Proceedings, The Eleventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-15)

A Lightweight Algorithm for Procedural Generation of Emotionally Affected Behavior and Appearance

Yathirajan Manavalan[1] and **Vadim Bulitko**[1] and **Marcia Spetch**[2]
Departments of Computing Science[1] and Psychology[2]
University of Alberta
Edmonton, Alberta, T6G 2E8, CANADA
{yathi.bm|bulitko|mspetch}@ualberta.ca

Abstract

Displaying believable emotional reactions in virtual characters is required in applications ranging from virtual-reality trainers to video games. Manual scripting is the most frequently used method and enables an arbitrarily high fidelity of the emotions displayed. However, scripting is labour intense and greatly reduces the scope of emotions displayed and emotionally affected behavior in virtual characters. As a result, only a few virtual characters can display believable emotions and only in pre-scripted encounters. In this paper we implement and evaluate a lightweight algorithm for procedurally controlling both emotionally affected behavior and emotional appearance of a virtual character. The algorithm is based on two psychological models of emotions: conservation of resources and appraisal. The former component controls emotionally affected behavior of a virtual character whereas the latter generates explicit numeric descriptors of the character's emotions which can be used to drive the character's appearance. We implement the algorithm in a simple testbed and compare it to two baseline approaches via a user study. Human participants judged the emotions displayed by the algorithm to be more believable than those of the baselines.

1 Introduction

Emotionally affected behavior of virtual characters in video games and training environments can significantly impact the user's experience. In movies and books characters are often remembered by their emotional reactions. We expect James Bond to be confident and suave, Master Yoda to be mysterious and wise and Sherlock Holmes to be calm and rational. Similarly, the witty and narcissistic antagonist GLaDOS from the *Portal* video game series (Valve 2007) stays with the player long after the game is over. The emotional reveal by Pagan Min at the climax of *Far Cry 4* (Ubisoft 2014) brings a resolution to the player's struggle during the forty-some hour campaign. The cool and collected acknowledgment "Moving to position" of the non-playable special-force companions in *Tom Clancy's Ghost Recon 2* (Ubisoft 2004) becomes the strained "Move accomplished!" as the game's combat heats up. And how can one forget the gleeful "Yest, Kapitan!" of a battle station officer acknowledging a torpedo launch aboard a Russian attack submarine in *Sub Command* (Strategy First 2001)?

Similarly, virtual reality trainers have long attempted to incorporate emotionally believable characters in their training simulations. In the early nineties, the US Navy IDCTT trainer used live acting recordings to create the emotionally charged atmosphere of damage control aboard a ship under attack (Johnson 1994). In the mid two thousands, an Iraqi police captain Farid, a non-playable character in a virtual reality trainer by the Institute for Creative Technologies, would be offended by the player's inappropriate chitchat (Solomon et al. 2008).

As the graphical fidelity of virtual characters have increased substantially, the gap between their realistic appearance and unrealistic behaviors has widened. This is in part due to two trends currently seen in the video game industry. First, modern video games are moving towards giving the player more agency. While most commercial video games still do not allow the player to free type like *Façade* (Mateas and Stern 2003) or use their voice like *The Restaurant Game* (Orkin and Roy 2007), more options are becoming available (e.g., the interaction wheel of the *Mass Effect* series). Second, even the traditionally story-oriented studios such as BioWare are making their games more open-world (Campbell 2014) where the player is actively encouraged to roam massive worlds and interact with hundreds of AI-controlled virtual characters in them (Pramath 2015).

Combined, the two trends result in a progressively larger number of AI-controlled non-playable characters (NPCs) the player can interact with and a larger number of ways to do so. As a result, even with tens of millions of dollars in development budgets, it is impossible to hand-script emotionally affected reactions and appearances of each character in response to each possible way the player may interact with them. Thus, a handful of characters may be fully scripted for the specific ways the player is allowed to interact with them whereas hundreds of others will deliver generic one-line reactions to actions the player takes toward them. The discrepancy breaks immersion and reminds the player that she is playing a game. For instance, Amita, a key story character in *Far Cry 4*, becomes a generic in-world NPC after the single-player champaign ends and the game transitions to an endless open-world simulator. Previously

Copyright © 2015, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

talkative and emotional, Amita now idly walks around the world, cycling through generic animations, mute and oblivious of her prior interactions with the player. While merely immersion-breaking in a video game, such lack of realism may be intolerable in a VR trainer (Traum et al. 2003).

