Intelligent Narrative Technologies and Social Believability in Games:
Papers from the AIIDE 2015 Joint Workshop

# Aesthetic Interleaving of Character Performance Requests

**Daniel Shapiro, Larry LeBron, Andrew Stern, Michael Mateas**

Computational Media, University of California at Santa Cruz
dgs at ucsc.edu

**Abstract**

We have constructed a system that supports unscripted social interaction between a player and virtual characters, where the participants pursue internal agendas and respond to one another in real-time. Our emphasis on *unscripted interaction* means that the characters must accept dynamically generated performance requests, while our concern with *social interaction* implies that the characters must interleave performances with an attention to natural flow that encourages social engagement. We present initial work on a performance management mechanism that produces this interleaving. It initiates and suspends character performances by allocating animation resources to requests via a utility function representing aesthetic concerns. That function weighs extrinsic factors reflecting the purpose of taking an action against intrinsic ones that concern features of a given performance. We show, via multiple short videos, that the features are individually material to the aesthetic quality of the result and that the mechanism can produce aesthetically pleasing performances on par with the best hand-generated prioritization scheme. We argue, anecdotally, that the parameters of the model are easy to identify, suggesting that the feature vocabulary is both intuitive and useful for shaping character performances.

## Introduction

Every system that supports human interaction with virtual characters must interleave actions that respond to player input, pursue character intent, and potentially advance a scenario. However, these motivations can be in conflict. Consider a moment within an experience where a character could deliver a line to advance the narrative, answer the player's question, or display irritation at the player's interruption consistent with the character's type. These actions are in tension at the level of character or designer intent, and in hard conflict over animation resources.

Most gaming systems never frame such alternatives as a meaningful, automated choice. Instead, they require authors to tightly control interactions by identifying and encoding a single next action into the character's behavior sequence. Game engines provide a variety of formalisms for expressing the requisite reasoning and control flow, including rules, scripting languages, and behavior trees, Hierarchical FSMs, choice points, and detailed transition logic. However, these techniques all pass the burden of interleaving externally motivated and internally motivated character action to authors of the interactive experience.

In contrast, we consider an approach that delegates interleaving decisions to an automated system. It calls on the author to identify a set of actions the character should perform in the current game state, and relies on an algorithm to sequence them appropriately, taking into account the importance of the action to the narrative, the character, and to the player's interactive experience. This approach has several potential benefits; it requires less programming effort through delegation of the interleaving task, it weakens the constraints on allowed player input common in interactive experiences, and it supports more emergent flow by generating performance sequences instead of encoding a branching scenario in advance. The potential downside is a loss of authorial control, which we address through inclusion of a tradeoff function.

While delegating the interleaving task has broad benefits, it is especially relevant in our application context, which is to employ whole body interaction with virtual characters to train social skills (Shapiro et al 2015; Shapiro et al. 2013). This application stresses the interleaving task. First, social characters must perform many small communication tasks, such as brief glances, acknowledgments, and attention shifts, and fold them into ongoing activities. More broadly, the characters must exhibit a consistent personality and pursue a social agenda, which generates a stream of internally motivated behaviors. Our context also demands a high degree of player control over scenario flow, as social discourse is inherently less constrained than other, function-oriented interactions (such as navigation or combat). In addition, whole body interaction produces many gestural signals that translate into a stream of externally motivated demands for NPC response. Finally, the training objective imposes an application purpose for char-

acter actions, e.g., to deliver content or create situations that afford learning, which impacts the importance of behavioral choices. These aspects of social interaction made it natural for us to adopt a generative stance, where characters independently nominate social moves, the player and other NPCs cause further actions to be nominated, and the actual sequence is resolved via a prioritization method. Here, the overriding goal is to interleave actions with an attention to natural flow that facilitates social engagement with the player.

