A Real-Time Simulation Mechanism for Unlimited Variability of Situations Based on Dynamic Event Coupling and Parameterized Animation against a Background Virtual World

Jun Seong Choi and Jong H. Park

***Abstract*—While the story plots in conventional RPG games are focused on its dramatic development, a system for education based on virtual world pursue diversity and unpredictability. The recent popularity of Open-World games such as Grand Auto Thief 5 over RPG games is due mainly to the fact their users tend to be immersed in the virtual world as its residents The monolithic authoring used in conventional narrative systems is not appropriate for edugames for language learning, our target application, which should provide the users with diverse experience. In an attempt to accomplish story plot variability and to solve this authoring scalability, we propose a situation simulation method based on dynamic coupling among independent events in contrast to monolithically pre-authored situations. An overarching event corresponding to an agent's goal is decomposed and meaningfully connected into a multi-event plan in a search-based planning, which tend to be coincidentally coupled with other agents' plans in their execution against the background world. This simulation is implemented with multiple real-time priority queues corresponding to agents' plans. The actions constituting those queues each are assembled in terms of flexible parameterized motions which allow many actions to be performed in parallel. To reflect the potential discrepancy between the background world conditions and their corresponding knowledges of the agents, our simulation conducts the validity testing and considers those agents' recognition on the preconditions of events in execution. We implement several complex multi-event situations to demonstrate the variability and scalability of situations resulting from our simulation mechanism based on the dynamic event coupling via the background world.**

***Index Terms*—Authoring scalability, background world, dynamic event coupling, situation variability.**

I. Introduction

While the story plots in conventional RPG games are focused on its dramatic development, a system for education based on virtual world pursue diversity and unpredictability [1], [2]. The recent popularity of Open-World games such as Grand Auto Thief 5(GTA 5) over RPG games is due mainly to the fact their users tend to be immersed in the virtual world as its residents [3], [4]. Still, NPC's in Open-World games are not designed to affect the virtual world just like in conventional RPG games, but only their users can make changes to the virtual world [5], [6]. As a result, the events that can happen in the game are rather limited, which weakens sustained immersiveness of its user. The solution of this limitation is crucial for successful edugames for language learning [7]. To pursue variability in event flow, Interactive Storytelling(IS) has developed diverse techniques such as Parameterizing Behavior Tree, Hierarchical Task Network Planning, Motion Repair [8]-[10]. These techniques all author events in a monolithic manner so they need to pre-author every possible flow of the plot along the timeline with all the fixed factors relevant to event development in multi-event situations. Consequently, they suffer from authoring scalability as the situation scale grows along with the problem of limited expression of connections among events. To alleviate this problem, our model employs dynamic coupling among events in contrast to the pre-planned paradigm. That is, the (static) pre-planned models author the events in 'instance' form, but our dynamic model authors in schematic form in reference to the background world. This dynamic approach allows numerous multi-event situations to be generated by instantiating the (small set of) schematic events [11].

The situations in our model unfold by pre-planned and unforeseen events being coincidentally coupled via the background world. Those events are substantiated by instantiating their associated schematic events with the background world factors as the accumulated results of the events that have so far occurred. Those events may be dynamically coupled via their relevant associations such as causality. These couplings generate their results on the background world, which in turn may create another coupling, and recursively. That is, individual events are designed independently and connected later in execution, if necessary, via the background world, avoiding the authoring identical or similar situations. Further, the (changing) relevant factors in situations are naturally referenced via the background world, allowing the author not only to construct the situations without paying extra attention to their contexts but also to pursue unlimited variations at the intersection of background world conditions and event trigger timing. Real-time execution and high action flexibility required to implement multi-event situations in our model is achieved by coupling agents' planned actions based on priority queues. The actions belonging to different plans constitute multi-event situation and arranged in their associated queues corresponding agents. All events in their priority queue are in contention to be animated according to their time-varying priority. Consequentially, variation of priority enables dynamic coupling by interleaved events, requiring re-planning to account for discrepancy between current background world conditions and the agent's world knowledge. The dynamic coupling in our model requires real-time performance and high animation flexibility in the

Manuscript received December 9, 2016; revised March 17, 2017.

