An Agent-Based Simulation Framework for Location-Based Games

Thomas Heinz and Christoph Schlieder
University of Bamberg
An der Weberei 5
Bamberg, Germany
{thomas.heinz, christoph.schlieder}@uni-bamberg.de

Abstract

There is considerable interest in substituting at least some of the tests in the early design phases of location-based games by simulations because field-testing in outdoor environments comes with high costs. Unfortunately, available agent-based simulation frameworks provide insufficient support for that task. The paper presents a simulation framework for location-based games together with the underlying generic player model. Features of the framework include functionality for importing geographic data and for simulating pedestrian locomotion. As a proof of concept, we implemented the framework as an extension to the Repast agent-based simulation environment. A simulation of a location-based game demonstrates the functioning of the framework.
Keywords: agent-based simulation, location-based games, pedestrian routing.

1 Introduction

Location-based games have attracted considerable interest from player communities and game designers alike, but they are still in their infancy when it comes to commercial success [1]. This could be a direct consequence of a number of difficulties developers face designing this kind of games [2]. Compared to video games especially, the development and balancing of the game mechanics is a lot more challenging. The spatial methods of game analytics, such as trajectory- or behavioural analysis all require player data [3]. Obtaining such data for location-based games involves extensive field studies in which test players physically move through the outdoor environment of the game. Such tests are expensive in terms of human resources as well as time budget.

Simulation models can provide a remedy for this problem but building simulations that integrate geographic information systems (GIS) to analyse problems which incorporate location, mobility and environment-interaction aspects is not an easy task and poses various key challenges [4]. While there exist agent-based simulation toolkits with limited GIS functionalities, such as AnyLogic[1], Mason[2], Netlogo[3], and Repast Simphony[4], they "do not extend to the point where very detailed spatial models can be built within their structures" [5].

In this paper, we present an agent-based simulation framework based on Repast Simphony, which provides the spatial modelling features needed for testing location-based game mechanics. Our framework comes with functionality for creating typical game elements such as geographic places and software agents (players) that trigger game events by moving from one place to another. In their game play, the agents may follow different tactics, as this would be the case with real players. The interaction between agents and the environment is constrained by geographic data imported from OpenStreetMap at the start of the simulation. Since much of the work in designing a location-based game concerns temporal balancing, a realistic simulation of pedestrian locomotion constitutes a central requirement for the simulation framework. Even in urban environments, we cannot expect to find only street-bound locomotion. In open spaces such as public parks, players move along less constrained trajectories. Our pedestrian routing engine uses the geographical data to support free-space navigation as well as network-bound navigation.

The remaining paper is structured as followed: In section 2 we present a player model which improves upon the reasoning-action model described in [6]. Our model incorporates more complex player reactions to game state changes and the possibility to switch player behaviour on the fly. In section 3 we present the agent-based simulation framework which implements the player model and proposes an approach for pedestrian routing compatible with the requirements of the player model. As a proof of concept, we demonstrate the functioning of the framework by simulating the player behaviour in the location-based game GeoTicTacToe in section 4. Finally, we conclude with a discussion of lessons learned and give an outlook on future research (section 5).

2 A Player Model for Location-Based Games

The purpose of a player model is to abstract from the details of specific location-based games by providing generic descriptions of the game elements needed for modelling a wide range of game mechanics. Game elements include players and places. Most importantly, the player model describes the reasoning-action cycle of the player and the state space relevant for simulating the game.

Our starting point is the player model for Geogames described in [6], which has been successfully applied to a variety

[1] www.anylogic.com
[2] cs.gmu.edu/~eclab/projects/mason
[3] ccl.northwestern.edu/netlogo/docs
[4] repast.sourceforge.net

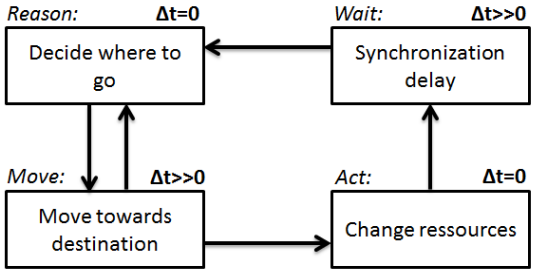
of different location-based games. We use essentially the same set of basic game elements but introduce a more general type of state space:

- *Places:* a finite set of (immobile) regions of interest in the geographic environment. Any actions involving game resources are only possible at these places.
- *Players:* autonomous agents that move through the environment picking up game resources at some place and depositing them at some other place. Each player has a geographic position at each point in time.
- *Resources*: virtual token with a game-specific meaning. For instance, a player could claim a place by depositing her or his ID-token. Resources are allocated either to a place or to a player.
- *State space:* the geographic location of the players and the mapping of resources onto places and players determine the state of the game. A single state is distinguished as the starting state of the game and a game-ends condition characterizes one or more states in which the game stops.

