Please cite as follows

Kiefer, P., Matyas, S. (2005). The Geogames Tool: Balancing spatio-temporal design parameters in location-based games, In: Mehdi, Q., Gough, N. (eds.): Proceedings of the 7th International Conference on Computer Games: Artificial Intelligence, Animation, Mobile, Educational and Serious Games (CGAMES 2005), Nov.28-30, 2005, Angoulême, France, Univ. of Wolverhampton, School of Comp. and Inf. Techn., pp. 216-222

THE GEOGAMES TOOL: BALANCING SPATIO-TEMPORAL DESIGN PARAMETERS IN LOCATION-BASED GAMES

Peter Kiefer and Sebastian Matyas
Laboratory for Semantic Information Processing
Otto-Friedrich-University Bamberg
96045 Bamberg, Germany
{peter.kiefer, sebastian.matyas}@wiai.uni-bamberg.de

KEYWORDS

Mobile gaming, game design tool, game balancing, spatio-temporal MinMax algorithm, location-based games

ABSTRACT

Taking advantage of the full potential of mobile gaming, location-based games let the player totally immerge in the game experience through physical movement of the whole body (locomotion) in an outdoor environment. Although this offers a variety of new possibilities to the game designer, the task of balancing a location-based game to be fair and challenging is nearly unsolvable with traditional methods from video game design: The real world setting neglects play testing and leaves only "offline" methods from game theory as a possible solution. The real-time aspect of concurrent moves and the spatial aspect of a real-world game board pose new problems for a game theoretical analysis. We propose a spatio-temporal MinMax algorithm as a solution for these problems. Our algorithm is embedded in a tool for balancing the spatio-temporal parameterization of a certain subclass of location-based games called "geogames", allowing a game designer to evaluate value ranges for a challenging location-based game.

INTRODUCTION

Although mobile gaming promises to the game designer a whole new world to play with (Aarseth, 2003), most of today's games for mobile devices are still mainly adoptions of single-player computer games. These games, even though restricted in computational power, graphics and I/O, are good for killing one's time while waiting for the train or standing in line. But while these standard mobile games disregard main features of mobility, location-based games make use of localization technology like GPS and build the player's current position and motion path into the game. Moving and acting in an outdoor environment involves the player through physical movement of the whole body (locomotion) and lets him or her totally immerge into the gaming experience; prominent examples are *Can You See Me Now* in Flintham et al. (2003) or *Botfighters* in Sotamaa (2002).

One major problem in the design of location-based games consists in balancing the various parameters influencing the course of the game. In general, a balanced game design involves two aspects: On the one hand, a game should be fair and favor none of the players by default. On the other hand, a fair game that always ends in draw is fair but boring, so the second demand is to make the game challenging. Both of these aspects will be addressed in our paper.

Traditionally, a balanced design of video games is achieved through repeatedly play testing the game with test users in parallel runs until no more unfair conditions are detected. Obviously, this process is not practicable for location-based games because parallel test runs in the real world turn out to be difficult (or even impossible) given the size of the game area. For most location-based games the game area is not smaller than the area of a town; see *CityPoker* in Kiefer et al. (2005) for an example. To solve this problem, we suggest game balancing to be addressed "offline" before playing on the streets. For classic board games, game theory provides the right tools to analyze the entire game state for fairness. However, location-based games pose two problems for a game theoretical analysis. First, algorithms like the widespread MinMax algorithm (see Russell and Norvig, 2003) are well suited for turn-based games but are not able to handle concurrent move decisions in real-time games. Second, game theory neglects the spatial dimensions of the game board, because in board-games every possible move costs the same amount of time and physical effort. In location-based games, however, a player should deliberate thoroughly whether to invest the time and effort of moving to a very far location. Accordingly, a game theoretical analysis for location-based games should integrate the spatial dimension of the game board.

The contribution of this paper consists in a tool for balancing a class of location-based games, called geogames, first introduced in Schlieder et al. (2005), which enables a game designer to balance his game "offline". This tool uses an extended MinMax algorithm to handle the spatio-temporal parameters involved in the design of geogames. To illustrate the design process, we use *GeoTicTacToe*, also introduced in Schlieder et al. (2005), as our running example. We will evaluate two scenarios and the corresponding design parameter values for challenging game design settings.

