RealAct Manual

Updated 17/10/2016

Maryam Saberi  
PhD Candidate  
School of Interactive Art and Technology  
Simon Fraser University  
msaberi@sfu.ca

# Overview

“RealAct” is a computational framework for expressive, personality-based, non-verbal behaviour for affective 3D character agents. To make the virtual characters behave in a believable and consistent manner, the system controls non-verbal behavior such as gaze, facial expressions, gestures and postures in order to give the impression of a specific personality type). The design and development of different modules of the RealAct system, e.g. for controlling the behaviour and generating emotion, is directly derived from existing theoretical and computational literature in the area of personality models impression of personality from nonverbal behaviour for both humans and 3D humanoid characters. In addition to these core modules, the RealAct system contains a library of modules that are specifically geared toward real-time behavior control needs such as sensory inputs, scheduling of behaviour and controlling the attention of the character.

RealAct provides the following capabilities for a virtual 3D character in real-time:

1. A comprehensive empirical, psychological and statistical data on biological humans’ behaviour, personality, and human to human interaction is explored to extract patterns of human behaviour which affect the perception of personality. These “soft” data sets are then reduced integrated, and categorized to definitive data useful to computational modeling. The categorized and structured material on the association of personality and nonverbal behaviour can be a useful repository for cognitive and affective computing researchers.
2. The structured and categorized data are then extensively programmed into computational framework using artificial intelligence and computer simulation based techniques to create the RealAct system with the following features:
   1. I proposed a novel **hybrid** structure for the RealAct system (‎Figure ‎3.1) to follow two distinct patterns of human behaviour: 1) plan-based and logical (RealAct’s Event-based module), 2) reflexive and emotional (RealAct’s Emotionally-Continuous module).
   2. Several sophisticated real-time bio gesture and movement **sensor** **systems** are tested, setup and eventually implemented in RealAct to create an emotive real-time character responsive to users’ movements, facial expression, and hand gestures.In addition, RealAct is based on our Movement + Meaning (m+m) architecture [167], which is a software platform which facilitates adding new modules to RealAct for capturing and recognizing the movement data.
   3. To promote the future use by other researchers, I programmed the RealAct system as a set of encapsulated and reusable blocks saved in the open source RealAct library . In addition, RealAct used an open standards system for sending behaviour commands to the animation toolkit which can be used by other animation engines (Behaviour Markup language [129]). The RealAct framework and its documentation are available online at [[ivizlab.sfu.ca/research/thesis/saberi](http://ivizlab.sfu.ca/research/thesis/saberi)].The following is the summary of available blocks of the RealAct library, and the novel incorporation of personality traits “extraversion” and “emotional-stability” in them:
      1. **Gaze controller** is a combination of eye, head, torso, chest, back and blink behaviour control module. It refines the Eyes Alive model of gaze [100] to create a gaze behavior following the human ocular behaviour. The expression of personalities is reinforced by controlling the following gaze parameters: chance of occurrence of averts or mutual gazes, gaze direction, duration of avert and mutual gazes, and speed of head movements (Table ‎3.1).
      2. **Postures and gestures controller** proposes the following expressivity dimensions to reflect emotional-stability and extraversion personality traits: 1) Posture-shift behaviour, 2) Self-adaptor behaviour, 3) Leaning behaviour (lean forward, no lean and lean backward) 4) Twitches (true or false) and 5) Spacious gestures (true or false). These dimensions can have three different frequencies (high, mid, low) and three speeds (fast, mid, slow) (Table 2).
      3. **Facial expressions** **controller** adapts Boukricha et al’s model to associates the emotional valance and arousal values [82], generated in “emotion generation module”, with facial muscle movements [71]. The impression of personality is created through changing the intensity of emotions, filtering of emotions, and facial twitching (Table ‎3.3).
      4. **Emotion generation module** uses three kinds of triggers to elicit the emotional valence and arousal [51]: 1) triggers activated during interaction with the user and environment, 2) triggers regarding the interaction scenario, and 3) internal triggers when no external event is happening. Based on their importance, triggers can have different impacts on the generation of arousal and valence (see Figure ‎3.4). Personality affects the generation of emotion (see Table ‎3.4). New triggers with desirable impacts can easily be added to RealAct.
      5. **Attention controller** module makes the 3D character attentive to sudden environment changes, and events regarding the scenario of the interaction. Based on the body parts involved two attention types are proposed: gaze, body. If the attention signal only requires the attentiveness of the gaze, other body parts can continue with their idle behaviour. The same rule applies to the body.
      6. **Behaviour scheduler** prioritizes and selects a behaviour from multiple behaviour requests generated by behaviour controllers using three priority queues for high, mid and low priority behaviour (Figure ‎3.15‎). The behaviour with the highest priority is then sent to the animation engine.

Table of **Contents**

[Overview 2](#_Toc464483085)

[Downloading RealAct 5](#_Toc464483086)

[RealAct Architecture 6](#_Toc464483087)

[1.1. Stateflow Subcharts 8](#_Toc464483088)

[1.2. Behaviour Controllers Modules 9](#_Toc464483089)

[1.2.1. Gaze Controller 10](#_Toc464483090)

[1.2.2. Gestures and Postures Controller 13](#_Toc464483091)

[1.2.3. Facial Expression Controller 17](#_Toc464483092)

[1.3. Emotion Generation Module 18](#_Toc464483093)

[1.4. Behaviour-Scheduler Module 21](#_Toc464483094)

[1.5. Attention-Controller Module 25](#_Toc464483095)

[1.6. Personality as a Parameter Influencing the System 27](#_Toc464483096)

[1.6.1. Personality Influence on Emotion Generation 27](#_Toc464483097)

[1.6.2. Personality Expressive Gaze 30](#_Toc464483098)

[1.6.3. Personality Expressive Poses and Gestures 34](#_Toc464483099)

[1.6.4. Personality Expressive Face 35](#_Toc464483100)

[Rock/Paper/Scissors Game Scenarios 37](#_Toc464483101)

[Measurements used for the Real-time Experiment 40](#_Toc464483102)

# Downloading RealAct

The SmartBody source code can be downloaded from [[ivizlab.sfu.ca/research/thesis/saberi](http://ivizlab.sfu.ca/research/thesis/saberi)].

# RealAct Architecture

Following the pattern of human behaviour, the RealAct’s hybrid structure is designed to support both a 1) logical behaviour of the virtual character moving through states of the interaction and 2) continuous updates of the emotional expressions of the virtual character depending on feedback from interactions with the environment. Matlab Stateflow [125] and Matlab Simulink [126] are selected to implement respectively Event-based and Emotionally-Continuous behaviour of the RealAct. To promote the future use by other researchers, I programmed the RealAct system as a set of encapsulated and reusable blocks saved in the open source RealAct library.

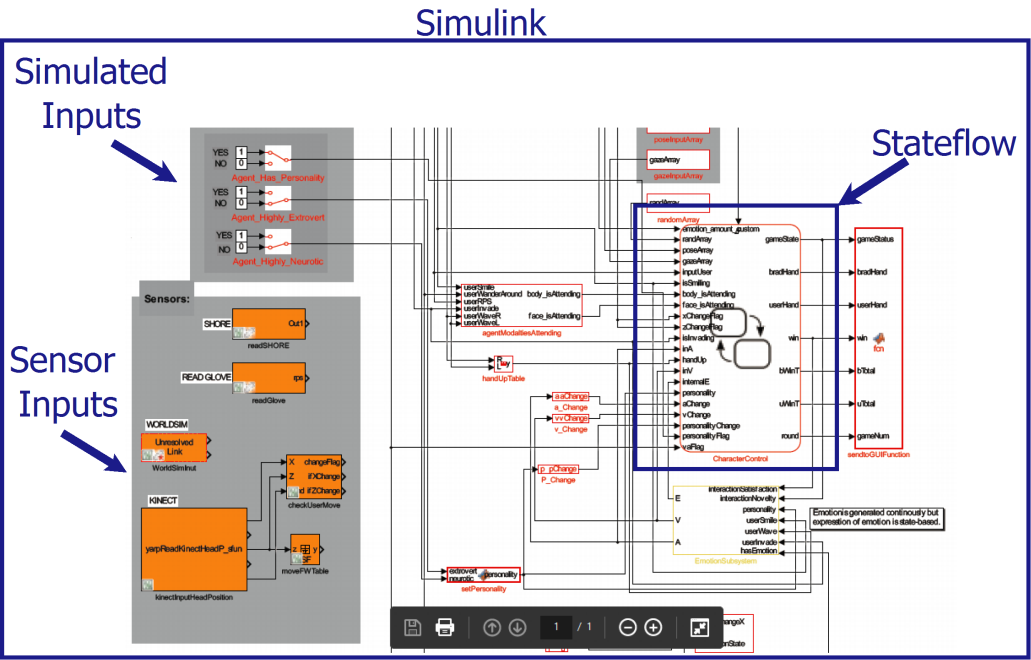


Figure ‎0.1. Screenshots of the system (Matlab Simulink/Stateflow)

The Emotionally-Continuous system is implemented using Matlab Simulink [126] which facilitates a real-time continuous framework. Programming with the Simulink platform allowed me to give RealAct the possibility of interactively simulating while continuously viewing the results of the selected variables of the system using scopes and graphical displays. Simulink also allowed me to use integration algorithms that compute system dynamics over time.

Within the RealAct system, Simulink sends data to Stateflow and receives data in return. Personality type selected by the software designer, sensor data and emotional valence and arousal generated in Emotion Generation module flows from Simulink into a chart, dynamically during the simulation, via input ports on the chart. Output data, such as interaction phase and status of the interaction, flows from a chart into Simulink via output ports on the chart (Figure ‎3.2). I took the advantage of Simulink data stores memory blocks to create global variables across the Simulink and Stateflow such as Behaviour Markup Language (BML) strings [129] created in behaviour control modules and sent to animation engine from Behaviour-Scheduler module (section ‎‎3.9).