We propose to address the content bottleneck by procedurally generating both emotionally affected actions and emotional appearance descriptors of computer-controlled characters in video games and virtual reality trainers. Procedural generation is a promising approach with advances made in level generation (Valtchanov and Brown 2012), NPC population (Booth 2009) and gameplay difficulty adjustment (Hunicke and Chapman 2004). Furthermore, work exists on visualizing emotions via procedurally animated facial expressions (Nvidia 2013) and body language (Desai and Szafron 2012). In this paper we implement and evaluate a recent algorithm that can procedurally generate emotionally charged NPC actions and explicit emotional descriptors for visualization. While procedural models of emotional reactions have been explored over the past few years, this algorithm is intentionally lightweight for an easy integration in video games and trainers, especially running on mobile hardware and consoles.

The rest of the paper is organized as follows. We formulate the problem in Section 2 and follow with a discussion of related work in Section 3. We present details of our implementation of the algorithm in Section 4. We then present the results from a user study comparing the algorithm against two baseline models in Section 5. We discuss the limitations of the algorithm and possible future work Section 6.

2 Problem Formulation

Our problem is to procedurally generate NPC emotions that can be used to drive the NPC appearance and behavior. The criterion of success is whether such displays of emotions are deemed believable by a human observer. As more and more games are moving towards the open-world model with hundreds to thousands of AI-controlled characters, an ideal solution to the problem will be computationally lightweight, allowing a number of NPCs to be driven simultaneously on modest hardware such as game consoles. Additionally, the procedural methods should minimize character authoring time and, ideally, be accessible to a non-programmer.

3 Related Work

There have been three primary approaches to achieving emotionally believable artificial characters in video games and other settings. The first is manual scripting, where the designers script the actions, facial expressions and dialogue of a virtual character for every possible in-game interaction. As a result, the designers have complete control on how the virtual character is going to behave and emote in each scenario. This can yield realistic movie-like experience but requires a substantial amount of time and effort on the part of the designers. Consequently, most video games and virtual training environments necessarily limit the number of characters the player/trainee can interact with and/or the scope of the interactions available. For instance, BioWare's *Mass Effect* series (Bioware 2007) constrains the player's actions with a dialogue wheel and quicktime events and only fleshes out the primary characters in specific encounters. Other characters tend to utter generic one-liners and behave less believably when the player attempts to interact with them. Even the primary characters exhibit emotionally unbelievable behavior beyond pre-scripted encounters: the implausibly repetitive "Wrex. Shepard." dialogue from *Mass Effect* has become an Internet meme (Linguica 2008).

The second approach replaces hand-scripted emotional NPC responses with a procedural model that takes in the context and the player's actions and outputs a numerical descriptor of the NPC emotions. A few representatives of this approach use an appraisal model of emotions which postulates that emotions arise from the character's appraisal of the current state with respect to the character's goals (Scherer, Schorr, and Johnstone 2001). For instance, when achieving a desirable goal is uncertain the character may experience hope. A looming undesirable state elicits fear and so on. A well known computational implementation of this approach is *EMotion and Adaptation* (EMA) (Gratch and Marsella 2001; 2004a; 2004b; Marsella and Gratch 2009) which can be expensive to run and daunting for game developers to implement at its full scale. A lightweight subset of EMA has been proposed (Bulitko et al. 2008) but it can only produce emotional descriptors (e.g., the amount of fear an NPC is feeling) and not the related coping actions.

The third approach forgoes procedural generation of emotion descriptors and instead generates emotion-related actions. For instance, an NPC fearing for its life may run away, causing the player to *attribute* the emotion of fear to it (Isla 2005). The psychological theory of resource conservation (Hobfoll 1989) suggests that character actions stem from protecting and gaining resources that the character deems valuable. Computational implementations of this approach (e.g., COR-E (Campano et al. 2013) and *Thespian* (Si, Marsella, and Pynadath 2005)) have been proposed. While computationally lightweight and less daunting to embed into a game, COR-E does not generate explicit emotional descriptors and therefore is unable to drive facial (MOVA 2009; Nvidia 2013) and non-facial (Desai and Szafron 2012) NPC emotional displays or modulate the NPC voice (Black and Lenzo 2003).

