HORIZON2020 FRAMEWORK PROGRAMME

ICT – 21 -2014

Advanced digital gaming/gamification technologies

Gamification of Prosocial Learning for Increased Youth Inclusion and Academic Achievement

D4.8

2nd Expressive Virtual Characters
This document extends the description of expressive virtual characters in games for prosocial skills development, focusing on the technical realization of the Virtual Character Controller and surrounding components.

Author(s)  Chengjie Li (KTH), Fangkai Yang (KTH), Christopher Peters (KTH)

Contributor(s)  Robin Palmberg (KTH), Kostas Apostolakis (CERTH), Kosmas Dimitropoulos (CERTH)

Reviewer(s)  Kosmas Dimitropoulos (CERTH)

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Modified by</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>14/11/2016</td>
<td>Christopher Peters (KTH)</td>
<td>Integration of preliminary content into deliverable format</td>
</tr>
<tr>
<td>0.2</td>
<td>01/02/2017</td>
<td>Christopher Peters (KTH)</td>
<td>Integration of background work section</td>
</tr>
<tr>
<td>0.3</td>
<td>22/03/2017</td>
<td>Christopher Peters (KTH)</td>
<td>First completed draft of deliverable ready</td>
</tr>
<tr>
<td>0.4</td>
<td>31/03/2017</td>
<td>Christopher Peters (KTH)</td>
<td>Final version</td>
</tr>
<tr>
<td>0.5</td>
<td>05/04/2017</td>
<td>Pilar Perez (ATOS)</td>
<td>Format review final version</td>
</tr>
</tbody>
</table>
### List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC</td>
<td>European Commission</td>
</tr>
<tr>
<td>FSM</td>
<td>Finite state machine</td>
</tr>
<tr>
<td>NPC</td>
<td>Non-player character</td>
</tr>
<tr>
<td>RPG</td>
<td>Role-playing game</td>
</tr>
<tr>
<td>AI</td>
<td>Artificial intelligence</td>
</tr>
</tbody>
</table>
Executive summary

This document presents background and technical details concerning the application of Expressive Virtual Characters to prosocial game scenarios. It builds upon a previous description of the roles and potentials of virtual characters in prosocial skills development through games to detail the technical components towards their realization. An example demonstrator has been detailed which shows how a small prosocial game featuring expressive virtual characters can be built using the tools described here. The document extends and elaborates upon D4.7 1st Expressive Virtual Characters and has links to D2.2 Prosocial Game Scenarios, D2.6 Prosocial Game Design Methodology, D3.1 User data acquisition and mapping in game environments and D4.3 1st Prosocial Game Mechanics.
Index

1 Introduction ......................................................................................................................... 6
   1.1 Purpose of the document ............................................................................................... 6
   1.2 Scope and Audience of the document ........................................................................... 6
   1.3 Structure of the document .............................................................................................. 6
2 Related Work .......................................................................................................................... 7
   2.1 The Transporters ............................................................................................................. 7
   2.2 Mind Reading .................................................................................................................. 8
   2.3 Peppy Pals ....................................................................................................................... 9
   2.4 Social Relationships Game ............................................................................................ 10
   2.5 SARA ............................................................................................................................ 10
3 The Virtual Character Controller ............................................................................................ 12
   3.1 Introduction and purpose ............................................................................................... 13
   3.2 Overview ......................................................................................................................... 13
4 Core System Components ..................................................................................................... 16
   4.1 Player and Camera Controllers ..................................................................................... 16
   4.2 Face controller ............................................................................................................... 18
   4.3 Body controller ............................................................................................................... 19
   4.4 Behaviour Controller ..................................................................................................... 21
   4.5 2D Sprite Support .......................................................................................................... 23
   4.6 Formation Controller ..................................................................................................... 25
   4.7 Conversation Controller ............................................................................................... 27
   4.8 Copy controller ............................................................................................................... 29
5 Avatar Authoring Tool .......................................................................................................... 31
6 Exemplar: Emotions with Friends ......................................................................................... 33
   6.1 Technical overview ........................................................................................................ 33
   6.2 Scenario customisation ................................................................................................. 35
   6.3 Difficulty level and adaptation ..................................................................................... 36
   6.4 Additional uses of the demonstrator ........................................................................... 37
7 Conclusions ........................................................................................................................... 38
8 References ............................................................................................................................. 39
9 Appendix ............................................................................................................................... 40
1 Introduction

1.1 Purpose of the document

This document provides a detailed technical description of expressive virtual characters for prosocial skills development in games, with a focus on software, with the Virtual Character Controller component at its core, for enabling and accelerating the development of prosocial games involving expressive virtual characters.

It presents:

- Related work concerning existing uses of multimedia and ICT in emotional and social demonstration and skills development with expressive virtual agents and real people.
- The purpose and role of the Virtual Character Controller in the creation of prosocial games containing expressive virtual characters.
- The technical subcomponents of the Virtual Character Controller involved in the creation of virtual characters for prosocial applications.
- Details about a demonstrator game involving expressive virtual characters that has been created using the Virtual Character Controller.

The document extends  and elaborates upon D4.7 1st Expressive Virtual Characters and has links to D2.2 Prosocial Game Scenarios, D2.6 Prosocial Game Design Methodology, D3.1 User data acquisition and mapping in game environments and D4.3 1st Prosocial Game Mechanics, which are referenced throughout.

1.2 Scope and Audience of the document

The dissemination level of this document is public. The final outcome of this deliverable is a second stage description of the role and operational aspects involved in the design and use of expressive virtual characters in prosocial game scenarios.