The authors are with the School of Electronics engineering, Graduate School, IT College, Kyungpook National University, Daegu, Korea (e-mail: daegulink@naver.com, jhpark@ee.knu.ac.kr).

doi: 10.18178/ijiee.2017.7.2.661

visualization level. The minimum coupling unit of schematic action is instantiated into diverse motions in reference to the background world in the planning phases. That is, it shows animation reflecting changes in the background world regardless of its former motion unlike the conventional monolithically authored animation. This schematic animation mechanism is designed for both resolving the scalability problem of monolithic animation authoring and coping with the uncertainty of each situation being changed according to background world conditions. Our agent model implements not just the dynamic (inter-event) coupling in the event level, but the intra-event coupling according to essentiality of the body parts. In the intra event coupling, an agent extracts two or more motions from multiple actions (e.g., phoning walking) to effectively animate parallel actions providing animation scalability. Combining the techniques above, we propose a new approach to edugame based on virtual world for language learning. We justify its validity by implementing example multi-event situations.

II. Event Planning

Planning is performed based on the planning agent's schematic knowledge on events and background world conditions. The planning is in two phases: horizontal and vertical. The horizontal planning starts with an overarching event associated with the goal, and the event is ramified recursively into subsidiary events along their associated candidate paths. Those paths are dictated by real-life associations such as causality, deonticity, procedurality, etc.. These paths are searched for in the agent's knowledge, and the search is completed successfully only when their associated background world conditions are satisfied. The vertical planning decomposes each event into its associated subsidiary events until every subsidiary event is an atomic event or an action. The overall planning process is as follows:

1) Finding an event with the goal in its effects

Unless the goal is satisfied with the given conditions a case-based search is needed to find events which each can produce the goal situation. Of those candidate events, if any, the best one is selected according to some criterion, and its precondition is identified.

2) Identifying preconditions and effects of events according to their composition

To identify the precondition of an event, those of its subsidiary events need to be identified first. The composition of each event is described in its associated ontology or planner's world knowledge. Since each subsidiary event itself is another event it is to be successively decomposed into its own subsidiary events until they all reduce to actions [12]. As a consequence, this step needs to be invoked every time any new event is derived, so they become interleaved with the other steps.

The precondition and effect of event $A = \prod_{i=0}^{M}(A_i)$ can be computed respectively as:

$$S_P = \bigcup_{i=1}^{M} S_P^i - \bigcup_{k=1}^{M} \overline{S_F^k}$$
$$\overline{S_F} = \bigcup_{i=1}^{M} \overline{S_F^i} - \bigcup_{k=1}^{M} S_P^k \quad (1)$$

where $A_j, A_k \in A$, M = the number of subsidiary events.

3) Adding events as stipulated by association of deontic type, but customary type generally applied to decomposition for vertical planning performed in 2).

4) Searching for the events required to satisfy the goal, and arranging them into a plan.

Once the overall event has been identified in terms of its precondition and effects, the initial overall plan is to be laid out via backward reasoning with respect to the precondition [13]. Specifically, the causality or the premise relation is exploited to deduce the events for a goal in backward search as $\{A \mid S_P \rightarrow | A \cup S_P \Rightarrow A,\ S_F \cap S_G \neq \emptyset\}$. The foregoing search and selection is recursively applied after setting as an intermediate goal each element of the precondition in the plan until all the derived intermediate goals are satisfiable with the average background conditions. At every round of the backward search its respective set of candidate events is deduced shown as the following pseudo code in Fig. 1.

```
GetEvent(Agent_Index, Goal_Index)
{
    If((Current_Situation<-Search(Goal_Index)) = NULL)
        Go to Point_Beyond_Function;
    EndIf
    For(Current_Situation.Essential_Preconditions)
        If(Check_Not_Satisfied(Current_Situation.Essential_Preconditions)
            Get_Event(Agent_Index, Current_Situation.Essential_Precondition_Index);
        EndIf
    EndFor
    For(Current_Situation.Alternative_Preconditions)
        If(Best_One(Current_Situation.Alternative_Preconditions))
            Best_Index = Current_Situation.Alternative_Precondition_Index;
        EndIf
    EndFor
    If(Best_Index ≠ NULL)
        Get_Event(Agent_Index, Best_Index);
    EndIf
    Enqueue(Current_Situation.Actions)
    Return NULL;
}
```

Fig. 1. Pseudo code for planning based on multi-dimensional search.

