Viktor Wendel, Marc-André Bär, Robert Hahn, Benedict Jahn, Max Mehltretter, Stefan Göbel, Ralf Steinmetz:
Automatic situation recognition in collaborative multiplayer Serious Games. In: Academic Conferences Limited: Proceedings of the 7th European Conference on Game-based Learning , p. 610-619, acpi, October 2014. ISBN 978-1-910309-55-1.

Automatic situation recognition in collaborative multiplayer Serious Games

Viktor Wendel, Marc-André Bär, Robert Hahn, Benedict Jahn, Max Mehltretter, Stefan Göbel, Ralf Steinmetz
Technische Universität Darmstadt, Multimedia Communication Labs, Darmstadt, Germany
viktor.wendel@kom.tu-darmstadt.de
marc-andre.baer@live.de
r.hahn90@gmail.com
benejahn@web.de
max@mehltretter.net
stefan.goebel@kom.tu-darmstadt.de
ralf.steinmetz@kom.tu-darmstadt.de

Abstract:

With the establishment of Serious Games during the last decade, various approaches and concepts for multiplayer Serious Games have been proposed in the last years, many of those for use in training groups or in classroom. One major problem in this field is the adaptation of a multiplayer Serious Game to the needs of a whole group of players, including difficulty, content adaptation, and game pace. This can be done by a human instructor who is responsible for various tasks like coaching, moderating, or guiding the learners, to take care of game pace, difficulty and game related problems. All actions and measures performed by an instructor are usually based on human reasoning. The instructor processes information about the game – the game state, player actions and the overall situation – and fells a decision about if and how to intervene if necessary. Whereas a human instructor can rather easily recognize and judge what a player or a group of players is doing at a certain moment during the game session by observing the scene, his/her background knowledge of the game, and human reasoning, this is extremely difficult to recognize automatically. Especially in game genres where players can move freely in a game world, deciding for themselves about where to go next, what to do and how, it is a challenge to automatically recognize what a player – or a team - is doing at a certain moment.

In this paper, we propose an approach for automatic situation recognition in multiplayer Serious Games. The goal is to automatically recognize what a single player or a group of players are likely doing at a certain point in a game, using information about their locations, their movements, their actions, their interactions, and the game state. Therefore, we define an interface to access elementary and abstract game states, current and past player actions, game quests and (learning) tasks, and game relevant attributes. Moreover, an algorithm is designed to calculate probabilities of possible game situations using the specified game data. We implemented our concept as an extension of the existing collaborative multiplayer Serious Game *Escape From Wilson Island* and performed an initial study to evaluate the soundness and correctness of our approach with promising results. Our approach can generically be transferred to similar multiplayer games (open world, avatar-based action adventure-like games).

Keywords: Serious Games, Collaborative Learning, Game Mastering, Adaptation

1 Motivation

Especially for game-based collaborative learning scenarios major fields of research are adaptation of learning content, difficulty, as well as game-pace and content. First approaches to address problems in those fields have been proposed, focusing on human instructor support and Game Mastering. Many of those approaches consider what information needs to be presented to the instructor and what adaptation mechanisms are necessary and should be regarded.

A different approach is automatic adaptation of multiplayer Serious Games. Any adaptation algorithm, however, needs knowledge about the game state, the learner/player state, and player progress and behavior. In particular, an algorithm needs to be aware of problems and misunderstandings during the game. Whereas a human Game Master can rather easily recognize and judge what a player or a group of players is doing at a certain moment during the game session just by observing the scene, his/her background knowledge of the game, and human reasoning, this is extremely difficult to recognize automatically. Especially in game genres where players control an avatar in a rather open

world and can move freely in this world, deciding for themselves about what to do next and how (like *Second Life*, mods of commercial roleplaying games, or collaborative multiplayer Serious Games like *Woodment* or *Escape From Wilson Island*), it is a challenge to automatically recognize what a player – or a team - is doing at a certain moment. The scenario observed here focusses on non-scene-based open world action-adventure-like games using avatars to (re-)present players.