1.3 Structure of the document

This document is divided into the following sections: Section 2 presents an overview of related work in multimedia concerning the use of virtual characters and real people for the purposes of emotion and social demonstration and skill development. In Section 3, we describe the purpose and role of the Virtual Character Controller in the creation of prosocial games containing expressive virtual characters. Section 4 provides technical details concerning the subcomponents of the Virtual Character Controller which are involved in the creation of expressive prosocial characters. Section 5 gives an overview of an avatar authoring tool for customizing the appearance of virtual characters, while Section 6 presents a concrete exemplar of a small prosocial demonstrator game created using the technical components described in Section 4. The conclusion is presented in Section 7, while the Appendix includes some technical details relating to the labels for behaviours used in the system.
2 Related Work

This section presents a selection of related efforts that apply ICT to the demonstration and enhancement of emotional and social abilities, both in commercial and research contexts. The purpose of this section is to provide background for identifying some of the general technical capabilities commonly required from animated virtual characters based on their historical usage (and the use of recordings of real people) in prosocial contexts. While some of the efforts are aimed primarily at those with Autism Spectrum Conditions, many note also their application to those with learning difficulties and to the more general population.

Please refer to Section 2 of D4.7 1st Expressive Virtual Characters for an overview of the general area of animating expressive virtual characters, which forms a technical basis for realizing many of the core components of the tools and games described in this Section.

2.1 The Transporters

The Transporters is a 3D animated series aimed at children between the ages of 3 and 8 with autism spectrum conditions (ASC). The series consists of 5 minute episodes, each of which focusses on a specific emotion or mental state. The emotions are happy, sad, angry, afraid, disgusted, surprised, excited, tired, unfriendly, kind, sorry, proud, jealous, joking and ashamed, covering the ‘basic’ emotions and more complex, developmentally appropriate emotions, in addition to emotions and mental states deemed important for everyday social functioning.

Children can either watch the episodes in a fixed order or choose which episodes to view, and can engage in a number of quiz games such as matching faces to emotions and matching situations to faces. Parents and caregivers are provided with a detailed guide to the DVD and encouraged to participate in and reinforce the child’s learning. Notably, the characters in the series are mechanistic vehicles that have real human faces placed on them with varying ages, sexes and ethnicities in order to enhance generalization and skills in processing real faces in real situations (Golan et al. 2010).

An evaluation study using The Transporters was conducted with 20 children with ASC aged between 4 and 7 over a four week period (Baron-Cohen et al. 2009). They were tested on their emotional vocabulary and emotion recognition before and after intervention using the series. The study

\[ \text{http://www.thetransporters.com} \]
concludes that using *The Transporters* significantly improved emotion recognition in children with ASC in comparison to two control groups, 18 children with ASC and 18 normal-developing children. Notably, while improvement in the intervention group was seen over four task levels demonstrating generalization of knowledge beyond the situations that they were trained on, generalization to social functioning in real life settings remains an important challenge requiring further investigation. However, anecdotal evidence suggested from the parents of the intervention group suggests that the children were more willing to discuss emotions and more interested in facial expressions.

Potential factors that are mentioned as contributing towards generalization in the game included presentations of emotions and mental states in context; the use of an entertaining and intrinsically motivating media supporting incidental rather than direct learning; and the use of animated vehicles, noted as an obsession for many children with ASC, with real faces which made the series more appealing to children.

### 2.2 Mind Reading

*Mind Reading: The Interactive Guide to Emotions*[^1] is an encyclopedia of emotions that is primarily aimed at people on the autistic spectrum, who may have difficulty recognizing emotions (Baron-Cohen et al. 2002). It also includes groups such as difficult-to-manage children, people with learning difficulties or even those working in dramatic arts.

Based on a thesaurus review of emotion-related words in the English language, 412 distinctly defined emotion words were identified and ranked to estimate the ages at which typically-developing children and teenagers would understand their meaning. Emotion words were then assigned to 24 groups e.g. afraid group.

![Figure 2 – Mind Reading: The Interactive Guide to Emotions](http://www.jkp.com/mindreading)

This information was used to create an Emotions Library consisting of 6 different actors (to support generalization) portraying each of the 412 emotions. Playback consists of a video of their acted facial expression and audio from their vocalisations. An explanation is also provided, through six stories, of the contexts in which specific emotions typically arise. A Learning Center gives guided tutorials explaining the top 20 or 100 emotions and enables the user to engage in quizzes related to matching emotion words and their contexts.

[^1]: http://www.jkp.com/mindreading
emotion words to faces, emotions in intonation with emotion in the face and emotion words to voices. Finally, there is a Games Zone in which player learn about emotions indirectly while playing card games and puzzles.

In a study evaluating Mind Reading (Golan et al. 2006), a group of 19 adults diagnosed with Asperger syndrome and high-functioning autism were tested on the recognition of faces and voices at three levels of generalization. In a first experiment, they used the software at home for 10-15 weeks while a second experiment included weekly group meetings with a tutor. A control group of 22 adults were assessed without any intervention. The intervention group in both experiments performed significantly better than the control group on close generalization tasks, although there were no significant differences between the control and invention groups for distant generalization tasks.

2.3 Peppy Pals

Peppy Pals\(^2\) is a set of apps and e-books designed for children between 2 and 8 years old about emotions, empathy and friendship through storytelling. The scenarios consist of five gender neutral animal friends (rabbit, horse, owl, dog and cat) that children interact with in order to inspire discussions about feelings and empathy. For example, children may be asked ‘How do you think Gabby (rabbit) feels when Sammy (horse) doesn’t share the hay?’ The games fit a point-and-click interface approach.

The aim of Peppy Pals is to help teach kids empathy, emotions and collaboration through humor and hugs and to build bridges between children and parents, teachers and guardians. Notably, Peppy Pals is universal, in the sense that there is no text or language: the animals express themselves through non-verbal full-body and face animations.