The structure of this paper is as follows: In section 2 we summarize Schlieder et al. (2005) with the definition of geogames and use the example of *GeoTicTacToe* to explain the problems arising when creating location-based games from classic board-games. Section 3 introduces the spatio-temporal MinMax algorithm and gives a description of the geogame tool architecture. With this tool, two scenarios of GeoTicTacToe are analyzed in section 4 to illustrate how a game designer can balance the spatio-temporal parameters of a geogame. In the last section we conclude with a discussion of related work and an outlook on future research.

GEOGAMES FRAMEWORK

Designing fair and challenging location-based games is not a trivial task. In Schlieder et al. (2005) a location-based game is considered *challenging*, if it equally demands the players' acting and reasoning skills to win the game. Consequently, neither a pure chase game nor a live version of chess would fulfill this definition.

A transition of classic board games into location-based games, named *spatialization*, provides a rich pool of challenging games, if one major problem is being solved. In the line of Nicklas et al. (2001), "*lifting turn-based restrictions can make a game unfair*", consider a location-based variant of TicTacToe displayed in Figure 1. Like in the classic board game two players, X and O, are trying to place three marks, X or O, in a row, a column or one of the two diagonals to win the game. Note that in the location-based variant, *GeoTicTacToe*, the game board is split on separate geographic locations not necessarily maintaining the appearance of the classic game board, see Figure 5. Furthermore, we determine for Figure 1 that player X moves significantly faster than player O.

Without turn-based restrictions this leads to a simple winning strategy for player X and lets the game deteriorate to a non-challenging race: Player X can simply run from location 1 over 4 to 7 without player O having any chance to hinder him from winning the game.

A surprisingly simple solution is proposed. To design a challenging geogame a game designer must include a *synchronization time interval (syncTime)* in his rule set. Players now must wait at a location until the syncTime is over before they can move again. This syncTime parameter must be chosen individually for each geogame to keep it challenging. SyncTime does not necessarily have to be implemented directly as idle wait time, but can also be integrated indirectly through other game elements. Think, for example, of solving mini games before moving on or searching for elements hidden on the real-world game board, e.g. for playing cards in *CityPoker* (Kiefer et al., 2005).

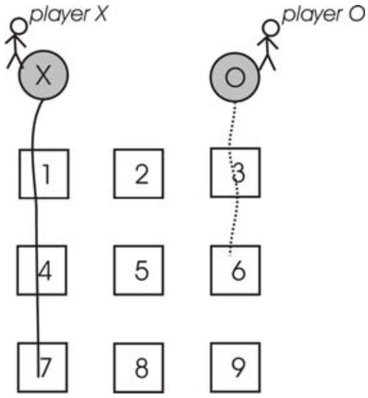

Figure 1: Spatial version of TicTacToe

Geogames are a special class of location-based games with common game elements, defined in the following way: A fixed number of *players* move between a fixed number of *locations* taking up and putting down *resources* when they reach a new location. A *resource* can be anything that the players can pick up and dispose at another location, including *virtual resources*, like the X and O-marks in GeoTicTacToe. However, *resources* cannot move around without any involvement of a player, which is one basic constraint for geogames (*spatial coherence*). The *state of a game* is defined by the players' locations and the distribution of resources over players and locations. *Actions* describe the transitions between game states. A second constraint (*temporal coherence*) asserts that performing an action needs time at least as long as the synchronization interval. For a formal definition see Schlieder et al. (2005).

From the *spatial* and *temporal coherence* two design parameters for geogames can be derived. *Spatial coherence* assures that the players actually move through the game area. As shown in the example of Figure 1, the difference in physical ability, measurable as speed, can alter the challenge of the game dramatically. This phenomenon can be observed on town or football sized game boards as well as backyard sized ones and is not a problem of different arrangements of locations and starting points. Therefore the players' speed will be one important parameter for a game designer. We measure the players' speed by the time they need to move from one location to another. *Temporal coherence* includes the syncTime parameter, which addresses the real-time aspect of geogames. It is the second important design parameter for a game designer, as we already have seen in the example of Figure 1. For different constellations of these two parameters a location-based game is considered to fall into one of three categories: either a *race game*, in which being faster is the only strategy for winning, a *board-game* with strictly alternating move behavior and exclusive emphasis on reasoning, or a *challenging geogame*, where winning demands both, a good strategy and good physical abilities. Parameter values defining a challenging geogame are of most interest for a game designer. An example of a challenging game play style of GeoTicTacToe will be demonstrated at the end of section 4.