This paper presents initial results of a priority-based method for interleaving action requests. We consider a resource-constrained set of animations and dialogue tasks and employ a weighted linear combination of features to determine action priorities. We allocate animation resources to actions in rigid priority order. The features reflect the reasons for taking action as well as the properties of a given performance, while the net ranking is designed to generate smooth player experiences. We show that:

- No fixed, content-insensitive execution policy will reliably generate high quality performances.
- A factored priority model can produce performances equivalent to the best hand-selected behavior sequence.
- Each feature of the model is relevant, and material to the quality of the resulting performances.

In addition, we present anecdotal evidence that the parameters of the model are easy to determine, suggesting that the model offers an intuitive and useful feature base.

Given that the evaluation is fundamentally subjective, we employ a micro-scenario to set context, and multiple videos of performance sequences generated by the model to substantiate these claims. The following sections discuss related work, the micro-scenario and underlying simulation, the performance management algorithm and priority model, and nine experiments on model parameters. We conclude by discussing wider implications of the work.

## Related Work

Very few systems for interacting with virtual humans delegate performance-shaping decisions. On the surface, this seems odd, since the factors that shaped our approach are quite common; players routinely impose demands for action on virtual characters, those characters simultaneously pursue internal agendas, and authors want, in parallel, to shape the interaction to achieve narrative objectives. Our focus on social interaction highlights the need for automatic interleaving, but the problem exists at some level across game and virtual character systems at large.

SIMS 4 contains a multi-tasking system that is possibly the closest in spirit to our work (Ingebretson and Rebuschatis 2014). It addresses a similar problem of selecting short performances from a set of possible steps, e.g., to grab a drink, watch TV, sit, or read a book, where the actions carry substructure, preconditions, and resource constraints. However, interleaving in SIMS 4 concerns multitasking subject to constraints, vs. sequencing actions for aesthetic purposes. It does not evaluate action choices relative to narrative purpose, character intent, or performance readability, as in our work.

Work on abstract control of animations is a close cousin to the type of delegation we discuss in this paper. For example, SmartBody (Thiebaux et al. 2011; Feng, Xu, and Shapiro 2012) provides authors with a markup language for describing performances that it transforms into character animation and synchronized audio. It also automates actions to reach, grasp and touch arbitrary objects in a dynamic space. In contrast, we focus on externalizing the criteria that shape good performances vs. abstractly specifying the performance content.

More generally, work on interleaving concerns performance blending. SmartBody lets authors merge the influence of parallel controllers on the same virtual body joints (Kallmann and Marsella 2005), while Unity, and other virtual character engines support related blending curves. The Assassin's Creed games rely heavily on blending to maintain the momentum of a player character as it climbs on surfaces or jumps through trees (Laidacker and Dumas 2013). While this work directly addresses aesthetic performance goals, it emphasizes blending across animations associated with known sequential states, vs. generating those sequences by interleaving performance requests.

Utility functions have been a staple of game programming for many years (Mark 2009) Here, the goal is to automate some aspect of behavior generation via a tradeoff function, but our application to sequencing a known set of performance requests seems new.

In summary, utility-based choice is common in game AI, linear models are well explored, and the desire to build smooth user experiences is ubiquitous. Given that the elements of our approach are individually available, our framing of this task in terms of delegation is possibly the main contribution of our work.

## The Give-Gift Experience

Our work on social interaction with virtual characters has produced several interactive experiences of some depth, but we have extracted a micro-scenario for use in this paper (as shown in Figure 1). It contains two NPCs, Yunos, and Basuki, who each have a social agenda, and a player who interjects a request at a critical time. In overview:

- Yunos attempts to give a piece of fruit to the player, which he steals from Basuki. When the player fails to respond (a given in this example), Yunos takes offense.

*Figure 1. A Snapshot of the Give-Gift Interaction*

- Basuki senses his fruit is stolen. He reacts with irritation towards Yunos.
- 1 second after Basuki becomes irritated, the player shows the NPCs a photo. Both NPCs respond to the player's implicit request to identify the person involved.

We have uploaded a ~30 second video illustrating a high-quality run-through of this interaction [1]. This paper explores a number of variants of this sequence, generated by different prioritization of the performances involved, and by altering the timing of the player's intervention.