The Geogame player model of [6] works with a much more specific state space where the state of the game is defined as a mapping of players and resources to the places. In other words, the model takes a snapshot of the game actions which assigns every player to a place. Players who are moving between places are assigned to the last place they have left. There is no explicit modelling of player movement. While this simpler model has the advantage of reducing the size of the state space for a search-based analysis, it is clearly not adequate for an agent-based simulation approach, which aims at an explicitly reproducing pedestrian locomotion.

In our player model, at each moment in time, geographic coordinates specify the position of each player. By doing so, we switch from a discrete to a continuous event handling system. Consequently, the player model supports a more complex reasoning action loop (Fig. 1). A player first acts on the resources of the place, which is assumed to take almost no time (ACT phase) in comparison to other phases of the cycle. Some sort of optional in-game task, e.g. a mini-game, might delay the player from moving on (WAIT phase). Based on information about the state of the game, the player then takes a decision which place to move to (REASON phase). We assume that the time needed for reasoning is negligible compared to the time spend on in-game tasks and locomotion. Finally, the player starts to move towards a place (MOVE phase).

Figure 1: Reasoning-action loop



The reasoning-action loop of the Geogame model of [6] forces the players actually to reach the destination before taking a next decision. In our more general model, however, players continuously re-evaluate their decision based on information about the state of the game. This reflects the fact, that real players often modify their decision to move to a certain place, when they learn that another player will reach that place before them.

3 The Agent-Based Simulation Framework

For completing the simulation framework, we need to address two more tasks: (1) the implementation of the player model with an appropriate agent-based simulation environment, (2) the implementation of geo-processing functionality, most importantly, pedestrian routing and OSM data import.

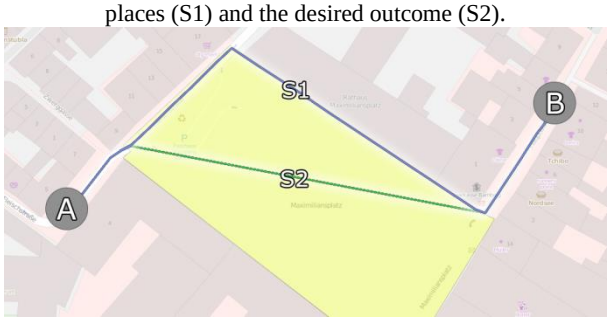
The open source toolkit Repast Simphony was chosen as the simulation engine since it not only provides state of the art components for all required elements of agent-based simulation but also supports extensions because of its modular architecture. Repast also offers basic support for the modelling and the visualization of geographic information.

In the implementation of the player model, the agents have direct access to the pedestrian routing functionality described below as well as to the geographic data. This reduces the complexity of developing a simulation considerably. The game designer just needs to adjust or extend the existing game element classes according their requirements. She or he can model the game resources using the Java interface or via scripting in Groovy or through graphic modelling.

In order to simulate a realistic locomotion of players between the places of the game, the framework supports importing data from OpenStreetMap (OSM). The game designer wanting to create the simulation of a game in a particular geographic environment, simply specifies a bounding box or a more complex API request and the data is automatically acquired through the OSM Web API.

The GraphHopper library provides the foundation for the routing functionalities (A* and Dijkstra shortest path algorithms). However, the library does not support free-space navigation, which is a serious shortcoming for the simulation of pedestrian locomotion behaviour in location-based games. Even shortcut situations as the one depicted in Fig. 2 cannot be handled by network-bound navigation.

Figure 2: GraphHopper routing behaviour over traversable places (S1) and the desired outcome (S2).



We therefore extended the routing library by free-space navigation based on an efficient visibility graph algorithm [7]. All

required data for this process is already present within the imported geographic data. The algorithm returns the Euclidian shortest path for a free-space navigation problem with obstacles. This is generally considered a sufficient approximation of how players move in the environment when subject to tight temporal constraints. In addition, an abstract version of the algorithm permits to implement customized best path criteria.

Fig. 3 shows the components involved in generating a route for a player agent. The framework imports OSM data and forwards it to the GraphHopper library which builds the routing graph. A special feature parser accesses both, the OSM data and the routing graph, to scan for characteristic features like traversable places. Router implementations then make use of the routing graph to calculate routes for the player agents.