4 Implementing ACORE

In this paper we adopt a recent combination of COR-E and EMA known as ACORE (Manavalan and Bulitko 2014). Similarly to COR-E, ACORE uses a representation of resources valuable to the NPC. This allows ACORE to generate actions that the player can interpret as the NPC's displays of emotions (e.g., fleeing from danger may be attributed to fear). Like CEMA and unlike COR-E, ACORE also uses an appraisal model of emotions to compute explicit emotion descriptors. For instance, if a valuable resource is threatened, ACORE will generate a numeric value for the emotion of fear which can be used to drive the NPC appearance and even color a display of the protective action.

To make this paper more self-contained, we will now algorithmically describe ACORE. In-game non-playable characters, NPCs, are controlled by Algorithm 1. Lines 1 through 3 initialize the NPCs' data. Then, as long as the game is not over (line 5), we model each NPC's action selection by invoking the resource conservation module $\text{ACORE}_{\text{action}}$ (line 7, detailed in Section 4.2). We then compute NPC's emotional descriptors by invoking the appraisal module $\text{ACORE}_{\text{emotion}}$ (line 8, detailed in Section 4.3). Resource values are updated using the actions of this and other NPCs (line 9). Emotion descriptor values decay over time (line 10).

Algorithm 1: ACORE

```
1  for each NPC do
2      initialize: resources r̄_1, resource weights w̄_1
3      initialize ē_1 = (0, 0, 0, 0)
4  t ← 1
5  while game running do
6      for each NPC do
7          act: a_t ← ACORE_action(r̄_t, w̄_t, A_t)
8          display emotions: ē_{t+1} ← ACORE_emotion(r̄_t, w̄_t, ē_t)
9          update resources: r̄_{t+1} from r̄_t, a_t and other NPCs'
           actions
10         decay the emotion values: ē_{t+1} ← ē_{t+1}/2
11     t ← t + 1
```

4.1 Resources

ACORE associates a set of valuable resources (e.g., health, reputation, gold) with each NPC. Mathematically, each NPC has a vector of resources, which at time t is denoted by $\bar{r}_t = (r_t^1, \ldots, r_t^N)$ where each $r_t^i \in \mathbb{R}$ is a scalar representing the value of i-th resource at time t. The NPC assigns different importance or value to each resource and this is represented by N-dimensional weight vector $\bar{w}_t = (w_t^1, \ldots, w_t^N) \in (0, 1]^N$. A higher weight indicates higher importance of the resource to the NPC. Generally speaking, the weights may change over time but remain in $(0, 1]$.

To illustrate we will walk through the algorithm on a concrete example. Consider an NPC standing in a line of other NPCs, waiting to purchase a product (Figure 2). Each of the NPCs has three resources: health, reputation and rank, which is inversely proportional to the NPC's place in line. Assume that at some time t, the resources of our NPC are $\bar{r}_t = (1, 1, 0.5)$. Suppose the NPC values the health resource at 0.26, the reputation resource at 0.17 and the rank resource at 0.44: $\bar{w}_t = (0.26, 0.17, 0.44)$. We will now run the NPC step-by-step through ACORE.

4.2 Computing NPC Actions

Each NPC has a set of actions (called *behaviors* in COR-E) available to it at time t, denoted by A_t. The actions can be primitive or complex, consisting of a control policy. Each action can affect the resources the NPC holds (e.g., getting into a fight may negatively affect the NPC's health). NPCs select actions that are expected to increase their cumulative weighted resource value (lines 2 and 3 of Algorithm 2).

The value of an action is the sum of the resource value deltas weighted by the resource weights, conditional on the

Algorithm 2: $\text{ACORE}_{\text{action}}$

inputs : current resources $\bar{r}_t$, resource weights $\bar{w}_t$, available actions A_t
outputs: selected action: a_t

```
1  for a ∈ A do
2      compute action value: V(a) ← w̄_t × (E[r̄_{t+1}|a] − r̄_t)^T
3  select action: a_t ← arg max_{a∈A_t} V(a)
```

action. In our example, an NPC standing in line has two actions available for it: $A_t = \{a^{\text{pass}}, a^{\text{wait}}\}$. By taking the a^{pass} action, the NPC will attempt to pass the NPC standing in front of it, thus improving its rank in the line but possibly losing reputation or even health (if the NPC being passed physically protests the pass). Suppose that by passing the NPC expects its resources to change from $\bar{r}_t = (1, 1, 0.5)$ to $E[\bar{r}_{t+1}|a^{\text{pass}}] = (0.95, 0.6, 1)$. Weighting the expected delta $E[\bar{r}_{t+1}|a^{\text{pass}}] - \bar{r}_t$ by $\bar{w}_t = (0.26, 0.17, 0.44)$ the NPC computes the value of passing as $V(a^{\text{pass}}) = 0.139$ (line 2).