The initial event is selected as one with its effects containing the goal, and recursively decomposed into subsidiary events with intermediate goals. The planner is given only those situations it can perform against its background (ability, states, etc.) A situation is characterized by the effects of the current situation, the essential and alternative preconditions and their associated actions needed for their animation. To perform a subsidiary event as well, the agent checks its preconditions, essential conditions first.

When any essential precondition is not satisfied it is decomposed into subsidiary events in a further attempt to satisfy it. Unlike an essential one, a set of alternative preconditions are first to be evaluated to select the best one under the current background world conditions. When there is no precondition unsatisfied, those resulting actions are enqueued into the planner's queue in a reverse order of that in their decomposition, generating an instantiated overarching event.

III. Priority-Queue Based Implementation of Dynamic Event Coupling

Our execution mechanism for multi-event situations is based on a priority queue using parameterized action functions. Each queue corresponds to an agent's schedule with events and their associated actions. Event sets from

different events are arranged in the queue according to their priority. Our execution mechanism is composed in three layers, event, action and motion layers, to realize dynamic coupling for action diversity and motion variation. First, in the event layer, (pre-planned or unforeseen) events are ordered according to their inter-event priority, and in the action layer the actions belonging to each event are arranged in their associated event section according to their priority in the planning. This action-level assignment of event into the priority queue allows any event to be uniformly animated in terms of parallel or concurrent actions regardless of whether it is of pre-planned or unforeseen event type [14]. The Motion layer allows an action to be diversely animated using its parameters corresponding to the background conditions. That is, each action is composed of the reusable parameterized primitive motions. The inter-event priority among independent events in a situation (or global) level is observed in a sequential or an interleaved manner with respect to their associated sequences of actions, the intra-event precedence among subsidiary events (or actions) of each overarching event (or plan) as identified and scheduled in its respective intra-event planning is strictly (or temporally) maintained across their associated queues. As a result, the outcome expected with each intra-event planning is guaranteed regardless of the adjusted order due to the inter-event priority unless interrupted by unforeseen events or disruptive conditions. Specifically, the priory of each action in the queue is formulated to reflect both the precedence among the events within an overarching event (or a plan) and the real-world priority among those overarching events (or plans.) Those priorities are rearranged possibly to a new order if additional exogenous events are (recognized to have been) joined to the situation. An active action in a queue becomes inactive if preempted by an event of a higher priority. The action remains inactive until that preemptive event finishes executing, and may be resumed on that event's finishing. Meanwhile, those events whose pre-conditions are not currently satisfied are set to disabled in order not to (allow their agents to intend to) contend with exogenous events for their chance to be executed. Eventually, the actions in those agents' queues are de-queued in order to be visualized, consequently their associated plans being executed in animation often against environmental (natural or social) events. As a result, independent events are arbitrarily interleaved according to their priority leading to coincidental coupling among those events.