## Social Simulation

A full description of the simulation that generates these performances is outside the scope of this paper, although its basic elements are relevant here. It contains a Kinect-based gesture recognition system that outputs "embodied events" (like *extendHand* and *pointAtHand*), a perception process that abstracts these events into social signals (such as *requestAnswer* and *physicallyDisengaged*), an intent formation process that models the resolution of character motivations into individual actions, and a bottom-up response process that maps player-generated social signals into choice points for character action. The simulator casts these actions as "Social Interaction Units" (SIUs), each of which implements an isolated social purpose that spans 2 - 10 seconds of clock time, and supports interaction with other characters or the player. While SIUs can call one another, they are composed from a collection of performance behaviors that invoke individual animations. SIUs are written in a behavior-oriented language (Mateas and Stern 2002).

In a bit more detail, Yunos' fruit theft performance is generated by the SIU, GiveGift:ObjectTransaction(Yunos, Player, Fruit). This SIU invokes performances to locomote to the object, acquire it (e.g., via theft), offer it (in this case to the player), wait on the player's response (here, to ignore the offer), and express Yunos' reaction to that response (i.e., to take offense). Basuki's irritation is initiated by the SIU, RequestStepBack(Yunos, fruit), which encompasses Yunos' reaction to that social communication. The player's action to show the photo motivates calls to additional performance behaviors that implement brief glances and more thorough displays of attention.

From the perspective of the performance management system, each of these invocations corresponds to a performance request that arrives at an unpredictable time, and must be interleaved with ongoing performances as animation resources permit. Table 2 identifies the complete list of performance behaviors utilized by this scenario.

## Performance Management

In the scenario described above, NPCs, acting as virtual humans, pursue multiple social interactions at the same time. The associated behaviors can sometimes be performed in parallel. However, if actions conflict over animation resources (body parts), they must be performed in sequence over time, in an order that depends upon their computed importance. The performance manager imposes this priority structure, and mediates among performance requests to choose an interleaving. It implements this task:

**Given:**
- a set of {performance, resource requirement} requests
- a subset of those requests executing at the current time
- a decision opportunity, defined as the arrival of a new request, or the completion/termination of a prior request

**Choose:** a set of requests to execute, and a set to suspend.

The algorithm imposes a strict, priority ordered resource allocation scheme. We discuss the algorithm, followed by our method of assigning priorities to performance requests to reflect aesthetic criteria, below.

### Priority based resource allocation

A performance request needs to control some subset of the virtual character's body in order to execute. We have defined nine body resources: BaseFullBody (legs, spine), ArmLeftLayer, ArmRightLayer, HeadTrackingLayer, SpeechLayer, HeadLayer, FacialExpressionLayer, BodyPosture, and FeetPosAndBodyHeldWeight.

The more "efficient" a performance can be, i.e. the fewer body resources it requires to express its meaning or accomplish its task, the more likely that other performances will be able to mix in and enact simultaneously (due to the absence of resource conflicts). Our algorithm distinguishes two types of resources: necessary, and optional, where necessary resources are required for execution and optional resources are simply nice to have. Behavior authors annotate every performance with its required resources, optional resources, a timeout of how long to wait for resources before giving up, and whether to retry if interrupted. For example a brief performance to acknowledge a player's interruption (i.e., turning to look and saying "hmm?") receives the following annotations:

```
performanceWME.requires(baseFullBody);
performanceWME.requires(headTracking);
performanceWME.requires(speech);
performanceWME.setTimeout(3000);
performanceWME.setBRetry(true);
```

Given this data, the resource allocator executes these steps:

1. Place all new/active performances in descending priority order. Break ties by placing the oldest performance first. Delete suspended performances that have timed out.
2. Iterate down the list of priority-sorted performances. For each request, claim all of its required resources if available and mark it as performable. Otherwise claim none.
3. Consider each performable request in priority order. Assign it all requested, available optional resources.
4. Delete any performance that has lost its required resources, and was marked to not retry if interrupted.
5. Execute all performable performances, in parallel.