Figure 3: Components involved in pedestrian routing

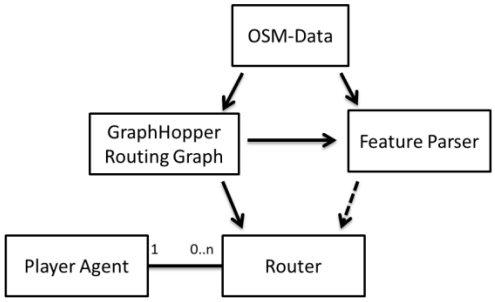
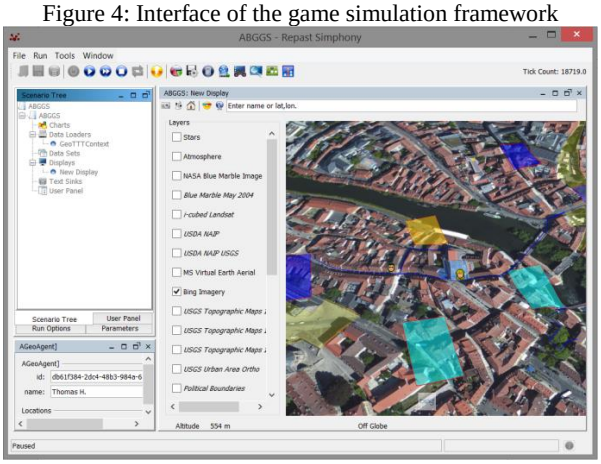


Figure 4: Interface of the game simulation framework

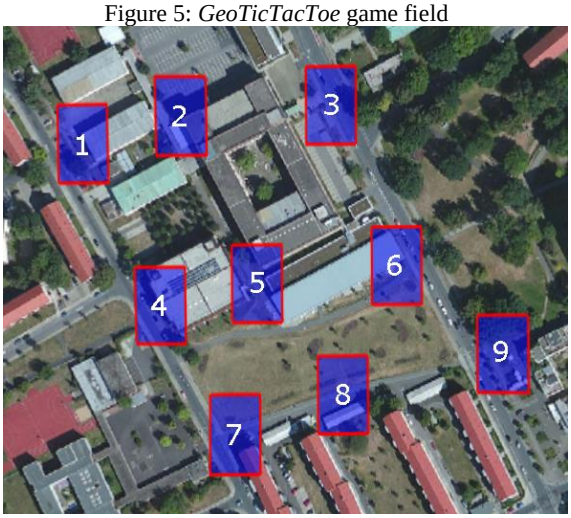


4 Proof of Concept

A major advantage of testing location-based games by agent-based simulation is that the simulation permits to model different player tactics and to let them play against each other. In contrast, the analysis proposed in connection with the Geogame player model of [6], that is, MinMax-search in the state space of the game, assumes uniform (and optimal) player decisions.

As a proof of concept, we apply the simulation approach to the GeoTicTacToe game for which the Geogame player model had originally been developed. This well researched location-based game is a spatial variant of the classic paper-and-pencil game for two players. Fig. 5 shows a typical spatial layout for *GeoTicTacToe*. The blue rectangular patches, numbered 1 to 9, depict the 9 places of this game. The first player to enter physically a place marks it with his or her token (X- or O-token). As in the paper-and-pencil version, the first player to put three token in a row (e.g. 4-5-6), a column (e.g. 3-6-9) or a diagonal (1-5-9 or 3-5-7) wins the game. As most location-based games, GeoTicTacToe is played in real-time: players do not take turns but move and act independently (but not agnostic) of each other.

Figure 5: *GeoTicTacToe* game field



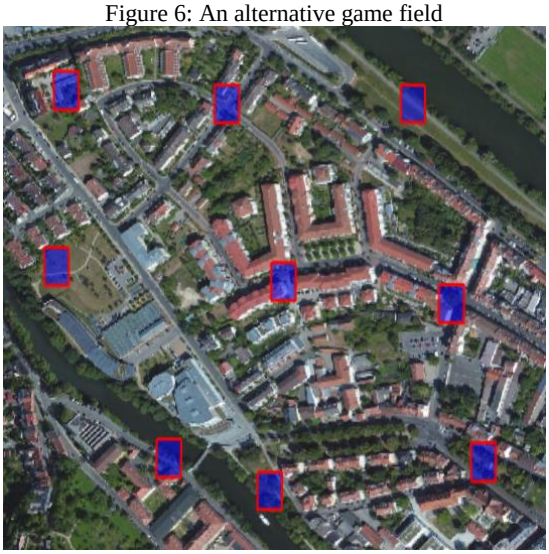
Despite its simplicity, the game illustrates that the spatial layout of the places has a huge impact on the balance of a location-based game. Imagine a row of places 1, 2, and 3 very close to each other while the other places have much larger mutual distances. Different from the paper-and-pencil version where the best first move always consists in taking the centre (place 5), with the spatial layout described, one of the places 1, 2 or 3 might be a better choice. The implications of a particular spatial layout are difficult to anticipate for a game designer and it becomes even more complex when the designer tries to balance the game for players adopting different tactics.