An alternative action is to wait in line which does not immediately change the resource vector. Hence $V(a^{\text{wait}}) = 0$. Then, in line 3, the NPC will select the action with the maximum value: $a_t = a^{\text{pass}}$. Note that a different set of resource weights (e.g., $\bar{w}_t = (0.5, 0.5, 0)$ for an NPC that cares about its health and reputation but not about its place in the line) would have made the value of the passing action negative and led to the NPC selecting the wait action instead.

4.3 Computing NPC Emotions

ACORE simplifies the emotion model of EMA (Gratch and Marsella 2001; 2004a) by assuming that each NPC has only a single goal: to improve its weighted resource values. The appraisal process is thus limited to considering changes in the resource vector. In line with CEMA (Bulitko et al. 2008), ACORE models only four emotions: hope, joy, fear and distress. At time t, the NPC represents its emotional state as a four-dimensional vector $\bar{e}_t = \left(e_t^{\text{joy}}, e_t^{\text{hope}}, e_t^{\text{distress}}, e_t^{\text{fear}}\right)$ where each e_t represents the *intensity* of the corresponding emotion. The initial emotion vector $\bar{e}_1$ is $(0, 0, 0, 0)$, set in line 3 of Algorithm 1.

The intensity of each emotion with respect to a resource is computed as the product of the NPC's desirability α of the change in the resource caused by the action selected by $\text{ACORE}_{\text{action}}$ (Section 4.2) and the certainty of the change β. The desirability is the product of the resource weight and the expected change in the value of a resource caused by the action (line 3, Algorithm 3). The certainty of the change is the probability of the change in the direction specified by α (lines 5 and 7). A desirability of zero leads to no changes to the agent's emotions.

The total intensity of each emotion is the sum of its intensity for each resource i (the loop in line 2). Which emotion gets the update depends on the relation between the desirability and certainty. Desirable but uncertain changes in a resource contribute to hope (lines 10 and 11), desirable and certain changes contribute to joy (lines 8 and 9). Likewise, undesirable resource changes contribute to fear (when they are uncertain, lines 14 and 15) or distress (when they are certain, lines 12 and 13).

Algorithm 3: ACORE$_{\text{emotion}}$

inputs : current resources $\bar{r}_t$, resource weights $\bar{w}_t$, selected action a_t, current emotion $\bar{e}_t$

output: emotion $\bar{e}_{t+1} = \left(e_{t+1}^{\text{joy}}, e_{t+1}^{\text{hope}}, e_{t+1}^{\text{distress}}, e_{t+1}^{\text{fear}}\right)$

1 start with the current emotion $\bar{e}_{t+1} \leftarrow \bar{e}_t$
2 **for** resource index $i = 1, \ldots, N$ **do**
3   compute desirability: $\alpha \leftarrow w_t^i \cdot (E[r_{t+1}^i | a_t] - r_t^i)$
4   **if** $\alpha > 0$ **then**
5   compute certainty: $\beta \leftarrow \Pr(r_{t+1}^i > r_t^i \mid a_t)$
6   **else**
7   compute certainty: $\beta \leftarrow \Pr(r_{t+1}^i < r_t^i \mid a_t)$
8   **if** $\alpha > 0$ & $\beta = 1$ **then**
9   compute joy: $e_{t+1}^{\text{joy}} \leftarrow e_{t+1}^{\text{joy}} + \alpha$
10   **else if** $\alpha > 0$ & $\beta < 1$ **then**
11   compute hope: $e_{t+1}^{\text{hope}} \leftarrow e_{t+1}^{\text{hope}} + \alpha \cdot \beta$
12   **else if** $\alpha < 0$ & $\beta = 1$ **then**
13   compute distress: $e_{t+1}^{\text{distress}} \leftarrow e_{t+1}^{\text{distress}} - \alpha$
14   **else if** $\alpha < 0$ & $\beta < 1$ **then**
15   compute fear: $e_{t+1}^{\text{fear}} \leftarrow e_{t+1}^{\text{fear}} - \alpha \cdot \beta$