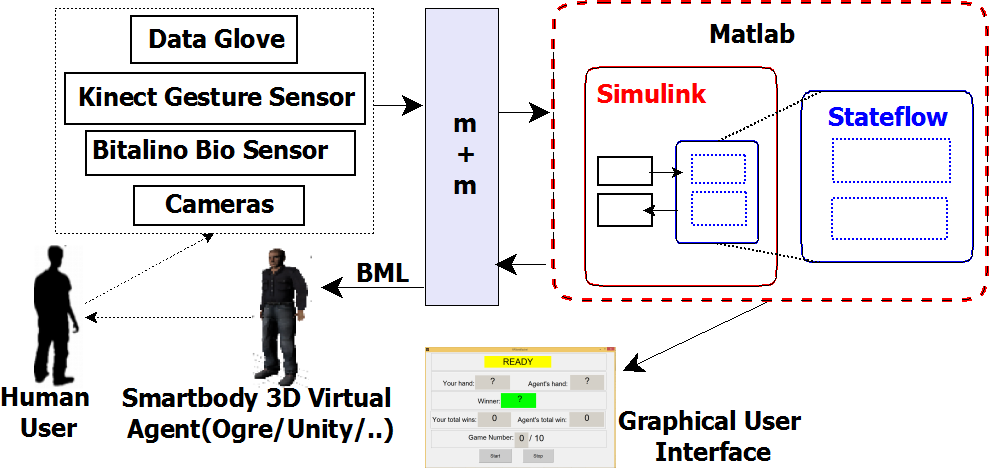


Figure ‎0.2.a Matlab Simulink/ Stateflow is depicted as an implementation tool for the hybrid structure of the system (m+m [167] stands for Movement and Meaning framework is explained in section ‎3.10). For the subsystems inside the Simulink and Stateflow see the next figure.

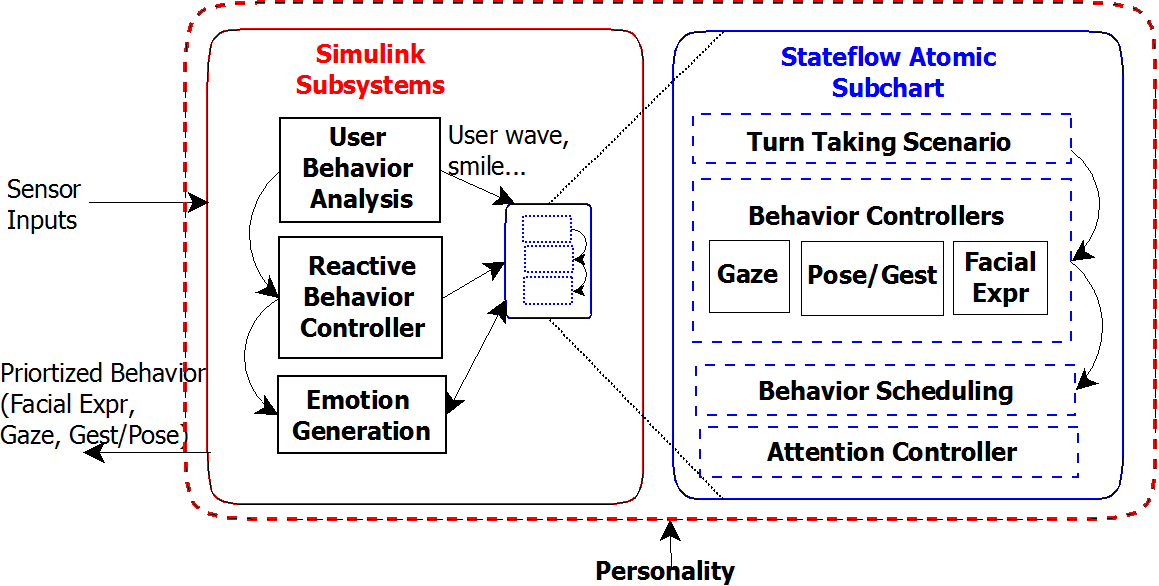


Figure ‎0.3.b Detailed structure of the components of Simulink and Stateflow (Note: The User-Behaviour-Analysis module is not supported in this version, but is suggested here as an important module to be implemented in future. However, the current system supports a few simple functionalities such as understanding and mimicking user’s smile and wave gesture)

As depicted in Figure ‎0.2.a, the inputs of the system are received from sensors installed in the environment. The character reacts dynamically and in real-time to these inputs. The outputs of the system are facial expressions, postures, and gestures of the 3D virtual character, which are dynamically generated and fed to the animation toolkit and before being displayed to the user.

## Stateflow Subcharts

Figure ‎3.3.b, demonstrates that event-based behaviour of the character is dynamically generated and scheduled in the Event-based component of RealAct. On the other hand, facial expressions and reactive gestures of the character are controlled continuously and based on the continuous feedback from the environment in the Emotionally-Continuous component. Parameters of the architecture, such as specified personality for the character, can be tuned by the designer of the system at the beginning of the interaction or dynamically during the interaction. Additionally, the behavioural scope of the architecture is limited to strategic turn-taking interaction between the character and the user.

## Behaviour Controllers Modules

The outputs of the RealAct system are behaviour commands sent to the animation engine for three modalities: facial expressions, postures/gestures, and gaze movements. These commands are dynamically fed to the animation engine, and performed by the virtual character during the simulation. For each of the three modalities, three kinds of behaviour are defined: 1) idle, 2) communicative and 3) reactive. The idle and communicative behaviour is generated in the Event-based system while reactive behaviour is controlled with the Emotionally-Continuous system (Figure ‎3.1). The idle behaviour is mainly a dynamic set of animation sequences that involves subtle movements of all the body parts such as gaze, head and hand movements. It is defined to ensure the character does not look motionless while standing and doing nothing. People tend to keep moving their body even if they are doing nothing e.g. they shift their body, scratch their body parts or move their head around. For the virtual character, an idle behaviour is a behaviour the character shows when not following any goal or not responding to any environment changes.

The communicative behaviour is triggered in response to goals (usually goals dictated by the interaction scenario) such as character showing rock hand gesture (e.g. throwing out their hand in a fist position) in a rock/paper/scissors game (RPS). The communicative behaviour has a higher priority than idle behaviour and will replace it if they are triggered at the same time. The reactive behaviour is usually in response to a sudden or unexpected change of environment and an automatic or not planned behaviour. For instance, if the user gets too close, the character automatically moves back and adjusts her personal space. The reactive behaviour has the highest priority. For the full explanation of the defined priorities and the logic behind them see Behaviour-Scheduler Module section. Moreover, the character’s idle, responsive or communicative behaviour is influenced by the character’s personality. This sophisticated and lifelike hierarchy of prioritized and intergraded behaviour layers which can be dynamically augmented by personality type and reaction to long term and immediate user and environmental events are the main contributions of my research and the RealAct system. If for example, a character is an extravert, its idle behaviour consists of faster gestures, postures, and head movements. In addition, Gaze, Pose/Gesture, and Facial Expression controllers are developed in the way that can be used separately. For instance, researchers can use the gaze module, without getting involved with the rest of the system.

Following is a general description of RealAct’s main behaviour channels (gaze, pose/gesture, and facial expression), how they function and their implementation details. All of the behaviour channels (Gaze, Pose/Gesture, and Facial expressions) have their own Matlab functions and Truth-table functions. Stateflow’s Truth-tables are used to implement combinatorial logic design behaviour to categorize and select the behaviour control commands. Each of the decision columns combines an outcome for each condition (usually specified based on personality type) with a logical AND into a compound condition that is referred to as a decision. The decision refers to select and performing one or multiple BML commands to control the behaviour. Prioritizing and sending behaviour commands are centralized in Behaviour-Scheduler module. All these sub-systems are designed separated from and in the same hierarchical order with the scenario of interaction. Some of these sub-charts are themselves include inner sub-charts of their own. For instance, gaze consists of two sub-charts executing in parallel for controlling the eye and head of the character.

### Gaze Controller

Gaze behaviour is a combination of movements of character’s eye, head, chest, back and torso, dynamically controlled by the system. The blink frequency of the character is also controlled as a part of gaze behaviour. To make the gaze behaviour of the character appear natural, the gaze module produces eye movements which are consistent with human ocular behaviour. In addition, head movements’ model is also implemented based on imperial, psychological and statistical data on biological humans’ human’s head movement behaviour, and how personality affects it. Personality affects the gaze behaviour on multiple dimensions: head speed, blink frequency, type, direction, frequency and duration of the gaze (see ‎3.8.2).

In RealAct’s design, three categories of gaze behaviour are utilized: 1) reactive, 2) communicative, and 3) idle behaviour (Figure ‎3.5). The reactive category is when the gaze responses to user’s movements and in general environment feedback. Communicative category addresses gaze behaviour that communicates a meaning to the user or is triggered by a change in the state of the interaction. For instance, in a rock-paper-scissors game, character gazes at the graphic user interface (GUI) which has real-time updates on the game statistics, to create the impression the character is checking the ongoing results of the game.

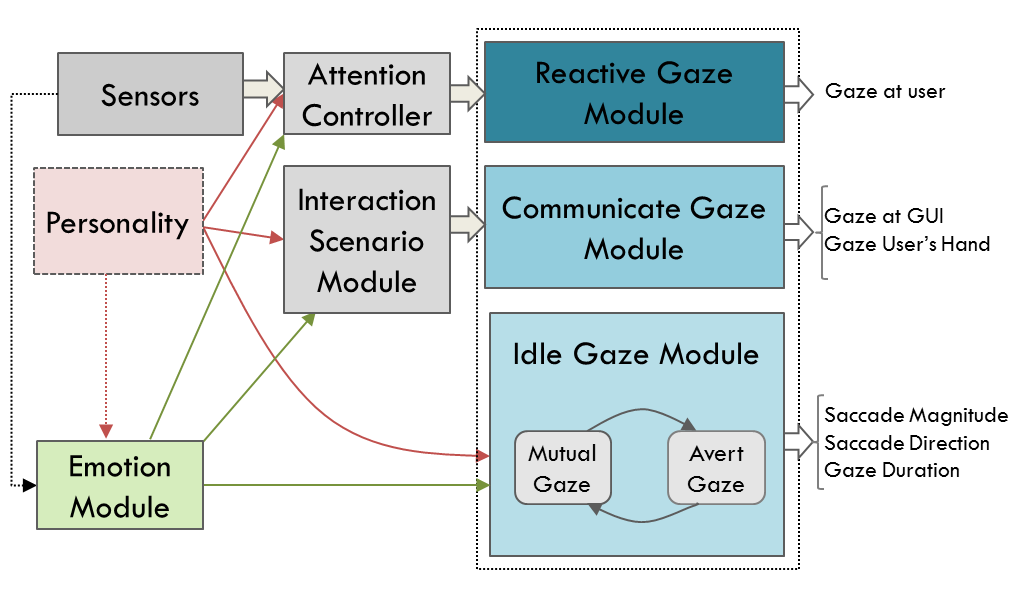


Figure ‎3.5. The hierarchical structure of gaze behaviour which is affected by generated emotion and selected personality for the virtual character. Reactive gaze has a high level in the hierarchy and is responsive to environment changes. Communicative gaze is responsive to character’s goal and scenario of the interaction. Idle gaze has a low level of importance and other gaze behaviour will overwrite them.