In this paper, we propose an approach for automatic situation recognition in multiplayer Serious Games. The goal is to automatically recognize what a single player or a group of players are likely doing at a certain point in a game based on information about their locations, their movements, their actions, and their interactions. Therefore, we define an interface to access elementary and abstract game states, current and past player actions, game quests and (learning) tasks, and game relevant attributes. We then create state diagrams regarding dependencies between states and use different kinds of criteria to calculate probabilities of situations. We define criteria based on space, time, and state, like local or global criteria, distance criteria or criteria based on game states or attributes. We designed an algorithm which calculates which criteria are fulfilled using the data gathered from the game. Based on that, the algorithm calculates probabilities for players being in certain situations, like trying to solve a certain task or exploring the level, etc.

We implemented our concept as an extension of the existing collaborative multiplayer Serious Game *Escape From Wilson Island (EFWI)* which offers group tasks designed in a way such that players need to work together and communicate in order to succeed (Wendel et al., 2012). A Game Master Frontend has been implemented to enable an instructor to perform instructor tasks from inside the game at runtime (Wendel et al., 2012). The situation recognition described in this paper is a direct extension to this Game Master framework with the objective of enabling the Game Master framework to automatically recognize game relevant situation in order to inform and support the Game Master and automatically perform adaptations based on the recognized situations

An initial study to evaluate the soundness and correctness of our approach has been carried out comparing the recognized situations with the situations recognized by a real GM. The initial results are very promising. The situation recognition was able to correctly recognize the defined game situations in almost every case. However, further evaluation with a bigger set of users is required.

2 Related Work

2.1 Collaborative Learning

(Roschelle and Teasley, 1995) define collaboration as "*a coordinated, synchronous activity that is the result of a continued attempt to construct and maintain a shared conception of a problem*". The concept of Collaborative Learning is used widely in e-learning and game-based learning. (Dillenbourg, 1999) defined Collaborative Learning as "*a situation in which two or more people learn or attempt to learn something together*". Various parameters define the success of collaborative learning: One core element is the group of learners, characterized by its size and its composition. In this work, the focus is on small learner groups (2-6 players) suitable for cooperative learning scenarios. (Johnson and Johnson, 1999) define five essential elements of cooperation which are a prerequisite for cooperation to take place in cooperative learning scenarios: Positive Interdependence, Individual Accountability and Personal Responsibility, Promotive Interaction, Appropriate Use of Social Skill, and Group Processing.

Computer-supported collaborative learning (CSCL) is the combination of computer technology and collaborative learning concepts. The fields of application are communication, coordination, cooperation in groups, and cooperative learning rooms (especially virtual learning rooms) (Haake et al., 2004, p. 358). Whereas first virtual learning rooms were CSCL applications specifically designed for a CSCL purpose, most often integrating a chat system and a shared screen, later versions used existing virtual worlds like Second Live or Massively Multiplayer Online Role-Play Game (MMORPG) worlds (Eustace et al., 2004).

2.2 Serious Games for Collaborative Learning

In Recent years, first CSCL Serious Games have been designed and implemented. They incorporate the CSCL principles and combine them with Serious Games principles resulting in first multiplayer Serious Games for collaborative learning (Zea et al., 2009). (Hämäläinen, 2011) describes an approach of a collaborative game for vocational learning focusing on design elements essential for

collaboration. Reuter et al., 2012) describe an approach for designing and authoring multiplayer adventures for collaborative learning deriving concepts for puzzle design in multiplayer games.