An evaluation study involving Peppy Pals (Bohné 2014) involved a pre-study and main study involving twenty-five children between the ages of 4 and 6 and four parents from two different preschools in Sweden. The main-study focused on two of the scenarios from the game that had storylines (the Pie and Slide scenarios). Observation and interviews were used to collect data concerning the interactions between the children and the game. The results of the study indicated that children really enjoyed playing the game and were able to clearly articulate the emotions of the animals and understood the emotions in the context of the scenarios. However, the study reports that they did

\(^2\) http://peppypals.com/
not have to actively use emotions in order to play the game and often tried a trial and error approach to playing the game, leading Bohné to compare the play style to an interactive book where the purpose of the interaction is to advance to the next part of the story.

### 2.4 Social Relationships Game

Ho and Dautenhahn (Ho and Dautenhahn, 2009) extend previous work (Watson et al. 2009) towards the development of a Virtual Learning Environment and intelligent virtual agents for facilitating and better understanding the development of social relationships between children. Interactions take place in a playground and canteen scenario and allow the investigation of the social structures that emerge from play, especially with respect to group formations e.g. F-formations (Kendon, 1977) and social distance (Hall 1966). The aim is to allow users to explicitly perceive and explore the social dynamics emerging from user-agent and agent-agent interactions.

![Figure 4 – Social Relationships Using IVAs with Social Group Dynamics](image)

### 2.5 SARA

The SARA project (Arellano et al. 2014) is based on an open-source agent platform for rapidly creating and prototyping animated virtual characters using FAT (Facial Animation Toolset) and the Facial Action Coding system (FACS). Automatic speech recognition and synthesis capabilities are provided by SemVox and SVOX respectively, third-party software integrated in the framework. Of particular interest in the project is the use of stylized virtual characters that enable psychologists to simplify the detail in different regions of the face in order to adapt them to the feedback of users.

![Figure 5 – The SARA project brings together facial animation, abstractions in the form of non-photorealistic rendering, and research on Autism (from [Arellano et al. 2014])](image)
A notable study conducted using this framework include DECT, the Dynamic Emotion Categorization Test reported in (Rauh and Schaller 2009), which was an experiment using human participants and virtual agents to examine the ability of emotion recognition with dynamic physical stimuli in children with autism.
3 The Virtual Character Controller

In this section, we describe the usage and implementation of virtual characters as components in games surrounding prosocial skills. An important note is that the focus of this work is not to provide a single prosocial scenario or set of characters, but rather a development kit for creating a wide range of scenarios surrounding prosocial activities that support a large variety of character appearances: from cartoon-like to realistic, 3D to 2D, humanoid to fantasy creatures. The Virtual Character Controller is the primary technical component providing this functionality and it will be described in this document in detail. Two general examples of the types of prosocial scenarios that can be created are shown below.

![Figure 6 – Prosocial scenario example 1: Maintaining social distance. The character in dark clothes moves backwards in order to maintain a comfortable social distance with respect to the other character.](image1)

![Figure 7 – Prosocial scenario example 2: The ‘cold shoulder’. The character in dark clothes orients their back towards the other in a display of impoliteness.](image2)

In the first example, the virtual character controller may be used to ensure that characters maintain social distance with respect to each other. In the second scenario, a character orients its body in order face away from a second character, in what could be interpreted as demonstrating impolite behavior or an unwillingness to engage in conversation. This document does not concern the pedagogical utility of such scenarios in the wider context of classroom teaching and learning, but rather the technical realization of these behaviours.
3.1 Introduction and purpose

The *virtual character controller* is a technical toolkit for the development of virtual character behaviours aimed specifically towards prosocial game scenarios. The virtual character controller has two main objectives:

I. Enabler

Allow users with varying levels of technical expertise to create sophisticated prosocial games involving 3D virtual characters. The development of requires the development of basic animation skills for the faces, bodies and formations of virtual characters. Learning those skills can be time-consuming and daunting, even for technically proficient developers. The virtual character controller aims to substantially decrease the complexity of integrating sophisticated 3D virtual characters, face and body animation, and formation control into prosocial game scenarios with respect to the default options provided by game engines such as Unity 3D.

II. Accelerator

Accelerate the development of PSL games involving virtual characters for those who have technical expertise in virtual character animation. The virtual character controller aims to substantially decrease the amount of time required to create sophisticated 3D virtual characters, face and body animation, and formation control for use in prosocial game scenarios. An important rationale for this is to allow more time to be devoted towards the design of pedagogical game scenarios and to take the emphasis away from fundamental technical development work involved in programming the animation of expressive virtual characters.

We hope that both of the objectives above will allow game developers and other prosocial scenario developers to put more emphasis on pedagogical game design issues and also to be able to consider the use of more sophisticated technologies that may have scientific and pedagogical importance in scenarios that involve prosocial skill development.

Throughout the document, we use the *MCS Female character* by *Morph3D* as an example embodiment to demonstrate various possibilities of the system. This model was chosen since it has a good degree of visual realism, has detailed 3D face and body animation capabilities and is also available in a free version. Ultimately, the appearance and nature of specific characters for a prosocial game will be at the discretion of the game designer: the virtual character controller contains a number of ways to plug animations from different character types into the system. These are described in more detail below.

3.2 Overview

Operationally, the virtual character controller manages the control of the face, full-body and formation behaviours of individuals and groups of virtual characters. It is especially focussed towards behaviours relating to prosocial scenarios. The controller also supports a number of peripheral features required for working with virtual characters, such as camera, player and navigation control, and a number of special features, such as the *copy controller*, which enables and determines how sets of human facial expressions are to be mapped onto the faces of virtual characters in real-time.