GEOGAMES TOOL

The geogames tool helps the game designer in tuning his location-based game to a challenging location-based game. Any location-based game that is an instance of the geogames class can be analyzed with the following steps: 1) Map the location-based game to the geogames framework. 2) Determine the main parameters that are decisive for the excitement of the game (like syncTime). 3) Explore the parameter space by running the geogames tool for different parameter combinations and finally: 4) Choose a parameter combination that is likely to make up a challenging game.

In the remainder of this section we show how to map a location-based game to the geogames framework, explain the spatio-temporal MinMax algorithm, which is the central component of the geogames tool, and shortly describe the architecture of the geogames tool.

Mapping of a location-based game to the geogames framework

In the following we illustrate the ten steps necessary for mapping a location-based game to the geogames framework with our example GeoTicTacToe and the game board displayed in Figure 1.

a) Define the *players P*. In our case: $P = \{P_x, P_o\}$ In general the geogames framework and tool allow more than two players.
b) Define the *locations L*. For GeoTicTacToe we have 9 locations representing the game board and the two starting locations for the players: $L = \{L_1 \ldots L_9, L_X, L_O\}$.
c) Define the *resources R*. Each mark that a player can set in GeoTicTacToe is one resource. At least after his sixth mark a player will have three in a row, column or diagonal, so each player may set a maximum of six marks: $R = \{X_1,\ldots, X_6, O_1,\ldots, O_6\}$.
d) Define the *movingtimes*: $L \times L \times P \rightarrow IR^+$, i.e. the time players need for moving from one location to another. Note that this may vary for each player to model fast and slow players. Furthermore, this does neither need to be proportional to the Euclidean distances, nor symmetrical (e.g. moving up or down a hill). For our example of GeoTicTacToe, we assume symmetrical movingtimes and the Euclidean distances from Figure 1 as time units, e.g. movingtimes(L_1, L_2, P_X) = 2.
e) Define the *starting state*, i.e. the starting resource distribution and the starting location for each player: location(P_x) = L_X, location(P_o) = L_O, resource_pos(X_i) = P_X , resource_pos(O_i) = P_o
f) Define the *final condition*. If a game state is reached that fulfills the final condition, the game ends. In our case: All nine locations L_1 - L_9 contain one resource or three resources of the same type (X/O) are in one column, row or diagonal.
g) Define the *state evaluation:* Each player must have an individual evaluation function for comparing the final states. For GeoTicTacToe each player prefers a winning situation to a draw and a draw to a loss. Furthermore, players strive to win preferably early (with setting only few marks) and to lose preferably late. We call the number of marks that has been set when the game ends *depth of the game*. The final state of Figure 1, for example, has a depth of 5. Obviously, depth may vary between 3 (one player wins before the other could set a mark) and 9 (all locations have been marked).
h) Define the *possible change actions*, i.e. how players may drop and take resources at the locations. In our case marks may not be removed, so that players may never pick up any resources. They are allowed to drop exactly one resource at any location L_i ($i \in \{1..9\}$), but only if that location is empty. L_X and L_O are only starting locations where no actions are allowed.

The two main parameters of the game we will vary with the geogames tool are:

a) The above-mentioned *syncTime*, i.e. the time a player is forced to wait after changing resources. For small syncTime we expect a non-challenging race style of game, for high syncTime the game should deteriorate into a non-challenging board-game style, according to the definition of challenge for geogames in section 2.
b) The personal *speed-factor* describing the difference in speed between the players. In our case we model P_o as a slower player by assuming movingtimes(l_y, l_z, P_o) to be movingtimes(l_y, l_z, P_X) multiplied by speed-factor (for any locations l_y and l_z).

Note that syncTime is given by the rules and therefore the parameter with which the game designer can influence the game flow. On the other hand, speed-factor is not induced by the rules but underlies the estimations of the game designer, e.g. “the difference of speed between the players in my game will not be higher than p%”. Certainly, any estimation on speed-factor is some kind of simplification, because the difference in speed is usually not constant for the entire game. Speed-factor rather expresses a medium difference in speed. The geogames tool will help in finding rules like “if one player is p% slower than the other player, a syncTime of *s* should be chosen to keep the game challenging” or “for a given syncTime *s*, a player needs to be at least p% faster to win the game”. Given these rule sets, the game designer can determine a region in the spatio-temporal parameter space where his geogame will probably remain challenging for a predefined syncTime, although he is lacking knowledge about the exact speed-factors.