On completion of this algorithm, it is possible that (a) a new performance will not receive required resources, and will be queued until such time as they appear, (b) an existing performance will lose its required resources and will be suspended, or deleted if marked not to retry, (c) a suspended performance will time out, and never run, (d) a performance will start or restart execution, and (e) many performances will receive resources, and subsequently execute in parallel. At least one performance (the first on the list) will receive all of its required resources.

### Linear prioritization

We define the priority of performance requests via a multi-attribute utility function that combines features of interest. We employ a linear function of the form, $v = W \cdot F$, where W is a vector of weights in the range (0 , 1), and F is a vector of features, also normalized to the (0, 1) range.

Before examining the structure of F, it is important to clarify the purpose of the value function; v exists as input to the performance manager for use in interleaving performance requests. So, v(W,F) must select for readable performances and select against unnatural/jarring ones.

We have engineered two classes of features into the vector F; extrinsic features capture the purposes a performance serves, while intrinsic features capture properties of an individual performance relevant to smooth/readable flow. We define nine features, as shown in Table 1.

We incorporate extrinsic features into the priority function following the intuition that, given the choice, characters (and people) will select actions that are more vs. less important to their intentions. Conversely, we assume that observers will interpret character actions as windows into their intent, and perceive goal-consonant behavior as more natural than less goal-relevant choices. The same comment extends to extrinsic features beyond character utility; we want character choices to communicate the fact that the selected performance has dramatic value, scenario import, or application significance (e.g. within a tutoring system).

| Extrinsic Features | Intrinsic Features |
|---|---|
| Character Utility of the action | Performance Continuity |
| Dramatic value of the action | Performance Urgency |
| Scenario importance of the action | Performance Duration |
| Application value of the action | Autonomic content |
| | Performance Inertia |

*Table 1. Features of the Performance Prioritization Model*

Among the intrinsic features, Continuity captures relevance to previous action, i.e., if the performance is a part of a multi-step sequence. The intuition is that natural performances tend to exhibit unity of intent over time. Urgency reflects the time available for initiating action if it is to read well (it makes no sense to raise your hand to catch a ball if the ball has flown by). Duration measures the time required to execute the action. The intuition is that observers will expect characters to perform short/immediate actions vs. delay those actions until longer activities complete. To simplify the problem of predicting an interactive performance's length, we take the need for locomotion as a surrogate for long duration. Duration itself is a surrogate for some sense of efficient flow (e.g., to pick up an object while near the table, vs. walk away and return).

The Autonomic quality of a performance reflects the degree to which the action is involuntary. An emotional tirade is high on this scale. The intuition is that observers will find it natural to see involuntary communications interrupt many voluntary ones. Note that a high Autonomic performance could have high or low Duration (a tirade vs. a single grimace), and high or low Urgency (a fight or flight reflex vs. a smile in response to a kind word).

Finally, the Inertia of a performance describes the extent to which an action, once initiated, should not be interrupted if it is to read well. Walking is an example, as it would feel unnatural for a character to stop and start a movement multiple times. An emotional tirade is similarly awkward to suspend and resume, so it has high Inertia as well.

## Aesthetic Experiments

Our work examines the conjecture that we can delegate interleaving decisions to an automated system and produce aesthetically pleasing results. Our experiments to investigate this claim all rely on subjective assessments of character performance videos, and as there can be no gold standard for this evaluation, we have posted the results on-line, and invite readers to view the videos. Note that we are concerned with automatic generation of nuanced performances, so some variations of interest are subtle.

| Performance Behavior | Urgency | Duration | Autonomic | Inertia |
|---|---|---|---|---|
| acquireObject_approach | 0 | 1 | 0 | .5 |
| acquireObject_pickUp | 0 | 0 | 0 | .5 |
| requesterRequestStepBack | .5 | 0 | .5 | 1 |
| requesteeRequestStepBack | .5 | 0 | 0 | 0 |
| offerObject_giver | 0 | 1 | 0 | .5 |
| rejectOfferedObject_giver | .5 | 0 | .5 | 1 |
| showObject_showee_initialGlance | .5 | 1 | 0 | .5 |
| notComplyWithRequest_requesterRequestStepBack | .5 | 0 | .5 | 1 |
| notComplyWithRequest_requesteeRequestStepBack | .5 | 0 | 0 | 1 |
| showObject_showee | .5 | .5 | 0 | 0 |