Since the style and needs of each game differ, the expectation is that game designers will eventually design, rig and create predefined animations for their own characters that they will associate with

---

4 https://www.morph3d.com/
predefined behavior labels (see Appendix) in the asset. The process for doing this involves binding the facial expressions and full body animations of the new characters into the Virtual Character Controller. This process is described in each of the related subsections for the face and body. A further section details the process for the use of 2D sprites, since this requires a different approach to 3D face and body animation.

![Diagram of components related to the virtual character controller](image)

**Figure 8 – Overview of components related to the virtual character controller**

The virtual character controller consists of the following components:

**Player and Camera Controllers:** These enable the player to be controlled by mouse and keyboard, also the selection of views in the game and methods for the player to shift between them.

**Face Controller:** Play face related expressions or movements based on the mesh-based blendshapes through predefined PSL_FACE_ labels (see Appendix) and facial expressions database.

**Body Controller:** Play 3D body animations using Unity animator component through predefined PSL_BODY_ labels (see Appendix).

**2D Sprites Controller:** Play 2D Sprite animations using the Unity animator component through predefined PSL_2DSprites_ labels.

**Behaviour Controller:** This is a combination of both the face controller and body controller. It invokes the corresponding controllers based on the requirements of the developer as specified in a behaviour profile database. Each agent has a behaviour controller with a behaviour profile that defines their behavioural tendencies. Behavior profiles may be loaded from XML files and edited externally to the system. A default behavior profile is included as standard in the asset, but more can be defined by the developer.
Formation Controller: The formation controller initiates the positions of each virtual character in groups and alters their positions based on different behaviour states.

Conversation Controller: This consists of an automatic conversation model in which speakers and listeners use a Finite State Machine (FSM) to simulate the transaction between seven conversational states.

Relationship Manager: Defines agents’ relationships which affect their interactions, for example, their behaviours when they join a group.

Copy controller: Defines support for and mapping between the facial expressions of humans in real-time and the faces of virtual characters.

Each of these controllers is described in more detail in Section 4.
4 Core System Components

4.1 Player and Camera Controllers

The Player controller specifies the control of the agent through the mouse and keyboard. In the third-player and first-player views (see Camera Controller), the agent is controlled by keyboard. Navigation is done by the player who controls the agent. On the other hand, if the view is shifted to the perspective view, the motion control is done by mouse clicks, i.e. left click to set the destination.

![Player Controller Parameters](image)

Figure 9 – Parameters associated with the player controller.

The camera controller allows the player to shift among different views in the game. There are four views in the game as shown below.

![Camera Views](image)

Figure 10 – Example of four camera views within the game. From left to right, top to bottom: Third player view, first player view, free roaming and perspective view.

Note that if the formation controller takes the control of the player, it will also take the control of camera controller.

The various camera types can be enabled through the PSL_CAMERA_VIEW_TYPE settings as follows:
<table>
<thead>
<tr>
<th>Agent inside the conversation</th>
<th>PSL_FIRSTPERSON_CAMERA</th>
<th>First person camera follows the agent which can be rotated by mouse. In this camera view, the agent cannot be controlled by the user.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent outside the conversation</td>
<td>PSL_THIRDPERSONORBIT_CAMERA</td>
<td>Third person camera follows the agent from behind which can orbit the agent controlled by mouse. In this camera view, the agent is controlled by the keyboard.</td>
</tr>
<tr>
<td></td>
<td>PSL_PERSPECTIVE_CAMERA</td>
<td>Third person camera follows the agent from top-down perspective view which can orbit the agent controlled by mouse. In this camera view, the agent is controlled by mouse click.</td>
</tr>
</tbody>
</table>
4.2 Face controller

The body animations of virtual characters are controlled by the Unity animator through the body controller. The 3D animated facial expressions of virtual characters are controlled by mesh-based blendshapes. In order to add facial expressions that can be played on 3D characters, it is necessary to complete a binding process, which maps the blendshapes defined for a specific character to a set of predefined PSL facial expression labels (see Appendix). The PSL_FaceExpressionLookupTable contains the mappings between predefined PSL facial expression labels and user defined facial expressions (see Appendix).

Another important component is the PSL_FaceExpressionDataBase, which is the database where the values of facial expression blendshapes are stored. When PlayFaceAnim(PSL_FACE_LABEL) is called, the related face expression is found according to the PSL_FACE_LABEL from the lookup table and then the values of blendshapes with the same name of the face expression are loaded into the character from the expression database.
The body animations of virtual characters are controlled by the Unity animator through the body controller. The main part of the process involves binding the animations that are specific to the virtual characters into the system through the PSL_BODY_ labels (see appendix). The process for doing that is as follows. The PSL_BodyAnimLookupTable contains the mappings between predefined system labels (see Appendix) and the user-defined body animations. The source animations should have the same names as those defined by the user in the lookup table. Initially, the CSV_BodyLabelLoader.Load function loads the label mapping into a runtime body label lookup database. When PlayBodyAnim(PSL_BODY_LABEL) is called, the corresponding animation with label PSL_BODY_LABEL is activated based on the lookup table. All the animations should be attached to the animation controller of the character beforehand.
The following example uses the default *MCS female character* to illustrate how desired body animations can be played. Firstly, all of the animations should be attached to the Animator Controller of the character manually before the program runs.

<table>
<thead>
<tr>
<th>LABEL</th>
<th>ANIMATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSL_BODY_Neutral</td>
<td>Basicidle</td>
</tr>
<tr>
<td>PSL_BODY_HappyLowIntensity</td>
<td></td>
</tr>
<tr>
<td>PSL_BODY_HappyHighIntensity</td>
<td>Greeting</td>
</tr>
</tbody>
</table>

*Figure 15 – Example of the Body Label lookup table (left) and resultant animation ‘Greeting’ (right) which has been bound to the PSL_BODY_HappyHighIntensity animation label.*

The name of the animations can be defined by the developer, but the body labels lookup table should be filled in with the same as the real animations. The example above shows the binding for a developer defined animation called ‘Greeting’, which is bound in the system to the PSL_BODY_HappyHighIntensity animation label.