Spatio-temporal MinMax algorithm

The central component of our tool is a variant of the standard MinMax algorithm. Standard MinMax (see Russell and Norvig 2003) explores the game state of a deterministic full-information two-person game with two players MIN and MAX who always move alternately. As mentioned above, avoiding strict alternation distinguishes a challenging geogame from a board-game style location-based game. More precisely, the spatio-temporal parameterization of a geogame decides about the alternation behavior, leading to the necessity to integrate spatio-temporal parameters into standard MinMax, an algorithm we call *spatio-temporal MinMax*.

This algorithm builds on the following assumptions: Players always take the shortest path between locations. They do not change destination before arriving at a location and do not wait longer than syncTime at any location. The players always have full-information on the current game state and move optimal in any case. Consequently, players in a geogame behave in the following way (Figure 2). They first decide which location to move next (several possibilities), then they move towards that location and arrive after some time. Now they select which resources to change, before they finally have to wait syncTime and afterwards move on to the next location.

With these assumptions we implement a MinMax analysis like illustrated in Figure 3. Each node in the tree is one state of the game and each state is described by the *distribution of resources* and the players' *position*. A position is a 3-tuple consisting of the player's state (waiting or moving), the location where he is waiting (or in state moving the location he is heading for) and the time units until end of waiting (or until arrival respectively). The player to decide about the next action does not alternate like in standard MinMax, but is determined by the time units, which introduces the temporal aspect into the algorithm. The following rules are applied: If no player has time units 0 in his position (everybody is waiting or moving), subtract from all positions the minimum time units, so that at least one player now has time units 0. If exactly one player has time units 0, he is the one to decide. If more than one player has time units 0 and they are in state moving and heading for the same location – in other words: if more than one player arrives at the same location at the same time – a dice node is inserted (randomized MinMax algorithm, see Kovarsky and Buro (2005)). If more than one player has time units 0, but they are at different locations or not all in state moving, it does not matter who will decide first, so one is selected by chance.

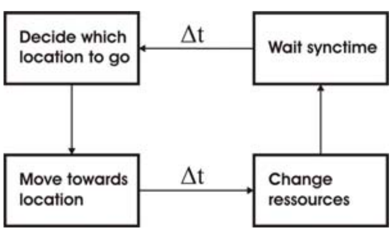

Figure 2: Behavior sequence of players in geogames

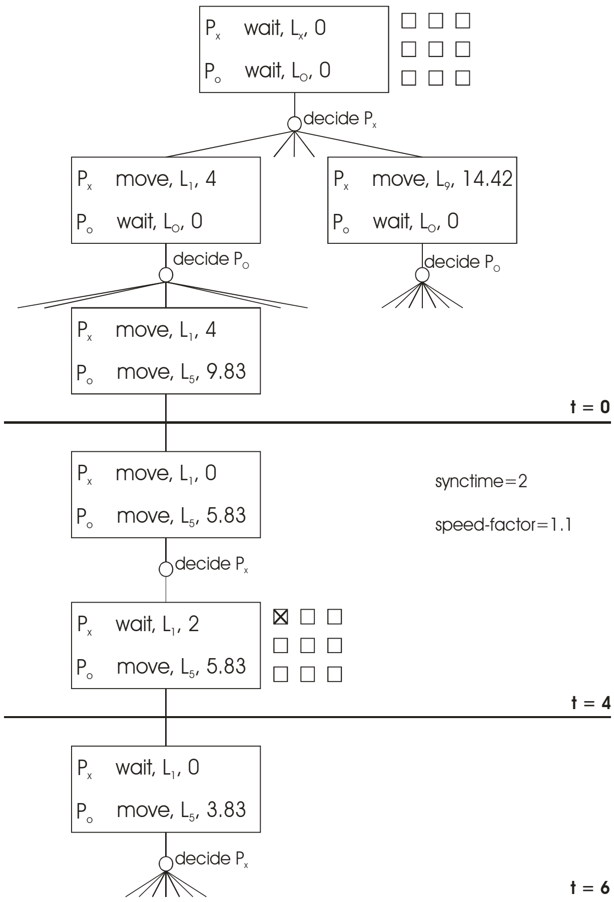

Figure 3: Example game state tree

Our use case holds some simplifying properties: The speed-factor of 1.1 with the Euclidean distances prevents two players from arriving at the same location at the same time, so we do not have any dice nodes, in the pictured example of Figure 3. Furthermore, when players change resources they do not have more than one possibility, namely drop a resource (see Figure 3 at t=4), which reduces the branching factor. Bottom-up evaluation with depth-first search of standard MinMax is applied, whereas the evaluation function takes the depth of the game into account (see section 3.1). The small state space of TicTacToe allows us to prune without a heuristics. Nevertheless, for games with more complex state space, the sophisticated pruning strategies that have already been developed for board games should be applied.

