Mission-Based Scenario Modeling and Generation for Virtual Training

Linbo Luo[1], **Haiyan Yin**[1], **Jinghui Zhong**[1], **Wentong Cai**[1], **Michael Lees**[1], **Suiping Zhou**[2]
[1]School of Computer Engineering, Nanyang Technological University, Singapore
[2]School of Science and Technology, Middlesex University, United Kingdom
{lbluo, hyin1, jinghuizhong, aswtcai, mhlees}@ntu.edu.sg, s.zhou@mdx.ac.uk

Abstract

Automated scenario generation for virtual training has become an emerging research problem, as manual authoring is often time consuming and costly. In this paper, we present a mission-based scenario modeling and generation framework for virtual training. In particular, we consider the issue of how the timing of the events in a scenario can impact the training process and how to incorporate such impact into the scenario generation. To this end, our framework is designed to explicitly capture the propagated effect of an event and its influence to other events. For representing mission-based scenarios, the concepts of mission objectives, events, and scenario beats are introduced. The generation process is designed to generate scenarios that can tailor to trainer's preference and adapt to different trainees' skill levels. The efficacy of the proposed framework is demonstrated through an empirical study of human players in a food distribution training game.

Introduction

In recent years, utilizing digital game as a pedagogical resource for serious applications has gained tremendous momentum. One common area of such application is in training, whereby people are tasked to play a game in a virtual world, so as to practice certain skills or complete a mission. In the development of a virtual training system, one of the challenging issues is how to effectively and quickly generate a variety of training scenarios that can meet different training objectives and, at the same time, be customized for individuals. This issue broadly pertains to the problem of content generation, which has been highlighted as a critical bottleneck in game production by many game designers and AI researchers (Togelius et al. 2011; Zook et al. 2012). In fact, it is also a major concern of the government agencies, who hope to leverage on AI and game technologies for various training purposes.

To tackle the scenario generation issue, some researchers (Magerko et al. 2006; Zook et al. 2012) have recently explored the various ways to automate the scenario generation process. Their systems are designed for generating not only a rich variety of scenarios, but also the scenarios tailored to trainee's abilities. However, one important issue that still needs to be addressed is how the timing of events in a scenario can impact the training process. By timing, we refer to the times to inject individual events of a scenario with the events arranged in specific orders. Given the same set of events, different timings can be achieved by shuffling the order of events and adjusting the events' injection times. We hypothesize that the difference introduced by the timing of events can affect a trainee's performance and thus influence the effectiveness of training. In practice, different timings of events may cause a trainee to make responses at different paces and require the trainee to handle either a burst of events or sparsely distributed events at certain time interval during training. All these may lead to the variation of trainee's performance. Therefore, when generating scenarios for virtual training, it is important to capture the impact caused by the timing of events, rather than only evaluating the impact of the events themselves. This is especially prominent when the effect of an event can propagate and influence the subsequent events.

This paper presents the design of mission-based scenario modeling and generation framework, which takes into account of the timing of events in a scenario and how it influences the training process. In our previous work (Luo et al. 2013), a scenario generation system has been designed. The objective of the system is to generate training scenarios that can tailor to the trainer's preferences and adapt to trainee's skill level. Our current design further extends previous work in the way that we explicitly capture the propagation of an event's effect and its impact on other events. The difference caused by the timing and relative order among the events is reflected in the fitness evaluation of our current design. Through an empirical study of human players, the impact of the timing of the events on the trainee's performance is examined. The results generally support our hypothesis and show that our system can provide a better estimation of a scenario's contribution towards various mission objectives.

Related Work

Procedural content generation has recently become an active and important research area in game AI. Various techniques have been explored for automatically generating game content of different types, including game levels (Sorenson et al. 2011; Shaker et al. 2012), virtual characters (Cavazza et

Copyright © 2013, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

al. 2002; Aylett et al. 2006), maps and terrains (Togelius et al. 2010; Doran and Parberry 2010), and narratives (Cheong and Young 2006; Thue et al. 2007; Yu and Riedl 2012). The narrative-based environment has also been successfully applied to the serious game for education (Rowe et al. 2011). A thorough survey on search-based procedural content generation for games is provided by Togelius et al. (2011). Compared to other types of content generation, research in the generation of scenario (i.e., the flow of events) is still a young and emerging area, where a limited number of approaches are proposed for online and offline scenario generation. In below, we focus on the review of offline scenario generation systems, as they are more relevant to our work.

