent goes to the ball if it is closest to it and attempts to kick or dribble the ball toward the opponent's goal. If our agent is not closest to the ball, it waits at a position two meters behind the ball as this was found in [12] to be an important supporting position. Since two agents of ours are always on the same team during the challenge, and often neither one of them is the player closest to the ball, they are often both trying to move to the same target position two meters behind the ball. In order to prevent collisions between the agents, they employ collision avoidance as described in [12].

The UTAustinVilla agent monitors communications from its teammates about the positions of themselves and the ball, and uses it to maintain a model of the world. Agents must localize themselves as they are not directly given their current locations in the world by the server. As not all agents are that adept at self-localization, our agent computes whether it thinks that another teammate's observations are accurate enough to be trusted based on its own observations. This assessment is done over time by recording whether or not communicated information from individual teammates falls within a reasonable accuracy threshold. Should the percentage of information deemed to be accurate from a particular teammate fall below a set level, our agent will no longer take information from that agent into account when building a model of the world. Unfortunately, as no other teams used the shared communication protocol for this challenge, this feature was never tested.

During kickoffs agents are allowed to teleport to anywhere on their own side of the field before play begins. Kickoffs allow for a team to kick the ball into the opponent's side of the field and, as reported in [13], can provide a substantial gain in performance for long kicks that move the ball deep into the opponent's side of the field. Before our own kickoff during the challenge our agents teleport to random positions on the field. After all teammates have teleported onto the field they then check if a teammate is right next to the ball. If no teammate is near the ball they then teleport next to the ball in order to take the kickoff. As we have two of our own agents on a team, and we don't want them both to try to teleport next to the ball to take the kickoff, the agent with the lower uniform number is the only one of the two allowed to teleport next to the ball to take the kickoff.

5 Challenge Results

This section presents the results of the SPL, 2D, and 3D drop-in player challenges held during the RoboCup 2013 competition.

5.1 SPL Results

In the SPL drop-in player challenge, six teams participated in four 5-minute games. As discussed in Section 3.1, the overall winner of this challenge was determined via two metrics: average goal difference and average human-judged score. The participating teams were assigned randomly to drop-in teams for this challenge; these teams, the game scores, and the judge-assigned scores can be seen in Table 1.

	Blue Team	**Red Team**
Game 1 1 – 0	1 B-Human: 7,7,6 2 rUNSWift: 8,7,7 3 UPennalizers: 5,6,0 4 rUNSWift: 5,5,7 5 UPennalizers: 3,4,8	1 UTAustinVilla: 4,5,4 2 Nao Devils: 5,9,7 3 UTAustinVilla: 6,6,7 4 Nao Devils: 5,7,7 5 Berlin United: 2,4,4
Game 2 2 – 0	1 Nao Devils: 7,7,6 2 Nao Devils: 7,6,7 3 UTAustinVilla: 6,5,7 4 UTAustinVilla: 6,8,8 5 B-Human: 6,7,9	1 UPennalizers: 1,6,0 2 UPennalizers: 4,7,5 3 Berlin United: 1,5,0 4 rUNSWift: 6,6,0 5 Berlin United: 4,5,2
Game 3 0 – 2	1 Berlin United: 5,6,0 2 UTAustinVilla: 7,5,6 3 UTAustinVilla: 6,5,8 4 UPennalizers: 6,2,6 5 Berlin United: 5,4,2	1 B-Human: 7,10,10 2 rUNSWift: 6,4,5 3 rUNSWift: 6,6,0 4 Nao Devils: 3,7,1 5 B-Human: 6,8,5
Game 4 0 – 0	1 B-Human: 3,8,5 2 Berlin United: 7,4,1 3 Berlin United: 1,3,6 4 UTAustinVilla: 6,6,5 5 B-Human: 6,5,5	1 Nao Devils: 8,9,6 2 Nao Devils: 7,6,4 3 rUNSWift: 5,3,8 4 UPennalizers: 5,4,8 5 UPennalizers: 7,7,0

Table 1: The teams, game scores, and judge scores for SPL.