The idle category is employed for the gaze behaviour when the character is doing nothing. These categories are prioritized so the higher level behaviour overwrites the ones with lower priorities. The highest priority gaze movement is reactive gaze. People, regardless of what they are doing, tend to look at the unpredicted movements they notice in the environment. Thus, gaze at unexpected environment changes and feedback, such as a sudden loud noise, is always in a higher priority that communicative gaze and will overwrite it if they are triggered at the same time. The idle gaze behaviour has the lowest priority and happens if no other gaze behaviour is triggered.

To incorporate the kinematic characteristics of the idle eye movement, RealAct gaze module adapted and updated Lee et al model of gaze (Eyes Alive) [100]. Lee et al developed an eye behaviour model which is derived from statistical models of eye-tracking data and is consistent with human ocular behaviour (see ‎2.2) [100]. In Eyes Alive model, two states of the gaze are defined: ‘avert’ and ‘mutual’. The mutual gaze is when the character gazes at the user, while avert is when the character is looking away. The character’s gaze rests at one of these states for a period of time and then moves to the other state. Additionally, idle gaze behaviour in Eyes Alive model is calculated as a function of 1) saccade magnitude, 2) saccade direction and 3) saccade duration. I am using the same terminology in my gaze model: ‘Saccades’ are defined in the literature as rapid and voluntarily movements of eyes [142]. The ‘magnitude’ of a saccade is the rotation angle of the eyeball. Saccade ‘direction’ is the position of the eye based on polar coordinates. Saccade ‘duration’ is the duration of an eye movement.

Usually, eye movements are followed by a head rotation in the same direction. In fact, the natural gaze is usually not more than 15 degrees [102]. Additionally, head and eye movements are synchronous [103]. Thus, to control idle head movements in RealAct’s design, if the generated amplitude is more than 15 degree, a head movement in the same direction with the eye will be generated. Smartbody’s procedural eye model supports the realistic eye and head coordination. When the head rotates to a new position, the eye movement is automatically generated and layered on top of the head and torso movement [101]. However, Smartbody characters’ head and eyes do not coordinate perfectly when the speed and magnitude of the head rotation are high (more than 25 degrees). Thus, I re-coded the head movements to limit the movements and speed to achieve a more natural behaviour. Furthermore, in the RealAct’s design the chance of the averting head is a function of time that is it increases as the time passes and the character gets increasingly bored.

Reactive gaze behaviour is implemented to be responsive to the sensors inputs. The character’s gaze follows the user as they move into the space in front of the character. Microsoft Kinect [127] and overhead camera are used to feed the user’s position to the system. Then, the eye and head of the character are moved to give users the impression that character’s gaze is reactive and responsive to the users’ movements. While generating this responsive gaze behaviour, I encountered an issue which is referred to as ‘Mona Lisa gaze effect’ in the literature [121]. Since the virtual environment, the character exists on is a two-dimensional space, looking at the character in the TV monitor from any direction gives users the feeling that the character is looking at them. This makes it hard to implement the impression of the character following the user moving in the room. Because, on one hand, a little bit of rotation of the head of character creates a drastic change of direction in the real world (e.g. if the head of the character is moved 10 degrees to the left, it looks like the character is looking at somewhere in 90 degrees to the left in the real world). On the other hand, if the head of the character does not move and stay still, it looks like he is not responsive to the movement of the user and environment events. To solve this problem, I coded the system to mainly move the torso and chest of the character in response to the movement of the user, while eyes and the head had a subtle movement in the same direction. The communicative and reactive gaze behaviours were not the main focus of the implementation, so a minimum functionality is implemented to maintain the believability and smoothness of interaction. However, they need to be explored in more detail in future.

### Gestures and Postures Controller

Similar to gaze behaviour, gestures can be reactive, communicative or idle (Figure ‎3.6). Reactive gestures are responsive to the environment, such as a 3D virtual character waving back at a user when that user waves first at them. Communicative gestures and poses are generated based on the scenario of the interaction, such as rock hand gesture in a rock-paper-scissors game. Same as the idle gaze, idle poses and gestures are performed while the character is not performing any specific task, which is at idle.

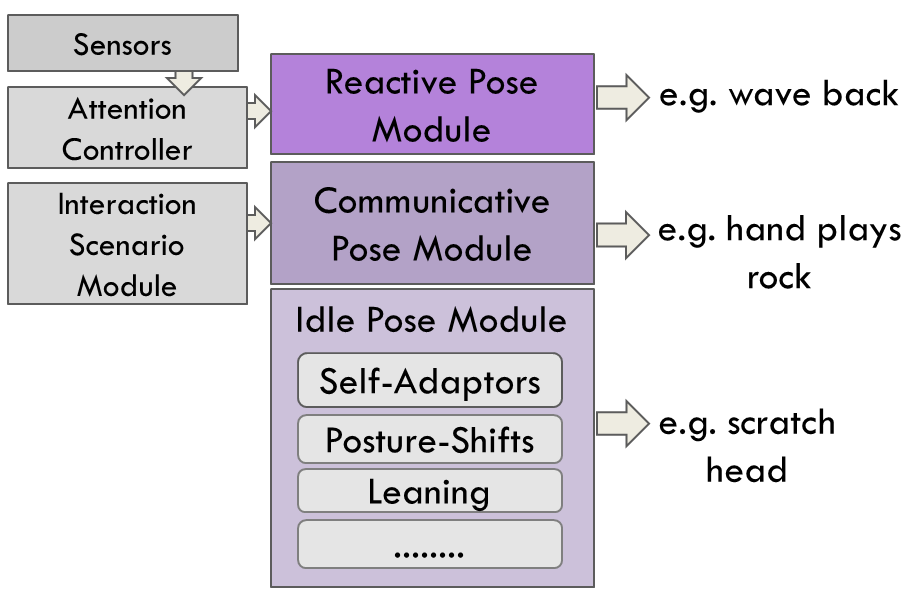


Figure ‎3.6. Gestures are categorized into three types: reactive, communicative and idle gestures and poses.

Reactive gestures/poses are tested in a simple scenario in which the character was mainly mimicking the behaviour of the user.

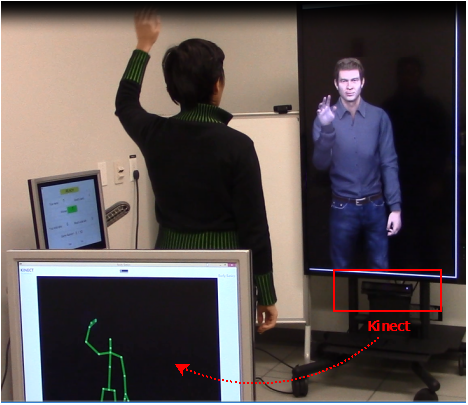


Figure ‎3.7. Using a Kinect sensor [127] the user’s waving hand gesture is recognized by the system, at which time the character responds with a wave back.

The scenario was only designed as a proof of concept and to show the flexibility of the system for the range of different scenarios. For instance, using the Kinect sensor [127], a user’s hand gestures such as waving is recognized by the system (Figure ‎3.7), at which time the character responds with a gesture such as waving back (more details on section ‎4.5.1).

Communicative gestures/postures are usually responding to social scenarios such as rock hand gesture in the RPS game (Figure ‎4.14). With these types of gestures, strong emotional reactions can be accompanied by a set of emblems (nonverbal signals that can be translated to words, see section ‎‎2.2.2). For example, if the character is very angry, he positions his hand into a fist while making frowning facial expression. Emblems, illustrators, and self-adaptors are proposed by Ekman et al’s as three classes for hand movements’ behaviour [106] (see‎ ‎2.2.2 for more details). Ekman et al presented a theatrically based categorization for interpretation of hand movements. Since my focus was on non-verbal behaviour, illustrators (used for dialog based conversation) are not addressed in the RealAct gesture/postures categorizations. Self-adaptors such as “scratching the head” and “holding two hands together” are also used in the RealAct.

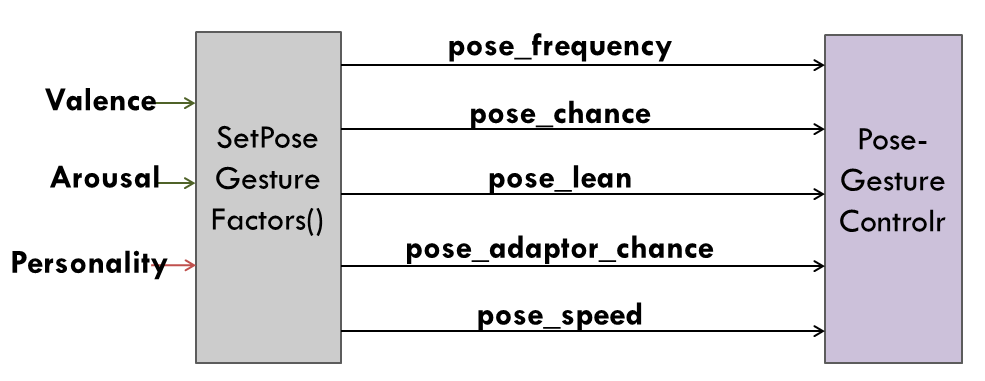


Figure ‎3.8. Drawing on psychological research, emotional state (valence-arousal) and personality of the character (extravert or stable) affects several dimensions of gesture and poses behaviour such as if the poses are occurring, their frequency occurrences and their speed [154][29][30][152].

The chance of selecting each of the gestural/postural animations from the mentioned categories is a function of 1) the simulation time, 2) emotional state of the character (valence-arousal) and 3) personality of the character (extravert or stable). These three parameters can specify if the pose is occurring, if it is either fast or slow and how frequent the occurrences of a category of gestures are (Figure ‎3.8).

A significant part of the research in this thesis was building a simple and implementable computational model of these mentioned categories into believable and verifiable dynamic and real-time gestural/postural animations in RealAct responsive character. The character’s hand movements, body gestures, and poses are controlled by pre-defined key-framed 3D parameterized animations. In order to synchronize multiple animations, Behaviour Markup Language (BML) provides sync-points and timing constraints which can be used to order the behaviour [129]. A subset of the animations used in the RealAct is selected from my animation toolkit’s default animations. The rest were created by an animation team we recruited from undergraduate students in iVizLab, Simon Fraser University. In addition, we recruited an actor and recorded him playing a list of scenarios needed to be animated (Figure ‎3.9). I divided the recorded videos into small portions, 3 to 5 seconds and sent them to 3D animators as a guideline alongside the 3D character’s skeleton. I worked with the 3D animators on integrating better animations for the system and built in the parameterization that made the animations dynamic (parameters such as speed of the movement). Cinema 3D and Poser were used to create the animations. Finally, I used fbx converter [101] to convert the animations received from animators back to fbx format and tested them in the system. The fbx format is a standard format and can be used by other animation toolkits.