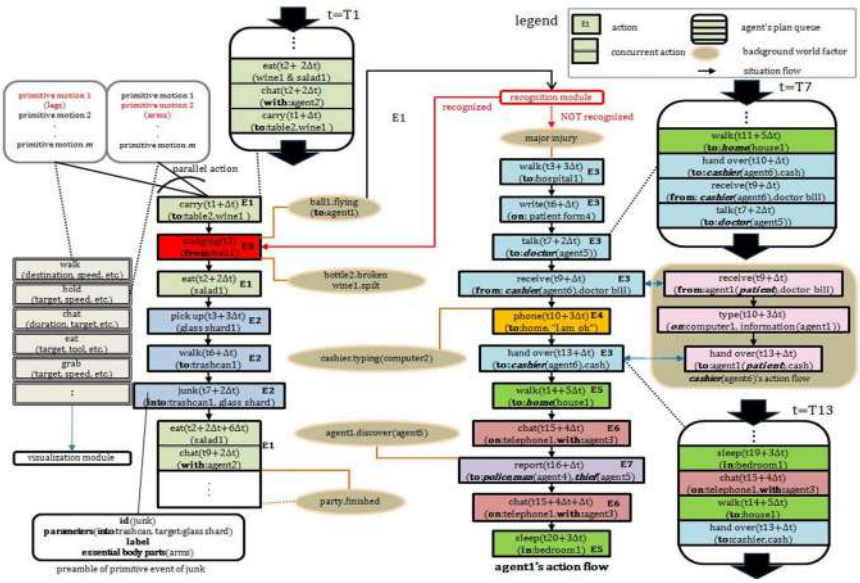


Fig. 2. A Multiple threads of situation progression involving coincidental event coupling via action-level interactions.

We will focus our description of the situation flow in Fig. 2. On those points involving the design issues we have presented so far. The actions Agent1 plans to execute are shown to be arranged according to their planned order in its queue at times T7, T13, and T1. Its plan can be modified anytime according to the associated background-world conditions. In the right action chain, the content of the queue at T7 is not realized as scheduled due to phone(to:home) coincidentally intervening before hand-over(:cash). Still, those actions such as receive(), write(), hand-over(), etc. comprising the event of treatment() or payment() are executed according to their original precedence, producing its planned effects. Agent1 on its way home at T14 could simultaneously conduct the event of 'chatting with Agent3' currently at the third priority in its queue. In this parallel execution instance of two events (i.e., going-home and chatting), the currently-executed action of walk() would be evaluated against all those contending actions (such as phone(), sleep()) in its queue with respect to their precedence and pre-conditions, therefore phone() being selected. Notice that this event-level parallelism considers occurrences of actions instead of schematic motions in the action-level parallelism [14]. The queue content is rearranged at T1 by an emergency event (denoted by a red box) of dodge(:ball1) in reaction to a ball flying toward. This dodge() action is superposed on the carry() to form a parallel action instead of preempting the currently-executed action of carry(), which corresponds to a reflexive reaction of the (human) agent. Further, carry() itself is composed of two primitive actions, walk() and hold(). Each action is animated by assembling the primitive motions pre-authored in a reusable cast [14], which allows parallel actions of one agent to be efficiently animated, e.g., carry() in terms of walk() & hold(). This real-time animation support becomes all the more significant in an

emergency situation, where the variable relative timing of concurring actions is a primary determinant of its subsequent narrative flow. Of course, the animation realism of a parallel action is proportional to how fine-grained their superposition is pre-formulated and animated according to their exact coinciding timings. In general, numerous aspects of the situation are sensitive to (occurrence) timing, adding to non-determinism of situation flow, such that when an event occurs may cause a change in its associated schedule, e.g, intervening at T16 of report(theft()) would cause chat() ongoing from T15 to be suspended until T16+Δt.

The event scheduling in our execution mechanism maximizes the body utilization as humans do, using the information on the body parts provided in the action level. When an agent is waiting for its subsidiary events to finish (e.g., preceding event being performed by another agent) or some body part is free or optional (e.g., arms are optional in 'walk'), he searches his queue for an action whose preconditions are satisfied, and essential parts do not overlap with other actions in order to perform the secondary execution method in parallel or concurrency. This method can give low-priority actions a chance to be executed earlier, maintaining the overall context and achieving efficient event scheduling. These primary execution method and secondary execution method together allow two actions from different events to be combined in an animation level.

As an example combined actions an agent 'walking while eating.' is shown in Fig. 3. To implement a parallel or concurrent event, it is more effective to superpose two actions extracted from different events rather than assembling primitive motions since a new composite action is constructed using pre-existing actions (i.e., pre-defined sequences of motions). This superposition is applicable to any two actions between which each uses its disjoint set of body parts. For example, 'eat' exclusively uses mouth and hand, while 'walk' exclusively uses legs. This principle applies likewise to composite events because each event in our planning is decomposed into actions.