We will now illustrate the operation of ACORE$_{\text{emotion}}$ with specific numbers, continuing our example from the previous section. Starting with the current emotion values $\bar{e}_t$ (line 1), our NPC computes an update to its new emotional state $\bar{e}_{t+1}$ for each resource in the loop in line 2. Suppose the current value of the NPC's health is $r_t^{\text{health}} = 1$. Under the action $a_t = a^{\text{pass}}$ selected by ACORE$_{\text{action}}$ in Section 4.2, the health resource is expected to decrease ($E[r_{t+1}^{\text{health}}|a^{\text{pass}}] = 0.95$) due to a possible physical opposition from the NPC being passed. The decrease in health in undesirable because health is positively weighted ($w = 0.26$). Thus, $\alpha = w \cdot \left(E[r_{t+1}^{\text{health}}|a^{\text{pass}}] - r_t^{\text{health}}\right) = 0.26 \cdot (0.95 - 1) = -0.013$.

The certainty β of the undesirable health decrease is then computed as $\Pr\left(r_{t+1}^{\text{health}} < r_t^{\text{health}} \mid a^{\text{pass}}\right)$ (line 7) which, generally speaking, depends on the action the NPC being passed will take. We model this in two stages. At the first stage we assume that our NPC has not yet observed the actions of other NPCs and thus uses a prior for the certainty values. We compute the resulting changes to the emotion model which can then be visualized with the NPC's appearance. In stage two, the NPC has observed the actions of the affected NPCs and updates its emotion model accordingly. The emotional appearance can then be visualized once again.

For the health resource before the NPC is able to observe actions of the NPC being passed, it uses the prior $\beta = 0.95$.[1] With this value, the conditions in line 14 will be satisfied and the intensity of emotion fear will be updated as $e_{t+1}^{\text{fear}} \leftarrow e_{t+1}^{\text{fear}} - \alpha \cdot \beta = e_{t+1}^{\text{fear}} - (-0.013) \cdot 0.95 = e_{t+1}^{\text{fear}} + 0.0123$ in line 15. In other words, our NPC is now slightly more afraid of the prospects of losing some of its health.

[1] In our implementation of ACORE we hand-coded expected resource changes and certainty values (Table 1).

The for loop in Algorithm 3 will then consider the next resource, reputation. The reputation is also predicted to decrease under the pass action: $E[r_{t+1}^{\text{reputation}}|a^{\text{pass}}] = 0.6$ because cutting in front of other people in line is disreputable thing to do. Our NPC cares about its reputation ($w = 0.17$) and so $\alpha = w \cdot \left(E[r_{t+1}^{\text{reputation}}|a^{\text{pass}}] - r_t^{\text{reputation}}\right) = 0.17 \cdot (0.6 - 1) = -0.068$. The prior probability of this undesirable loss of reputation is $\beta = 0.95$ which adds an extra $-\alpha \cdot \beta = -(-0.068) \cdot 0.95 = 0.0646$ to the emotion of fear.

The final resource is the rank in line whose current value is $r_t^{\text{rank}} = 0.5$ (i.e., the NPC is second in line). Under the pass action, the new value is expected to be $E[r_{t+1}^{\text{rank}}|a^{\text{pass}}] = 1$. Our NPC cares about its rank ($w = 0.44$) which means that $\alpha = w \cdot \left(E[r_{t+1}^{\text{rank}}|a^{\text{pass}}] - r_t^{\text{rank}}\right) = 0.44 \cdot (1 - 0.5) = 0.22$. The prior probability of this desirable gain of rank is $\beta = 0.95$ which adds $\alpha \cdot \beta = 0.22 \cdot 0.95 = 0.209$ to the emotion of hope (lines 10 and 11). Thus, at the end of the first stage, the NPC's fear is increased by $0.0123 + 0.0646 = 0.0769$ and the hope is increased by 0.209. The new emotion vector is then visualized via the NPC appearance.