Using these results and the team configurations, we can calculate the average goal differential for each team's drop-in players, where a player is credited with a goal if it was on the team that scored that goal. The combinations of the scores and rankings can be seen in Table 2.

Team	**Avg Goal Diff**	**Norm Goal Diff**	**Avg Judge Score**	**Final Score**	**Rank (Goal, Judge)**
B-Human	1.17	10.00	6.67	16.67	1 (1,1)
Nao Devils	0.57	4.90	6.24	11.14	2 (3,2)
rUNSWift	0.67	5.71	5.22	10.94	3 (2,4)
UTAustinVilla	-0.29	-2.45	6.00	3.55	4 (4,3)
UPennalizers	-0.57	-4.90	4.48	-0.42	5 (5,5)
Berlin United	-1.29	-11.02	3.38	-7.64	6 (6,6)

Table 2: The final scores and rankings for the SPL drop-in challenge.

5.2 2D Simulation Results

Nine teams participated in the 2D drop-in player challenge with seven games played in total. Each team contributed two players for each 7 on 7 game with each team also being given a canonical goalie agent.

As game results in the 2D simulation domain tend to be very noisy with high variance, only playing seven games does not allow for results to be statistically significant. Therefore, following the competition, we also replayed the challenge with the released binaries over many games including all combinations of the nine teams contributing two agents each. The number of combinations is $(\binom{9}{3} * \binom{6}{3})/2 = 840$ given that each drop-in team is made up of agents from three different teams. We played five games for each drop-in team grouping for a total of 4,200 games.

	RoboCup		Many Games	
Team	**AGD**	**Rank**	**AGD**	**Rank**
FCPerspolis	2.40	1	3.025 (0.142)	1
Yushan	2.25	2	2.583 (0.141)	2
ITAndroids	2.00	3	1.379 (0.152)	5
Axiom	1.20	4	1.315 (0.148)	6
UTAustinVilla	0.25	5	1.659 (0.153)	4
HfutEngine	-0.20	6	-2.076 (0.152)	7
WrightEagle	-1.60	7	-6.218 (0.129)	9
FCPortugal	-2.20	8	-3.379 (0.150)	8
AUTMasterminds	-2.80	9	1.711 (0.152)	3

Table 3: Average goal difference (AGD) with standard error shown in parentheses and rankings for the 2D drop-in player challenges.

Overall results of both the challenge held at RoboCup, and the more statistically significant challenge we ran afterward with many games played, are presented in Table 3. Looking at the results it is clear that only playing seven games does not reveal the true rankings of teams as the last place team during the challenge at RoboCup, AUTMasterminds, finished third overall when playing thousands of games. We do however acknowledge that as we are not running games on the same exact machines as were used at RoboCup, there is the potential for agents to behave differently in these tests.

The last place agent after playing many games, WrightEagle, often showed strange behavior where the agent would just stand still and no longer kick or go after the ball. When teammates passed the ball to a WrightEagle agent the other team was able to easily win possession of the ball. In comparison the other two agents with negative but significantly better average goal differences, FCPortugal and HfutEngine, often crashed and disconnected from the server forcing their teammates to play down an agent. Just leaving the game gave better performance than standing around as a decoy for teammates to pass to and lose the ball. This phenomenon illustrates a need for drop-in players to be able to evaluate and adapt to the skills shown by their teammates (learn not to pass the ball to a frozen teammate).

5.3 3D Simulation Results

There were ten teams that participated in the 3D drop-in player challenge consisting of four games played in total. Each team contributed 2 players for each 10 on 10 game (no goalies were used). Game results in the 3D simulation domain

also tend to have high variance, so playing only four games does not achieve results that are statistically significant. Therefore, we once again replayed the challenge using the released binaries using all combinations of teams. The total number of possible drop-in team combinations is $(\binom{10}{5} * \binom{5}{5})/2 = 126$ as each drop-in team is made of agents from five different teams. Each combination was played five times, resulting in a total of 630 games. Overall results of both the challenge held at RoboCup, and the more statistically significant challenge we ran afterwords with many games played, are presented in Table 4.