2.3 Adaptation

The actions and tasks of the instructor can essentially be categorized as assessment and adaptation. Adaption might occur for factors like difficulty, narration, players'/learners' preferences, or the need to compensate a deficit. In general, a common definition of adaptation is *"[the] ability to make appropriate responses to changed or changing circumstances*" (Kaukoranta et al., 2003). It is desirable to adapt games in various dimensions, like those stated above. "*Learning and adaptation are viewed by some as a having a crucial part to play in next-generation games*" (Charles et al., 2005). (Kickmeier-Rust and Albert, 2012) provide an overview over adaptation principles and techniques in Serious Games, like adaptive behavior of agents, motivational interventions, procedural and adaptive level and content generation. (Sweetser and Wyeth, 2005) defined the term '*GameFlow*', transferring the concept of '*Flow*' (Csikszentmihalyi, 1991) to games with the goal of designing and evaluating enjoyment in games. Their model includes the eight dimensions concentration, challenge, skill, control, clear goal, feedback, immersion, and social interaction. Chen, 2007) states that one fundamental condition for *Flow* in games is that the game must provide the right amount of challenges to match with the players' abilities. This, as well as the adaptation principles stated by (Kickmeier-Rust and Albert, 2012), requires that the game knows about the current game state and in what situation players are currently in.

3 Approach

3.1 Problem Statement

The scenario which is focused on in this paper is collaborative learning in small groups using multiplayer Serious Games. The Serious Games regarded here are considered to be non-scene-based, i.e. the games are open in terms of sequence and pacing. Therefore, it can be stated that it is impossible to predict the next game state from the current game state due to player freedom in terms of movement and player actions. This means that each player has full control over his/her character, thus it cannot be predicted what will happen next. This is contrary to e.g. scene-based games where it is much easier to define concrete game-states due to limited player freedom (in that type of game, usually player actions at a specific point in the game are clearly defined and limited).

The player freedom in the open world is the core problem for recognizing what situation a player – or a group of players – is in. Machine-based situation recognition needs to derive a situation from certain environmental conditions which are clearly defined. This is easy in some cases: E.g. a player gathering berries. If a player clicks on a berry bush, it can be stated that the player is in a 'gather berries' situation as this event can be clearly defined by 'clicking the "Gather Berries"-button'. In contrast, in *Escape From Wilson Island*, players need to surround a heron in order to hunt it. A human instructor can easily see when players are moving to hunt and surround the heron from the players' movements. As there are no hard triggers or events clearly defining the start of the hunting situation, this is very difficult to recognize automatically. The players' actions (including their movement) need to be evaluated algorithmically. Merely evaluating a player's position is not enough in this case as a player can be at a position for various reasons.

3.2 Game Interface

Our approach uses elemental game data like game variables, and player parameters, or elemental game actions, like moving, or triggering events by pressing a button. Based on those, we define game states and tasks. We define game situations using different kinds of criteria which indicate that the situation is present. Finally, all situations are evaluated periodically, calculating a probability for each situation to be present at a certain point in the game session.

Definition Game Variable: A *Game Variable* $v \in V$ is an elemental variable which describes a part of the game state. The set of all *Game Variables* V, i.e. the entirety of all *Game Variables*, forms the Game State. A *Player Parameter* $p \in P$ describes an attribute of a certain player, whereas P is the set of all *Player Parameters*. *Player Parameters* are a subset of all *Game Variables*: $P \subseteq V$; Player inventory can be presented through *Game Variables*.

Definition Action: An *Action* $a \in A$ is an elemental player activity which has a well-defined effect on the game (state), whereas A is the set of all *Actions*. An *Action* can be as simple as 'move forward' triggered by pressing the respective key, or 'gather berries' triggered by clicking on a berry bush game object.

Definition State: A *State* $s \in S$ describes a well-defined game status, whereas S is the set of all *States*. A *State* is 'active' if the boolean expression describing the *State* evaluates to true, else 'inactive'. A *Boolean Expression* be is either an elemental *Boolean Expression* or a composed *Boolean Expression*. An elemental *Boolean Expression* compares a *Game variable* $v \in V$ with a target value using an arithmetic test operator. The arithmetic test operators used are: $=, \neq, >, <, \geq, \leq$;
The target value is a concrete value of the respective *Game Variable*'s value range (e.g. x>5). A composed *Boolean Expression* combines two *Boolean Expressions* using a Boolean operator. The Boolean operators used to define a Game State are: and, or, not, xor;

$$f: be(V') \rightarrow \{true, false\} \; with \; V' \subseteq V$$
$$g: S \rightarrow B \; with \; g(be_s) = active \; if \; be_s = true, else \; inactive;$$

Whereas be_s is the well-defined *Boolean Expression* of state s which evaluates whether certain *Game Variables* have a certain value.

