TEAM-IT : Location-Based Gaming in Real and Virtual Environments

Spencer Frazier, Alex Newnan, Fotos Frangoudes,
Yu-Han Chang, Rajiv Maheswaran
University of Southern California
4676 Admiralty Way #1001
Marina Del Rey, CA 90292

Abstract

Location-based games are an emerging paradigm for training, simulation, entertainment, health and many other domains. In this paper, we consider the role of artificial agents in such games. We also examine how human teams perform when given the same game, played in both a real environment with mobile devices and also in a virtual environment that replicates the real environment. We perform the first direct comparison of real and virtual instantiations of the same location-based game. We show the similarities and differences in game play and then investigate how adding an advice-giving agent changes the experience.

Introduction

Location-based games are an emerging new genre in interactive digital entertainment. One motivating example is training and simulation of human teams in mission spaces such as disaster rescue. Another important domain is serious gaming addressing issues such as health where games are used to motivate exercise and activity. Games in virtual environments have certainly been able to capture the notion of geographical spaces but in the space of serious gaming or games made to understand human behavior, it is unclear how behavior in virtual environments maps to that in real environments (Williams 2010). Another open question is how we can use intelligent agents to either improve game play in the real world or facilitate other goals such as increased activity.

To investigate these questions, we created TEAM-IT , a location-based game that involves competition and cooperation between multiple human teams. The game is instantiated in both an iOS framework which can be played in the real world and also in Cosmopolis, a networked virtual environment. The game allows players to be either humans or software agents. All players have locations which are tracked and shared. In the real world version, human players are tracked via GPS location updates received through mobile devices whereas in Cosmopolis this is available from the environmental infrastructure.

TEAM-IT supports a variety of intelligent agents. Agents can be part of a team or have their own teams. They have the ability to negotiate and trade for cards with human players, however, we will not focus on this aspect of the game in this paper. Here, we focus on an agent's ability to give advice to the human and how this can be used to affect game play.

In this paper, we address two main issues: (1) the mapping of player behavior between a real environment and a virtual environment for the same geospatial area in a location-based game and (2) investigating the impact of advice-giving by an intelligent agent in a real-world instantiation of a location-based game. To achieve the first, we implemented a 3D model of the geospatial area of the real-world experiments into Cosmopolis and ran experiments in both environments, a non-trivial technical challenge. To address the second, we created agents that give players suggestions on task collections to pursue in real-time based on the current location of team members and investigated how it affected game play.

We ran a pilot experiment exploring both of these issues. The results demonstrate the mapping issue, showing how human behavior in the virtual game environment differs markedly from behavior exhibited in the real-world iPad-based version of the game. These differences illustrate the challenges in building virtual training simulations that promote the same behavior as in the real world, and the difficulties surrounding research that examines virtual behavior and uses it to draw conclusions about real world behavior. The results also show the benefit of adding advice-giving AI agents. The agents help the players to score more points while traveling larger distances. The players also enjoy the game more when accompanied by the advice-giving agent.

Related Work

Attempts have been made to create a system capable of running mobile, quasi-location-based games concerned with agent modeling. In one such game (Chang et al. 2011), a lightweight framework has been constructed using mobile phones and adherence to design principles which allow for efficient development on mobile platforms (Tan and Kinshuk 2009). The design principles include multi-platform adaption, minimal resource usage, mimimal human/device interaction, reduced data communication bandwidth usage, and no additional hardware. Environments like Colored Trails (Grosz et al. 2004) offer great insight into human-agent interaction by providing an interface to study (van Wissen, van Diggelen, and Dignum 2009) negotiation and

various behavioral models. It can be argued that robust game environments which allow accurate representation of agent interaction in real-world scenarios are still emerging. Where multi-agent systems will be operating in the real-world, it seems important to create a system which directly translates to such an environment.

Human-agent collectives have been useful for modeling expected outcomes in scenarios such as disaster relief (Jennings, Rogers, and Moreau 2011). Mixed reality games may be a powerful contribution to research by leveraging real-world environmental factors to extract more accurate behavioral models from events observed in game (Fischer et al. 2012). While certain environments like GameBots (Adobbati et al. 2001) have been created to simulate human-agent and multi-agent scenarios, in situations where actions are more important than emotional responses, the 3D environment can serve to complicate rather than complement research. Real-world environments may actually provide stronger emotional responses and suspension of disbelief in game scenarios. Other virtual environments (Ishida and Hattori 2009) seek to mimic human involvement by virtualizing data collected from real-world actors. Our approach takes the conveniences of virtualization and applies them to a real-world, spatial solution.