Figure ‎3.9. We recruited an actor and recorded him playing a list of scenarios needed to be animated. The recorded videos then were divided into small portions, 3 to 5 seconds used by 3D animators as a guideline. The right figure is a screen shot of the actor playing a wave. Left figure is a screen shot of a created animation for waving.

### Facial Expression Controller

For humans, emotions are activated by external or internal effects and each last for a short period of time [22][23]. Similarly, to perceive virtual humans as more natural and believable, it is crucial that they show proper facial expressions with respect to affective states which are recognizable by users. Achieving this level of believable expressive realism on a real-time and responsive 3D character is still a source of ongoing research [44][76][133][136]. In the RealAct system, the rich dynamics of emotional changes, triggered by external and internal stimuli, are continuously generated in the emotion generation module. The generated emotions then are expressed through facial expressions. For generating an emotion (internal), I am using the Circumplex model of affect in which each emotion can be understood as a linear combination of valence (pleasure) and arousal [51]. For the emotions that are revealed externally our character’s face to the user, generated values of valence and arousal were mapped to Ekman’s Action Coding System to generate facial expressions.

To map the valence and arousal and facial action units, I used the data of Boukricha et al’s study [82]. They applied reverse-engineering methods and statistical calculations to simulate a virtual human’s facial expressions with Pleasure-Arousal-Dominance models values. Their work is built around the data they gathered from 353 participants rating a set of generated facial expressions. While I was able to use Boukricha et al work as a foundation of RealAct emotional facial mapping algorithm, however, I needed to extend the work as well as test its validity in several areas. Since my system used the animation toolkit and open standards system of BML/Smartbody, it was important to extend the work so the valence and arousal parameters worked for the different facial action set of the SmartBody system including reinterpreting Boukricha et al’s defined face visemes (face movements defined for the speech) for use in expression for SmartBody conversation. In addition, for some of the action units, the mapping values for intensities were not high enough to be visible in Smartbody character’s face. Thus, I tuned some of the parameters used in Boukricha et al’s mapping [82] to generate higher intensity values for some action units. To test the validity of created emotions and make sure users perceived the same impression of the synthesized emotions as desired, I performed an experiment study in which 20 participants rated the valence and arousal of 9 still images generated from captured real-time output based on my updated emotional mapping (details are discussed in section ‎‎4.2).

## Emotion Generation Module

The RealAct design goal is to facilitate real-time, natural and human-like non-verbal behaviour which is responsive to the input streams from multiple sensors. As the result, in the RealAct system emotional responses are associated directly and as a linear function of immediate perceptual features and interaction scenario inputs, rather than responses based on the appraised interpretation of events which is of potential interest to models of strategic decision-making and action selection such as in [86][89][90].

In the current version of RealAct, three kinds of triggers elicit emotional valence and arousal: 1) triggers activated during interaction with the user and environment, 2) triggers regarding the interaction scenario, and 3) internal triggers when no external event is occurring. Several psychological studies on biological humans confirm that emotion is contagious [155]. Thus, the positive valence of the character increases if they sense a user’s positive emotion [155]. Moreover, a signal of potential gains increases valence while the signal of potential losses decreases valence [171][172]. In the RPS scenario, I assume satisfaction is increasing as the character wins and decreases as she loses in each round of the game. Thus, on one hand, positive feedback from the user (such as waving and smiling), and positive feedback regarding the interaction scenario (such as winning in the RPS game) increase the desirability of the interaction and generated valence. On the other hand, negative feedback from the user and interaction scenario decreases the desirability of the interaction and generated valence.

Likewise, generated arousal is a linear function of user and interaction scenario feedback. For instance, when a user invades the personal space of the character, this will increase the arousal of the interaction. Uncertain cues, competition, challenges, reward and punishment typically increases the arousal [16][57]. In addition, increasing the difficulty of the game leads to a higher arousal [157][57]. Since arousal is in direct relationship with the difficulty of the game, in RPS game scenario, I am assuming the character’s excitement increases as it gets closer to the “Go” state and decreases as it gets to the “Wait” state (more information about the procedure of the RPS game is discussed in section‎ 4.5.1). Since psychological data shows that repeating the game, again and again, decreases the experienced arousal [156][158], during the RPS game the repetition of the game cycles have a negative effect on the character’s experienced emotional arousal.

Additionally, depending on how important a trigger is in satisfying the character’s needs, it can have different types of impacts on generated arousal and valence. Based on Maslow’s “hierarchy of needs”, three categories of input triggers are defined: self-esteem, love/belonging and safety [123]. For RealAct I make the assumption that the triggers related to safety have more impact on arousal and valence changes than inputs related to love and belonging and they both have higher importance than inputs regarding the self-esteem based on work from [123]. For instance, a user invading a personal space of the character jeopardizes the need for the safety and has the highest impact in increasing the arousal while smiling back at user corresponds to need for being loved which has a lower importance. The values for the factors are specified merely to differentiate between the effects of different inputs. The impact is specified by multiplying a set of pre-defined weights to each of the triggers (Figure ‎3.4).

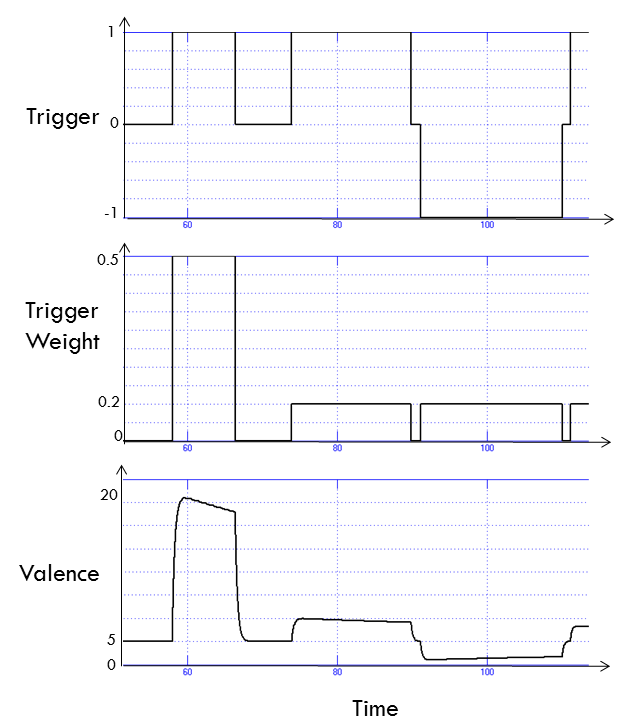


Figure ‎3.4. Generation of valence through the time is the function of triggers with different weights

Figure ‎3.4, shows the temporal change of input triggers, weight of each of the triggers and resulting valence changes, using a Simulink Scope block. The Simulink Scope block allows us to view signal data during the simulations while grouping data in this case triggers weights and valence signals which gives us a comprehensive view of the data with respect to the simulation time.

The RealAct Emotion-Generation module is designed in the way that future designers can add their desired input triggers and weights to the model easily. More details on the complete list of the triggers already implemented can be found in [[ivizlab.sfu.ca/research/thesis/saberi](http://ivizlab.sfu.ca/research/thesis/saberi)]. In the current version of the emotion generation module, emotional reactions are only considered to be self-centered. For example, the character gains positive valence when winning and negative valence when losing the game. However, people do not only show self-centered emotional reactions. They also tend to empathize with the other party of the interaction, based on their personality [160]. In future, the empathic reactions can be added to the emotion generation module which reacts to the user’s emotional expressions and user’s winning and losing based on the personality type.

## Behaviour-Scheduler Module

In daily life, people tend to pursue various goals e.g. having a well-paid job, buying a house, have an intimate relationship etc. Based on the urgency of the goals and availability of external and internal resources, people tend to prioritize these goals. Each of these main wants and needs can be a source of several compound goals and behaviour. Additionally, some of the goals can be pursued in parallel or in coordination (e.g. finding a job and buying a house), some the goals compete for the same resources (e.g. working or spending time with the loved ones) and some of the goals are in conflict (e.g. avoid rejection and create closeness and intimacy). These different types of complex goals create various tendencies in us which lead to a request for corresponding behaviour (e.g. being hungry requests for eating behaviour while a need for finishing a book requires studying). Humans prioritize and choose between these various requests for behaviour, based on their internal state (e.g. how hungry we are) and my past experiences [173][174][176]. Similarly, for virtual characters, various modules generate multiple behaviour requests and some of these requests can occur at the same time. Therefore, there is a need for a mechanism for prioritizing and selecting proper behaviour, based the research we understand of humans’ behaviour selection [175][176]. Some of the requested behaviour can be performed in parallel (e.g. head avert to a point and scratching the neck) while some are competing for the same body joins (e.g. waving for the user and scratching the chest). As shown in Figure ‎3.14, I defined four possible ways different behaviour can overlap: sequentially, switching, parallel and interrupting. In sequential condition, one behaviour is finished before the start of the next one. People also switch to another behaviour, especially if the behaviour is short, and then come back to the current activity. Some of the behaviour can be performed in parallel (e.g. head avert to a point and scratching the neck) while interrupting behaviour occurs when joints are competing for the same body joins (e.g. waving for the user and scratching the chest).

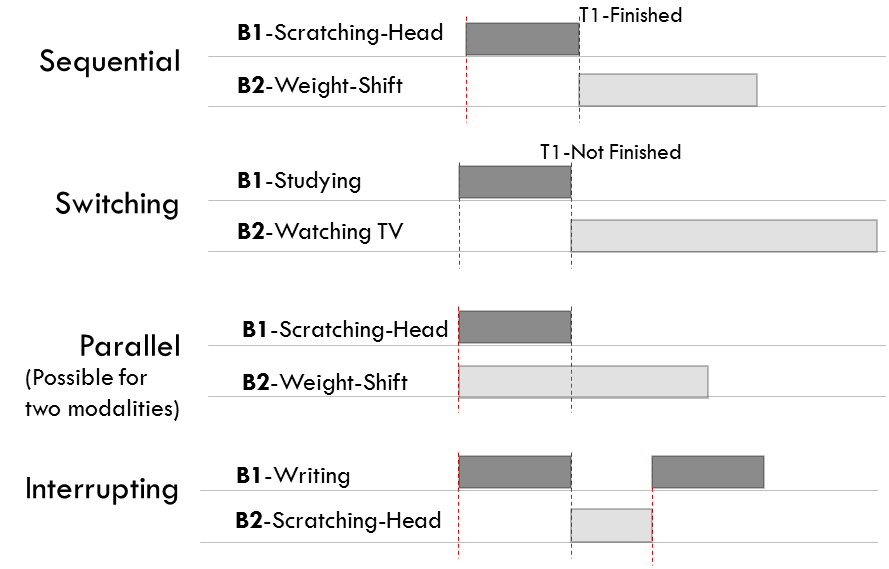


Figure ‎3.14. The four possible ways different behaviour overlap: sequentially, switching, parallel and interrupting. In sequential condition, one action is finished before the start of the next one. People also switch to another behaviour, especially if the behaviour is short, and then come back to the current activity. Some of the behaviour can be performed in parallel (e.g. head avert to a point and scratching the neck) while interrupting behaviour happens when joints are competing for the same body joins (e.g. waving for the user and scratching the chest).