The use of the agent-based simulation framework, however, easily permits to obtain valuable insights into the type of games that emerge from different spatial layouts. We compare the game field shown in Fig. 5 with that in Fig. 6. Obviously, the field in Fig. 6 covers a larger space, that is, the average time needed to move between places should be larger. The exact locomotion times, however, are difficult to estimate by visual inspection since they depend on a combination of network-bound and free-space navigation. We consider three different tactics for agents. The RANDOM tactic serves as a baseline, while the other two, the PAPER tactics and the NEAREST tactics are found in test games of human players:

- RANDOM tactics: The agent randomly chooses which place to move to next.
- PAPER tactics: The player chooses the next place to move to according to the optimal strategy in the paper-and-pencil game. This leads, for instance, to preferring the central place 5 to the corner places 1, 3, 7, and 9.

- NEAREST tactics: The player chooses the place, which she or he can reach in the shortest time.

Figure 6: An alternative game field



On both fields players following the three tactics were set up to compete against each other in 1000 simulation runs for each pairing. In all simulations, both agents started at the same location and moved with the same speed. Tables 1 and 2 display the winning ratio for each configuration.

Table 1: Wins after 250 games for game field from Fig. 5

		Player A		
		RANDOM	PAPER	NEAREST
Player B	RAND.	A wins: 115 B wins: 120 Ties: 15	A wins: 137 B wins: 97 Ties: 16	A wins: 231 B wins: 18 Ties: 1
	PAPER	A wins: 0 B wins: 250 Ties: 0	A wins: 111 B wins: 104 Ties: 35	A wins: 250 B wins: 0 Ties: 0
	NEAR:	A wins: 33 B wins: 212 Ties: 5	A wins: 0 B wins: 250 Ties: 0	A wins: 128 B wins: 122 Ties:

Table 2: Wins after 250 games for game field from Fig. 6

		Player A		
		RANDOM	PAPER	NEAREST
Player B	RAND.	A wins: 110 B wins: 112 Ties: 28	A wins: 156 B wins: 82 Ties: 13	A wins: 206 B wins: 36 Ties: 8
	PAPER.	A wins: 65 B wins: 173 Ties: 12	A wins: 115 B wins: 85 Ties: 50	A wins: 161 B wins: 89 Ties: 0
	NEAR.	A wins: 22 B wins: 222 Ties: 6	A wins: 71 B wins: 179 Ties: 0	A wins: 129 B wins: 121 Ties: 0

By comparing the simulation results of both game fields, it becomes apparent that the superiority of a tactic may depend on the spatial layout. On the game field from Fig. 5 the NEAREST tactics has huge advantages. Agents adopting this strategy did never loose against agents adopting the PAPER tactics. This is valuable information for the game designer. Someone designing a tourist game who does not want to frustrate naïve players (PAPER strategy), may discard the game field from Fig. 5 based on this result.

5 Discussion

We have presented different components of an agent-based simulation approach to testing and balancing location-based games. Our player model improves upon the Geogame player model of [6] in that it permits more complex player reactions to game state changes. We have described the implementation of the player model as well as the approach taken for pedestrian routing. Finally, a simulation of player behaviour in the location-based game GeoTicTacToe has demonstrated the functioning of the framework.

The comparison of player tactics in the simulation framework provides interesting insights to designers. Such a comparison has not been produced by the search approaches to state space analysis and it is difficult to see how this would be possible. Certainly, such an analysis is beyond the visual inspection capabilities of human game designers.

In further studies we would like to explore more sophisticated player models and game tactics and we are interested in applying them to more complex games. Another area for further research is the systematically identification of geographical features that bias player behaviours.

Acknowledgement

The work reported here has been funded in part by ESRI within the Geogames and playful Geodesign project.

References

[1] F. von Borries, S. P. Walz, M. Bottger, D. Davidson, H. Kelley, and J. Kücklich, *Space Time Play*: Birkhauser, 2007.

[2] J. Jacob and A. F. Coelho, "Issues in the development of location-based games," *International Journal of Computer Games Technology*, 2011.

[3] M. El-Nasr, A. Drachen, and A. Canossa, *Game analytics: Maximizing the value of player data*: Springer, 2013.

[4] A. Crooks, C. Castle, and M. Batty, "Key challenges in agent-based modelling for geo-spatial simulation," *Computers, Environment and Urban Systems*, vol. 32, pp. 417-430, 2008.

[5] A. J. Heppenstall, A. T. Crooks, L. M. See, and M. Batty, *Agent-based models of geographical systems*: Springer, 2012.

[6] C. Schlieder, P. Kiefer, and S. Matyas, "Geogames: Designing Location-Based Games from Classic Board Games," *Intelligent Systems, IEEE*, vol. 21, pp. 40-46, 2006.

[7] M. Kallmann, "Dynamic and Robust Local Clearance Triangulations," *ACM Trans. Graph.*, vol. 33, pp. 1-17, 2014.