The accuracy and scalability (Hajarnis et al. 2011) limitations of mixed-reality and alternate-reality games are ameliorated here due to the campus network. The inaccuracy of GPS data (Chang et al. 2011) can be overcome by enlarging the game space, executing trials in an ideal outdoor environment and utilizing robust wireless and cellular data networks where possible. In terms of scalability, we recognize the value of being able to load agents asynchronously into the game environment and as such, have made this action possible in our game scenarios. Wearable (Kleiner, Behrens, and Kenn 2006) and physical (Lund and Pagliarini 2011) solutions for multi-agent, human-interactive games provide robust data collection but may prove difficult to configure both on the interface side as well as the server side. TEAM-IT seeks to be configurable in any environment with any number of agents without the need for prolonged training periods while still maintaining comprehensive data tracking.

Game Scenarios

In TEAM-IT , each player belongs to a team. The player can be human or a software agent. Each player is assigned a number of skills. In our current implementation, skills are represented by a shape-color pair. Players can move around the world and discover and complete tasks which give points to their team. Endgame scenarios can be specified to depend on point thresholds or any other rules based on the game state. For this paper, we use a game-timer which fixes the duration of the game where the team with the most points in that duration wins.

In the geospatial environment, task collections or "boxes" are placed at various locations. A player can view these boxes if they are within a discovery radius of the location. This discovery radius is modifiable for each game instance. When a player is within a (typically smaller) application radius, they can open the box to see the task collection there.

Figure 1: TEAM-IT iPad Map View (left) which shows player team, skills, points, time remaining, locations of various boxes and their accessibility and locations of players. (Right) Five players using iPads to play the game.

Each task in a task collection is associated with a set of skills that must be applied simultaneously for a given duration in order for the task to be completed. When completed, points are awarded to teams as specified by the task. It is possible for a task to give points to multiple teams for completion at different values but for this instantiation of the game, we do not leverage this feature. The latter is useful to investigate environments where one would want to investigate simultaneous cooperation and competition between teams.

To apply a skill to a task, a player must move to within the application radius and use the required skill. If they move away from the application radius before the required duration for the task, the task is reset to being incomplete. Often it may require skills from multiple players to complete a task. Thus, teams must coordinate to have players in the right places at the right times to get points.

iPad Implementation

Each player in the game plays through an iOS application on their device. The number of players is limited by the number of devices available to the scenario organizers. Once a device ID is mapped to a player ID, it is set for the duration of the scenario but can be changed dynamically by the server.

Using the GPS and WiFi capabilities of the iOS devices, player location can be monitored and recorded with a precision radius ranging between 2 and 10 meters. Accuracy depends on signal interference, weather conditions and availability of cellular data. The relative accuracy of the device antennas allows dynamic content accessibility depending on geospatial context. A player must be physically close to an objective, agent or player in order to interact with that actor in game. We restrict our interaction radius to a distance of 30 meters, though this adjustable in the client build of the game.

Figure 1 shows the *map view* of the TEAM-IT Interface on the iPad. In the top bar, the players team is identified,

Figure 2: TEAM-IT iPad Tasks View shows the tasks in a task collection box, skills required to complete the task, points received for completing it, and status of each task.

their current score is given along with the time remaining in the game. At the top right, the player is shown which skills they have and how many "cards" of that skill they have. In the main part of the panel, a map has information tagged at various locations. Boxes within the discovery radius are shown as orange squares. Boxes within the interaction radius are shown as larger orange squares with exclamation points. One is also able to see all players on the screen as a circle with their player ID and agents which are circles marked with an "A". The color of the circle indicates the team. In this view, there are two agents, one for each team.

If one clicks on a box that is within the interaction radius, meaning they are sufficiently physically close to the box location, it will take them to the task view for that box. Figure 2 shows the *task view* of the TEAM-IT Interface on the iPad. Each row is a task and the symbols in that row denote the skills that need to be applied simultaneously within the interaction radius of the box in order for the task to be completed. At the top right of each row, the points awarded to each team for completing the task are displayed. Completed tasks and tasks in-progress are marked dynamically.

Intelligent Agents

TEAM-IT supports having intelligent software agents being part of the game. They have the same capabilities as players in terms of begin able to move, discover boxes and apply cards. Agents can be assigned to any team just as any player can. Figure 3 shows agents in the Map view of the iPad implementation as well as in Interaction views.