Thus when `PlayBodyAnim(PSL_BODY_HappyHighIntensity)` is called by the system, the body animation with the label PSL_BODY_HappyHighIntensity (i.e. “Greeting” animation) will be played.
4.4 Behaviour Controller

The Behaviour Controller manages the control of the face and body of virtual characters. It also plays a role in the type of animations used during encounters with others characters in formations. A behavior profile defines the behaviours (i.e. animations) that virtual characters make in different situations, thus allowing different characters to have different reactions to similar events. They define the related animation labels (and thus, animations) that will be called in specific interaction situations, phases, relationships and states.

Each virtual character is assigned a single behavior profile. Profiles are stored in the Behaviour Profile Database. The user can customize and add their own profiles to the database according to some pre-defined rules in the system. An example of part of the behavior profile for a virtual character during the joining phase of a social interaction (i.e. when a character joins a group of characters) is shown below. While default behavior profiles exist for the characters in the system, new profiles may be created and it is expected that the game developer will tailor the behavior profiles according to the needs of their specific scenario.

It should also be noted that all references in the behavior controller are to standard PSL_FACE and PSL_BODY labels. Therefore behaviours defined in the behavior controller have a level of indirection with respect to the specific characters used and their animations, with the result that the same behavior profiles can be applied to different virtual character embodiments.

Figure 16 – Flow diagram of the Behavior Controller.
Figure 17 – Example of the Join Group behavior definition for an agent (left) and corresponding animation (right) of a high intensity happy face animation and arm wave when the agent has a positive relationship with the other. Details are also shown for the same Join Group behaviour when there is a negative relationship with the other agent, in which case a facial expression of contempt is displayed.

4.4.1 FSMs for character control

A stack-based finite state machine (FSM) is implemented to control the states of each character. Each state is linked to a separate function which is called when the character is in that state and associated events are triggered. A template for the stack-based FSM is provided and the developer only needs to modify trigger events and the state functions if they wish to make changes to the system. An example of the FSM relating to joining a group is shown below.

Figure 18 – Example of the Join Group FSM defined for the agent wishing to join a group.
4.5 2D Sprite Support

2D animated sprites are also supported by the system. In many cases, the use of such sprites will enhance the performance of the character animation system, especially on platforms that do not have substantial graphics capabilities. In many cases, the use of 2D sprites will provide a comparative level of visual quality to their 3D counterparts, although the flexibility of such systems is more limited.
The 2D Sprite animations are manipulated by the Unity animator through the 2D Sprites controller. There is one important data file: `PSL_2DSpritesLabelLookupTable` which contains the mappings between the predefined sprite labels and the user-defined 2D Sprites. Initially, the `CSV_2DSpritesLabelLoader.Load` function is called. It loads the label mapping into a runtime 2D Sprite labels lookup database. When the function `Play2DSpritesAnim(PLS_2DSprites_LABEL)` is called, the corresponding animation with label `PSL_2DSprites_LABEL` will be activated based on the lookup table. All the animations should be attached to the animation controller of the character beforehand.

![Figure 21 - Illustration of a 2D animated sprite. In some cases, 3D characters and the associated processing costs for rendering and animation can be reduced through the use of 2D sprites.](image)

Here is an example of how 2D sprite animation can be played for the MCS female character. Given a set of sprites for a happy animation that has been created by the developer (here we assume the animation has been named `HappyLow`), an empty game object is created in the scene. Create a new animation with the name `HappyLow`. Drag all the sprites into the clip editor and tune samples rate to make a proper speed `HappyLow` animation.
Then attach one sprite to the Sprite Renderer as the starting sprite of the animation. Fill the 2D Sprites lookup table, assigning the name HappyLow to the label PSL_2DSprites_HappyLowIntensity.

4.6 Formation Controller

The formation controller consists of a group generator and an interaction controller: The group generator manages aspects of the group such as the formation type and range at which a player is considered to join the group. The interaction controller manages the processes relating to the player joining, leaving and maintaining interaction with a group of virtual characters.

4.6.1 Group controller

The developer can set a list of parameters to initialize a group as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of agents in this group.</td>
</tr>
<tr>
<td>Radius</td>
<td>Radius of group.</td>
</tr>
<tr>
<td>Detect Range</td>
<td>Range to detect new coming agent who can possibly interact with this group.</td>
</tr>
<tr>
<td>Formation Type</td>
<td>The type of the formation, i.e. circle or semi-circle.</td>
</tr>
</tbody>
</table>

A number of predefined formation templates are supported by default in the system. They are set according to the PSL_FORMATION_TYPE variable. The available formation templates are as follows:
<table>
<thead>
<tr>
<th>PSL_FORMATION_FULL_CIRCLE</th>
<th>Generated formation with agents standing in a circle.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSL_FORMATION_SEMI_CIRCLE</td>
<td>Generated formation with agents standing in a semi-circle</td>
</tr>
</tbody>
</table>

The initial positions of each agent in the group are determined by evaluating the similarity between randomly positions (with constraint) and real data. The positions are generated repeatedly until the similarity is below threshold.

### 4.6.2 Interaction controller

There are three main states in the interaction between new coming agent and the group: join, leave, and maintain. The interaction controller takes control of the new coming agent (if it is previously controlled by the player). The transition between each state is accomplished by choosing different options in the pop-up dialogue and buttons.