Suppose then the NPC being passed protests the pass. Thus, in stage two, the passing NPC observes the protest and updates its α and β values. Retrieving the values from Table 1, ACORE computes the new increase of fear as $-\alpha \cdot \beta = -(-0.013) \cdot 0.5 = 0.0065$.

In stage two, the change in the reputation resource remains the same leading to $\alpha = 0.17$. However, since the NPC is now aware of its pass action being protested, the expectation that the reputation will decrease is reduced to $\beta = 0.5$. Thus, the stage-two increase in fear due to possible change in reputation is $-\alpha \cdot \beta = -(-0.068) \cdot 0.5 = 0.034$.

For the rank resource, the expected value of the resource change remains the same as in stage one which leads to $\alpha = 0.22$. The certainty is now reduced to $\beta = 0.5$. Thus at stage two the NPC's hope is increased by $\alpha \cdot \beta = 0.22 \cdot 0.5 = 0.11$. Thus, at the end of the second stage, the NPC's the NPC's fear is increased by $0.0065 + 0.034 = 0.0405$ and its hope is increased by 0.11. The new emotion vector is visualized. It should be noted that the β values for certainty of change in Table 1 are for the moment in time the actions are taken. Once the outcome of an action is known the β of a change can be 1. For instance, once an NPC has successfully passed another NPC, it will experience joy since its position in line (the rank resource) has improved.

5 Evaluation

We evaluated our implementation of ACORE with a user study in which we compared ACORE with COR-E and a random emotion model as baselines.

5.1 Experimental Testbed

Adapting the scenario from COR-E studies (Campano et al. 2013) we considered a line of six people waiting to purchase a video game (Figure 2). Each person in line is represented by an NPC controlled by ACORE and has three resources: health, reputation and rank. Each person has two basic actions available to them at each time step: pass the person in

Table 1: Domain dynamics.

Resource r_t	**Action** a_t	**Expected resource change** $E[r_{t+1}\|a_t] - r_t$	**The other NPC's action**	**Certainty of change** β
health	pass	-0.05	wait	0.95
reputation	pass	-0.4	wait	0.95
rank	pass	$1/(1/r_t^{\text{rank}} - 1) - r_t^{\text{rank}}$	wait	0.95
health	pass	-0.05	protest	0.5
reputation	pass	-0.4	protest	0.5
rank	pass	$1/(1/r_t^{\text{rank}} - 1) - r_t^{\text{rank}}$	protest	0.5
health	protest	-0.05	pass	0.5
reputation	protest	-0.1	pass	0.5
rank	protest	$1/(1/r_t^{\text{rank}} + 1) - r_t^{\text{rank}}$	pass	0.5
health	wait	0	wait	0.95
reputation	wait	0	wait	0.95
rank	wait	0	wait	0.95
health	wait	0	pass	0.5
reputation	wait	0	pass	0.5
rank	wait	$1/(1/r_t^{\text{rank}} + 1) - r_t^{\text{rank}}$	pass	0.5

Figure 1: Neutral expressions of the 18 individuals whose portraits were used in the user study.

front of them (if he/she exists) or wait in line. Additionally, a person being passed can choose to protest the passing action or allow it to happen (i.e., continuing waiting in line).

At every time step, the person at the head of the line purchases the video game and leaves the line. The simulation was stopped when the line becomes empty. Visually, each person in line was represented with a photograph showing their facial expression (only the highest intensity emotion was shown; Figure 3), their name and the three resources. Health was visualized with a bar underneath the image. The reputation was shown by the color of frame around their portrait. The rank was shown by the position of the person in line (Figure 2). Additionally, people in line uttered one-line remarks shown as text above their heads (Table 2).

There were eighteen individuals (Figure 1) from which a line of six was randomly populated (without repetition).

5.2 The Experiment

For the user study we recruited 94 participants (30 males, 64 females; mean age 20). For their participation, each participant received a partial credit in an undergraduate psychology course. After a briefing and signing a consent form, each participant was exposed to each of the following conditions.

Figure 3: An individual showing hope (top left), joy (top right), fear (bottom left) and distress (bottom right).

Experimental condition (E) presented a participant with a line of people whose actions and appearances were controlled by the ACORE algorithm.

Control condition (C) was identical to E except the appearances were not controlled. Thus each person in line maintained their neutral expression throughout the simula-

Figure 2: Individuals standing in line to purchase a video game. Jim is attempting to pass Vince.