Sending two behaviour requests which share some of the joins to the animation engine can lead to ignoring one of the behaviours or blending the behaviours in an undesired way. This can be problematic especially for behaviour which is time-based and crucial. To elaborate more, consider the scenario of a virtual character playing the rock-paper-scissor game with a user. Several modules are sending behaviour commands to the animation engine. For example, a module responsible for idle poses and gazes generates random poses e.g. head scratch at random times of the interaction. If this occurs at the time the user and the character want to play the hand, the idle module will send a head-scratch command and this command can overwrite the RPS hand gesture in which case the character will scratch her head instead of playing the hand! Such an unfortunately synced coincidence can negatively affect the believability of the experience.

To avoid problems like these and control the overlapping behaviour with different priorities four techniques have been designed and implemented: if two behaviours have different priorities, the one with a higher priority will be selected. If a higher priority task is followed by a lower priority task, RealAct’s attention module will assure that the lower priority task does not affect or blend with the higher priority behaviour. If a low priority behaviour is followed by a higher priority/ or low priority behaviour, it will blend to the new behaviour (Figure ‎3.15).

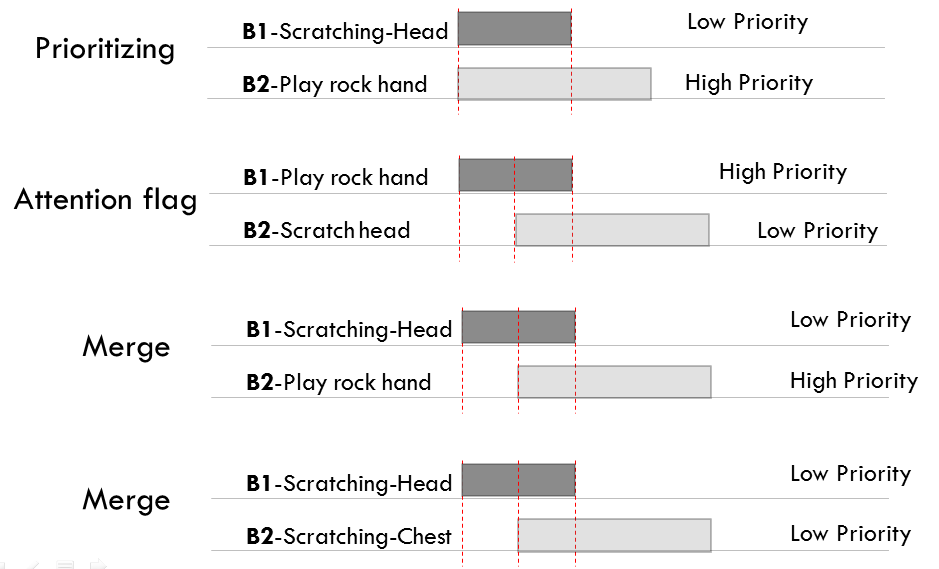


Figure ‎3.15. The four possible ways behaviour with different priorities can overlap and how they are managed in the RealAct: if two actions have different priorities, the one with a higher priority will be selected. If a higher priority task is followed by a lower priority task, the RealAct’s attention module makes sure the lower priority tasks does not affect or blend with the higher priority behaviour. If a low priority behaviour is followed by a higher priority/ or low priority behaviour, it will blend to the new behaviour.

When synchronizing and prioritizing the behaviours of the character, RealActis responsive to external events while considering the physical constraints of the 3D character. For instance, whenever the user smiles or waves at the 3D character, the character smiles back or waves. In addition, the character’s behaviour stays physically reasonable. For instance, body parts do not move too fast or slow and different channels of behaviour behave in synch. . .

The RealAct system includes three priority queues for high, mid and low priority behaviour (Figure ‎3.16). High-priority behaviours need to be performed immediately after the generation. They are usually behaviour responsive to the environment and the user inputs or immediate reactions needed to the interaction scenario. For mid-priority behaviour, even if the behaviour cannot be scheduled to the specific time, it still needs to be scheduled it as close as possible. The low-priority behaviour is usually idle behaviour which does not correspond in a synced way to any outside events so their delay or removal will not affect the perceived experience. Behaviour is inserted to the corresponding queues in multiple behaviour controller modules. The Behaviour-Scheduler then sends the selected behaviour (with the highest priority) to the animation engine. The Behaviour-Scheduler module is responsible for deciding which behaviour is selected. It will check the behaviour queues by order of their priorities. The low-priority queue is checked, only if the high-priority queue is empty (Figure ‎3.17).

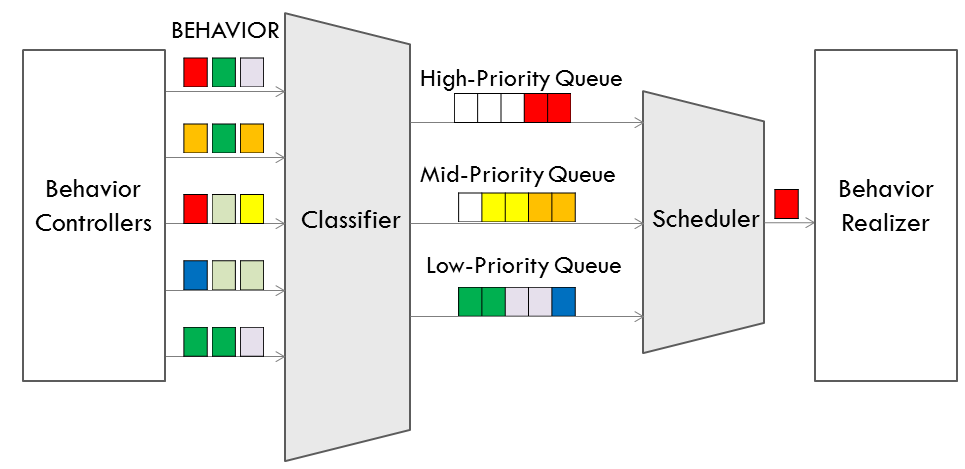


Figure ‎3.16. The RealAct system includes three priority queues for high, mid and low priority behaviour. Behaviour is inserted to the corresponding queues in multiple behaviour controller modules. The Behaviour-Scheduler then sends the selected behaviour (with the highest priority) to the animation engine.

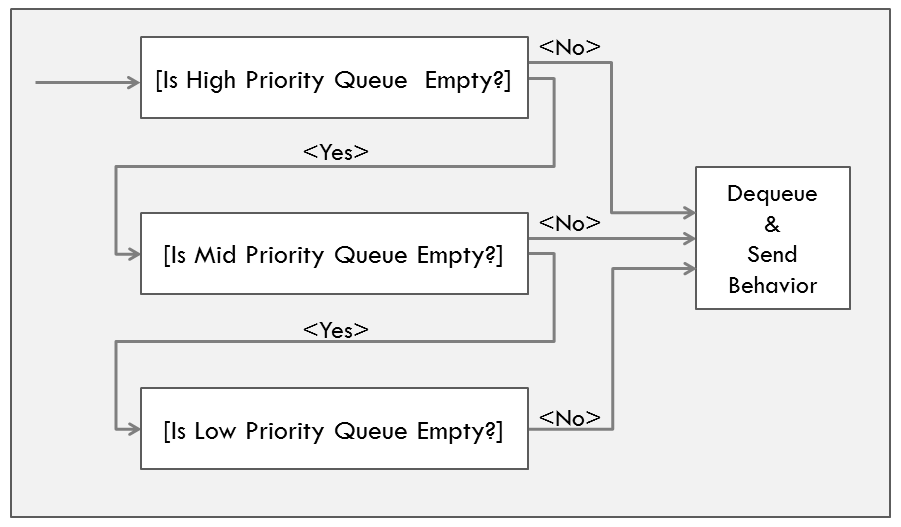


Figure ‎3.17. A demonstration of the RealAct’s Behaviour-Scheduler design (First high-priority queue, then mid and at last the low-priority queue is checked for behaviour commands. If each of the higher queues is empty, the lower priority queue is checked. )

Smartbody animation toolkit is used as the RealAct animation engine [101]. Behaviour Markup language (BMLs) is used to send behaviour commands to the Smartbody character [129]. I am using two approaches for controlling Smartbody character’s behaviour: 1) procedural control: involved joints will maintain until a new behaviour is sent to the character and 2) animations: involved joints return to default pose when the behaviour run their course. For example, since Smartbody’s gaze behaviour is procedural if a character is asked to gaze at a certain target, the gaze will stay on the moving target until the new gaze behaviour command is sent to the animation engine. In order to synchronize multiple behaviours together, I align sync-points provided by BML[129]. Additionally, BML also provides timing constraints which can be used to order the behaviour.

## Attention-Controller Module

A virtual humanoid character needs to be able to direct its attention to people and objects in the environment in a believable and human-like manner [139]. In most of the existing behavioural computational models, attention module is employed for controlling the characters’ gaze to attend a person or an object [139] [143]. In the RealAct design, attention module, for all the modalities, is to prevent idle and other nature behaviour from disrupting the more deliberate and at times immediate reactive and communicative behaviour. In fact, the idle behaviour needs to be paused if an event which requires immediate attention occurs. There are two types of events that require immediate attention: sudden environment changes, and events regarding the scenario of the interaction. For instance, if the user starts to wave at the character, the character cannot show idle gaze behaviour of averting the head; or during the RPS game, it does not make sense if the character averts its gaze when playing a hand.

Additionally, if a behaviour command for a specific joint of the character enters the animation engine right after another command for the same joint, the result will be a blend movement of the first animation to the second one. For example, if the character is showing the rock hand gesture and a scratching head gesture comes after that, the rock gesture blends to a head scratch which is not desirable. The attention module is designed to avoid such unwanted blending. RealAct then via the Attention-Controller module attempts to balance the realism of blended natural, ongoing and complex behaviour with the more deliberate reactionary behaviors.