Definition Task: A *Task* is a well-defined assignment which has to be fulfilled by one or more players. A *Task* consists of at least two *States* (start and end). The start *State* defines conditions to be met before the *Task* can be started (or for the *Task* to be assigned). The end *State* is reached when the *Task* is fulfilled. There can be *States* between start and end *State*. State transitions can be triggered by player actions or game events. Due to the clear unambiguous structure of a *Task*, it is possible to use state transitions to predict players' most probable next *Actions* and subsequently the next situation.

Definition Region: A well-defined area in the game world (e.g. defined by a square or a circle).

In order to be able to evaluate which situation is currently present, a set of criteria is used. A situation is defined to be true if all of its criteria are fulfilled.

Definition Criterion: A *Criterion* $c \in C$, whereas C is the set of all *Criteria*, is an item which defines if a game condition is fulfilled. Criteria can be partially fulfilled (e.g. distance). The following types of *Criteria* are being defined:

- *Atomic Criterion*: A *Criterion* which is directly retrievable from the game (state), i.e. from a *Game Variable* or *Action*. Examples are: 'Is player x moving?'
- *Spatial Criterion*: A *Criterion* which makes use of player or object positions.
  - *Local Criterion*: Is considered fulfilled if the respective player is in the related *Region*.
  - *Global Criterion*: Is based on the *Local Criterion*. Contains a set of *Regions* and defines the relationship between them. Example: A set of *Regions* has to be visited in a given order.
  - *Distance Criterion*: Based on the distance between a player and another player/object.
- *Temporal Criterion*: *Criterion* which is based on a point in time or on a time interval.
  - *Time Criterion*: The *Criterion* can only be fulfilled at a certain point in time (this point can actually be a time span, like e.g. 'at night').
  - *Interval Criterion*: Is used to measure the distance since the last occurrence of an event or *Action*.
- *State-oriented Criterion*: *Criterion* based on a certain state.
  - State Criterion: Uses a concrete *State*, i.e. is considered fulfilled if the respective *State* is active.
  - *Attribute Criterion*: Is based on *Player Attributes*. If a *Player Attribute* has a certain value, is above or below a certain threshold, or is in a certain range, the *Criterion* is considered fulfilled.
  - *Inventory Criterion*: This *Criterion* is used to model the necessity of an item to be in a player's inventory.
  - *Task Criterion*: This *Criterion* is used to make a task a prerequisite of a situation. The *Task* needs to be either in the state 'not started', 'ongoing', or 'finished'.

Definition Situation: A *Situation* sit is a point of interest in the game which can be considered interesting or relevant due to its meaning for the game or game purpose. Formally, a situation is a well-defined shaping of the Game State. It is defined via a set of *Criteria* C_{sit} defining this situation:

$$sit := C_{sit} \in C$$

Following, two examples will be presented showing the use of criteria to define a situation:

Example 1: Skyrim – Buying a house in the city of Whiterun:

- *Local Criterion*: Is the player in Whiterun?
- *Task Criterion*: Is the player 'Thane' of Whiterun?[1]
- *State Criterion*: Did the player already slay the first dragon?
- *Distance Criterion*: Is the player close to the Non-Player-Character Proventus Avenicci?
- *Inventory Criterion*: Does the player possess more than 5000 pieces of gold?

If all of the criteria listed above are fulfilled, there is a high probability that the player is on his/her way to buy a house in Whiterun. If however, only some of the criteria are fulfilled, the player however is not Thane, it is still possible that he/she is trying to buy the house, but just does not yet know that he needs to be Thane. Or, he/she approaches the NPC because he/she wants to buy something else. Subsequently, the probability that the player is about to buy the house is still relatively high.

Example 2: Building the log hut:

- *Local Criterion*: Are the players close to the hut building area?
- *State Criterion*: Is the first part of the hut build already?
- *Atomic Criterion*: Is a large palm being carried?