*Table 2. Intrinsic Feature Values*

In more detail, we varied the feature weights of the linear prioritization model, and examined the impact of those changes on the perceived performance quality of the "Give gift" micro-scenario. To do so, we annotated the underlying performance behaviors with sensible, fixed intrinsic feature values, and annotated each performance request with a reasonable, fixed, aggregate value for extrinsic features. Table 2 illustrates the annotations involved. Note that the value of Continuity is determined from context; it is set to high if the SIU calling this performance was itself called by another SIU. Of these behaviors, the first supports locomotion to the fruit, or towards the photo. Yunos employs Object_pickUp to take the fruit. Basuki employs requesterRequestStepBack to express his initial irritation, while the Yunos calls the requestee version to perform his initial reaction (a glance). The notComplyWithRequest actions perform the detailed back and forth of their argument. This naming convention should clarify the meaning of the remaining behaviors.

The following material presents nine experiments on this model that demonstrate our claims regarding the centrality of a content-based action selection policy, and the power of our delegation model to generate aesthetic performances.

## Fixed selection strategies

We executed the Give-Gift scenario with simple queue-processing policies in place of a content responsive prioritization model. As shown in [2], the First In First Out policy creates non-responsive characters; Yunos ignores Basuki's irritation (time 0:03), and executes his agenda to offer fruit to the player. He only responds to Basuki's irritation (at time 0:18, with the phrase "it's nothing" in Esperanto) when that sequence completes. Similarly, Yunos fails to react to the player's request to identify the image in the photo (represented by the raised, ghost hand holding a picture), since too much time has passed before he's free to perform the reaction. In contrast, the Last In First Out policy [3] creates highly distractible characters; here, Basuki's complaint and Yunos' reaction to it get interrupted mid-sentence to respond to the photo (time 0:06). Their interaction never completes for repeated distractions.

By extension, any fixed interleaving policy can fail to respond to an important request, or complete some important agenda item. As a result, the interleaving policy must be content-responsive (whether or not it is priority based).

## High quality interleaving

One of our prior demonstrations of social, virtual characters included the Give Gift micro-scenario as a component part. That work categorized performance requests into fixed priority tiers as input to the resource allocation algorithm described above, and produced the best available hand-specified performance sequence. We chose weights in the linear model to duplicate those ordinal values in numeric terms, resulting in a nearly identical sequence of actions with the same performance quality. This is the base case referenced earlier [1]. Since many vectors in a 9-dimensional space can produce this effect, we leveraged the feature semantics to choose values. This made it easy to identify appropriate weights.

This weight vector, and associated performance serves as the base case for the remaining experiments, which perturb individual feature weights. That is, we increase a factor weight to 1, or decrease it to 0, and analyze changes in the quality of the resulting performance.

## Impact of Extrinsic and Intrinsic Features

We annotate each performance request with a single number that summarizes its external importance. Actions that respond to the player have a non-zero extrinsic feature value. When the weight on extrinsic features is elevated relative to the base case, the characters treat the player as overly important [4]. For example, Basuki and Yunos interrupt their argument mid-sentence to look at the photograph, as

in the LIFO policy (although they are able to return to the argument at time 0:12, here). This excessive attention to external purposes is unreasonable even if the player were King, implying that extrinsic features are material to performance quality as a whole.