The attentive body parts in the RealAct’s Attention-Controller can be gaze based, body based or both. If the attention signal only requires the attentiveness of the gaze, other body parts can continue with their idle or ongoing behaviour. The same rule applies to the body. Similar to Sidner et al.’s work, the design leads to three phases for attention: establishing, maintaining and closing the attention [146]. When attention-seeking events occur, establishing the attention occurs by triggering a flag for gaze, body or both. During the maintaining phase, the FSM time-based attention flag will stay on for a fixed period of time and then turns off automatically (closing) unless another attention-request is received. In the maintaining phase, depending on whether it is the gaze or body attention flag that is in the on position, their corresponding idle behaviour pauses until the flag turns off.

## Personality as a Parameter Influencing the System

The RealAct’s personality subsystem is built such that its generated nonverbal behaviour creates the impression of personalities ‘extraversion’ and ‘emotional-stability’ (traits from Big Five). These traits are widely used in affective computing [6] and have biological basis [47]. In addition, there is empirical evidence regarding the link between them and nonverbal behaviour [29] [31]. As depicted in Figure ‎3.1, personality traits affect different parts of RealAct system. People with different personalities tend to filter their emotions differently e.g. extraversion tends to be correlated with expressing the feelings more easily and do less filtering. In addition, personality affects the gestures, postures, and expression of facial emotions. Another aspect through which personality is revealed is the coping mechanisms and emotional reactions of the characters to users’ behaviour. To limit the scope of the research, this aspect of personality is not implemented in the current version of the RealAct.

The RealAct’s hierarchical parameterizedstructure (see Figure ‎3.1) facilitates the encapsulation and separation of the emotion generation, behaviour controllers, and personality parameters. Thus, the designer of the system can choose to incorporate the effect personality in the behaviour modules, or not. Likewise, the designer can select if the emotion generation is affected by the personality or not. How the emotion generation and behaviour controller modules work were explained in the previous section. Here, I review how I implemented the effect of personality traits on emotion generation and behaviour controllers (i.e. gaze, pose, gesture, and face). As a combination of two traits extraversion and emotional-stability, four personality types are considered: High Extraversion-Low Stability (HELS), High Extraversion-High Stability (HEHS), Low Extraversion-Low Stability (LELS), and Low Extraversion-High Stability (LEHS).

### Personality Influence on Emotion Generation

Emotional reactions of extravert and stable characters are different in several aspects, including coping mechanism and reaction to positive and negative feedback. Table ‎3.1, lists differences between the fours personalities, implemented in RealAct, in terms of generation of valence and arousal. For instance, highly extravert-highly stable characters are highly responsive to positive stimuli [32]. In addition, based on psychological studies, extraverts experience more positive emotions in general [41]. Moreover, psychological data shows that extravert individuals’ coping mechanism involves acting rationally and thinking positively [43] while low stable coping mechanism is to withhold and be passive [43]. A coping mechanism is not mentioned in the table since it is not implemented in this version of the RealAct. However, RealAct’s encapsulated framework, with hierarchically-interconnected and well-defined modules, facilitates an incorporation of future components such as the copying controller.

Table ‎3.1. Summarizing the findings from psychology, five dimensions for the effect of personality parameters on the valence and arousal are defined (for emotional valence, initially experienced value of emotion, frequency of chance of valence and reaction to stimuli; and for emotional arousal, initial value and arousal change in response to positive and negative stimuli)

|  | VALENCE | | | AROUSAL | |
| --- | --- | --- | --- | --- | --- |
|  | INITIAL VALUE | REACTION TO STIMULI | CHANGE FREQ | INITIAL VALUE | REACTION TO STIMULI |
| HELS | 0 [41] | High to positive & negative [32] | High [36] | Positive [36] | High [36] |
| HEHS | Positive [41] | High to positive [32] | Low [36] | Negative [36] | Low [36] |
| LELS | Negative [41] | High to negative [32] | High [36] | Positive [36] | High [36] |
| LEHS | 0 [41] | Low to positive & negative [32] | Low [36] | Negative [36] | Low [36] |
| NEUTRAL | 0 [41] | Normal to positive & negative [32] | Low [36] | 0 [36] | Normal [36] |

Figure ‎3.10, shows a Simulink Scope Block which compares the emotional valences generated by the RealAct’s emotion generation module for four personalities. Since, extraverts show stronger emotional responses to the positive feedback (for example, when the character is winning in a game), when the feedback is positive, extraverts’ valence curve has a high exponential rise. However, individuals who are score high in extravertion are not typically sensitive to negative feedback so when feedback is negative, the decrease of valence is not as fast. Low stables, on the other hand, typically show a stronger response to negative stimuli than a positive one [32]. Therefore, the valence value is decreased with a higher rate in response to negative feedback. Low stables experience more negative emotions in general [36].

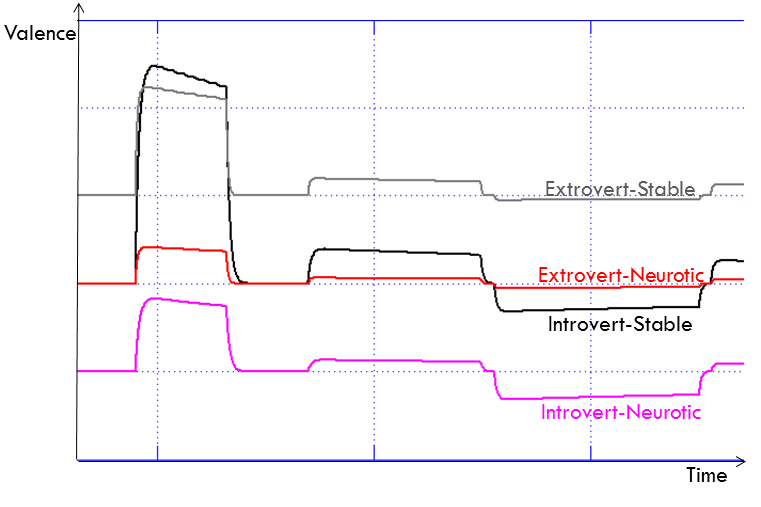


Figure ‎3.10. Comparison between the RealAct’s generated Valence for four different personalities

To confirm that the resulting valence and arousal for four personalities are compatible with the psychology literature, I compare the results with Heller Adoption of Heller & Eysenck Model [159] which offered a map between valence-arousal and Personality traits Extraversion and Neuroticism (emotional instability) (Figure ‎3.11). To be able to compare, I plotted the RealAct’s temporal changes of valence and arousal for four personalities using Simulink 2D plots with x dimension as valence and y dimension as arousal (Figure ‎3.12).

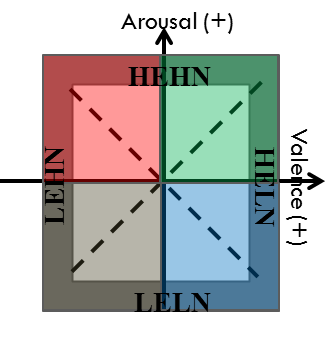


Figure ‎3.11. An adoption of Heller & Eysenck Model (A map between valence and arousal and Personality traits Extraversion and Neuroticism [159])



Figure ‎3.12. Temporal valence changes of different personality types ( Gray: Extravert-Stable, Black: Introvert-Stable, Red: Extravert-Low Stable, Pink: Introvert-Low Stable) using Simulink 2D plots. For all of the plots x dimension is valence and y dimension is Arousal. (need a better figure)

The RealAct system generated valence and arousal appeared to be compatible with Heller’s model in terms of the regions Extraversion and Emotional-stability cover in valence-arousal dimensions. For instance, highly extravert-highly stable individuals seem to experience mostly positive valence and a combination of negative and positive arousal.

### Personality Expressive Gaze

A set of factors were added to the gaze model to control the impression of personality. Due to two distinct natures of gaze behaviour, the personality factors change the parameters of the gaze in one of these ways: 1) some of the factors are multiplied with the parameters of the gaze, to continuously impose the effect of personality on the behaviour, 2) some of the personality factors update the gaze parameters, only once, when the personality is changed (specified by ‘personality-changed’ and ‘VA-changed’ flags in Figure ‎3.13). For instance, the system constantly generates new amplitudes and duration to gaze at a new coordination for a short duration of time. Therefore, a personality factor is multiplied by the duration of gaze each time to increase or decrease the duration based on personality type specified on that time of the simulation. On the other hand, blink frequency is only specified one time and in the start of the simulation. Thus, it is enough to multiply the blink frequency value by a factor if the personality changes during the simulation. Moreover, emotional changes (how aroused or pleased someone is) have definite effects on the gaze behaviour. Although I limited the scope of the research to personality, in future I explore and implement the effect of emotional changes on gaze.

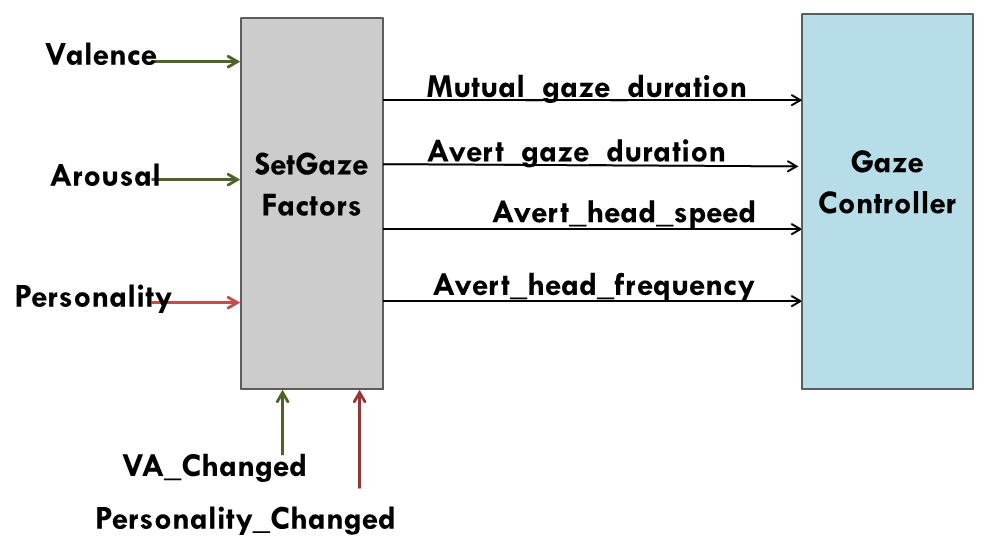


Figure ‎3.13. Personality and emotional state can affect various dimensions of gaze behaviour such as duration, direction and speed of the gaze (To limit the scope of the research, in this version of RealAct’s implementation, effect of emotional valence and arousal on gaze is not implemented. However, in future it will be added to the system.)