If all of those criteria are fulfilled, it can be assumed with a very high probability that players are trying to finish building the log hut. If, however, only a part of those criteria is fulfilled, the players are probably trying something else (like building the raft, or carrying logs to make firewood) or they, for example, do not know where they have to build the hut. Therefore, the probability that the players are actually building the log hut is lower, but significantly above zero.

3.3 Situation Evaluation

Evaluation of all defined situations is performed on a cyclic basis. This can happen every frame, but for performance reasons it can be reduced. It is sufficient to evaluate situations once per second.
For each *Criterion* c in a *Situation*'s set of *Criteria* C, a value between [0;1] is assigned, depending on to which degree the *Criterion* is fulfilled. For a *Criterion* with a binary value range (like the State condition) the value can only be exactly 0 or 1. For *Criteria* with a continuous value range (like the *Distance Criterion*), the resulting value is normalized to [0;1]:

$$f(c): C \rightarrow [0; 1]$$

A *Situation*'s evaluation value $e(sit)$ is defined as the weighted sum of the evaluation of its criteria:

$e(sit) = \frac{\sum_{i=1}^{n} w_i * f(c_i)}{i}$, with c_i:i-th criterion of the situation, w_i: weight for the i-th criterion.

Thus, it is possible to assign a probability value (between 0 and 1) for each defined *Situation* in the game. The evaluator then orders all *Situations* according to their evaluation value in a descending order, trimming those situations where $e(sit) < \delta$ with δ being a threshold to define a minimum probability.

[1] The player can become Thane of Whiterun (a title of nobility) by solving a quest

4 Implementation

We implemented our approach as a Unity3d-based library coded in C# and used the existing Serious Game *Escape From Wilson Island* for a first evaluation. Due to its Game Mastering interface, EFWI already provides access to its *Game Variables* and *Actions*. That provides all necessary information for the situation evaluation.

4.1 Escape From Wilson Island

EFWI is a multiplayer Serious Game focusing on collaboration and teamwork. The narration can be described as a collaborative 'Robinson Crusoe scenario'. The tasks to be solved by the players are designed in a way such that players need to work together and to coordinate their actions. The overall task is to escape from a deserted island where the players stranded. In order to achieve this goal, the players need to build a log hut for shelter, establish a food supply, build a raft, find and fill a gas bottle, steer the raft towards the second island, and ignite a signal fire. The single tasks require a high amount of teamwork and coordination. Carrying a palm to build the log hut, for example, requires three players to coordinate their movement to not let the palm fall down. More detailed information about the game can be found in (Wendel et al., 2012). The Game Master concept and frontend is described in (Wendel et al., 2012).

4.2 EFWI Tasks

In EFWI, players need to solve several subtasks, like building the log hut, or filling the gas bottle, in order to finally fulfill the overall task of escaping from the island.