While the online scenario generation systems (Magerko et al. 2006; Niehaus and Riedl 2009) focus on the dynamic adjustment of game events during game-play, the offline scenario generation works on the generation prior to any gameplay and explores how to create globally optimized/customized scenarios. Hullett and Mateas (2009) described a planning-based generation system to generate scenarios from the selected pedagogical goals. Martin et al. (2010) used functional L-systems to specify generation rules that expand training objectives into scenario elements. Our work also focuses on the offline scenario generation. But, it differs from the above work not only in the generation methodology, but also in the evaluation of scenarios. Our scenario generation system considers multiple mission objectives can be exercised in a single scenario and the system evaluates how different events in the scenario can contribute to each mission objective.

Our work is inspired by the work proposed by Zook et al. (2012). In their work, scenario generation is treated as a combinatorial optimization process, where the scenarios are evolved and evaluated based on a set of criteria, such as the diversity of scenario and fitting to learner's ability. To tailor to an individual, the scenario is evaluated by matching the target performance specified by trainer with a learner model. The learner model is used to predict the trainee's performance independently against each skill event in the scenario. In our work, we also adopt a similar optimization approach. However, we consider the scenarios where the effects of events can propagate over time and the timing of events in a scenario may affect the trainee's performance. To model such kind of scenarios, we investigate how to model the propagation of the event's effect and its influence to the subsequent events. The difference introduced by the timing of the events is reflected in the fitness evaluation during our scenario generation process.

Mission-based Scenario Modeling

In this work, we focus on the generation of mission-based scenarios (Reimer 2008), where a trainee is required to undertake a specific mission (e.g., distribute food in peacekeeping). In such a kind of scenario, the injections of events can contribute to the training of certain mission objectives. For example, in a food distribution scenario, the event of adding violent civilians can be used to practice the mission objective of controlling crowd anger. To accomplish the mission, a trainee may need to take actions (e.g., pacify angry civilians and arrest instigators) at any time across the entire scenario, rather than just at the occurrence of individual events. The impact of an event in such a scenario can propagate over time. For example, the event of "add violent civilians" can cause a group of crowd become angry, which in turn can make other people angry. The situation can get worse, when subsequent events (e.g., attack) occur. Thus, the trainee needs to take actions whenever the situation has hindered the achievement of mission objectives.

The modeling and generation of mission-based scenarios is a challenging task, as it requires to capture the dependency among the events in terms of the impact to trainee's performance. Our scenario generation system is specifically designed to address this issue. In this section, we first introduce a conceptual model for representing mission-based scenarios, which includes the concepts of *mission objectives*, *events*, and *scenario beats*. For illustrative purpose, we will use the food distribution scenario as the running example to explain these concepts. The details of scenario generation process will then be explained in next section.

Mission Objectives

Each mission-based scenario in our design is associated with single or multiple mission objectives (MOs). The concept of MOs is similar to the training objectives as described by Martin et al. (2010), which define *what* is to be trained. Each MO specifies certain mission-specific tasks or abilities to be practiced throughout the training. Examples of MOs in the food distribution scenario are to practice how to 1) control the escalation of crowd anger level (MO_1), 2) identify instigators (MO_2), and 3) timely control the distribution process (MO_3). At the design stage, the trainer or domain expert needs to define or select the desired MOs and associate them with the scenario to be used for training.

Apart from selecting appropriate MOs, we also consider the extent to which each selected MO should be exercised in the given scenario. This is modeled using a MO intensity vector as $\overrightarrow{I} = [x_1, x_2, ..., x_n]$, where $\overrightarrow{I} \in \mathbb{R}^n_{\geq 0}$, n is the number of MOs in the scenario. Each element in $\overrightarrow{I}$ represents the training intensity for the corresponding MO. Our scenario model is centered on the concept of mission objectives, in the way that the generation system is designed to generate scenarios that can match the desired MO intensities, according to the trainer's preference and trainee's existing skill level.

Events

Events are the primitive units to form a scenario in our model. A given scenario can include a number of different types of events. Examples of the event types in a food distribution scenario may include "add instigators", "add violent civilians", "rumor spreading" and "food supply disruption". Each event type is characterized by certain parameter(s). By assigning value to the parameter(s), the event instances of a particular type can be created. For example, the event type of "add instigators" can create different event instances by setting the parameter (e.g., the number of instigators) with different value.