The following, statements 1 to 5, are used in the RealAct to calculate gaze parameters. The base for the most of these statements is Eyes Alive model [100].However, I adapted and updated the formulas to be responsive to personality changes, and characteristics of Smartbody animation toolkit. In Eyes Alive model Saccade magnitude, (statement 1), is an inverse of an exponential fitting function of a chance of occurrence (P). I defined animation\_engine\_factor variable so the resulting magnitude maps the character gaze’s degree of freedom (since for the original P value some of the magnitudes were too small to be seen).

(1)

Percent chance of no change from the mutual state (mutual\_cont\_chance) and the percent chance of staying in averting gaze (avert\_cont\_chance) are polynomial fitting functions of elapsed time and are respectively estimated by formula 2 and 3. I used the same second-degree polynomial curves defined in Eyes Alive models to calculate the chance of no change while adding factors to reinforce the expression of personalities. Factors ‘mut\_freq\_facor’ and ‘avert\_freq\_facor’ are set dynamically based on specified personality type. For example, introversion is correlated with a higher chance of staying in a avert gaze state. Thus, if the character is highly introvert, avert\_freq\_facor will automatically set to a higher value by the system, which led to a higher chance of continuing the averted gaze. In addition, the amount of time passed affects the chance of mutual and avert in Eyes Alive gaze model (t). In the RealAct, t is set to the current time of simulation.

(2)

(3)

In my implementation, a Matlab function named ‘GazeFunction’ accepts the calculated gaze parameters as inputs and returns gaze behaviour commands. In the updated Eyes Alive model, saccade direction (named gaze\_dir in 4 and 5) is assumed to be one of the following eight directions: Up, Down, Left, Right, Up-Left, Up-Right, Down-Left, and Down-Right. There is 15% chance of pure directions (Left, Right, Up and Left) and 8% chance of others. In the RealAct, I manipulated the chance occurrence of the directions based on the personality type, utilizing two factors ‘avert\_gaze\_dir\_chance’ and ‘mut\_gaze\_dir\_chance’. For instance, since the literature shows a correlation between introversion and the higher chance of looking down, down-left and down-right [4], I increase the chance of occurrence of Down, Down-Left, and Down-Right, and decrease the chance of Up, Up-Left, and Up-Right.

Duration of the gaze is calculated using the formula 4. Similar to Eyes Alive model, D is assumed to be 2.4 milliseconds per degree and D0 is assumed to 25. Using variables ‘avert\_gaze\_duration\_factor’ and ‘mut\_gaze\_duration\_factor’, Personality affects the duration of averting and mutual gazes. For example, related psychological studies show a correlation between the extraversion and higher duration of the mutual gaze which is set to a higher value of ‘mut\_gaze\_duration\_factor’.

(4)

(5)

Table ‎3.2. This table summarizes the findings from psychology on the expression of two personality traits through the gaze behavior. The columns of the table are Idle (for neutral personality) and four different personalities: Highly Extravert-Low Stable (HELS), Highly Extravert- Highly Stable (HEHS); Low Extravert- Low Stable (LELS); Low Extravert- Highly Stable (LEHS). Six dimensions of gaze behaviour are addressed: head speed, blink frequency, type, direction, frequency and duration of gaze. U, D, C, L, R respectively stand for Up. Down, Center, Left, and Right.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | GAZE TYPE | GAZE DUR | GAZE FREQ | GAZE DIR | BLINK FREQ | HEAD SPEED |
| HELS | Both Mutual & Avert [4][3] | Long to Short[16][15] | High to Low [17] | U/C/L/R [14][9][1][38] | High [105] | High [5] [29] |
| HEHS | More Mutual [4][3] | Long to Med[16][15] | High to Med[17] | U/C/L/R [14][9][1][38] | Low [105] | High [5] [29] |
| LELS | More Avert [4][3] | Short [16][15] | Low [17] | D/L/R [14][9][1][38] | High [105] | Low [5] [29] |
| LEHS | Both Mutual & Avert [4][3] | Short to Med [16][15] | Low to Med [17] | D/L/R/C [14][9][1][38] | Normal [105] | Low [5] [29] |
| IDLE | Both Mutual & Avert [100] | Normal [100] | Normal [100] | U/D/L/R/C [100] | Normal [104] | Normal [100] |

Table ‎3.4, gives a details description of all the gaze parameters set, to generate different gaze behaviour, for a different type of personalities. The direction, duration, frequency of the gaze, and speed of the head movements are controlled to give different impressions of personality. For instance, if the character is low extravert-low stable, it averts its gaze more frequently [17], but for short periods of time [16], mostly gazes down left and right [14][9] and moves its head with a low speed [29].

For the blink behaviour in the RealAct system, I used the available statistical data for human blink behaviour and how emotional state and personality affects that. Based on Itti et al [104], people blink 11.6 times per minute. Additionally, highly aroused people tend to blink more frequently [105]. Thus, same as other parameters of the gaze, I defined a factor to change the blink frequency, during the simulation, based on changes in the personality. Communicative gaze is triggered based on the scenario of interaction. Although these behaviours do not contribute to the expression of personality, they are necessary to have an active realistic interaction. For instance, right after playing the hand in RPS game character looks at the hand of the user to see what they played, or looks at the GUI to see and confirms the result.

### Personality Expressive Poses and Gestures

Compiling the related psychological and embodied research material I reduced the human-based data for computational use by extracting expressivity parameters for describing the physical realization of gestures and poses. For instance, high extravert-low stable character shows a high frequency of self-adaptors, shows twitches, moves its body parts faster, and shows more gestures and poses in general [154][29][30]. Additionally, the compiled data showed that leaning forward communicates a positive attitude and leaning backward communicates a negative attitude [152][153]. Highly stable individuals lean backward reclining while head leans sideway or down. Posture-shifts are more frequent in beginning of the interaction and decrease in later interactions [107].

I introduce five dimensions for expressive idle gestures and posture in the RealAct system: 1) Posture-shift behaviour, 2) Self-adaptor behaviour, 3) Leaning behaviour (lean forward, no lean and lean backward) 4) Twitches (true or false) and 5) Spacious gestures (true or false). These five dimensions can have three different frequencies (high, mid, low) and three speeds (fast, mid, slow). The map between these dimensions and personality types is mentioned in details in Table ‎3.2. For instance, highly extravert-low stable characters show idle poses and gestures more frequently and faster. Highly extravert characters tend to lean back more and use more space when gesturing and posing [107]. Low stable characters show twitches and jerkiness in their poses and gestures [31]. These five expressivity dimensions for gestures and poses were chosen since 1) these dimensions are important in creating the impression of personality, 2) these dimensions have in general or in parts been used in several studies on affecting computing and for creating impression of personality and emotion [44][45][131][137], and 3) it was feasible to synthesize these dimensions using my animation toolkit’s provided features.

Table ‎3.3. This table summarizes the findings from psychology on the expression of personality through gestures and poses. Eight dimensions for gestures/postures are proposed: Frequency of gestures and postures in general, Leaning behaviour, Posture-Shifts behaviour frequency and speed, self-adaptors behaviour frequency, and speed and whether twitching and spacious poses and gestures are present. In order to differentiate the four modelled personalities, the above expressivity parameters are used to adjust the behaviour.

|  | GEST/POSE FREQ | LEAN | W.S. FREQ | W.S. SPEED | S.A. FREQ | S.A. SPEED | TWITCHES | SPACIAL EXTEND |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| HELS | High [154][131] | Back [152][153] | 10% [107]  [30] | High [5][29] | 90% [45][31] | High [29][30] | Yes [57] | Yes [152][131] |
| HEHS | Low [154][131] | Fw [152] [153] | 90% [107]  [30] | High [5][29] | 10% [45][31] | High [29][30] | No [57] | Yes [152][131] |
| LELS | High[131][154] | Back [152] [153] | 10% [107] [30] | Low [5][29] | 90% [45][31] | Low [29][30] | Yes [57] | No [152][131] |
| LEHS | Low [131] [154] | Fw [152] [153] | 90% [107] [30] | Low [5][29] | 10% [45][31] | Low [29][30] | Yes [57] | No [152][131] |
| NEUTRAL | Norm [154][131] | Norm  [152]  [153] | 60% [107]  [30] | Norm [5][29] | 40% [45][31] | Norm [29][30] | No [57] | No [152][131] |

### Personality Expressive Face

As shown in Table ‎3.4, in the RealAct process flow, personality affects the expression of emotion in three ways: intensity of emotion expressed, how much emotion is filtered out, and if the character shows facial twitching. For instance, extraverts tend to express their emotions more freely and do less filtering [38]. On the other hand, there is a correlation between low stability and the presence of involuntary face twitches and head jerkiness [57].

Table ‎3.4. Based on the personality type of the character, the amount for activated action units are adjusted. In addition, based on personality type, if the amount assigned to an action unit is less than a threshold it will be filtered out (considered as internal and not expressed emotional states).

|  | ACTION\_UNIT\_AMOUNT | FILTER THRESHOLD | FACE\_TWITCHES |
| --- | --- | --- | --- |
| HELS | High [57][36] | Low [38] | Yes [57] |
| HEHS | High [57][36] | Low [38] | No [57] |
| LELS | Low [57][36] | High [38] | Yes [57] |
| LEHS | Low [57][36] | High [38] | No [57] |
| NEUTRAL | Normal [82] | Normal [38] | No [57] |

# Rock/Paper/Scissors Game Scenarios

The rock-paper-scissors (RPS) game scenario was designed to provide a chance for interaction between a user (biological human) and a virtual human in real-time with no need to conversation (Figure ‎4.14). The turn taking nature of the interaction is controlled by RPS game module. During the game, the virtual character plays multiple rounds of the rock-paper-scissors game with the user. A GUI (graphical user interfaces) is used for synchronizing the RPS game. GUI is updated based on the states of the interaction (Ready, Go, Hands, Result) (Figure ‎4.13). In the “Go” state, a random hand gesture is generated for the character.



Figure ‎4.13. Graphical User Interface for synchronizing the rock-paper-scissor game played between a user (biological human) and a virtual human in real-time

Gestures and poses used in Rock Paper Scissor Scenario are mainly pre-defined animations that can be grouped, blended and partially dynamically controlled. Some of the poses and gestures used are provided by Smartbody animation toolkit and some are generated by iVizlab’s animation team. Some of the parameters of the animations such as speed, starting and ending time of the movements can be controlled dynamically in the simulation. Personality type, which is an input parameter to this module, affects the speed of the gestures. For example, highly extravert character shows faster rock, paper, and scissors hand gestures.

