Which game are you playing? - An interactive and educational video show

Matjaž Poljanšek[1], Nataša Knez[1], Sonja Vuk[2], Borut Batagelj[1]
[1]Faculty of Computer and Information Science, University of Ljubljana, Tržaška 25, 1000 Ljubljana, Slovenia
[2]Academy of Fine Arts and Design, University of Ljubljana, Erjavčeva 23, 1000 Ljubljana, Slovenia
E-mail: matkocopatko@gmail.com, natasa.knez@gmail.com, sonja.vuk@gmail.com, borut.batagelj@fri.uni-lj.si

Abstract - *Project "Which game are you playing?" is an interactive video show, which has an educational purpose. Video records projected on the three walls are swapped according to where a person has sat down, in other words, according to a person's »ego« role: child, parent or adult. Videos show conflicts among young people and their resolution according to the person's »ego« role. We made the project with two programs: Smart Wall and vvvv. For the realization of this project we need two computers, which are connected with an ethernet cable, two cameras, one placed on the ceiling and the other in front of the »ego« chairs, and three projectors.*

Keywords – *Game, Computer Vision, Ego, Smart Wall, vvvv*

1. INTRODUCTION

The project "Which game are you playing?" is an interactive video show that has an educational purpose. The work consists of two parts: three main projections and an installation in the space. Three chairs symbolizing particular »ego« roles (child, adult, parent) are placed near the wall. Other chairs (four, five or six) are placed randomly in the space. Chairs should look like school chairs, so the space is like a simulation of a classroom.

The records showing different situations from the life of the youth (bullying, provocations) are projected on the three walls. One projection is activated by sitting on one of the randomly placed chairs. It shows the conflict itself. The type of the conflict is changed by a decision to sit on one of the randomly placed chairs. The other two projections are activated by sitting on the »ego« role chair – a parent, an adult or a child. When a person sits on one of the chairs, the projection, showing conflict among youth seen from the standpoint of that particular role, begins on a near-by wall. At the same time, on the second wall the same situation is played from the non-violent conflict resolution position.

In the lower part of the projection the record of a sitting person is shown. Fig. 1 shows the visual presentation of our project.

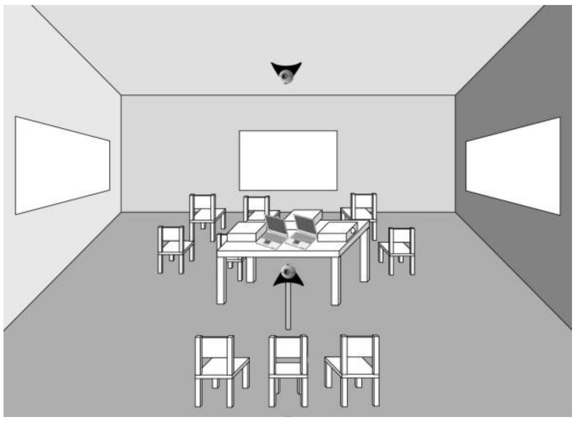


Fig. 1. Visual presentation of the project

2. MATERIALS AND METHODS

We made the project with two programs: Smart Wall [1] and vvvv [2]. Smart Wall is a software package for controlling the showing of interactive content in accordance to movement in space. vvvv is a toolkit for real time video synthesis. It is designed to facilitate the handling of large media environments with physical interfaces, real-time motion graphics, audio and video that can interact with many users simultaneously. vvvv uses a visual programming interface. Therefore it provides a graphical programming language for easy prototyping and development.

2.1. Part one

First part of the project is about projecting video on the middle wall of the room, when a person sits on one of the randomly placed chairs and swapping to another video, when a person sits on another of these randomly placed chairs.

This part is realized with a camera placed on the ceiling, so that the whole area of the randomly placed chairs is captured. Using the Smart Wall (the smartwall_hotspot_definition_interface feature), we can select »hotspots« for each chair by clicking on the area, which we want to observe. The data of these hotspots are registered in a xml file *hotspots.xml*. This file is loaded in the main program of the Smart Wall. Then the capturing of the video from the camera is activated and in short time intervals the Smart Wall periodically generates an extra xml file *hotspotOutput.xml,* which contains the data about the occupancy of an individual hotspot (if the value of a hotspot is 0 then the chair is not taken, if it is 1 it is taken).

Then a vvvv program reads this file and plays or switches to another video – it depends on the values of the hotspots. If a change of a hotspot has been done, a video number increases, which actually represents a sequential video number in a list of video files. This video number is registered in a special file, which is read by the other computer in the other part of the project. The two computers are connected with an ethernet cable. Namely, when a video number increases, a video has to be replaced with another video and at the same time two projections on the side walls, which represent the solution of this conflict, have to be swapped.