	RoboCup		Many Games	
Team	**AGD**	**Rank**	**AGD**	**Rank**
BoldHearts	1.50	1	0.178 (0.068)	4
FCPortugal	0.75	T2	1.159 (0.060)	1
Bahia3D	0.75	T2	-0.378 (0.068)	7
Apollo3D	0.75	T2	0.159 (0.068)	5
magmaOffenburg	0.25	5	0.254 (0.068)	3
RoboCanes	-0.50	6	-0.286 (0.068)	6
UTAustinVilla	-0.75	T7	0.784 (0.065)	2
SEUJolly	-0.75	T7	-0.613 (0.066)	9
Photon	-0.75	T7	-0.425 (0.068)	8
L3MSIM	-1.25	10	-0.832 (0.065)	10

Table 4: Average goal difference (AGD) with standard error shown in parentheses and rankings for the 3D drop-in player challenges.

6 Challenge Analysis

This section presents further analysis of the results of the SPL, 2D, and 3D drop-in player challenges held during the RoboCup 2013 competition.

6.1 SPL Analysis

The winners of the drop-in challenges should be the players who displayed the best teamwork abilities. These players may or may not have the best low level skills. Especially in the SPL, where the six teams contributing drop-in players utilized at least four different gaits, the fear was that the team with the best low level skills might dominate the drop-in player challenge if only goal differential were used to determine the winner. Hence, three human judges were asked to judge each drop-in player's teamwork abilities. The question to be considered after the competition for the SPL was: did the drop-in player with the best teamwork abilities win or did better low level skills cause a team with inferior teamwork abilities to win?

Table 5 shows the main RoboCup competition rankings, records, and goals scored for each SPL team that participated in the drop-in challenge. The top four finishers in the challenge all claimed to implement the drop-in communication protocol in personal communications with the authors after the competition, and it is our understanding that all four teams ran a different strategy in the drop-in player competition than in their main games at RoboCup 2013.

Team	Drop-In Rank	Main Rank	Main Win/Tie/ Loss	Main Goals
B-Human	1	1	8/0/0	62
Nao Devils	2	T5	4/1/2	18
rUNSWift	3	4	5/0/4	18
UTAustinVilla	4	3	6/0/2	30
UPennalizers	5	17	4/0/2*	17*
Berlin United	6	T9	1/1/3	3

Table 5: The final rankings in both the drop-in challenge and the main competition at RoboCup 2013 for the SPL drop-in participants. 22 teams participated in the SPL at RoboCup 2013. (*3 wins and 12 goals of the UPennalizers were in the Consolation Cup between the six teams that were eliminated after the first round of pool play.)

UPennalizers and Berlin United finished near the bottom in the drop-in challenge, and they also finished in the lower part of the main competition. Perhaps weaker low level skills, vision algorithms, or localization algorithms caused them to perform poorly in both competitions. However, it could also be that their teams did not spend as much effort on their drop-in player challenge strategies as more work was required for their team in the main competition. Notably, B-Human performed best in terms of their drop-in and main ranks as well as in the human judges' scores, indicating that their teamwork and adaptability performed well in both the main competition as well as in the drop-in challenge.

6.2 2D Analysis

To further analyze the results from the 2D drop-in player challenge we again compare teams' performances in the drop-in player challenge to their performance in the main competition. As the main tournament only includes a relatively small number of games, and thus rankings from the main competition are typically not statistically significant, we ran 1000 games of our team's main competition binary against each of the other teams' released main competition binaries who participated in the drop-in player challenge. This process gives us a statistically significant baseline for comparing the performance of teams on the task of playing standard team soccer. Results are shown in Table 6, using the drop-in rankings from the larger analysis.

While Table 6 shows that there is not a direct correlation between ranking in the drop-in player challenge compared to standard team soccer, there is a trend for agents that perform better at standard team soccer to also perform better at the drop-in player challenge. Excluding the outlier team WrightEagle which clearly showed a bug during the drop-in player challenge, the top half of the teams for the drop-in player challenge had an average rank of 4.25 when playing against UTAustinVilla while the bottom half of the teams for the drop-in player challenge had an average rank of 6.75 against UTAustinVilla.