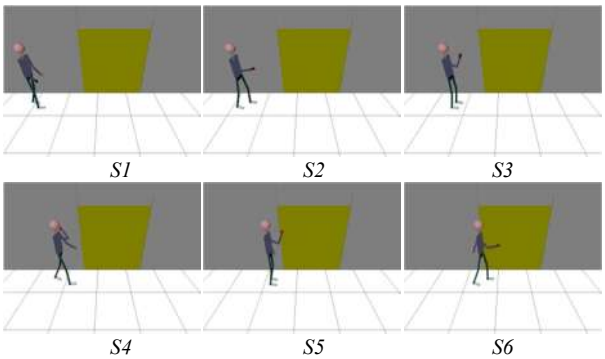


Fig. 3. A parallel action of 'eating while walking'.

IV. Validation Checking and Variability from Limited Recognition

Though an overarching event is executed with all the preconditions satisfied, the preconditions of events in a multi-event situation may change over execution leading to discrepancy between planning phase and execution phase. That is, if the background conditions referenced by the actions have changed from the planning time, re-planning may be needed or some events may need to be cancelled. For this re-planning the agents need to update their knowledge on background world conditions. In case the agent's knowledge has changed to reflect its sensing of the background world, validation check needs to be performed on his corresponding event queue.

In case of invalid conditions, there are two positions re-planning would start according to whether the parameter is essential or not. If the precondition is essential, all its parent's events become inexecutable. If it is alternative it seeks another alternative event in its own line regardless of its overarching event type. That is, since a parent event is dependent on its child event having effects needed for achieving its goal a re-planning is performed a re-planning is performed from the point where an influence of invalid event is not existed. Also, when the agent recognizes those discrepancies determines how long the unnecessary actions leave their impacts on the background world. As the re-planning updates the agent's knowledge through his recognition the situation variability deepens through the discrepancy between the up-to-second world condtions and the agent's knowledge. Notice, the recognition timing as a part of dynamic coupling could significantly affect story flows.

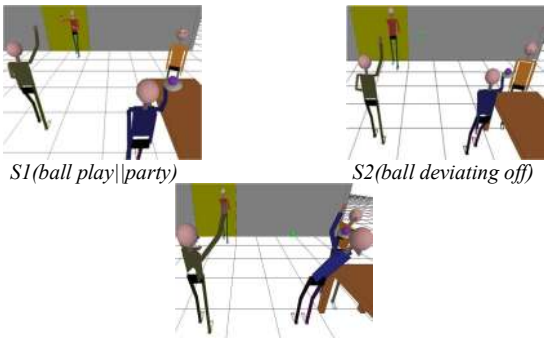


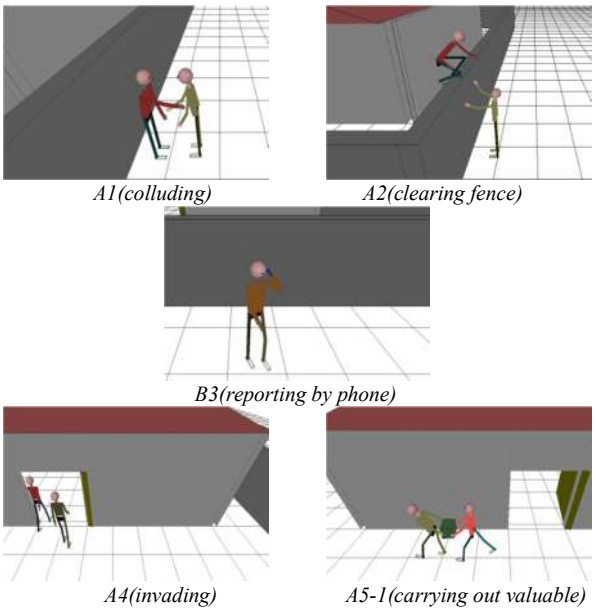
Fig. 4. An Emergency situation involving 'carry' action and reactive action of 'dodge'.