![Figure 23 – Virtual characters in a group make space for the new character.](image)

The interaction controller acts not only as a trigger for the behavior controller, but also handles the dynamics of the inner group. For example, in the join state, if the agent tries to join a position in the group formation in which there is not enough space, the agents in the group will move aside to make space. After the agent leaves the group, other agents in the group will move back to the previous positions.
In the *maintain* state, the developer can set behaviors of each agent through the behavior controller. In the default case, the introduction behavior is played in which a character in the group introduces the newcomer. In this mode, the user is prompted to select two characters in the group who will engage in the introducing behavior: the first selected character introduces the second one. By default, two close-ups of the faces of each character are displayed on the screen in order to clearly show the facial expressions being made during the introduction process. Both face and body animations also play in the main scene window.

### 4.7 Conversation Controller

The conversation controller manages the turn-taking behaviours of virtual characters that are engaged in a conversation in a group. It offers two options of control: *automatic* and *user defined*.
4.7.1 Automatic control

This is applied when a player is not actively engaged in interaction with a group of characters (when they are animated in the background for example). In this case the conversation of the group is managed automatically. The type of behaviors each character makes, order in which each takes its turn to speak, duration for which it speaks, and so on, are determined by a turn-taking conversation model based on the work of Ravenet et al. 2015. This model is a joint of speakers and listeners using Finite State Machine (FSM) to simulate transitions among seven conversational states. These states and transitions between them depend on the attitudes between each agent and the rest of the group. Attitude is depicted on a two-axis space: status and affiliation. The status axis ranges from submissive attitude to dominant attitude, while the affiliation axis ranges from hostile to friendly. The attitude of each agent triggers the transitions between the different states. A schematic and in-game view of the turn-taking FSM transaction framework is shown below.

![Diagram of the conversational turn-taking FSM model from Ravenet et al. 2015 (left) and an image of the model in operating within a small group. The automatic model is employed for fully automated groups of virtual characters, such as those in the periphery or background of an ongoing game scenario.](image)
4.7.2 Manual control

Once the player joins a group of characters, the automatic turn-taking model is switched off and a user-defined conversation is activated. The behaviors, order to speak, speech content, and so on are based on pre-defined data from the behavior controller.

4.8 Copy controller

The copy controller manages the translation of human motion from an input data source and analysis module to the realization of a similar or alternative behavior on the virtual character in real-time. For example, the face motions of the virtual character may be driven from high-level emotion labels derived from analyzing the face of the user through a web-camera. Examples of a direct mapping of labels are shown below, in which expressions of happiness and anger on a user face are mapped to predefined animations of happiness (PSL_FACE_HappyHighIntensity) and anger (PSL_FACE_AngryHighIntensity) on the face of the virtual agent, both in the close-up view and in the situated 3D environment.

<table>
<thead>
<tr>
<th>Input emotion labels</th>
<th>Virtual Character emotion labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neutral</td>
<td>PSL_FACE_Neutral</td>
</tr>
<tr>
<td>Angry</td>
<td>PSL_FACE_AngryLowIntensity</td>
</tr>
<tr>
<td>Angry (high intensity)</td>
<td>PSL_FACE_AngryHighIntensity</td>
</tr>
<tr>
<td>Sad</td>
<td>PSL_FACE_SadLowIntenstiy</td>
</tr>
<tr>
<td>Sad (high intensity)</td>
<td>PSL_FACE_SadHighIntensity</td>
</tr>
<tr>
<td>Surprise</td>
<td>PSL_FACE_SurpriseLowIntensity</td>
</tr>
<tr>
<td>Surprise (high intensity)</td>
<td>PSL_FACE_SurpriseHighIntensity</td>
</tr>
<tr>
<td>Smile</td>
<td>PSL_FACE_HappyLowIntensity</td>
</tr>
<tr>
<td>Smile (high intensity)</td>
<td>PSL_FACE_HappyHighIntensity</td>
</tr>
</tbody>
</table>

Table 2 – Example of the Behaviour Mapping Table, which maps input emotion labels to the corresponding emotion labels of the virtual character.

The Copy Controller does not define a specific module for user behavior analysis, but rather it is chosen and added to the system by the developer as third-party software. Note that the capabilities of detected expressions and the robustness with which they are detected largely depends on the capabilities of the analysis software used. An example of a tracker that has been used with the system is Affdex SDK⁵, which has the option of a basic free license.

In order to support a variety of possible prosocial game scenarios, a Behaviour Mapping Table defines the mappings from input emotion labels to the behaviors that are realized by the character. Thus, it is possible for the developer to define mappings that vary the input and output modalities, for example, to render the user’s facial expression as a body motion with a corresponding emotion by the virtual character. Other possibilities include mapping multiple input user expressions onto

⁵ http://developer.affectiva.com/v2_3/unity/
fewer or more intense/subtle emotional expressions and to map specific user facial expressions to alternative emotions to be expressed by the character (for example, an angry facial expression made by the user becomes a happy facial expression made by the character).

Figure 28 – Example of the copy controller mapping facial expressions of happiness (top) and anger (bottom) between a user and the face of the virtual agent in both close-up and perspective views in real-time.
5 Avatar Authoring Tool

Along with the virtual character controller, a simple authoring tool has been developed to allow ProsocialLearn game developers to create assets with different looks for use in their games. As was explained in D4.7, the core avatar authoring tool is based on the RAAT JavaScript 3D library (Apostolakis and Daras, 2013), which is an extension upon Three.js\(^6\). RAAT includes support for standard node.js\(^7\) platform setup. The REVERIE Avatar Authoring Tool (RAAT) is a JavaScript-based 3D library of algorithms, tools and templates designed for the creation and deployment of web-based 3D character creation applications. The library accommodates for interchangeable character 3D assets, including meshes and textures, providing the means to switch between parts, customize specific parts and select specific part textures. One of the major features incorporated is an encapsulated framework for generating character face textures from user uploaded face images, called the Photographic Texture Composer. This component receives an image of the user’s face as input and can, on demand a) generate an appropriate texture for a 3D character’s face mesh asset; b) re-configure the asset’s geometry by structuring the mesh vertices to better resemble the appearance of the user’s face; and c) appropriately recolor the character’s full body skin tone to match the calculated histogram of the user’s skin, resulting in a seamless matching of varied texture interchangeable body parts.