Architecture

The geogame tool has a flexible architecture with the layers illustrated in Figure 4.

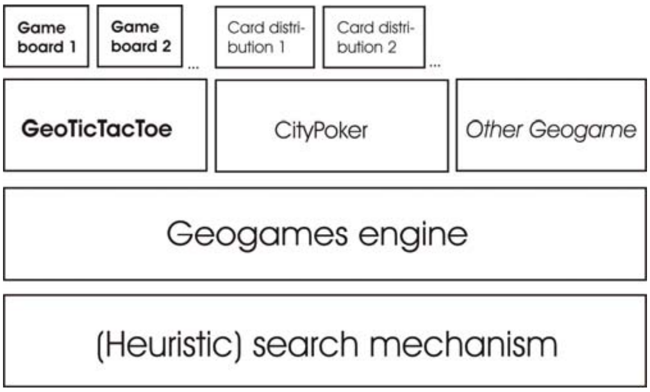

Figure 4: Architecture of the geogame tool

(Heuristic) search mechanism: This layer is a generalization of MinMax and can be used for all kinds of MinMax problems. Like mentioned above, our implementation includes the possibility of dice nodes to handle concurrent resource change decisions. This problem does not bother us in the use case we analyzed for this paper, so we do not go any further into the problem of concurrent resource change decisions here.

Geogames engine: This layer defines all concepts of the geogame framework and combines them to data structures that are taken as input for the underlying search mechanism layer.

Concrete geogame: In this layer, we map the rules of a concrete game to the geogame framework. Most of the parameters of geogames are fixed, for example the number of the locations *L*.

Parameterized concrete geogame: Finally, we are able to define different variants of a concrete geogame like different coordinates for the locations in GeoTicTacToe or different starting card distributions for CityPoker.

RESULTS

The results we present in this section will clarify the benefit of balancing a game with the geogames tool. Figure 5 displays two GeoTicTacToe game boards with different geographic footprints.

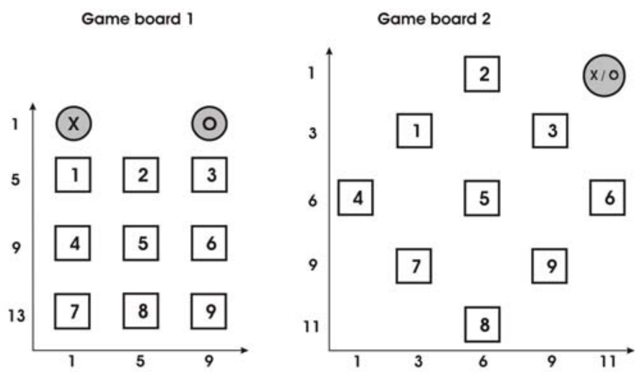

Figure 5: Two spatial variants of GeoTicTacToe

Game board 1 (left) is similar to that of Figure 1 with a standard TicTacToe board and two different starting locations. Game board 2 (right) is a distorted version of the standard board with the four locations 2, 4, 6 and 8 dragged away from the center and a common starting location for players X and O. We analyzed both game boards with a parameterization of syncTime ranging from 0 to 12 in steps of 0.5 and speed-factor between 1.01 and 1.20 in steps of 0.01 and obtained for each test run three results: First of all the *outcome* of the game, X-wins or draw; player O was never able to win because of his disadvantage in speed. Second one *optimal path* through the MinMax-tree, i.e. the course of the game if both players act optimally. Usually, more than one optimal path exists. Third, the number of marks that is set if both act optimally, i.e. the *depth of the game* on the optimal path.

The depth of the game for game board 1 is shown in Figure 6 for all possible parameter settings. We detect three possible depths: At depth 5 we have a course of the game like that of Figure 1, so we call this area *race game*. Even though a depth of 9 could either be a draw or a win for X, in our scenarios a depth of 9 was always associated with a draw, so we call the right area *board-game*, where the players move strictly alternately. The *challenging geogame* we strive to achieve can be found in the center at depth 7. Note that speed-factor 1.0 (both players have same speed) is not displayed. This would make the game end in a draw for every value of syncTime, which indicates the fairness of TicTacToe. Anyhow, in reality two players will never have exactly the same speed and arrive exactly at the same moment at a location.