Table 2: One-line utterances.

Condition	Utterance
At the head of the line	Can I get a copy of Destiny?
Being passed	Stop! Where are you going? You shall not pass!
Having been just passed	Not fair! I will get back at you!

tion. This condition approximates COR-E which modeled NPC actions but not their emotional appearances.

Random condition (R) was identical to E except the facial expressions were uniformly randomly selected from four images we had for each character.

Each participant saw one of the six possible permutations of the conditions (i.e., ECR, ERC, CER, CRE, REC, RCE). Then he/she saw the same permutation again. For instance, one participant may have seen ECRECR whereas another may have seen RCERCE. The permutation order was assigned randomly to participants. After each of the six conditions each participant saw, he/she was required to fill out a questionnaire. The primary statement the participants were asked to respond to was *"The virtual characters showed believable emotions."* The answers were on a 1 (strongly disagree) to 7 (strongly agree) point scale.

5.3 Analysis and Discussion

Table 3 shows the believability rating for each condition averaged over all participants and both presentations of that condition to the participant.

Table 3: Mean believability of the conditions.

Condition	Mean $\pm$ standard error
Experimental	4.87 ± 0.12
Control	3.28 ± 0.14
Random	4.09 ± 0.13

A one-way repeated-measures analysis of variance (ANOVA), with Believability as the dependent measure and Condition (E, C, R) as the independent variable revealed a significant effect of Condition, $F(2, 358) = 39.51$, $p < 0.001$, $\eta_p^2 = 0.181$. Bonferroni-corrected pair-wise comparisons confirmed that all three conditions were significantly different from each other.

The participants found the experimental condition to be more believable than the control or the random condition. Thus, explicitly computing (via an appraisal model) and visualizing (via facial expressions) emotions appears to add to the believability our control condition that approximated the previous algorithm (COR-E).

Interestingly, the participants rated the random condition as more believable than the control condition which appears to suggest that even random facial expressions, unrelated to the actions and utterings of people in the simulation, are better than a constant neutral expression.

6 Current Limitations and Future Work

Our implementation of ACORE used *ad hoc* hand-coded values for expected resource changes as well as the associated certainty values. They had not been validated and may have had a negative impact on the overall believability of our ACORE implementation. Future work will generate such values procedurally via stochastic simulation.

Second, we used static images for the facial expressions which were photographs of actors acting out the emotions. Future work will investigate the effectiveness of procedurally generated facial expressions (MOVA 2009; Nvidia 2013), driven by ACORE.

Finally, our simulations were non-interactive as the participants watched the simulation unfold on a computer screen. We are currently working on incorporating ACORE into a video game where it will control actions and appearances of NPCs interacting with the player's avatar.

7 Conclusions

This paper presented the first implementation and evaluation of ACORE, a recent lightweight computational model of emotions. The results of a user study suggest that ACORE can increase believability of AI-controled virtual characters relative to the previous model, COR-E.

8 Acknowledgments

We would like to gratefully acknowledge the funding provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), the time volunteered by the actors of Die Nasty: The Live Improvised Soap Opera, the valuable feedback provided by members of the IRCL Research Group and the help volunteered by Mathew Fritze in developing the experimental setup.

References

Bioware. 2007. Mass Effect Series. http://masseffect.bioware.com/.

Black, A. W., and Lenzo, K. A. 2003. Building synthetic voices. *Language Technologies Institute, Carnegie Mellon University and Cepstral LLC*.

Booth, M. 2009. The AI Systems of Left 4 Dead. http://www.valvesoftware.com/publications/2009/ai_systems_of_l4d_mike_booth.pdf.

Bulitko, V.; Solomon, S.; Gratch, J.; and van Lent, M. 2008. Modeling culturally and emotionally affected behavior. In *The Fourth Artificial Intelligence for Interactive Digital Entertainment Conference*, 10–15. The AAAI Press.

Campano, S.; Sabouret, N.; de Sevin, E.; and Corruble, V. 2013. An Evaluation of the COR-E Computational Model for Affective Behaviors. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 745–752. International Foundation for Autonomous Agents and Multiagent Systems.

Campbell, C. 2014. How Skyrim is helping to shape Dragon Age: Inquisition. http://www.polygon.com/2014/11/3/7151567/how-skyrim-is-helping-to-shape-dragon-age-inquisition.