To use the avatar authoring application, users can visit the dedicated portal via a standard, WebGL-compliant web browser\(^8\). Users are greeted to a realistic 3D view of a classroom, to which an avatar

\(^6\) http://threejs.org/
\(^7\) https://nodejs.org/
\(^8\) As WebGL is part of the Khronos Group (https://www.khronos.org/) rather than the W3C (http://www.w3.org/), some web browsers are known not to be compatible with its functionality while
blank “canvas” can be added following the instructions on the tool GUI. This GUI, or Toolbox, is a set of options that offers users the means to a) create a new character; b) cycle through available face/body customization options; c) make changes to the current avatar style i.e. change clothing/apparel; d) character export; and e) information about the RAAT and developer credits. A screenshot of the application (visuals under development) can be seen in Figure 29.

others might offer partial or full support, depending on graphics drivers, OS and platform. Google Chrome and Mozilla Firefox have offered sufficient WebGL and 3D Canvas graphics functionality for desktop use for several ongoing iterations of their products, while other browsers might offer limited or full support in the near future. Mobile browser support has been a work in progress and, at the time of writing these lines, is being offered for several popular browser apps’ latest versions. A complete list of the latest supporting desktop and mobile browsers can be retrieved in Wikipedia’s WebGL entry: http://en.wikipedia.org/wiki/WebGL.
6 Exemplar: Emotions with Friends

6.1 Technical overview

An initial demonstrator version of the Emotions with Friends game was presented in D4.7 1st Expressive Virtual Characters, 4.3.1. The demonstrator concerns the recognition of social cues and emotions based on the expressions of virtual characters: children observe a face or body expression made by a virtual character and must then subsequently select the emotion word that they feel best describes the expression they just witnessed.

![Program flow in Emotions with Friends](image)

**Figure 30 – The program flow in Emotions with Friends.** At program start, the trials to be played are loaded in XML format. They include details about the emotion labels to be used, which are automatically placed by a layout generator, and the virtual character, which is automatically loaded into the scene with the selected animation.

The initial pedagogical aim of the demonstrator has been to raise awareness and discussion possibilities with students about face and body expressions and to enrich their emotional vocabulary i.e. through the use of a range of emotion labels. However, the longer term goal within the project is to investigate the potentials of the game for actively developing the emotion and social recognition skills of students.

Continued development of the game has focused on optimising its performance. This has been accomplished especially through the use of 2D Sprite animation control (see Section 4.5) instead of real-time 3D characters. A number of other improvements have been made, such as allowing developers and users to customise their own scenarios and adding more possibilities in relation to automatic difficulty adjustment as a basis for personalized adaptation.
Figure 31 – Example of a trial within Emotions with Friends. A virtual character in the center of the screen makes a facial expression. The player must choose the nearest matching emotion word from a selection of labels surrounding the character. The number and content of labels can be customized in order to create more difficult games.

6.1.1 Database

An important aspect of the game is that the answers given by players of the game are ranked with respect to the answers already present in a database of previous answers. Rather than attempting to provide specific judgment about whether an answer made by the player is correct or incorrect, the player instead sees how their answers ranks with respect to the answers made by their peers. This peer group could consist of the student's class, adults from the same region or country as the student, or even qualified behaviour experts. The content of the ranking database is therefore open to many possibilities and focusses on supporting and provoking discussions between students and teachers concerning their choices.
Figure 32 – Example of the ranking screen in Emotions with Friends which is displayed after the player has selected an emotion label. In this case, the player chose happiness, which was the highest ranked answer in the database for that facial expression.

6.2 Scenario customisation

It is now possible to customise the scenarios that are shown in the game, allowing anybody (student, teacher, developer) to specify a scenario. The scenario creator can select a set of candidate emotion labels in any language and a corresponding animation (including from a set of predefined standard animations – see Appendix) that will be played by the virtual character. Answers for the scenario are provided by having a group (e.g. teachers) play the game in order to populate the database. The internal format of the trials is shown in Figure 33.
Figure 33 – Format for the definition of each trial in the game. Labels refer to emotion words in the local language in which the game will be played, while the animation relates to the animations supported in the system by the virtual character controller (see Appendix). User defined animation can be added if desired.

6.3 Difficulty level and adaptation

The demonstrator has also been used to provide a test bed for techniques related to varying the challenge level for individual users. This is a prerequisite for enabling the game to adapt to the skill levels of individuals, maximising learning potential and alleviated boredom. D4.7 1st Expressive Virtual Characters described a number of ways in which the complexity of tasks involving virtual characters could be automatically altered, including behaviour complexity and intensity and appearance complexity. Other ways in which the difficulty level of the game can be altered include the number of emotion labels to choose between and their relationship with the animation being made.

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embodiment</td>
<td>Cartoon-like versus realistic faces and bodies.</td>
</tr>
<tr>
<td>Variety</td>
<td>The number of faces and bodies of the same embodiment type used for playback.</td>
</tr>
<tr>
<td>Complexity</td>
<td>Basic versus complex emotional expressions.</td>
</tr>
</tbody>
</table>
6.4 Additional uses of the demonstrator

The demonstrator has additionally been used to:

1) Investigate the viability of prosocial games that use virtual characters for emotion and social cues recognition with teachers and children. Especially, to investigate what aspects of a lesson should be encapsulated in the game itself and which should be external to the computer e.g. as part of a larger classroom activity taking place away from the computer.