For each event instance, we model its contribution towards the exercise of the MOs by using a MO intensity contribution vector $\overrightarrow{I^{c}} = [x_1^{c}, x_2^{c}, ..., x_n^{c}]$. The values in $\overrightarrow{I^{c}}$ indicate the extent to which the given event instance can contribute to the exercise of the mission objectives of the scenario. To determine the value in $\overrightarrow{I^{c}}$, we assume the existence of a suitable technique, which will certainly vary depending on the scenario being modeled. In our current implementation, we use an evaluation function, which takes the event type and the parameters' values of the event instance as the inputs and calculates intensity values for different MOs based on the event's relevance with respect to each MO. For instance, the events of "add instigators" will be attributed a relatively high intensity value for MO_2 (as described previously), as compared to the intensity value for other MOs.

Scenario Beats

As a higher level construct, the scenario beat is introduced to organize the events of a scenario. Compared to story beat in Façade storytelling system (Mateas and Stern 2005), our scenario beat is relatively light-weighted (each beat usually consists of less than 10 events) and it does not require action/reaction pairs for narrative coordination. Scenario beat is used to represent the typical situation in real life, which may involve the occurrence of multiple events in a certain period of time. Formally, the scenario beat is represented as a tuple: $b =< E, O, \delta >$, where E is the set of events contained in the beat, O is the set of ordering constraints on events in E, δ is the maximum time interval in which all the events in the beat must be executed. To form a scenario, a beat sequence can be constructed by selecting a set of scenario beats from the beat collection. Each beat sequence is a variable-sized, ordered set of beats, represented as: $B = (b_1, b_2, ..., b_m)$.

The usage of introducing scenario beats in our scenario model is two-fold. Firstly, scenario beats serve as the key building blocks of a scenario. To design a training scenario, we can leverage on the knowledge of domain experts about commonly observed situations in real-life scenarios and encode such situations into various beats. Secondly, compared to all the possible event instances of different event types, the number of the modeler-defined scenario beats is relatively small. As our scenario generation system operates on scenario beats rather than primitive events, it helps to reduce the search space for scenario generation.

Scenario Generation Process Design

Based on the model of scenario presented in the previous section, our scenario generation system is designed to automatically generate the mission-based scenarios using a genetic algorithm (GA)-based searching process. In overall, the system aims to generate scenarios that can tailor to the trainer's preference on different MOs and adapt to the trainee's skill level. This is achieved by allowing a trainer to provide two key inputs to the system. The first input is a MO intensity vector $\overrightarrow{I^{\text{in}}} = [x_1, x_2, ..., x_n]$, where $\overrightarrow{I^{\text{in}}} \in \mathbb{R}^n_{\geq 0}$, n is the number of MOs. In the current design, each MO intensity is specified in the range from 0 to 10. A trainer high or low values in $\overrightarrow{I^{\text{in}}}$, so that she/he can implicitly control the types of event instances to be generated, which are related to the training of different MOs. The second input from the trainer is a skill level vector $\overrightarrow{L^{\text{in}}} = [l_1, l_2, ..., l_n]$, where $\overrightarrow{L^{\text{in}}} \in \mathbb{N}^n$. It represent the trainee's estimated skill level with respect to each MO. The specification of $\overrightarrow{L^{\text{in}}}$ can be based on *a priori* knowledge about the trainee or a player model built based on the trainee's previous performance data. In the current design, a trainee's skill level is graded in the range from 1 to 5 for each MO.

The desired MO intensity vector $\overrightarrow{I^{\text{d}}}$, which is used to determine the fitness of scenarios, is derived with the combination of $\overrightarrow{I^{\text{in}}}$ and $\overrightarrow{L^{\text{in}}}$. Each element x_i^{d} in $\overrightarrow{I^{\text{d}}}$ is proportional to the product of the corresponding element in $\overrightarrow{I^{\text{in}}}$ and $\overrightarrow{L^{\text{in}}}$, that is, $x_i^{\text{d}} \propto x_i l_i$. The purpose of doing this is to compensate the difference in trainees' skill levels. Pedagogically, a trainee should be assigned with a scenario, which is appropriate to her/his existing skill levels. Thus, given the same MO intensities specified by the trainer, the desired intensities are adjusted according to different trainees' skill levels.