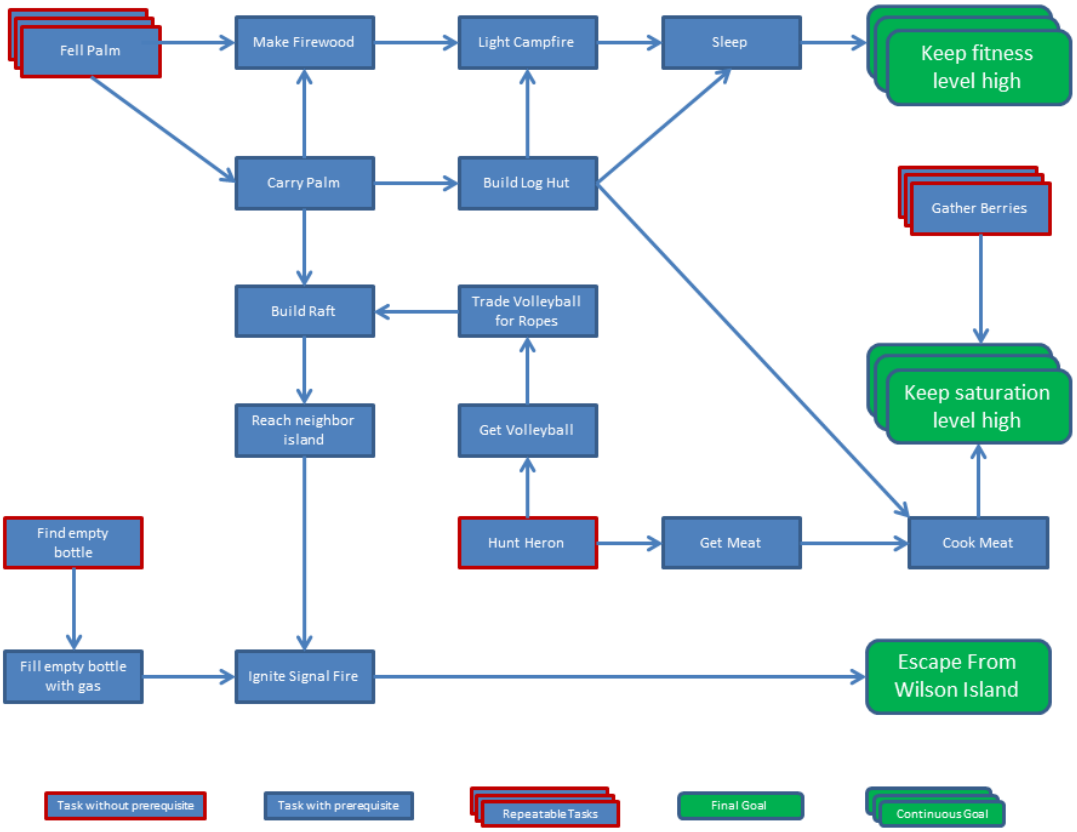

Figure 1: Schematic of all *Escape From Wilson Island* tasks

4.3 EFWI Example Situation

The following example describes the definition and evaluation of the *Situation* 'Filling Gas Bottle'. Table 1 shows a list of the defined *Criteria*.

Table 1: Criteria of the' Filling Gas Bottle' situation

Criterion Name	*Criterion* Type	Explanation	Value Range
DayNight	*TimeCriteria*	Day-night-Cycle	[0;0.2]
FalshlightInInventory	*InventoryCriteria*	Flashlight in inventory?	[0;0.2]
EmptyBottleInInventory	*InventoryCriteria*	Empty Bottle in inventory?	[0;0.2]
GeyserRegion	*LocalCriteria*	Player in geyser Region?	[0;0.2]
FlashlightUsed	*StateCriteruia*	Flashlight being used?	[0;0.2]
FilledBottleInInventory	*InventoryCriteria*	Full Bottle in inventory?	[0;-1]

The gas bottle can only be filled at night. Therefore the first *Criterion* evaluates to 0 if it is day, and to a value between 0 and 0.2 at night, with 0.2 at midnight. A player needs to provide light to find the correct geyser at night. Therefore, a player needs to have a flashlight in his/her inventory. If this is the case, the second *Criterion* is evaluated to 0.2, otherwise to 0. Players also need to have the empty bottle in their inventory. So this criterion evaluates to 0.2 or 0, if not. Players can only fill the bottle, if they are in the 'Geyser *Region'* of the island. If players are close to the geysers, this *Criterion* evaluates to 0.2. The farther away players are, the lower this value becomes, until it reaches 0 when players are far enough away. Finally, the flashlight needs to be on, so that players can fill the bottle. So this *Criterion* evaluates to 0.2 if the flashlight is on, and to 0 if not. Thus, if all of those *Criteria* are fulfilled completely, the *Situation* would be evaluated to a value close to 1.
Once players filled the bottle with gas, they won't try to fill it again. Therefore, the FilledBottleInInventory *Criterion* is added which is evaluated to -1 if a player already has a filled gas bottle, zeroing out all positive evaluations of the other *Criteria*. So, even if players are moving around the geysers at night, having a flashlight on, they probably are not trying to fill the bottle, if it is already filled.