Fig. 4 shows an emergency situation involving two independent events (i.e., party and ball-play) in Fig. 2, both of which an agent plays roles (i.e., cook and victim) in. This emergent situation conditionally occurs only when the cook carrying dish for the party happened to be on a collision course with the stray ball. A reflexive action of dodge() is superposed on the currently-performed action of carry() to make a parallel action on the agent. This situation is implemented using the case-based reasoning scheme and the real-time animation technique [14]. Notice this emergency situation involves all those three focal issues on dynamic event coupling, that is, coincidental interaction between agent and object, each agent's handling of multiple roles, and prompt reaction to abrupt change of an ambient condition.

In Fig. 5, we apply in a comprehensive manner our simulation method to the multi-event situation involving the events of a party, a burglary and a (potential) police arrest. We demonstrate in animation how these independently-planned events can be dynamically inter-coupled to generate countless, many unforeseen, situation flows without pre-authoring every flow variation. All the possible scenes are animated in terms of a small set of

basic actions such as 'phone', 'talk', 'walk', 'grasp', 'push and 'climb′. Further, each agent plays different roles in different situations, such as Agent1(party host, victim, reporter), Agent2(thief, suspect, escapee), Agent3(thief, suspect, escapee) and Agent4(policeman, arrester, chaser) along with props like phone, door and valuable, where bold Italic indicates roles. Of those possible flows, a few would progress along A1, A1→A2, A1‖B3→A2→A4→C6-1, A1→A2→A4→A5-1‖B3,

A1→A2‖B3→A4→A5-1→C7-1→C7-2, where ‖ indicates parallel occurrence; A, B and C indicate independent events; i of Ai-j indicates a chronological order of the sub-event within an event, and j indicates order within the sub-event Ai. All these variations depend on the conditions of those agents and the rest of the background world. To name a few, a crucial condition to dictate the flow in the example situation is the 'report' event with respect not only to whether or not that event is initiated, but also to when it is performed. In fact, its occurrence time could be anywhere along the entire progression of this multi-event situation, e.g., during A1, before A2, long after A5-1, etc. each leading to a different flow. If it has been initiated at all, its progression may not be as expected due to various unfavorable conditions, and accordingly its results would vary non-deterministically. In general, each sub-event in any flow could go awry for diverse reasons, e.g., the phone Agent1 uses to report could be malfunctioning, creating new background-world conditions. A successful completion of the theft (along a flow up to A5-1 unrestrainedly) could result from a number of different conditions, e.g., Agent2 or Agent3 is not detected by Agent1 in the first place, or Agent4 was too far away to get to the scene in time. Conversely, the successful sequence of sub-events for the theft could be severed any time before its completion to divert to another situation flow. If Agent4 or any other policeman happened to be nearby on a patrol, the police could stop the theft in a collusion phase, and the flow would be as short as A1 (surely, this sub-event itself is intricate enough to ramify into a number of different flows.)



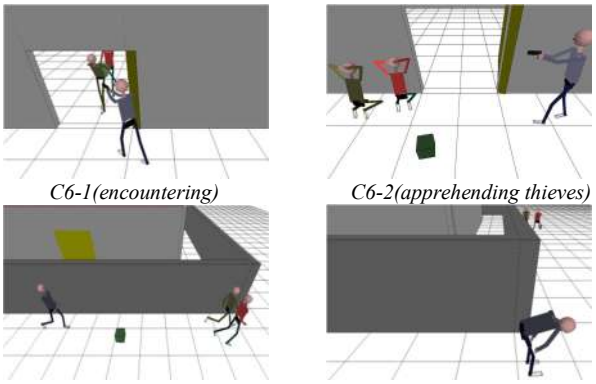


Fig. 5. Branched sequence of scenes due to 'theft'.

V. Flexible Animation Model

Our multi-event simulation cannot be realized without an efficient real-time animation technique in action and motion levels. In conventional IS systems, the actions and motions are authored monolithically so they are not interruptible [6], [15]. In contrast, our model employs reusable primitive motions to assemble complex actions and parameterizes motions via the background world in the event level to enhance animation variation. The parameterized motion is designed to proceed to achieve the goal as a result of parameterizing regardless of the agent' current motion state. The motion of a body part is formulated as follows:

$$\prod_{n}(f : g_i \rightarrow m_i) \tag{2}$$

where Π denotes a temporal sequence, g denotes an angle of a joint, and m denotes a primitive motion.