Once the trainer's inputs is received and $\overrightarrow{I^{\text{d}}}$ is derived, the GA-based generation process starts by constructing the initial population. From the collection of scenario beats, the scenarios, in the form of beat sequences, are first randomly generated. Each individual of population is then constructed by assigning the scheduling times for the beats in a given beat sequence. The individual is thus represented as $B_t = (< b_1, t_{b_1} >, .., < b_k, t_{b_k} >, .., < b_m, t_{b_m} >)$, where t_{b_k} is the time, at which the beat k is scheduled in the simulation. Note that the beat scheduling must satisfy the constraint that: $t_{b_k} + \delta_k \leq t_{b_{k+1}}$, where δ_k is the time span of beat k. Within each beat, the scheduling times of the events in the beat are also assigned as $b_k = (< e_1, t_{e_1} >, .., < e_j, t_{e_j} > .., < e_q, t_{e_q} >)$, where t_{e_j} is the time when the event j in the beat k is scheduled, and $t_{e_1} = t_{b_k}$.

Given the initial population being constructed, the population needs to be evaluated and iteratively evolved. To evaluate each individual scenario, we aggregate the MO contribution vector $\overrightarrow{I^{c}}$ of each event in the scenario. Here, instead of a simple linear summation of individual $\overrightarrow{I^{c}}$s of all the events in the scenario, our current design derives the aggregated MO intensity vector $\overrightarrow{I^{a}}$ by considering the impact of the timing of the events. With different timings of the events, we examine how the effect of an event, which is injected at a particular time, can propagate and influence other events. Specifically, each element x_w^{a} in $\overrightarrow{I^{a}}$ for a corresponding MO_w is calculated as:

$$x_w^{\text{a}} = \sum_{i=1}^{r} (P_i + \sum_{j=1}^{i-1} \triangle P_{ij}) \tag{1}$$

where r is the number of the events in the given beat sequence, P_i is used to measure the propagated effect of the event i, $\triangle P_{ij}$ is used to measure how the effect of a previous event j affect the current event i.

To model a mission-based scenario, we consider that the effect of an event can propagate over time, rather than occur instantaneously. The propagated effect P_i of the event i for MO_w is computed as:

$$P_i = \int_{t_{i_s}}^{t_{i_e}} x_i^w f_{\text{p}}^v (t - t_{i_s}) dt \quad (2)$$

where t_{i_s} is the event i's scheduling time, t_{i_e} is the time at which the effect of event i ends, x_i^w is the MO intensity for MO_w in the event i's $\overrightarrow{I^{\text{c}}}$, and $f_{\text{p}}^v(t)$ is the propagation function of a particular event type v. $f_{\text{p}}^v(t)$ is used to describe a propagation curve for the effect of a baseline event, which is of the same type as event i and has the unit intensity. The propagation curve of event i is scaled from $f_{\text{p}}^v(t)$ by the value of x_i^w. Figure 1 shows an example to illustrate the propagated effects of two events with the same event type, with $f_{\text{p}}^v(t)$ modeled as a bell-shaped curve. In practice, we estimate the $f_{\text{p}}^v(t)$ by observing how the effect of the event actually changes in the simulation. For example, for the event type of "add instigators", its effect to the MO of crowd anger control is measured by the number of angry agents caused by instigation. Thus, we measure how the number of angry agents changes over time, when the baseline event of "add instigators" is injected. We use polynomial regression to fit the measured data and transfer into $f_{\text{p}}^v(t)$.

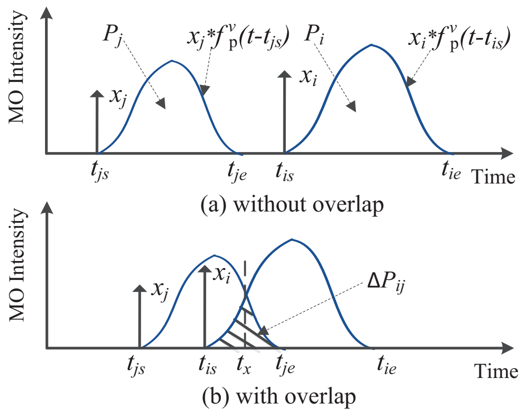

Figure 1: Example of the propagation of the two events' effects, with $f_{\text{p}}^v(t)$ modeled as a bell-shaped curve