2.2. Part two

In the second part of the project, the simultaneous projection of two videos takes place on two parallel walls. One video shows the resolution of a conflict, depending on the »ego« role, while the other shows its non-violent resolution. The face of the person sitting on one of the three »ego« role chairs is also projected on the wall.

Using a separate camera, placed in front of the three »ego« role chairs, and Smart Wall, we detect whether a person has sat on a chair (this is done similarly as in the first part of the project). Then the selection and playback of videos is realized using the program vvvv.

Firstly, the xml file *hotspotOutput.xml* is read and the information about the status of the three hotspots is extracted from it.

Then the number of the video currently playing on the middle wall is extracted from a file, which is created in the first part of the project and is located on the first computer.

Using the hotspot status information, the video number and a combination of logical functions and arithmetical operations, two videos are selected from two lists. One list contains the paths to videos showing the resolution of a conflict, depending on the »ego« role, the other contains paths to videos showing a non-violent resolution of a conflict. The two selected videos are then simultaneously played and projected on two parallel walls.

The videos on the two lists must be sorted and named by a certain key. Here is an example and an explanation:

Solution depending on »ego« role:	Non-violent solution:
EMPTY	EMPTY
Video_Child_1.avi	Video_Nonviolent_1.avi
Video_Child_2.avi	Video_Nonviolent_2.avi
Video_Parent_1.avi	Video_Nonviolent_3.avi
Video_Parent_2.avi	Video_Nonviolent_4.avi
Video_Adult_1.avi	Video_Nonviolent_5.avi
Video_Adult_2.avi	Video_Nonviolent_6.avi

Table 1. Sorted lists of video files

The number at the end of each video name refers to the number of the video/conflict, currently playing on the middle wall. If the number of the video playing on the middle wall is 1, then *Video_Nonviolent_1.avi* must be played on one of the parallel walls, while the video playing on the other parallel wall is chosen according to the chair that is currently taken:

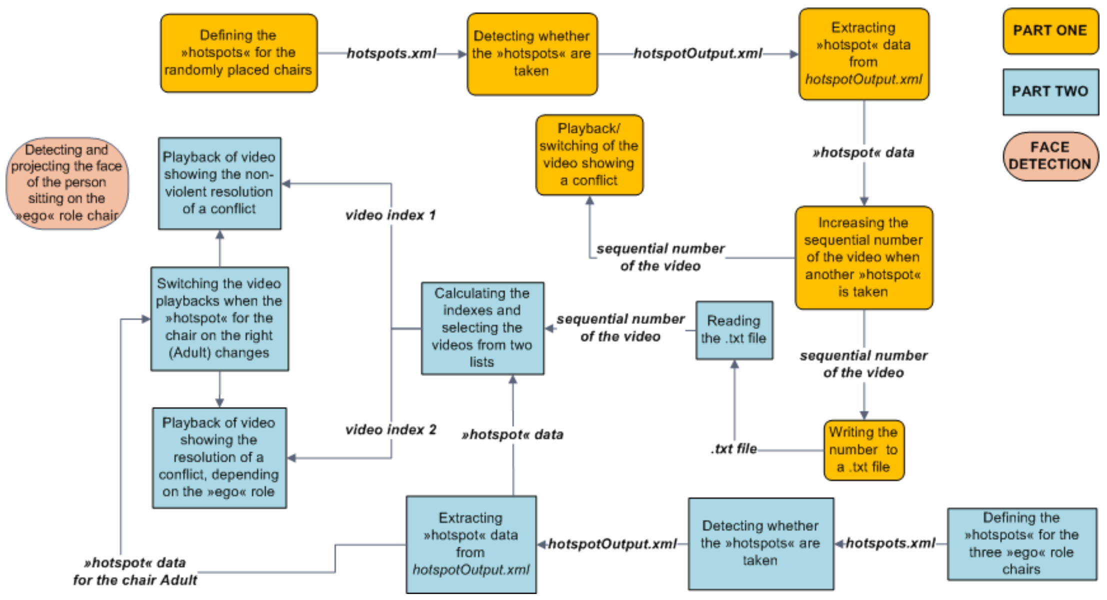
- if the 1st hotspot is active (Child – left chair), *Video_Child_1.avi* must be played,
- if the 2nd hotspot is active (Parent – middle chair), *Video_Parent_1.avi* must be played,
- if the 3rd hotspot is active (Adult – right chair), *Video_Adult_1.avi* must be played.