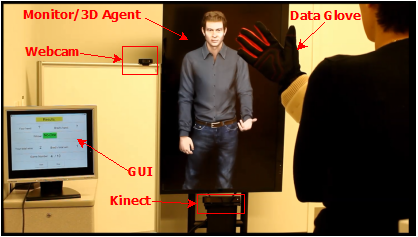


Figure ‎4.14. Experiment setup

In addition to minesweeper and RPS scenario, I developed a few simple scenarios, as a proof of concept for a large range of possible applications of the RealAct system. I developed a set of modules in RealAct to receive streams of inputs of sensors such as Kinect [127], an overhead camera, camera, EMG (muscles facial electromyography) [98], and data from SHORE application [97].SHORE, developed by Fraunhofer research center [97], receives the input stream from a webcam and forward information such as emotional expression of the user’s facial expressions. Using these inputs, the virtual character can react to the facial expressions and gestures of the users such as adjusting his personal space with the user knowing user’s location in the space.

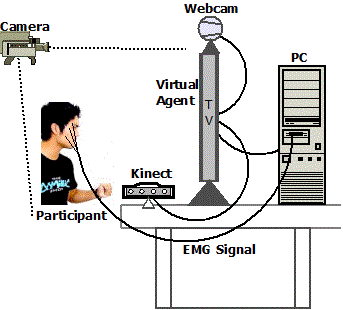


Figure ‎4.15. A participant plays with the virtual character (right) which behaves based on commands in receives from the RealAct system. A video camera records the interaction of both sides.

In this hand on user interaction study, I investigated if a virtual character that shows contingent smile behaviour will create an impression of low social status. Using EMG sensors or SHORE emotional face recognition inputs the system recognized the smile of the users. In the Responsive session, the virtual character mimics the smile of the human participants and in Non-contingent session, the character provides random smiles that are unsynchronized with the human participant’s smile. After each of these sessions, the user evaluates the social status of the character (Figure ‎4.15).

Independent variables (IV) of the presential experiment were 1) intended extraversion with two levels (low and high), and 2) intended emotional-stability with two levels (low and high). By intended, I mean the personality trait that is set as the input parameter of the RealAct system. The selected input parameter then controls the behaviour of the virtual character by affecting different modules of the system such as controllers of gaze, facial expressions, postures, and gestures. Dependent variables (DVs) of all the experiments are perceived extraversion and perceived emotional-stability. In addition to these main dependent variables, I also investigated the correlation between the IVs and other TIPI items (Openness, Conscientiousness, and Agreeableness), and dominance. I like to explore if participants will attribute other personality traits to the gestures, postures and facial expressions of the virtual character, generated by the RealAct system.

# Measurements used for the Real-time Experiment

In this section you can find two measurements used in the real-time presential experiment: TIPI measurement and the presence questionnaire.

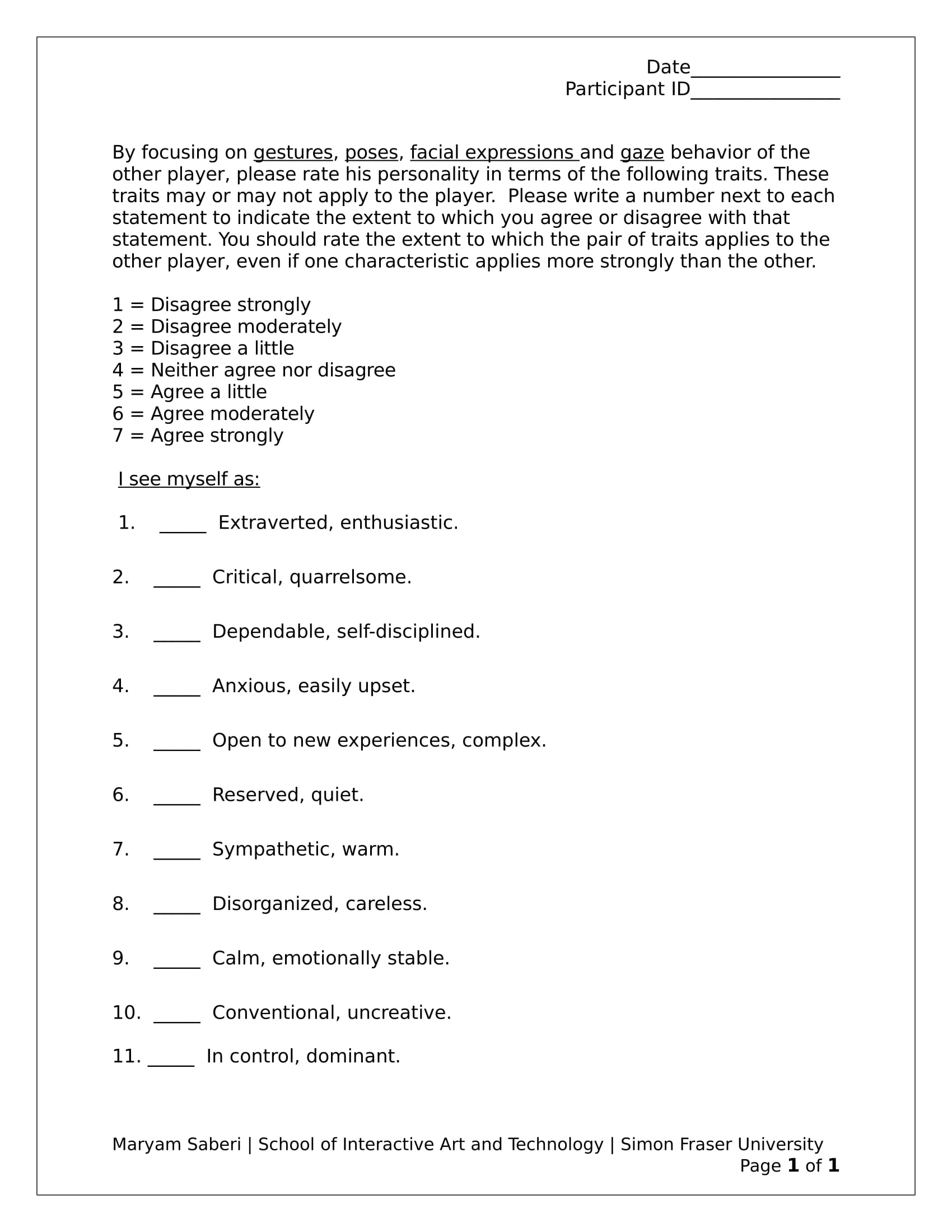


Figure ‎5.1. TIPI scale was used for measuring the personality of the virtual character

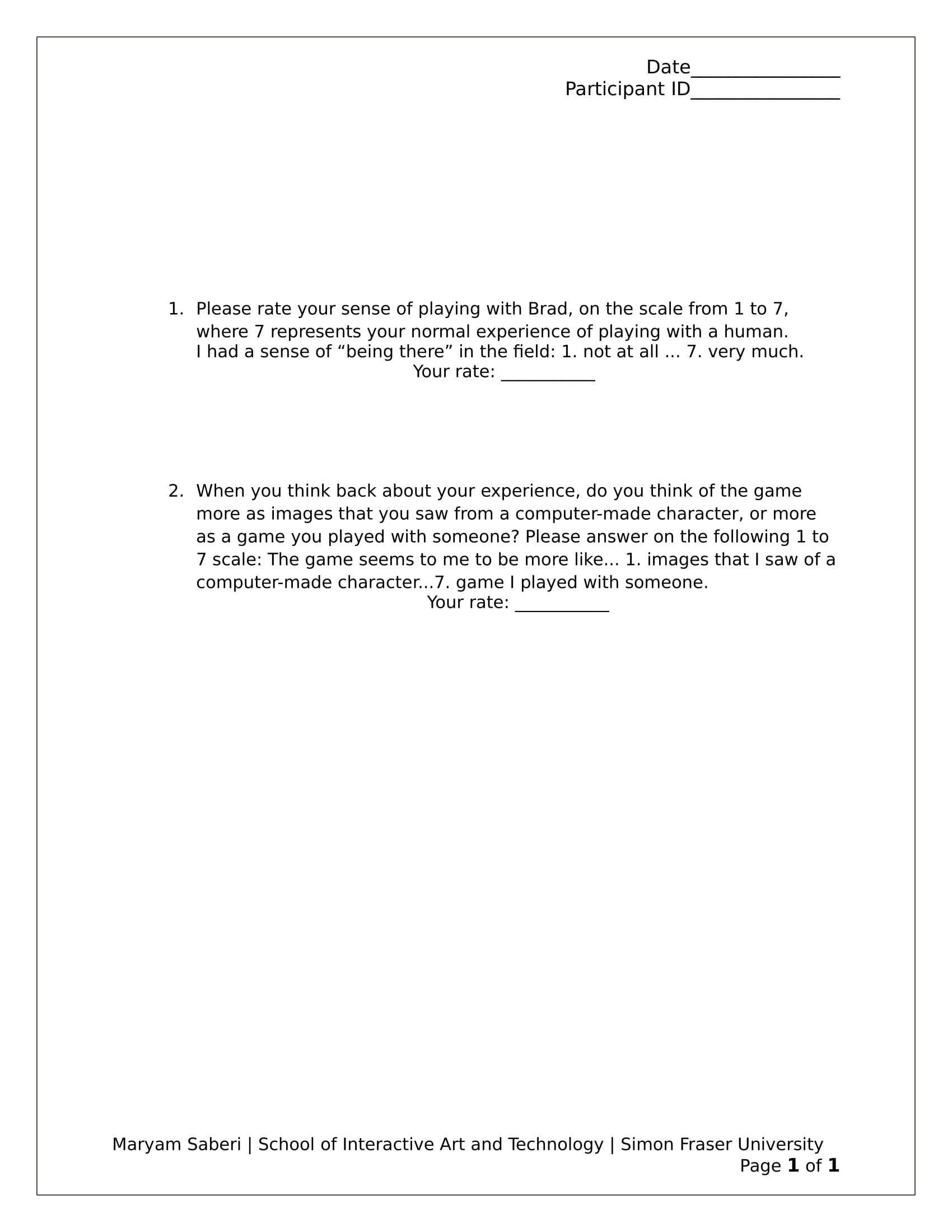


Figure ‎5.2 Presence questionnaire was adapted from [], and used to measure the sense of presence of the participants