We examined the impact of each intrinsic feature via similar perturbation experiments. Elevated Continuity weight favors the completion of performance sequences at the expense of responsiveness [5]. The result here is virtually identical to the FIFO video, because Yunos' full fruit offering sequence spans so much of the interaction. Yunos ignores the player's request to ID the photo (at time 0:01), and Basuki's complaint (at time 0:04), and only belatedly responds to Basuki's complaint after the player's refusal to accept the fruit becomes clear (time 0:18). Elevated Urgency weight favors reactionary performances [6]. In this video, Yunos begins the non-urgent fruit theft sequence (at time 0:00). However, both NPCs promptly react to the player's urgent request to ID the photo (from time 0:02 to 0:05), which interrupts the fruit theft and complaint sequence until these initial glances complete (at time 0:06). The loss (if any) in perceived performance quality is small, although, once again, the characters have difficulty pursuing longer term agendas. Elevated Inertia weight favors performance completion without interruption [7]. In the beginning of this video, Yunos steals the fruit, but Basuki performs his glance at the photo before reacting to the fruit theft, which is clearly unreasonable.

Reduced Duration weight removes the priority advantage given to short (non-locomoting) performances. We illustrate this effect by comparison to a version of the base case scenario where the player asks Yunos and Basuki to identify the photograph at the inception of the interaction [8]. Here, Yunos experiences a choice between stealing the fruit and looking at the photograph. He chooses to pick up the fruit, in part because that action is short (stationary) while the photo investigation is long (it involves walking towards the player). With Duration weight set low in the same scenario [9], Yunos chooses to examine the photo instead. Yunos' interjection, "ah" (at time 0:02) is the first step of this performance. He takes a step towards the player and gives the photo an initial glance (time 0:03). Next, he resumes the theft sequence, which requires taking a step towards Yunos (time 0:04). This results in somewhat jittery motion that affects performance quality, presumably because the character interrupts a short task to pursue a longer one (where the reverse is more natural).

Finally, a reduced Autonomic feature weight causes characters to be jarringly non-responsive [10]. To illustrate this case, we added two actions to the scenario; the player can rush forward at the NPCs, and the NPCs have a startle reaction with a high Autonomic feature value. In this setting, the NPCs repeatedly fail to respond to the player's rapid movement. They only become startled after the third trial, once the competing interaction between Yunos and Basuki over the fruit theft has terminated.

## Conclusions and Future Work

Our work on performance management supports an approach to unscripted interaction with virtual characters that relies on satisfying dynamically generating performance requests. This framework is novel with respect to the gaming industry, and most virtual character research. Given this commitment, our use of a value function to generate aesthetically pleasing performance sequences is straightforward. Our preliminary experiments have shown that a linear model composed of extrinsic and intrinsic features has the expressive power to generate nuanced performances, and that our specific features are material to the perceived quality of the resulting behavior. We have attempted to show that the feature vocabulary is at least reasonably intuitive through multiple examples. That said, we have not demonstrated completeness or independence of the feature set, and we have not tested the potential of this approach to scale up. Future work should examine whether behavior authors find it easy or hard to control the degrees of freedom present within the linear model.

An alternate perspective is that the extrinsic and intrinsic feature sets represent a natural vocabulary for describing performances, where the feature values are determined by performance content and purpose. In this view, authors delegate aesthetic choices to the performance manager by defining a tradeoff function via a vector of weights. If the feature set is adequate, and the weights reflect author preferences, the resulting behavior is by definition good. From our experience, authors will want the ability to override system decisions in specialized contexts, but the approach of delegating the vast majority of such decisions could easily scale. Authors might also choose to program via the weight vector, e.g., to generate a nervous character by instilling a tradeoff function that selects for impatient and distractible performances. Multiple changes to the weight vector could communicate mood shifts over time.

Overall, our work argues that it is feasible to delegate performance sequence decisions to an automated system, which in turn can facilitate a more abstract approach to authoring interactive experiences with virtual characters. Future work should test the generality of the delegation model, and its ability to scale. Given that the Give-Gift scenario was extracted from a social interaction experience 10 times its size, the obvious next step is to employ the linear prioritization method throughout the enclosing scenario. Annotating the underlying performance behaviors will require some effort, but the process will expose potential gaps in the feature base, and test the feasibility and

desirability of employing a delegation model to sequence large bodies of behavior.