The video showing the resolution of a conflict, depending on the »ego« role, is played on the wall closest to the taken chair. This means that when the 3rd hotspot is active (Adult – right chair), the video showing the resolution of a conflict, depending on the »ego« role, is played on the right wall, while the video showing the non-violent resolution of a conflict is played on the left wall. When any of the other two chairs are taken, the video showing the resolution of a conflict, depending on the »ego« role, is played on the left wall, while the video showing the non-violent resolution of a conflict is played on the right wall.

2.2.1. A descriptions of the vvvv components

Component ***Random*** generates numbers and constantly triggers the 1st input pin (*Update*) on ***Reader***, so that the latest version of the xml file *hotspotOutput.xml*, generated by Smart Wall, is read. The name of the xml file is given on the 1st input pin (*XML input*) on ***XPath***, while the 2nd input pin (*Xpath Query*) is given the query */hotspot/hotspot*, which finds the »hotspot« elements in the folder »hotspots« and returns their values, given between the tags <hotspot...> and </hotspot>.

AsValue converts the values of hotspots from strings to numbers, and passes them to ***IOBox***, which displays them.



Defining the »hotspots« for the randomly placed chairs
hotspots.xml
Detecting whether the »hotspots« are taken
hotspotOutput.xml
Extracting »hotspot« data from hotspotOutput.xml
PART ONE
PART TWO
FACE DETECTION
Detecting and projecting the face of the person sitting on the »ego« role chair
Playback of video showing the non-violent resolution of a conflict
Playback/ switching of the video showing a conflict
»hotspot« data
video index 1
sequential number of the video
Increasing the sequential number of the video when another »hotspot« is taken
Switching the video playbacks when the »hotspot« for the chair on the right (Adult) changes
Calculating the indexes and selecting the videos from two lists
sequential number of the video
Reading the .txt file
sequential number of the video
Playback of video showing the resolution of a conflict, depending on the »ego« role
video index 2
»hotspot« data
.txt file
Writing the number to a .txt file
Extracting »hotspot« data from hotspotOutput.xml
hotspotOutput.xml
Detecting whether the »hotspots« are taken
hotspots.xml
Defining the »hotspots« for the three »ego« role chairs
»hotspot« data for the chair Adult

Fig. 2. Schematic diagram of the operation of the project

Each of the three hotspots values is then passed to its own ***IOBox*** (1: child, 2: parent, 3: adult) using three ***GetSlice*** components. This is done by connecting the 1st input pin (*Input*) on each ***GetSlice*** to the output pin (*Y Output Value*) on ***IOBox*** containing the hotspot values, and setting the 3rd input pin (*Index*) on each ***GetSlice*** to values 0, 1 or 2.

The three hotspot values and the number of the video currently playing on the middle wall (reading the video number is similar to reading the file *hotspotOutput.xml*) are used in combination with the multiplication components (*), sum components (+), and equivalence components (=) in order to generate an index used to choose an appropriate video from the list of videos showing the resolution of a conflict, depending on the »ego« role. The index is generated using these formulas:

Index = *Child* + *Parent* + *Adult*, where:

- Child = Value_Hotspot_Child × (Video_Number + Offset_Child)
- Parent = Value_Hotspot_Parent × (Video_Number + Offset_Parent) × Child_Taken
- Adult = Value_Hotspot_Adult × (Video_Number + Offset_Adult) × Child_Taken × Parent_Taken

Offset_Child, Offset_Parent and Offset_Adult are set according to the number of videos related to each of the three chairs. The value of Child_Taken is 1 if the chair Child is not taken, and 0 if it is taken. The value of Parent_Taken is 1 if the chair Parent is not taken, and 0 if it is taken.

In the case, where all the chairs are taken, the priority of the chairs is as follows: Child→Parent→Adult. The video related to the chair Child will therefore be played always when the chair Child will be taken. The video related to the chair Parent will be played only when the chair Child is not taken (for this purpose we have to multiply with Child_Taken). The video related to the chair Adult will be played only when the chairs Child and Parent are not taken (for this purpose we have to multiply with Child_Taken and Parent_Taken).

For selecting the video from the list of videos showing a non-violent resolution of a conflict, only the number of video currently playing on the middle wall is used.

Playback of videos and switching the projections:

The 1st input pin (*Play*) on both ***FileStream*** components is connected to the output on the ***IOBox*** component containing the index of the video from the list of videos showing the resolution of a conflict, depending on the »ego« role. By doing this, both videos are played only when a certain chair is taken. If none of the chairs are taken, the value of the index will be 0 and playback will be disabled.