An important aspect of the drop-in player challenge is for agents to adapt to the behaviors of their teammates. For example, if most of an agent's teammates are assuming offensive roles, that agent might better serve the team by taking on a defensive role. As mentioned in Section 4.2, UTAustinVilla's dynamic role

	Drop-In	Main	Against UTAustinVilla	
Team	**Rank**	**Rank**	**Rank**	**AGD**
FCPerspolis	1	5	4	3.127 (0.059)
Yushan	2	2	3	4.034 (0.065)
AUTMasterminds	3	4	2	5.111 (0.117)
UTAustinVilla	4	8	8	0.000 (self)
ITAndroids	5	7	7	0.505 (0.063)
Axiom	6	3	5	1.803 (0.074)
HfutEngine	7	9	9	-6.027 (0.184)
FCPortugal	8	6	6*	*
WrightEagle	9	1	1	6.176 (0.287)

Table 6: Average goal difference (AGD) with standard error shown in parentheses and rankings for the 2D drop-in player challenge, main competition, and playing against UTAustinVilla. *We were unable to run the FCPortual released binary from the main competition and thus approximated their rank against UTAustinVilla to be the same as the main competition.

assignment system [12] implicitly encourages this adaptation. We empirically found that most of the released agent binaries for the challenge used static role assignment thus underscoring the need to adapt to teammates' fixed roles (it is unlikely that teammates will adapt to roles assumed by you). We tested a version of the UTAustinVilla agent not using dynamic role assignment, but instead the default static role assignment included with `agent2d`, across the same 4,200 game extended drop-in player challenge used in Section 5.2. Not using our dynamic role assignment system hurt our agent's performance: its average goal difference dropped from 1.659 (+/-0.153) down to 1.473 (+/-0.157) when using static role assignments instead. This result shows promise for the dynamic role assignment system as not only a way to coordinate motion among one's own teammates, but also as a viable method for adapting to the behavior of unknown teammates in an ad hoc team.

6.3 3D Analysis

When analyzing results from the 3D drop-in player challenge we follow the same methodology used in the 2D analysis for comparing teams' performances in the drop-in player challenge to their performance in the main competition. To do so we ran at least 100 games of our team's main competition binary against each of the other teams' released main competition binaries who participated in the drop-in player challenge. Results for this are shown in Table 7, using the drop-in rankings from the larger analysis.

Similar to the results seen for the 2D simulation league in Table 6, Table 7 shows that there is not a strong correlation between rankings in the drop-in player challenge compared to standard team soccer. However, there is a trend that teams that perform better at drop-in player soccer also performing better at standard team soccer. The top half of the teams for the drop-in player challenge had an average rank of 3.4 when playing against UTAustinVilla, while the bottom half of the teams for the drop-in player challenge had an average rank of 7.6 against UTAustinVilla.

	Drop-In	Main	Against UTAustinVilla	
Team	**Rank**	**Rank**	**Rank**	**AGD**
FCPortugal	1	3	2	-0.465 (0.023)
UTAustinVilla	2	2	1	0.000 (self)
magmaOffenburg	3	T5	5	-1.447 (0.026)
BoldHearts	4	T5	6	-1.607 (0.029)
Apollo3D	5	1	3	-0.698 (0.027)
RoboCanes	6	T5	7	-1.828 (0.031)
Bahia3D	7	10	10	-9.800 (0.110)
Photon	8	8	8	-4.590 (0.081)
SEUJolly	9	4	4	-1.133 (0.027)
L3MSIM	10	9	9	-6.050 (0.098)

Table 7: Average goal difference (AGD) with standard error shown in parentheses and rankings for the 3D drop-in player challenge, main competition, and playing against UTAustinVilla.

To evaluate parts of the UTAustinVilla drop-in player strategy detailed in Section 4.3, we created the following variants of the UTAustinVilla drop-in player agent and then tested them across the same 630 game extended drop-in player challenge used in Section 5.3.

Dribble Agent only dribbles and never kicks.
DynamicRoles Uses dynamic role assignment.
NoKickoff No teleporting next to ball to take the kickoff.

Agent	**AGD**
Dribble	1.370 (0.064)
UTAustinVilla	0.784 (0.065)
NoKickoff	0.676 (0.065)
DynamicRoles	0.568 (0.071)

Table 8: Average goal difference (AGD) with standard error shown in parentheses for agents playing in the extended 3D drop-in player challenge.