## Acknowledgments

We gratefully acknowledge contributions by our colleagues at Raytheon-BBN Technologies, and SRI International at Princeton. This work was sponsored by the DARPA Strategic Social Interaction Modules program, under U.S. Army Research Office contract W911NF-12-C-0002. The contents of this paper are the responsibility of the authors, and do not necessarily reflect the position or the policy of the US Government. No official endorsement should be inferred.

## Notes

1. BaseCase https://www.youtube.com/watch?v=dnDlXpxJyNM&feature=em-share_video_user

2. FIFOpolicy https://www.youtube.com/watch?v=RWOCuWjwchM&feature=em-share_video_user

3. LIFOpolicy https://www.youtube.com/watch?v=Ifd604Ifvaw&feature=em-share_video_user

4. HighExtrinsicWeight https://www.youtube.com/watch?v=c4Opv7IJtyk&feature=em-share_video_user

5. HighContinuityWeight https://www.youtube.com/watch?v=_FApky5m24E&feature=em-share_video_user

6. HighUrgencyWeight https://www.youtube.com/watch?v=a-oE4ShCu-U&feature=em-share_video_user

7. HighInertiaWeight https://www.youtube.com/watch?v=EjPSTLBtDpw&feature=em-share_video_user

8. EarlyPhotoShow https://www.youtube.com/watch?v=6pfB5uer430&feature=em-share_video_user

9. LowDurationWeight https://www.youtube.com/watch?v=mB3ndoEVuWk&feature=em-share_video_user

10. LowAutonomicWeight https://www.youtube.com/watch?v=1oGBt2urd2A&feature=em-share_video_user

11 Feng, A. W., Xu, Y., and Shapiro, A. (2012). *An Example-Based Motion Synthesis Technique for Locomotion and Object Manipulation*, Symposium on Interactive 3D Graphics and Games, Costa Mesa, CA, March 2012

## References

Feng, A. W., Xu, Y., and Shapiro, A. 2012. An Example-Based Motion Synthesis Technique for Locomotion and Object Manipulation, Symposium on Interactive 3D Graphics and Games, Costa Mesa, CA, March 2012

Ingebretson, P., and Rebuschatis, M. 2014. Concurrent Interaction in SIMS 4, GDC Vault: http://www.gdcvault.com/play/1020190/Concurrent-Interactions-in-The-Sims

Kallmann, M., and Marsella, S. 2005. Hierarchical Motion Controllers for Real-Time Autonomous Virtual Humans, in *Proceedings of the 5th International Working Conference on Intelligent Virtual Agents* (IVA), 2005, pp 243-265.

Laidacker, A., and Dumas, R., 2013. AI Postmortems: Assassin's Creed III, Game Developer's Conference 2013, GDC vault http://www.gdcvault.com/play/1018058/AI-PostmortemsAssassin-s-Creed

Mark, D., 2009. Behavioral Mathematics for Game AI. Course Technology, a part of Cengage Learning, ISBN-13: 978-1-58450-684-3

Mateas, M., and Stern, A. 2002. A behavior language for story-based believable agents. *IEEE Intelligent Systems,* 17(4), 39–47.

Shapiro, D., Tanenbaum, K., McCoy, J., LeBron, L., Reynolds, C., Stern, A., Mateas, M., Ferguson, B., Diller, D., Moffit, K., Coon, W., and Roberts, B. 2015. Composing Social Interactions via Social Games, International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS) 2015, Istanbul, Turkey.

Shapiro, D., McCoy, J., Grow, A., Samuel, B., Stern, A., Swanson, R., Treanor, M., and Mateas, M., Creating Playable Social Experiences through Whole-body Interaction with Virtual Characters 2013. *Proceedings of the Ninth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (AIIDE-13), Boston, MA.

Thiebaux, M., Marsella, S., Marshall, A. N., and Kallmann, M. 2011. SmartBody: behavior realization for embodied conversational agents, Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems, 2011.