The following Table 2 contains game Variables and Actions which are being accessed to evaluate the *Criteria*:

Table 2: Example Game Variables, Player Actions, and Regions in EFWI

	Name	Type
Game Variables	GameTime	Float
	FlashlightInInventory	Bool
	EmptyBottleInInventory	Bool
	FullBottleInInventory	Bool
	Player Position (player attribute)	(float,float,float)
Player Actions	PickupBottle	
	Fill Bottle	
	Move	
	SwitchFlashlightOnOff	
Regions	GeyserArea	

4.4 EFWI Situation Recognition GUI

In the Game Master GUI (Figure 2), the highest evaluated situations are being displayed for each player and for the whole group. The GM can adjust how many situations to display for each player/group between only the most significant or the 8 most significant situations (see Figure 3).

Figure 2: Game Master overview over the game including the situation recognition

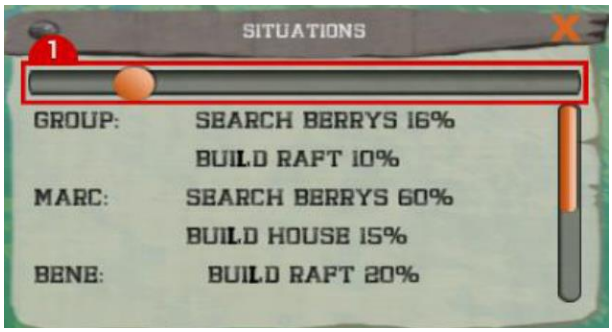

Figure 3: Situation recognition GUI with slider for number of situations per player/group (1)

5 Evaluation

An initial study to evaluate the soundness and correctness of our approach has been carried out comparing the recognized situations with the situations recognized by a real GM. Therefore, the following situations have been defined as relevant: ‘Build log hut’, ‘Build raft’, ‘Drive raft’, ‘Explore island’, ‘Fill bottle’, ‘Hunt heron’, ‘Idle’, ‘Ignite fire’, and ‘Search berries’. Those situations cover the first part of the game until steering the raft towards the second island.

5.1 Setup

Five game sessions (with two runs each) were played with random players. In the first run, a Game Master was watching the gaming session via the Game Master Frontend. The visualization of recognized situations was disabled.

The Game Master was instructed to write down what he/she thought what a player/the group was doing once per minute or whenever he/she thought players were doing something new. The notes were provided with a timestamp. Those notes were then compared to the situations recognized by the situation recognition. The situation recognition was always able to recognize the same situation for the group within 1-5 seconds of the actual situation happening. For single players, the correct situation was always among the three most significant evaluated situations.

After that, in a second game run with the same Game Master the visualization of the situation was enabled. Now, GMs again observed the game. After the game run, the GMs gave feedback on the situations which the situation recognition had proposed. All of the GMs stated that the situation they thought to be present was among the first three situations recognized for a player, and among the first two situations for the group.

5.2 Discussion

In sum, the initial study showed that the situation recognition worked very well for the game EFWI. A Game Master can use the information displayed as a means to reduce to cognitive load during the process of moderating a game session. The additional information might help to judge the current state of the game and the learning/gaming process and help to decide about adaptation measures.

In terms of automatic adaptation of multiplayer games, it can be stated that the automatic recognition of game situations can be viewed as a prerequisite of being able to make sound adaptations to a game. Therefore, the situation recognition presented here might be a first step towards a sound adaptation of non-scene-based multiplayer games with a high amount of player freedom. Being able to automatically recognize the situation, a group of players/learners is in, might help to recognize problems in the learning/gaming process and su7bsequently react with an appropriate game adaptation.

However, a comprehensive study under laboratory conditions is required for further, more precise statements. Moreover, it is necessary to evaluate the situation recognition framework in other similar games for evidence about the genericity of the concept.