The pictured results help the game designer in tuning his game. He can start by making estimation on the physical abilities of the players, “the difference in speed between players in my game will never be more than 5%”, and conclude on the necessary syncTime. In this example, he would have to choose a syncTime of at least 3.5, because for speed-factor 1.05 the race game ends at syncTime 3.0 and the challenging geogame starts at 3.5. Possibly he might make an additional demand like “the faster player needs to be at least 3% faster to win, for smaller speed difference the game should end up draw”. As we see in Figure 6, this additional constraint would lead to a syncTime of exactly 9.

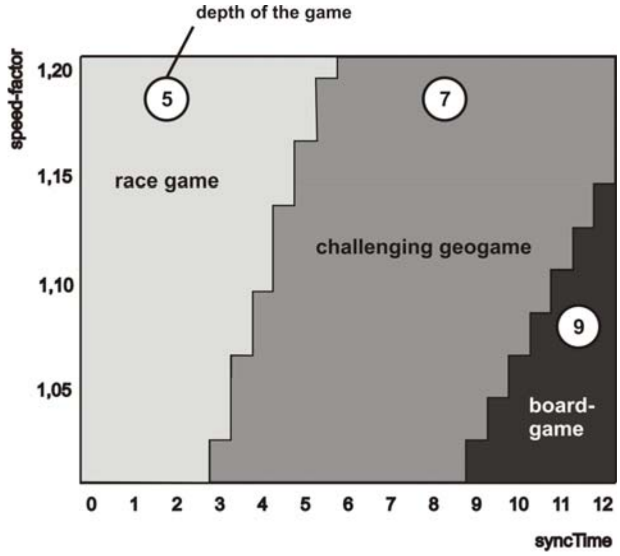

Figure 6: Results for game board 1

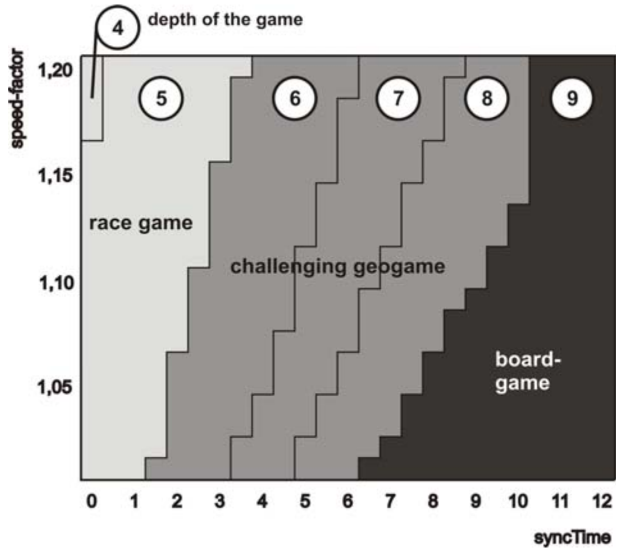

Figure 7: Results for game board 2

The results for game board 2 (see Figure 7) differ significantly from those for game board 1: All depths between 4 and 9 occur and challenging geogames can be found for three different depths, offering a variety of possible game flows. Figure 8 shows the optimal path for syncTime 5 and speed-factor 1.02 with depth 8 and illustrates the kind of strategic elements paired with physical movement that make a geogame challenging. At the beginning of this game it all looks like a race-style game. Player X starts running through the upper horizontal line, while player O occupies the middle spot 5. But because of the syncTime interval, player X is forced to wait and meanwhile player O can prevent a fast win by taking location 1. Player X in return prohibits player O the win by moving to location 9. This in turn forces player O to move to location 6 so that player X is not able to get three in a column. Finally, player X now can take advantage of his speed and takes locations 8 and 7 before player O can reach any of them.