When a player is close to another player, human or software, they can interact. One interaction, which is not used in this paper, is trading. Agents can trade "cards", i.e., resources, with other players.

A second interaction, which we do use in this paper, is the ability for agents to give advice to their team members. Software agents can be given with arbitrary knowledge of the game state in order to help game play. In this paper, agents were given full knowledge of the locations of all the boxes,

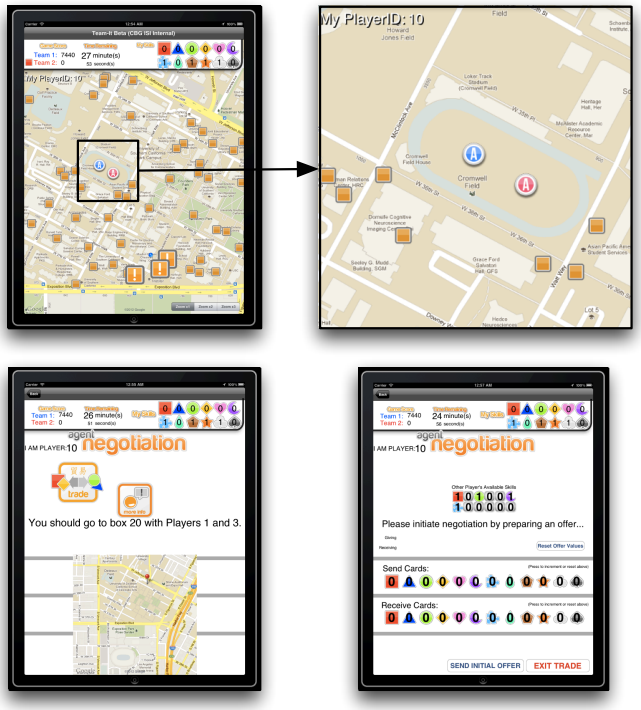

Figure 3: Agents in map (top) are marked by an "A". Example advice-giving and negotiation screens are shown.

the tasks they contained as well as the points they would yield. If asked by a member of their own team, they would give advice to team members on which location to travel to and with whom to go there. This advice would take into account the current location of all the members of the team as well as other potential goals of the team designer. In our paper, we had a secondary goal (after improving game play experience) of having the players move more than they would without advice. This is motivated by domains where facilitating health and exercise may be the goal of the designer.

The algorithm for the advice is as follows. The following procedure is run to generate candidate advice c.

1. Get random subset of players S_c including the player asking for advice and a random box B_c.
2. Calculate the point efficiency e_c as:

$$e_c = \left(\sum_{t \in B_c} p_t I_{S_c,t} \right) / |S_c| \cdot max_{i \in S_c} T(L(i), L(B_c))$$

which states that the efficiency is the sum of all the points p_t for all tasks in the box ($t \in B_c$) where the subset can accomplish the task with its skills ($I_{S(c),t}$) divided by the longest time that anyone in the the team will take to get from their current location ($L(i)$) to the location of the box ($L(B_c)$) and the number of players in the subteam ($|S_c|$). The locations of the players are obtained dynamically when the request is made. This is a measure of points per player-minute. If the time required is greater than the

Figure 4: TEAM-IT Cosmopolis: At the top left, we see how players see boxes and how the task screen is implemented. Players also have a minimap that echoes the iPad overhead view. At the top right, we show players during game play. At the bottom we show Cosmopolis screenshots compared to pictures taken from the real world.

time until the end of the game, the candidate advice is disregarded.

3. Calculate walk score as follows:

$$w_c = \sum_{i \in S_c} D(L(i), L(B_c))$$

which is the sum of all the distances the players have to walk to get to the box.

4. The candidate c is added to the candidate set with values (e_c, w_c).

Because we need to give advice in real-time the number of candidates generated can be adjusted to meet the computational limits and dynamics of game play. Once the candidate set is selected, we could simply give the advice with the highest efficiency, but since we also wanted to investigate changing player behavior, we gave the candidate that had the highest walk score that had an efficiency within 90% of the maximum efficiency. An example of advice is shown in the bottom left of Figure 3.