6 Conclusions

In this paper we presented an approach for automatic situation recognition in collaborative multiplayer Serious Games as a foundation for automatic adaptation of collaborative multiplayer Serious Games. Our approach uses basic information about the game, like elemental game variables and player parameters as well as player actions and game events. Situations are defined based on criteria which describe a game situation using elemental game information. An evaluation algorithm periodically evaluates the probability of a situation to be present. We implemented our approach as an extension of the existing collaborative multiplayer Serious Game *Escape From Wilson Island* and performed an initial evaluation. First results showed that our approach is able to correctly recognize the defined situations in congruence with human instructors. However, a more comprehensive study is required for more detailed results as well as information about the transferability of our concept to other games of similar type.

References

Charles, D., Kerr, A., McNeill, M., McAlister, M., Black, M., Kücklich, J., Moore, A., and Stringer, K. (2005). Player-centred Game Design: Player Modelling and Adaptive Digital Games. In *Proceedings of the Digital Games Research Conference*, Volume 285. Citeseer.

Chen, J. (2007). Flow in Games (And Everything Else). *Communications of the ACM 50*(4), 31–34.

Csikszentmihalyi, M. (1991). *Flow: The Psychology of Optimal Experience*. Harper Perennial.

Dillenbourg, P. (1999). What Do You Mean by Collaborative Learning? In P. Dillenbourg (Ed.), *Collaborative-learning: Cognitive and Computational Approaches*, pp. 1–19. Oxford: Elsevier.

Eustace, K., Lee, M., Fellows, G., Bytheway, A., and Irving, L. (2004). The Application of Massively Multiplayer Online Role Playing Games to Collaborative Learning and Teaching Practice in Schools. In R. Atkinson, C. McBeath, D. Jonas-Dwyer, and R. Phillips (Eds.), *Beyond the comfort zone: Proceedings of the 21st ASCILITE Conference*.

Haake, J., Schwabe, G., and Wessner, M. (2004). *CSCL-Kompendium - Lehr-und Handbuch zum computerunterstützten kooperativen Lernen*. Oldenbourg Wissenschaftsverlag.

Hämäläinen, R. (2011). Using a Game Environment to Foster Collaborative Learning: a Design-based Study. *Technology, Pedagogy and Education 20*(1), 61–78.

Johnson, D. and Johnson, R. (1999). Making Cooperative Learning Work. *Theory into practice 38*(2), 67–73.

Kaukoranta, T., Smed, J., Hakonen, H., and Rabin, S. (2003). Understanding Pattern Recognition Methods. *AI game programming wisdom 2*, 579–589.

Kickmeier-Rust, M.D. and Albert, D. (2012). Educationally Adaptive: Balancing Serious Games. *International Journal of Computer Science and Sport 11*(1).

Reuter, C., Wendel, V., Göbel, S., and Steinmetz, R. (2012). Multiplayer Adventures for Collaborative Learning With Serious Games. In P. Felicia (Ed.), *6th European Conference on Games Based Learning*, Reading, UK, pp. 416–423. Academic Conferences Limited.

Roschelle, J. and Teasley, S. (1995). The Construction of Shared Knowledge in Collaborative Problem Solving. In C. OMalley (Ed.), *Computer-supported Collaborative Learning*, Berlin, pp. 69–97. Springer-Verlag.

Sweetser, P. and Wyeth, P. (2005). Gameflow: A Model for Evaluating Player Enjoyment in Games. *Computers in Entertainment (CIE) 3*(3), 1–24.

Wendel, V., Göbel, S., and Steinmetz, R. (2012). Game Mastering in Collaborative Multiplayer Serious Games. In S. Göbel, W. Müller, B. Urban, and J. Wiemeyer (Eds.), *E-Learning and Games for Training, Education, Health and Sports - LNCS*, Volume 7516, Darmstadt, Germany, pp. 23–34. Springer.

Wendel, V., Gutjahr, M., Göbel, S., and Steinmetz, R. (2012). Designing Collaborative Multiplayer Serious Games for Collaborative Learning. In *Proceedings of the CSEDU 2012*.

Zea, N. P., Sánchez, J. L. G., Gutiérrez, F. L., Cabrera, M. J., and Paderewski, P. (2009). Design of Educational Multiplayer Videogames: A Vision From Collaborative Learning. *Advances in Engineering Software 40*(12), 1251–1260.