In Figure 1(a), the two events j and i are scheduled at t_{j_s} and t_{i_s} respectively. As the distance between t_{j_s} and t_{i_s} is relatively large, the effect of event j ends before the event i is scheduled. Thus, the event j has little impact on the event i in this case. However, in the case that two events are scheduled closely, as shown in Figure 1(b), we consider that the effect of the previous event j can have the impact on the subsequent event i. We model such impact using the overlap area of the two curves in Figure 1(b). $\triangle P_{ij}$, which corresponds to the overlap (shaded) area, is thus computed as:

$$\triangle P_{ij} = \int_{t_{i_s}}^{t_x} x_i^w f_{\text{p}}^v (t - t_{i_s}) dt + \int_{t_x}^{t_{je}} x_j^w f_{\text{p}}^v (t - t_{j_s}) dt \quad (3)$$

where t_{j_s} and t_{i_s} are the scheduling times of the event j and i, t_{j_e} and t_{i_e} are the times at which effects of the event j and i end, t_x is the time when two propagation curves intersect, x_j^w and x_i^w are the MO intensity for MO_w of the event j and i, and $f_{\text{p}}^v(t)$ is the propagation function.

To evaluate the fitness of a given scenario, each aggregated MO intensity x_w^{a} in $\overrightarrow{I^{\text{a}}}$ of the sequence is first derived based on the equations 1-3 as described above. The fitness of the scenario is then measured by the Manhattan distance between $\overrightarrow{I^{\text{a}}}$ and the desired MO intensity vector $\overrightarrow{I^{\text{d}}}$. The $\overrightarrow{I^{\text{d}}}$ is derived based on the trainer inputs of $\overrightarrow{I^{\text{in}}}$ and $\overrightarrow{L^{\text{in}}}$, as described earlier. The GA-based evolution thus aims to find the scenarios with their $\overrightarrow{I^{\text{a}}}$s best matched with the $\overrightarrow{I^{\text{d}}}$. To evolve the population, genetic operations including mutation, cross-over, addition and deletion are performed by 1) alternate the beats' scheduling times, 2) swap the beats in two beat sequences, 3) add beats, and 4) delete beats respectively. The typical GA settings are adopted with cross-over at 0.8, mutation at 0.1, insertion and deletion at 0.05. The evolution process terminates, when the best fitness in the population does not improve substantially for 20 iterations.

The GA-based fitness evaluation operates on a complete potential solution (i.e., a sequence of scenario beats). Thus, other global evaluation heuristics can also be incorporated for assessing the quality of a scenario. For example, the diversity of scenario in terms of distinct beats in the beat sequence can be added as the design criteria for evaluation. Moreover, it should be noted that although the mechanism to handle the causality among the events is not explicitly addressed in our current work, it is feasible to incorporate the causality constraints in the GA-based optimization process (Sorenson et al. 2011). The way to integrate the constraint satisfaction methods with our generation system will be explored in future work.

Experiment

We conducted an empirical study of human players using a simple food distribution training game. In our scenario generation design, we have taken into account of the difference caused by the timing of the events in a scenario, when evaluating candidate scenarios. In the experiment, we compare the scenarios generated by our system and the scenarios not considering the timing issue. By asking players to play different scenarios, we aim to investigate how these two kinds of scenarios can lead to the difference in trainee's performance and thus show the effectiveness of our scenario generation system. Below we describe the design of the training game, experiment procedure, and the results of experiment.

Training Game Design

We implemented a simple food distribution training game, in which the trainee acts as a solider to monitor the distribution process and issue the commands for pacifying the angry agents and arresting the instigators in the crowd. The game is implemented using the MASON multi-agent simulation toolkit[1]. The game interface is shown in Figure 2. It should be noted that this simple game mainly serves as a test-bed for our scenario generation design, rather than a real virtual training system.