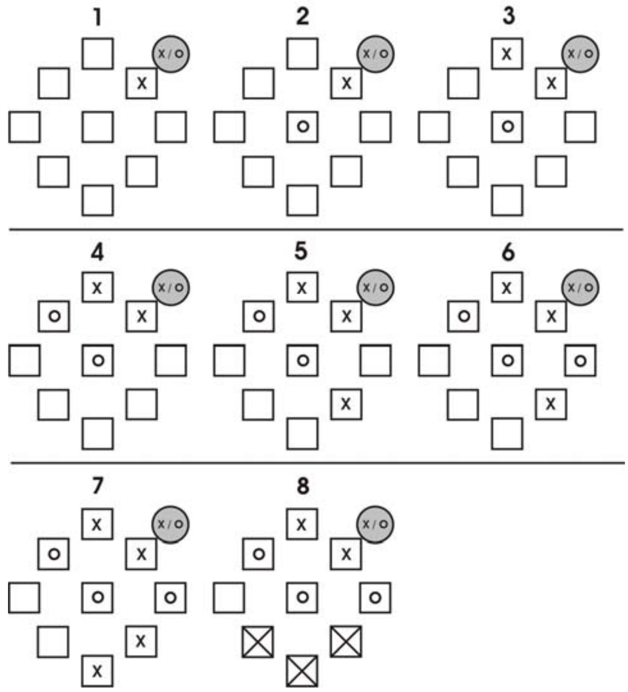

Figure 8: Game flow of a challenging geogame

RELATED WORK AND OUTLOOK

In this paper we presented the geogames tool for balancing geogames "offline" to be challenging and fair. We extended the standard MinMax algorithm to handle concurrent moves frequently occurring in real-time games and took into account the spatial dimension in which location-based games take place. The syncTime parameter and the players' physical abilities in form of the speed-factor are integrated into our spatio-temporal MinMax algorithm. With the resulting parameter space the game designer can decide how to balance the game taking different physical abilities of players into account.

Recently, adaptations of state space analysis have been proposed for real-time settings. An example is the sampling-based method using randomized alpha-beta trees proposed in Kovarsky and Buro (2005). Such approaches address the problem of planning an appropriate move at playing time. However, they do not solve the issue at design time where the designer wants to know how changes in the game's rules affect the game balance. AI techniques, like variants of MinMax-search, have been applied to board games and are constantly improved to create increasingly smart computer opponents, e.g. for Othello (Buro 1999). Although these results are interesting, they are out of the the focus of our paper which is not concerned with the development of optimal search algorithms or pruning strategies but with adapting search algorithms to handle geogames.

Location-based real-time games abandon the idea of turn-taking of classical board games and result in a synchronization problem. Nicklas et al. (2001) propose a solution which is inspired by methods for allocating machine resources to concurrent processes. Similarly, Natkin and Vega (2003) and Vega et al. (2004) show how to assist the game designer in finding dead locks in the game flow using

Petri-nets to describe the game. This type of research focuses on concurrency but does not address, let alone answer the problem of synchronization that characterizes the difference between race-style games, challenging geogames and classical board games.

A line of research similar in spirit to our approach is the study of game design patterns. Typically, a critical mass of existing games is examined to find common game patterns (Davidsson et al., 2004; Björk et al., 2003). Another empirical approach consists in analyzing team design of games (Björk and Åkesson, 2002). An even more holistic approach is followed by Konzack (2002) who distinguishes seven levels of game design: hardware, program code, functionality, game play, meaning, referentiality and socio-culture. Our analysis is clearly limited to the level of game play leaving it to the game designer to decorate the geogame once constructed with appropriate elements right up to the level of socio-cultural embedding. However, different from our emphasis, game pattern research seems to make little effort to gain a deeper understanding of the design parameters and their interaction.

As future work for the geogames analysis tool we plan to implement more complex games, like CityPoker or variants of chess like “progressive chess” or “double move chess” (see e.g. http://www.chessvariants.org/), which already lift turn-based restrictions to some degree. Although our geogames tool is able to handle multi-player geogames, good evaluation functions for concrete geogames have to be implemented and evaluated.

Furthermore, we plan to build into our model a parameter for the players’ cleverness. Imagine one player spending much time on reasoning but moving slowly, while the other player is moving fast but does not invest much effort in thinking. Simulating games with this constellation could make up an interesting case for testing the relationship between reasoning time and acting time. By varying one player’s search depth and the other’s speed, the balance of speed against reasoning could be emulated.

REFERENCES

Aarseth, E. 2003. "A Genre Analysis of Mobile Entertainment", Moore, K. & Rutter, J. (eds) Mobile Entertainment - Concepts and Cultures, pp. 15-26

Björk, S., Lundgren, S. & Holopainen, J.: "Game Design Patterns". *Proceedings of Level Up - 1st international Digital Games Research Conference* 2003, 4-6 November 2003 University of Utrecht, The Netherlands.