2) Allow us to investigate the challenges involved in creating a credible pedagogical tool that is also viable in its own right as a game. More specifically, implement and obtain feedback on some of the different issues surrounding skills development in the classroom, such as repetition, reward and generalisation.

3) Provide a concrete working example of a prosocial game focused on virtual characters as a basis for technical requirements analysis in the development of the Virtual Character Controller. The development of many aspects of the Virtual Character Controller were defined by the goals of easing and accelerating the creation of a game such as Emotions with Friends.

<table>
<thead>
<tr>
<th>Intensity</th>
<th>Subtle versus exaggerated expressions.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>Real-time versus slow motion playback of expressions.</td>
</tr>
</tbody>
</table>

Table 3 – Five character animation factors that may be automatically linked to difficulty level to support automatic adaptation to individuals and their skill levels. Note that the number of labels and their type also influences difficulty, but these have not been included here as they are external to the animation characteristics of the virtual character.
7 Conclusions

This document has presented the background and technical details concerning the application of Expressive Virtual Characters to prosocial game scenarios. It builds upon a previous description of the roles and potentials of virtual characters in prosocial skills development through games to detail the technical components towards their realization. An example demonstrator has been detailed which shows how a small prosocial game featuring expressive virtual characters can be built using the tools described here.
# References


Baron-Cohen S., Golan O., Ashwin E. Can emotion recognition be taught to children with autism spectrum conditions? Phil. Trans. R. Soc. B (364) 3567-3574, 2009


Bohne G. Emotions at play: gaining emotional knowledge using a video game, Master thesis, Master Programme in Human-Computer Interaction, Uppsala University, 2014


## 9 Appendix

Labels corresponding to the atomic units of virtual character behavior supporting by the virtual character controller. All or some of these may be linked to a proprietary virtual character that has been created by the game developer.

<table>
<thead>
<tr>
<th>PSL_BODY labels</th>
<th>PSL_FACE labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSL_BODY_Neutral</td>
<td>PSL_FACE_Neutral</td>
</tr>
<tr>
<td>PSL_BODY_HappyLowIntensity</td>
<td>PSL_FACE_HappyLowIntensity</td>
</tr>
<tr>
<td>PSL_BODY_HappyMediumIntensity</td>
<td>PSL_FACE_HappyMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_HappyHighIntensity</td>
<td>PSL_FACE_HappyHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_SadLowIntensity</td>
<td>PSL_FACE_SadLowIntensity</td>
</tr>
<tr>
<td>PSL_BODY_SadMediumIntensity</td>
<td>PSL_FACE_SadMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_SadHighIntensity</td>
<td>PSL_FACE_SadHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_AngryLowIntensity</td>
<td>PSL_FACE_AngryLowIntensity</td>
</tr>
<tr>
<td>PSL_BODY_AngryMediumIntensity</td>
<td>PSL_FACE_AngryMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_AngryHighIntensity</td>
<td>PSL_FACE_AngryHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_FearLowIntensity</td>
<td>PSL_FACE_FearLowIntensity</td>
</tr>
<tr>
<td>PSL_BODY_FearMediumIntensity</td>
<td>PSL_FACE_FearMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_FearHighIntensity</td>
<td>PSL_FACE_FearHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_SurpriseLowIntensity</td>
<td>PSL_FACE_SurpriseLowIntensity</td>
</tr>
<tr>
<td>PSL_BODY_SurpriseMediumIntensity</td>
<td>PSL_FACE_SurpriseMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_SurpriseHighIntensity</td>
<td>PSL_FACE_SurpriseHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_DisgustLowIntensity</td>
<td>PSL_FACE_DisgustLowIntensity</td>
</tr>
<tr>
<td>PSL_BODY_DisgustMediumIntensity</td>
<td>PSL_FACE_DisgustMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_DisgustHighIntensity</td>
<td>PSL_FACE_DisgustHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_HostileLowIntensity</td>
<td>PSL_FACE_HostileLowIntensity</td>
</tr>
<tr>
<td>PSL_BODY_HostileMediumIntensity</td>
<td>PSL_FACE_HostileMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_HostileHighIntensity</td>
<td>PSL_FACE_HostileHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_Agree</td>
<td>PSL_FACE_Agree</td>
</tr>
<tr>
<td>PSL_BODY_Disagree</td>
<td>PSL_FACE_Disagree</td>
</tr>
<tr>
<td>PSL_BODY_RotateBodywithHead</td>
<td>PSL_FACE_RotateBodywithHead</td>
</tr>
<tr>
<td>PSL_BODY_IntroduceSelf</td>
<td>PSL_FACE_IntroduceSelf</td>
</tr>
<tr>
<td>PSL_BODY_IntroOtherLeft</td>
<td>PSL_FACE_DissatisfiedLowIntensity</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>PSL_BODY_IntroOtherRight</td>
<td>PSL_FACE_DissatisfiedMediumIntensity</td>
</tr>
<tr>
<td>PSL_BODY_IntroduceThing</td>
<td>PSL_FACE_DissatisfiedHighIntensity</td>
</tr>
<tr>
<td>PSL_BODY_OfferThing</td>
<td>PSL_FACE_Talking</td>
</tr>
<tr>
<td>PSL_BODY_Listening</td>
<td>PSL_FACE_Blinking</td>
</tr>
<tr>
<td>PSL_BODY_Asking</td>
<td></td>
</tr>
<tr>
<td>PSL_BODY_Talking</td>
<td></td>
</tr>
</tbody>
</table>