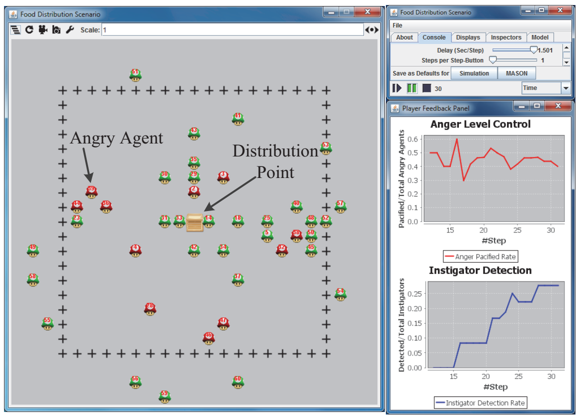

Figure 2: Game interface of the food distribution game

The agents in the game are of two types: civilians and instigators. These two types of agents are differentiable only by their behaviors, not by their appearances. The civilian agents, who are not in the angry state, move towards the distribution point to collect the food. As for the emotional behaviors, the civilian agents are further classified into two types: violent civilian and peaceful civilian. The violent civilians can become angry by self-triggering. The peaceful civilians only become angry passively, when they get "infected" by the surrounding angry agents (i.e., emotional contagion). Unlike civilian agents, the instigator agents always move randomly within the distribution area and purposely toggle between normal and angry state after a certain period of time. All angry agents are shown as red agents.

As a mission-based scenario, the scenario in the designed game is defined with two mission objectives: MO_1 - to control the escalation of crowd angry level, and MO_2 - to identify and arrest the instigators in the crowd. The trainee is given the control to *pacify* the angry agent using the left mouse click and *arrest* the instigator using the right mouse click. The *pacify* action will make an angry agent (civilian or instigator) back to normal state. Once pacified, the civilian may become angry again only if it is surrounded by other angry agents. The instigator, however, will purposely change to angry state again regardless of the states of other agents. The *arrest* action will remove an agent from the simulation, if the agent is an instigator. Otherwise, the agent will become angry, as a penalty for the false instigator identification.

The proposed scenario generation system is used to generate the scenario instances for the game. The scenario mainly includes two types of events: the instigation event (*IE*) - injection of instigators and the violence event (*VE*) - the injection of violent civilians. The number of these two types of events in the simulation greatly impacts how the trainee performs with respect to the defined MOs. The effects of these events are not static, but can propagate over time (e.g., through emotional contagion among the agents). Various scenario beats (35 in total) are created with different combinations of the event instances of two event types. Each scenario instance being generated is a sequence of the scenario beats with scheduling times.

[1]MASON: http://cs.gmu.edu/ eclab/projects/mason/

Experiment Procedure

Thirty participants (i.e., trainees) are recruited to play our food distribution mission game and they are randomly assigned into two groups (fifteen people for each group). Before the experiment, each participant in both groups reviewed a user manual and played the game using a set of sample scenarios. Once familiar with the player control of the game, the experiment was conducted in two phases. In the first phase, each participant in both groups was asked to play a set of pre-generated scenarios. The participant's performances against each MO were recorded, and a simple player model was used for estimating the participant's existing skill levels. The player model adopts the first-order, linear approximation of the trainee's skill level, based on their performances. For each participant, the average performance against each MO was obtained and mapped to a particular performance category. Based on the mapped performance category, the trainee's skill level for each MO is associated with an integer value ranging from 1 to 5.

In the second phase of the experiment, the scenarios were generated under two conditions respectively. We refer to the first condition as "customized", where the scenario for a participant was generated based on the estimated trainee's skill levels, which were obtained from the first phase of the experiment. We refer to the second condition as "rescheduled", where the scenario for a participant was first generated based on the estimated trainee's skill levels and a "rescheduling" operation was then performed on the generated scenario. The "rescheduling" operation randomly alters the injection times of scenario beats in the given scenario and also shuffles the order of some scenario beats. The scenario after "rescheduling" still preserves the same number of scenario beats, as in the corresponding (i.e., with the same inputs) "customized" scenario. In other words, the "rescheduled" scenario is equivalent to the corresponding "customized" scenario if we assumed the timing had no impact on the training.