The results in Table 8 show that the strategy of teleporting in to take the kickoff if no teammates are in place to do so improves performance. The results also reveal that by only dribbling, and not kicking, we would greatly improve our agent's performance in the drop-in player challenge. Considering that the UTAustinVilla 3D simulation team won both the 2011 [14] and 2012 [13] RoboCup world championships, and took second place at RoboCup 2013, it can be surmised that the UTAustinVilla agent likely has better low-level skills than most other agents on its drop-in player team. Therefore kicking the ball, and potentially passing it to a teammate, has a reasonable likelihood of hurting overall team performance.

Interestingly we see that using dynamic role assignment, instead of going near the ball in a support role, hurts performance. This result is in contrast to the results in the 2D simulation league where dynamic role assignment improved performance. Once again this result may be due to our agent performing better with the ball on average than its drop-in teammates, and thus should aggressively pursue the ball.

7 Related Work

Multiagent teamwork is a well studied topic, with most work tackling the problem of creating standards for coordinating and communicating. One such algorithm is STEAM [16], in which team members build up a partial hierarchy of joint actions and monitor the progress of their plans. STEAM is designed to communicate selectively, reducing the amount of communication required to coordinate the team. In [8], Grosz and Kraus present a reformulation of the SharedPlans, in which agents communicate their intents and beliefs and use this information to reason about how to coordinate joint actions. In addition, SharedPlans provides a process for revising agents' intents and beliefs to adapt to changing conditions. In the TAEMS framework [9], the focus is on how the task environment affects agents and their interactions with one another. Specifically, agents reason about what information is available for updating their mental state. While these algorithms have been shown to be effective, they require that the teammates share their coordination framework.

On the other hand, ad hoc teamwork focuses on the case where the agents do not share a coordination algorithm. In [11], Liemhetcharat and Veloso reason about selecting agents to form ad hoc teams. Barrett et al. [6] empirically evaluate an MCTS-based ad hoc team agent in the pursuit domain, and Barrett and Stone [5] analyze existing research on ad hoc teams and propose one way to categorize ad hoc teamwork problems. Other approaches include Jones et al.'s work [10] on ad hoc teams in a treasure hunt domain. A more theoretical approach is Wu et al.'s work [17] into ad hoc teams using stage games and biased adaptive play.

In the domain of robot soccer, Bowling and McCracken [7] measure the performance of a few ad hoc agents, where each ad hoc agent is given a playbook that differs from that of its teammates. In this domain, the teammates implicitly assign the ad hoc agent a role, and then react to it as they would any teammate. The ad hoc agent analyzes which plays work best over hundreds of games and predicts the roles that its teammates will play.

However, none of this previous research has evaluated their approaches with agents created by developers from around the world in a true ad hoc teamwork setting. The agents used in this work were created by a variety of teams from around the world, all trying to create intelligent ad hoc agents, but without any direct interaction with the other developers.

8 Conclusions

This paper describes the drop-in player challenges[10] run in three of the leagues at the 2013 RoboCup competition. These challenges serve as an exciting testbed for ad hoc teamwork, in which agents must adapt to a variety of new teammates

[10] Videos of the challenges can be found at `http://www.cs.utexas.edu/~AustinVilla/sim/3dsimulation/AustinVilla3DSimulationFiles/2013/html/dropin.html`

without pre-coordination. These challenges provided an opportunity to evaluate robots' abilities to cooperate with new teammates to accomplish goals in complex tasks. They also served to encourage the participating teams to reason about teamwork and what is actually necessary to coordinate a team.

The drop-in player challenges were announced to the competing teams fairly late in the year, so teams may not have prepared for them as much as they had desired to. In addition, it was difficult for teams to foresee how their robots would interact given that this was the first time the challenge was run. If the challenges are run again in future years, we expect that the players' strategies for dealing with previously unknown teammates will improve markedly.