A primitive motion refers to a movement involving only one joint, and a composite motion refers to one involving two or more joints. Many primitive motions may be assembled into a new motion. A sequence of motions constitutes an action with a specific goal. A primitive motion in an action is defined as a phase, and a set of parallel primitive motions constitute a composite phase. That is, the progression of a simulation related to actions is formulated as:

$$S(t+\Delta t) = s(t) + \prod_{i}^{M}\left(\sum_{j}^{N} m_j^i, \mathrm{A}_S \supset m_j^i\right) \tag{3}$$

where S denotes a state, m denotes a motion, A denotes an action, M denotes the number of phases and N denotes the number of motions in each phase, ∏ denotes a phasic development and Σ denotes a set of parallel motions.

Our model classifies the motions into intra-object and inter-object types according to whether their goal is motion itself or change in situation. The intra-object action like walk() including an intra-object motion is animated along a fixed trajectory. The primitive motions corresponding to its associated body parts are processed simultaneously in a parallel processing, after dynamically instantiated based on the parameter information in the action level. The inter-object action like catch() is realized by connecting the body parts to the goal based on the kinematic constraints (e.g.,

part length, permitted angle range) using the body part associated with its given termination condition (e.g., hold for catch()). That is, the inter-object action pattern is not executed by simply performing the primitive motions like in intra-object action, but following the trajectory from the current position of its associated body part to goal point designated in the termination condition. An inter-object action also can accommodate dynamically-changing background world conditions by re-targeting in real-time changing terminating conditions [16]. The actions used in parallel or concurrent event as described in Fig. 3 may be combined regardless of their types. We use a locking mechanism (with polling technique) to evaluate the parallelism among actions. Parallel and concurrent execution of actions are possible under the following conditions:

$$\begin{aligned} &If \quad P(\prod_{i=0}^{N} a_i) \cap P(\prod_{j=0}^{N} a_j) = \{\}, \\ &T(\prod_{i=0}^{N} a_i) \cap T(\prod_{j=0}^{N} a_j) \neq \{\} \rightarrow parallel \\ &If \quad P(\prod_{i=0}^{N} a_i) \cap P(\prod_{j=0}^{N} a_j) \neq \{\}, \\ &T(\prod_{i=0}^{N} a_i) \cap T(\prod_{j=0}^{N} a_j) \neq \{\} \rightarrow concurrent \end{aligned} \tag{4}$$

where a denotes action belonging to the agent's capability, P() denotes the body parts used in, and T() denotes time span of execution.

The difference between action and event levels of parallel (or concurrent) execution is similar to that between action and event in general. Since their composition is of a sequential set of actions, events cannot be juxtaposed for parallelism if those actions being combined are not in their turns (according to their planned precedence) or their preconditions conflict each other.

VI. Conclusion

We have developed a simulation method for realistic situations to provide abundant pedagogical experience. This simulation method affords very high variability of simulated situations involving many events, still achieves authoring (or implementation) scalability. Instead of planning for a story plot composed of pre-planned and monolithically-authored static situations along a single storyline, our inter-event planning method aims at simulating diverse situations each involving multiple concurrent events. Those events may belong to different storylines in their common ever-changing background world and be dynamically coupled via real-world conditions and inter-event associations. This dynamic and indirect event coupling not only practically resolves the scalability problem due to monolithic authoring, but enables originally-independent events to be coincidentally linked with each other during their execution time, naturally allowing unforeseen events to become part of an emergent situation without being pre-planned [17]. To realize dynamic coupling we implement two-dimensional (horizontal & vertical) planning in which events are meaningfully connected by the association rules and the background world conditions. To support an efficient implementation of our approach, we devise a real-time execution scheme based on multiple priority queues of animated actions particularly to visualize concurrent progression of many simultaneous events into a coherent situation. The inter-event priority among independent events in a situation (or global) level is observed in a sequential or an interleaved manner with respect to their associated sequences of actions, the intra-event precedence among subsidiary events (or actions) of each overarching event (or plan) as identified and scheduled in its respective intra-event planning is strictly (or temporally) maintained across their associated queues.