Each participant in the first group was asked to play a "customized" scenario for five runs, whereas each participant in the second group played a "rescheduled" scenario for five runs. The $\overrightarrow{I^{\text{in}}}$ was set to be the same for generating all the scenarios, with the $\overrightarrow{L^{\text{in}}}$ tailored to each participant. Participants in both groups spent the same amount of training time. To test the training effectiveness of the two groups, we measure the participant's performance gain (PrG) across the runs as:

$$PrG = \frac{P_{curr} - P_{prev}}{P_{prev}} * 100\% \tag{4}$$

where P_{prev} is the performance in the previous run, and P_{curr} is the performance in the current run. For each participant, the average value PrG across the five runs of the game play was derived. With the consideration of the timing the impact in our GA evaluation, we hypothesize that the customized scenarios generated by our system can better tailor the game's challenge to the trainee's skill level. Thus, our research hypothesis is that the average performance gain PrG of the first group of participants using the customized scenarios will be higher than the average PrG of the second group using the rescheduled scenarios.

Results

As the designed food distribution scenario contains two mission objectives, each MO is measured by a corresponding performance measurement. The MO_1 (crowd anger control) is measured by P_{MO_1} = no. of pacified civilians/no. of total angry civilians. The MO_2 (instigator identification) is measured by P_{MO_2} = no. of arrested instigators/no. of total instigators. P_{MO_1} and P_{MO_2} were recorded at the end of each round of the game. The performance gain of P_{MO_1} and P_{MO_2} were then derived using equation 4. For each participant in the two groups, the average performance gain across the five runs of game play was calculated.

Within each group, we divide the participants into four categories based on their estimated trainee's skill levels obtained in the first phase of experiment. Note that even though our skill levels are in the range of 1 to 5, we did not find any participant, whose existing skill levels can map to the skill level 5 for either MO_1 or MO_2. Figures 3 and 4 show the average performance gain for participants in each category. It can be seen from the results that for all categories, the participants playing the customized scenarios (i.e., the first group) have higher performance gains for both P_{MO_1} and P_{MO_1}, as compared with the ones playing rescheduled scenarios (i.e., the second group). Relatively large standard errors may be due to the simple player model being adopted. It can also be observed that for the customized scenarios, the participants with the low-estimated skill level (i.e., category 1 and 2) have higher performance gains, compared to the participants with the high-skill level. This is consistent with the common sense that the novice players usually improve more easily as compared to the advanced players.

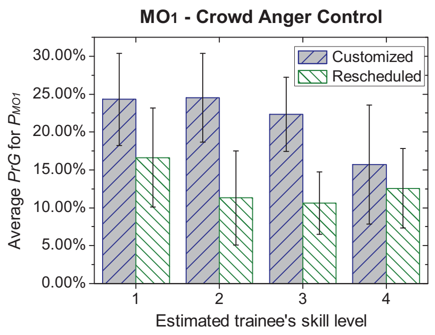

Figure 3: Average performance gain for anger control (P_{MO_1})

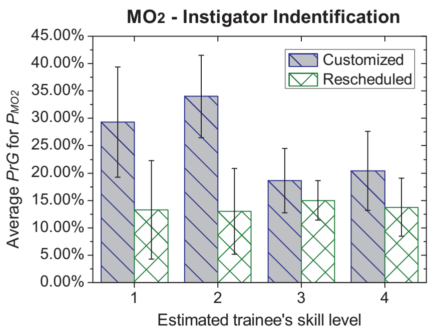

Figure 4: Average performance gain for instigator identification (P_{MO_2})

To test our research hypothesis mentioned previously, the average performance gains from the entire sample (i.e., 15 participants for each group) are calculated. Since the sample data from customized and rescheduled scenarios are independent (i.e., two separate groups of participants), the two-samples one-tailed t-test is conducted. Table 1 shows the average performance gains for the two groups and the p-value from the t-test. The small p-values strongly suggest that our research hypothesis (i.e., the average PrG of the first group playing the customized scenarios is higher than the average PrG of the second group playing rescheduled scenarios) is supported. The results indicate that it is beneficial to model the impact of the dependency among the events of a mission-based scenario in our scenario generation design, as it provides a better estimation of the scenario's aggregated MO intensities.

Table 1: The average PrG for the customized and rescheduled scenarios from the entire sample data

	Customized	Rescheduled	p-value
P_{MO_1}'s PrG	23.25%	11.86%	<0.0005
P_{MO_2}'s PrG	28.53%	15.46%	<0.005

Conclusion and Future Work

In this paper, we have introduced a scenario modeling and generation framework for mission-based scenarios. For generating mission-based scenarios, our framework considers the impact of the timing of the events in a scenario on the training process. We have proposed a conceptual model of mission-based scenario and designed a generation process for generating scenarios that match to both trainer's preferences and trainee's skill levels. Our experiment results have shown that the timing of events can lead to significant difference in trainee's performance and thus it is necessary to capture its effect in our design. We will continue our work by integrating the designed framework with a state-of-the-art human simulation system. The efficacy and scalability of our current approach will be examined in more complex scenarios. The method for MO intensity estimation will be further improved.