In the past, starting a new SPL team has required a substantial investment in at least five Nao robots, lab space for at least a partial field, and a dedicated team of programmers. Although multiple teams have done partial code releases, including UTAustinVilla [3], starting a new team still requires a significant amount of work. Beginning at RoboCup 2014, the SPL will encourage teams, especially new teams, to attend RoboCup and compete in a sizable, 20-game drop-in player competition even if they are not competing in the main team competition. We expect that this development will encourage teams to share code and concentrate their efforts on developing skills for teamwork. We hope that the drop-in player challenge will continue in the simulated robot soccer leagues and will be introduced in additional leagues in the coming years.

Acknowledgments

We would like to thank Duncan ten Velthuis from the University of Amsterdam for helping to run the Standard Platform League challenge, Aijun Bai from the University of Science and Technology of China for running the 2D challenge, and Sander van Dijk from the University of Hertfordshire for helping to run the 3D challenge. This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by grants from the National Science Foundation (CNS-1330072, CNS-1305287) and ONR (21C184-01). Patrick MacAlpine is supported by a NDSEG fellowship.

References

1. Akiyama, H.: Agent2d base code (2010)
2. Akiyama, H., Noda, I.: Multi-agent positioning mechanism in the dynamic environment. In: RoboCup 2007: Robot Soccer World Cup XI. Springer (2008)
3. Barrett, S., Genter, K., He, Y., Hester, T., Khandelwal, P., Menashe, J., Stone, P.: The 2012 UT Austin Villa code release. In: RoboCup-2013: Robot Soccer World Cup XVII (2013)
4. Barrett, S., Genter, K., Hester, T., Quinlan, M., Stone, P.: Controlled kicking under uncertainty. In: The Fifth Workshop on Humanoid Soccer Robots at Humanoids 2010 (December 2010)
5. Barrett, S., Stone, P.: An analysis framework for ad hoc teamwork tasks. In: AAMAS '12 (June 2012)

6. Barrett, S., Stone, P., Kraus, S.: Empirical evaluation of ad hoc teamwork in the pursuit domain. In: AAMAS '11 (May 2011)
7. Bowling, M., McCracken, P.: Coordination and adaptation in impromptu teams. In: AAAI (2005)
8. Grosz, B., Kraus, S.: Collaborative plans for complex group actions. Artificial Intelligence 86, 269–368 (1996)
9. Horling, B., Lesser, V., Vincent, R., Wagner, T., Raja, A., Zhang, S., Decker, K., Garvey, A.: The TAEMS White Paper (January 1999), http://mas.cs.umass.edu/paper/182
10. Jones, E., Browning, B., Dias, M.B., Argall, B., Veloso, M.M., Stentz, A.T.: Dynamically formed heterogeneous robot teams performing tightly-coordinated tasks. In: ICRA. pp. 570 – 575 (May 2006)
11. Liemhetcharat, S., Veloso, M.: Modeling mutual capabilities in heterogeneous teams for role assignment. In: IROS '11 (2011)
12. MacAlpine, P., Barrera, F., Stone, P.: Positioning to win: A dynamic role assignment and formation positioning system. In: RoboCup-2012: Robot Soccer World Cup XVI. Springer Verlag, Berlin (2013)
13. MacAlpine, P., Collins, N., Lopez-Mobilia, A., Stone, P.: UT Austin Villa: RoboCup 2012 3D simulation league champion. In: RoboCup-2012: Robot Soccer World Cup XVI. Springer Verlag, Berlin (2013)
14. MacAlpine, P., Urieli, D., Barrett, S., Kalyanakrishnan, S., Barrera, F., Lopez-Mobilia, A., Ştiurcă, N., Vu, V., Stone, P.: UT Austin Villa 2011: A champion agent in the RoboCup 3D soccer simulation competition. In: Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS) (June 2012)
15. Stone, P., Kaminka, G.A., Kraus, S., Rosenschein, J.S.: Ad hoc autonomous agent teams: Collaboration without pre-coordination. In: AAAI '10 (July 2010)
16. Tambe, M.: Towards flexible teamwork. Journal of Artificial Intelligence Research 7, 81–124 (1997)
17. Wu, F., Zilberstein, S., Chen, X.: Online planning for ad hoc autonomous agent teams. In: IJCAI (2011)