Desai, N., and Szafron, D. 2012. Enhancing the Believability of Character Behaviors Using Non-Verbal Cues. In *The Eigth Artificial Intelligence for Interactive Digital Entertainment Conference*, 130–135. The AAAI Press.

Gratch, J., and Marsella, S. 2001. Tears and fears: Modeling emotions and emotional behaviors in synthetic agents. In *Proceedings of the fifth international conference on Autonomous agents*, 278–285. ACM.

Gratch, J., and Marsella, S. 2004a. A Domain-independent Framework for Modeling Emotion. *Journal of Cognitive Systems Research* 5(4):269–306.

Gratch, J., and Marsella, S. 2004b. Evaluating the modeling and use of emotion in virtual humans. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, 320–327. IEEE Computer Society.

Hobfoll, S. E. 1989. Conservation of resources: A new attempt at conceptualizing stress. *American psychologist* 44(3):513.

Hunicke, R., and Chapman, V. 2004. AI for dynamic difficulty adjustment in games. In *Challenges in Game Artificial Intelligence AAAI Workshop*, 91–96. sn.

Isla, D. 2005. Handling Complexity in the Halo 2 AI. In *Game Developers Conference*, volume 12.

Johnson, M. S. 1994. Validation of an active multimedia courseware package for the integrated damage control training technology (IDCTT) trainer. Master's thesis, Naval Postgraduate School, Monterey, California.

Linguica. 2008. Wrex. Shepard. https://www.youtube.com/watch?v=u5Fjc1hBwuE.

Manavalan, Y. B., and Bulitko, V. 2014. Appraisal of Emotions from Resources. In *Interactive Storytelling*. Springer. 224–227.

Marsella, S., and Gratch, J. 2009. EMA: A process model of appraisal dynamics. *Journal of Cognitive Systems Research* 10(1):70–90.

Mateas, M., and Stern, A. 2003. Façade: An Experiment in Building a Fully-Realized Interactive Drama. In *Game Developers Conference (GDC03)*.

MOVA, O. 2009. Geni4. https://www.youtube.com/watch?v=0fF2pAsaaiw.

Nvidia. 2013. Nvidia Faceworks. https://www.youtube.com/watch?v=CvaGd4KqlvQ.

Orkin, J., and Roy, D. 2007. The restaurant game: Learning social behavior and language from thousands of players online. *Journal of Game Development* 3(1):39–60.

Pramath. 2015. This Is The Full World Map for the Witcher 3: Wild Hunt. http://gamingbolt.com/this-is-the-full-world-map-for-the-witcher-3-wild-hunt.

Scherer, K. R.; Schorr, A. E.; and Johnstone, T. E. 2001. *Appraisal Processes in Emotion: Theory, Methods, Research.* Oxford University Press.

Si, M.; Marsella, S.; and Pynadath, D. V. 2005. Thespian: Using multi-agent fitting to craft interactive drama. In *Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, 21–28. ACM.

Solomon, S.; van Lent, M.; Core, M.; Carpenter, P.; and Rosenberg, M. 2008. A Language for Modeling Cultural Norms, Biases and Stereotypes for Human Behavior Models. In *Proceedings of the 17th Behavior Representation in Modeling & Simulation (BRIMS) Conference, Providence, RI*.

Strategy First. 2001. Sub Command. http://en.wikipedia.org/wiki/Sub_Command.

Traum, D.; Rickel, J.; Gratch, J.; and Marsella, S. 2003. Negotiation over tasks in hybrid human-agent teams for simulation-based training. In *Second International Joint Conference on Autonomous Agents and Multiagent Systems*, 441–448. ACM.

Ubisoft. 2004. Tom Clancys Ghost Recon 2. http://en.wikipedia.org/wiki/Tom_Clancy's_Ghost_Recon_2.

Ubisoft. 2014. Far Cry 4. https://www.ubisoft.com/en-GB/game/far-cry-4/.

Valtchanov, V., and Brown, J. A. 2012. Evolving dungeon crawler levels with relative placement. In *Proceedings of the Fifth International C* Conference on Computer Science and Software Engineering*, 27–35. ACM.

Valve. 2007. Portal. http://www.valvesoftware.com/games/portal.html.