References

Aylett, R.; Dias, J.; and Paiva, A. 2006. An affectively driven planner for synthetic characters. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling*, 2–10.

Cavazza, M.; Charles, F.; and Mead, S. J. 2002. Character-based interactive storytelling. *IEEE Intelligent Systems* 17(4):17–24.

Cheong, Y.-G., and Young, R. M. 2006. A computational model of narrative generation for suspense. In *Proceedings of the AAAI Computational Aesthetic Workshop*.

Doran, J., and Parberry, I. 2010. Controlled procedural terrain generation using software agents. *IEEE Transactions on Computational Intelligence and AI in Games* 2(2):111–119.

Hullett, K., and Mateas, M. 2009. Scenario generation for emergency rescue training games. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, 99–106.

Luo, L.; Yin, H.; Cai, W.; Lees, M.; and Zhou, S. 2013. Interactive scenario generation for mission-based virtual training. *Computer Animation and Virtual Worlds* 24(3-4):345–354.

Magerko, B.; Stensrud, B.; and Holt, L. 2006. Bringing the schoolhouse inside the box-a tool for engaging, individualized training. In *Proceedings of the 25th Army Science Conference*.

Martin, G. A.; Hughes, C. E.; Schatz, S.; and Nicholson, D. 2010. The use of functional L-systems for scenario generation in serious games. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*.

Mateas, M., and Stern, A. 2005. Structuring content in the Façade interactive drama architecture. In *Proceedings of the 1st Artificial Intelligence and Interactive Digital Entertainment Conference*, 93–98.

Niehaus, J., and Riedl, M. O. 2009. Scenario adaptation: An approach to customizing computer-based training games and simulations. In *Proceedings of the AIED 2009 Workshop on Intelligent Educational Games*, 89–98.

Reimer, D. J. 2008. Training for full spectrum operations. Technical report, DTIC Document.

Rowe, J. P.; Shores, L. R.; Mott, B. W.; and Lester, J. C. 2011. Integrating learning, problem solving, and engagement in narrative-centered learning environments. *International Journal of Artificial Intelligence in Education* 21(1):115–133.

Shaker, N.; Yannakakis, G. N.; Togelius, J.; Nicolau, M.; and O'Neill, M. 2012. Evolving personalized content for super mario bros using grammatical evolution. In *Proceedings of the 8th Artificial Intelligence and Interactive Digital Entertainment Conference*, 75–80.

Sorenson, N.; Pasquier, P.; and DiPaola, S. 2011. A generic approach to challenge modeling for the procedural creation of video game levels. *IEEE Transactions on Computational Intelligence and AI in Games* 3(3):229–244.

Thue, D.; Bulitko, V.; Spetch, M.; and Wasylishen, E. 2007. Interactive storytelling: A player modelling approach. In *Proceedings of the 3rd Artificial Intelligence and Interactive Digital Entertainment conference*, 43–48.

Togelius, J.; Preuss, M.; Beume, N.; Wessing, S.; Hagelback, J.; and Yannakakis, G. N. 2010. Multiobjective exploration of the starcraft map space. In *Proceedings of 2010 IEEE Symposium on Computational Intelligence and Games (CIG)*, 265–272.

Togelius, J.; Yannakakis, G. N.; Stanley, K. O.; and Browne, C. 2011. Search-based procedural content generation: A taxonomy and survey. *IEEE Transactions on Computational Intelligence and AI in Games* 3(3):172–186.

Yu, H., and Riedl, M. O. 2012. A sequential recommendation approach for interactive personalized story generation. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, 71–78.

Zook, A.; Lee-Urban, S.; Riedl, M. O.; Holden, H. K.; Sottilare, R. A.; and Brawner, K. W. 2012. Automated scenario generation: Toward tailored and optimized military training in virtual environments. In *Proceedings of the 7th International Conference on the Foundations of Digital Games*, 164–171.