Our model achieves the animation variability by employing the motions as the atomic units to progressively animate composite motion, parallel and concurrent actions, and finally events.

All these techniques combined, diverse exogenous events can be derived and integrated coincidentally (still naturally) with the original events to form a wide range of emergent situations, enriching chances of interesting pedagogical experiences. These advantages fulfill the objective in our simulation of realistic multi-event situations only at an expense of a limited visual realism. Our planning method also suffers, if a much lesser degree than conventional narrative systems, from authoring burden in casting reusable actions in as flexible a form as to be applicable to all conceivable situations.

As a future research, we first need to develop a precise mechanism for coupling events via entities in the sense that each entity itself is modelled to be an active (and complex) concept. Our planning method also suffers, if a lesser degree than conventional narrative systems, from combinatorial explosion of authoring undertaking in constructing its background world.

References

[1] S. Marsella, W. Johnson, and C. LaBore, "Interactive pedagogical drama for health interventions," in *Proc. of AIED 2003*, Australia, 2003.

[2] A. Zook *et al*., "Automated scenario generation: toward tailored and optimized military training in virtual environments," in *Proc. the International Conference on the Foundations of Digital Games*, ACM, 2012.

[3] J. Kettunen, "Constructing identities in San Andreas: Characterizing the protagonists in Grand Theft Auto V," University of Jyvaskyla, Aug 2015.

[4] By Kyle Eric Perkins, "Successful storytelling in open-world games," Ohio University, Nov. 2010.

[5] N. Sgouros, G. Papakonstantinou, and P. Tsanakas, "A framework for plot control in interactive story systems," in *Proc. AAAI-96*, AAAI Press, 1996.

[6] M. Kapadia, S. Singh, G. Reinman, and P. Faloutsos, "A behavior-authoring framework for multiactor simulations," *Computer Graphics and Applications*, June 2011, pp. 44-55.

[7] S. D. Sigurðardóttir, "The use of games in the language classroom," April, 2010.

[8] A. Shoulson *et al.*, "Parameterizing behavior trees," *Motion In Games*, Springer, 2011, pp. 144-155.

[9] K. Erol, "Hierarchical task network planning: Formalization, analysis, and implementation," Dissertation, Univ. of Maryland, 1996.

[10] F. Charles *et al.*, "Planning formalisms and authoring in interactive storytelling," in *Proc. of TIDSE*., vol. 3, 2003.

[11] R. Aylett, M. Vala, P. Sequeira, and A. Paiva, "FearNot! - An emergent Narrative approach to virtual dramas for anti-bullying education," in *Proc. International Conference on Virtual Storytelling*, pp. 202-205, 2007.

[12] J. Park, "Formalization of cyber-microcosm," Tech. Report #99, AIMM Lab., KNU, 2013.

[13] M. Young, M. Pollack, and J. Moore, "Decomposition and causality in partial-order planning," *AIPS*, 1994.

[14] J. Choi and J. Park, "An effective implementation of agent's complex actions by reusing primitive motions," in *Proc. Simultech*, Vienna, Austria, 2014.
[15] D. Weld, "An introduction to least commitment planning," *AI Magazine*, vol. 15, no. 4, 1994.
[16] K. Choi and H. Ko, "On-line motion retargeting," *Journal of Visualization and Computer Animation*, vol. 11, pp. 223-235, 2000.
[17] M. Cavazza, F. Charles, and S. J. Mead, "Emergent situations in interactive storytelling," in *Proc. ACM Symposium on Applied Computing (ACM-SAC)*, Madrid, Spain, 2002.

Jun Seong Choi received his B.S. degree in electronics engineering from Kyuongpook National University, Korea in 2014. Currently, he is a PhD course student, in Kyungpook National University, Korea. His research interest is artificial intelligence, specifically animation techniques in multi-event situations and real-time simulation.

Jong Hee Park acquired his PhD in computer engineering from Univ. of Florida in 1990. He has since been on the Faculty at Kyungpook Nat'l Univ. in Korea. His research interests focuses on intelligent systems based on cyber-worlds. The specific research topics encompass intelligent tutoring system for immersed language learning, futuristic simulation games, and simulation of cyber-worlds for diverse applications.