Björk, S., J.H. and Åkesson, K. 2002. "Designing Ubiquitous Computing Games" A Report from a Workshop Exploring Ubiquitous Computing Entertainment, Journal of Personal and Ubiquitous Computing, Special issue on Ubiquitous Gaming, vol. 6.

Buro, M. 1999. "Experiments with Multi-ProbCut and a new high-quality evaluation function for Othello". H. J. van den Herik and H. Iida (Eds.): *Games in AI Research*

Davidsson, O., Peitz, J., Björk, S. 2004. "Game Design Patterns for Mobile Games". Project report to Nokia Research Center, Finland,

Flintham, M., Anastasi, R., Benford, S. D., Hemmings, T., Crabtree, A., Greenhalgh, C. M., Rodden, T. A., Tandavanitj, N., Adams, M., and Row-Farr, J. 2003. "Where on-line meets on-the-streets: experiences with mobile mixed reality games". *Proceedings of the CHI 2003 Conference on Human Factors in Computing Systems*, Ft. Lauderdale, FL, April,. ACM Press, New York.

Kiefer, P., Matyas, S., Schlieder, C.2005. "State space analysis as a tool in the design of a smart opponent for a location-based game". *Proceedings of the Games Convention Developer Conference “Computer Science and Magic”, Leipzig, Germany)*

Kovarsky, A. and Buro, M., 2005 "Heuristic Search Applied to Abstract Combat Games", *Proceedings of the The Eighteenth Canadian Conference on Artificial Intelligence*, Victoria

Natkin, S. and Vega, L. 2003. "Petri net modeling for the analysis of the ordering of actions in computer games". In Mehdi, Q. and Gough, N., editors, *GAME-ON, 2003, 4th International Conference on Intelligent Games and Simulation*, pp. 82-89.

Nicklas, D., Pfisterer, C., Mitschang, B. 2001. "Towards Location-based Games". In: Loo Wai Sing, Alfred (ed.), Wan Hak Man (ed.), Wong Wai (ed.), Tse Ning, Cyril (ed.): *Proceedings of the International Conference on Applications and Development of Computer Games in the 21st Century: ADCOG 21*; Hongkong Special Administrative Region, China, November 22-23.

Russell, S. and Norvig, P. 2003. *Artificial Intelligence A Modern Approach*, M. Pompili, Ed. Prentice Hall

Schlieder, C., Kiefer, P., Matyas, S. 2005. "Geogames: A Conceptual Framework and Tool for the Design of Location-based Games from Classic Board Games", *Springer Lecture Notes in Artificial Intelligence 3814 (INTETAIN conference, November 30-December 2, 2005, Madonna di Campiglio, Italy)*

Sotamaa, Olli 2002. “All The World's A Botfighters Stage: Notes on Location-Based Multi-User Gaming”. In Frans Mäyrä (eds) *Computer Games and Digital Cultures: Conference Proceedings.* Tampere: Tampere University Press.

Vega, L., Grünvogel, S. M., Natkin, S.2004. "A new Methodology for Spatiotemporal Game Design", Quasim Mehdi, Norman Gough (Eds.), *Proceedings of the CGAIDE'2004, Fifth Game-On International Conference on Computer Games: Artificial Intelligence, Design and Education,* pp. 109-113

BIOGRAPHY

Peter Kiefer commenced his studies in business information technology at the Otto-Friedrich-University of Bamberg in 2000. During his studies, he was a working student at Siemens ICN, SBS and Deutsche Telekom AG. In the autumn of 2003, he spent one semester abroad at the Lappeenranta University of Technology, Finland. In April 2005, he received his diploma in business information technology after nine semesters and immediately started his work as a research assistant at the chair for computation in the cultural science at the University of Bamberg. Striving for a dissertation in computer science, his main area of research is concerned with intention recognition from motion patterns. In this context, location-based games are analyzed as one use case for intention recognition.

Sebastian Matyas earned a diploma degree in Information Systems from the University of Bamberg, Germany in 2004. Currently, he works as a research assistant at the chair of computing in the cultural sciences (Prof. Schlieder). His Ph.D. research is concerned with semantic information processing in the context of location-based technologies. Furthermore, he is interested in the applications of location-based technologies, especially the design of mixed reality and geogames.