Cosmopolis Implementation

The development of the virtual TEAM-IT game was done using the Cosmopolis framework (Spraragen et al. 2012). Cosmopolis is a Massively Multiplayer Online Game (MMOG) designed as a research testbed for social and behavioral models." Researchers have the ability to develop their own subgames within an outer world framework. Some of the features of Cosmopolis include support of AI-based non-player characters, state-of-the-art graphics and effects, support of large cities and wilderness zones, as well as an in-game world editor tool which can be used to create new regions within the game. Because Cosmopolis was designed as a research testbed, it also includes a very flexible logging system, which can be extended by the developers of each subgame. This gives the ability to scientists to extract all the in-game data and make data analysis much easier. This is enabled by the event-based network model of the game which is used to manage communication between clients and the server.

For the purposes of this project, we created a new subgame within the Cosmopolis world and we tried to provide players with a similar experience as in the real-world counterpart. We used our campus as the test environment. To do this we used detailed 3D models that were created by another research lab focusing on geo-immersion. These files were stored in a KMZ format compatible with Google Earth and Google 3D Warehouse. We got the models and first converted them to a Maya compatible format (from KMZ). Then after converting the longitude and latitude to coordinates in the Cosmopolis world, we applied the required transformations to simulate the campus.

The configuration data for the game, like player positions, initial skills, box locations, tasks, etc. are imported in the beginning of the game from XML files. All the screens in the game, like task completion, trading, are made to resemble the ones in the mobile application in order to make the transition between the two easier for the players. Another feature in the game is the minimap which can be used for navigation purposes, as it pinpoints players the position of teammates, boxes and AI agents. Screenshots of the Cosmopolis implementation of TEAM-IT , game play and comparison with the real world are shown and discussed in Figure 4.

sectionExperiments and Results

The two questions we are interested in investigating are the differences between game play of the same location-based game in a real and virtual environment and the impact of adding advice-giving agents to a real-world location based game. We ran a pilot experiment where two teams of five players competed with each other in the same scenario. Each team played 3 games: (A) the iPad version without advice ("No-Agent"), (B) the virtual Cosmopolis version

Figure 5: Photos of players during experiments in the real world coupled with analogous screenshots from Cosmopolis.

Table 1: Summary of Game Statistics

Scenario	Team	Score	Total Dist. (m)	Tasks Completed
No-Agent	A	4550	4905	14
No-Agent	B	2810	4818	24
Virtual	A	15080	6363	29
Virtual	B	8420	5358	37
With-Agent	A	12660	8249	17
With-Agent	B	9610	5562	12

Table 2: Player Behavior in Virtual vs. Real World

	% Score Incr.	% Incr. in Dist.	% Incr. in Tasks
Team A	231%	29.7%	107%
Team B	199%	11.2%	54.2%

("Virtual"), and the (C) the iPad version with advice ("With-Agent"). The order of game play was ABC for the Team A and BAC for the Team B. The winning team for each game, i.e., the one with the higher score for that game, would receive a prize, in this case, movie tickets. The games were run for 30 minutes each, after some very basic introduction on game mechanics in each environment. Photos and screenshots of experiments can be seen in Figure 5. Table 1 shows a summary of the game statistics collected. Due to learning effects and obviously small sample size, we concentrate on the strong within-group differences exhibited in the data.

First, we discuss the results of the mapping experiments between the real and the virtual versions of the game. Our hypothesis that the activity levels in the virtual world would likely be higher is borne out. The number of tasks completed is much higher for both teams, as is the corresponding score. The distance traveled is also higher in the virtual game vs. the real world game (Table 2 and Figure 6). By observing the players during the game, we are also able to suggest qualitative explanations for these differences. Players in the virtual Cosmopolis game clearly took advantage of the "run" action much more often than players in the real world actually ran; players in the real world actually became tired and walked more frequently. Perhaps an even bigger difference is due to the physical limitations of the real world infrastructure. In Cosmopolis, all game communications worked flawlessly,

Table 3: Player Behavior With-Agent vs. No-Agent

	% Score Incr.	% Incr. in Dist.	% Incr. in Tasks
Team A	178%	68.2%	21.4%
Team B	242%	15.4%	-50%

in terms of recording task completion between the desktop clients and the central game server. In the real world, the iPad game clients required wireless communication with the game server, and it was often spotty. This caused players in the real world to spend much more time completing tasks, because tasks can only be completed when there is a connection to the server, and they often contended with poor wireless signals.

Second, we compare the results of the No-Agent base case and the With-Agent case (Table 3). The AI agent is able to promote much more player movement. One team traveled 68% more when using the agent's advice. This corresponded to each player walking or running 1.6km in half an hour, which is a good, mild exercise. By examining their velocity profile over time, we can see that it is actually an even better exercise routine, since it involves several fast sprints interspersed with rest periods. The agent is also helpful in that it guides players to the highest point value tasks, which happen to be located in a distant corner of the campus, relative to the starting location, as shown in Figure 6. In the No-Agent games, we can see that Team A happened to find this corner by chance, through their own exploration of the campus; while Team B wandered around several other parts of campus and never found this area. In the With-Agent games, we see that both teams consulted the advice-giving agent and ran directly to this area. This resulted in large score increases for both teams when assisted by the AI agents. In terms of number of tasks completed, Team A saw a moderate increase using the agent, while Team B actually only completed half the number of tasks. Partly this can be understood since the agent serves to increase the quality (point value) of the tasks completed, rather than increasing the quantity. In fact, by recommending distant tasks, it may actually lower the number of tasks that can be completed. Players also seemed to find the game more fun with the involvement of the agent. In written feedback we collected after the game, several respondents noted this; for example, one wrote: "The agent

Figure 6: Locations and point values of tasks completed by teams in the three scenarios. The star denotes the starting location.

was a great idea and certainly improves the way we play."

Conclusion

The experiments demonstrate the difficulty in designing virtual world games and training simulators that can accurately mimic real world situations. Player behavior is markedly different in the virtual version of the game. Our pilot trial also shows the benefit of adding an advice-giving AI agent. Players increased movement, scored higher, and enjoyed the game more. This paired virtual and real-world game framework promises to enable many more future investigations into the mapping problem and the role of AI agents in location-based games. For example, we are actively designing negotiation agents that trade skills with the players. We hope to release the framework for use by other researchers in the field.

Acknowledgments

Supported by AFOSR under Award No. FA9550-10-1-0569 and MURI Award No. W911NF-11-1-0332

References

Adobbati, R.; Marshall, A. N.; Scholer, A.; and Tejada, S. 2001. Gamebots: A 3d virtual world test-bed for multi-agent research. In *Proceedings of the 2nd Int'l Workshop on Infrastructure for Agents, MAS, and Scalable MAS*.

Chang, M.; Lu, C.; Kinshuk, D.; and Huang, E. 2011. Usability of context-aware mobile educational game. *Knowledge Management and E-Learning* 3(3).

Fischer, J.; Flintham, M.; Price, D.; Goulding, J.; Pantidi, N.; and Rodden, T. 2012. Serious mixed reality games. In *ACM Conference on Computer-Supported Cooperative Work*.

Grosz, B. J.; Kraus, S.; Talman, S.; Stossel, B.; and Havlin, M. 2004. The influence of social dependencies on decision-making: Initial investigations with a new game. In *AAMAS*, 782–789.

Hajarnis, S.; Headrick, B.; Ferguson, A.; and Riedl, M. 2011. Scaling mobile alternate reality games with geo-location translation. In Si, M.; Thue, D.; André, E.; Lester, J.; Tanenbaum, J.; and Zammitto, V., eds., *Interactive Storytelling*, volume 7069 of *Lecture Notes in Computer Science*, 278–283. Springer Berlin / Heidelberg.

Ishida, T., and Hattori, H. 2009. Participatory technologies for designing ambient intelligence systems. *J. Ambient Intell. Smart Environ.* 1:43–49.

Jennings, N.; Rogers, A.; and Moreau, L. 2011. Human-agent collectives: from foundations to applications.

Kleiner, A.; Behrens, N.; and Kenn, H. 2006. Wearable computing meets multiagent systems: a real-world interface for the RoboCupRescue simulation platform. In Jennings, N.; Tambe, M.; Ishida, T.; and Ramchurn, S., eds., *1st International Workshop on Agent Technology for Disaster Management at AAMAS06*.

Lund, H. H., and Pagliarini, L. 2011. An interactive tool for creating multi-agent systems and interactive agent-based games. In *AAMAS*.

Spraragen, M.; Landwehr, P.; Ranganathan, B.; Zyda, M.; Carley, M.; Chang, Y.-H.; ; and Maheswaran, R. 2012. Cosmopolis: A massively multiplayer online game for social and behavioral research. In *Proceedings of the 4th International Conference on Applied Human Factors and Ergonomics*.

Tan, Q., and Kinshuk. 2009. Client mobile software design principles for mobile learning systems. *iJIM* 32–37.

van Wissen, A.; van Diggelen, J.; and Dignum, V. 2009. The effects of cooperative agent behavior on human cooperativeness. In *AAMAS*.

Williams, D. 2010. The mapping principle, and a research framework for virtual worlds. *Communication Theory*.