The 3rd output pin (*Video*) on both ***FileStream*** components is connected to the 1st input pin (*Video*) on both ***VideoTexture*** components, which converts the video to a video texture (the ***VideoTexture*** component is needed because of the ***Switch*** component, which only accepts video in the form of video texture on the 2nd (*Input 1*) and 3rd (*Input 2*) input pins).

The ***Switch*** component has three input pins: the 2nd (*Input 1*) and the 3rd (*Input 2*) pin are video texture inputs, while the 1st input pin (*Switch*) selects which of the two videos/video textures goes to the ouput pin. If the value of the 1st pin is 0, the ***Switch*** component outputs the video on the 2nd pin, if the value is 1, it outputs the video on the 3rd pin.

The 1st output pin (*Texture Out*) from the left ***VideoTexture*** component is connected to the 2nd input pin (*Input 1*) on the left ***Switch*** component and to the 3rd input pin (*Input 2*) on the right ***Switch*** component, while the 1st output pin (*Texture Out*) from the right ***VideoTexture*** component is connected to the 2nd input pin (*Input 1*) of the right ***Switch*** component and to the 3rd (*Input 2*) input pin of the left ***Switch*** component.

To make sure the projections of the two videos are switched when ONLY the right (Adult) chair is taken, the value of the hotspot, related to this chair, is multiplied by a certain other value (0 or 1) and stored in the ***IOBox*** component. This value tells if the chair Adult is the only chair taken (1) and is calculated by summing the values of all three hotspots (using the sum component (+)) and comparing the sum to 1 (using the greater than component (>)). The output of the comparison is 1 if the sum is greater than 1, and 0 if it is lower or equal to 1. The output is then multiplied by the value of the hotspot relating to the chair Adult and negated using the ***NOT*** component.

The 1st input pin (*Switch*) on both ***Switch*** components is connected to the output pin on the ***IOBox***. When the value in the ***IOBox*** is 1 the projection switches and the video showing the resolution of a conflict, depending on the »ego« role is played on the right wall, while the video showing the non-violent resolution is played on the left wall. When any of the other two chairs are taken the projection is reversed.

The video texture from the output pin on both ***Switch*** components is then converted back to video, using the ***AsVideo*** components, and connected to the 1st input pin (*Video*) on two ***VideoOut*** components that output the two videos to two separate projectors.

2.3. Face detection

Face detection was already made in vvvv (the *detectObject.v4p* file in vvvv's manual folder *help*). The details are described in [3]. We only added two components: ***Quad*** and ***Renderer***. The 1st ouput pin (*Texture Out*) on ***VideoTexture*** is connected to the 3rd input pin (*Texture*) on ***Quad***. The ouput pin (*Transform Out*) on ***Transform*** is connected to the 5th input pin (*Texture Transform*) on ***Quad***. The ouput pin on ***Quad*** is connected to the 1st input pin (*Layers*) on ***Renderer***. By doing this, we capture and project only the face of the person sitting on a chair and not the whole image from the camera. The area of the face is determined by the coordinates of the rectangle, drawn around the face by the ***DetectObject*** component. These coordinates (x,y), together with the width and height of the rectangle, are then passed on to ***Transform***.

Since face detection (in vvvv) and the detection, whether a certain chair is taken (in Smart Wall), take place simultaneously, we used the program *WebCam Splitter Pro* [4], so that the video from the camera could be captured in both programs at the same time.

3. NEEDED HARDWARE

For the realization of this project we need two computers (one of them must have a dual video output), an ethernet cable to connect the two computers, two cameras, one placed on the ceiling and the other in front of the »ego« chairs, and three projectors.

4. CONCLUSION

The project was interesting from the point of learning to work with the programs – vvvv and Smart Wall, and their use in presentational and educational purposes. The most difficult part of the project was programming in vvvv, mostly the second part of the project – the realization of choosing the correct videos and swapping the projections between the two walls.

The first part of the project could also be changed, so that the projection of the video on the middle wall would begin when the first person steps in the area of the randomly placed chairs. This would be better as we would not have to wait for the first person to sit down and start the show.

ACKNOWLEDGEMENT

We would like to thank our advisor Borut Batagelj for helping and guiding us through the project.

REFERENCES

[1] Smart Wall: *http://ns2.nadzoruj.si/smartwall/?page=5* (last visit: March 4 2009)

[2] vvvv: *http://vvvv.org/tiki-index.php* (last visit: March 4 2009)

[3] Face detection: *http://vvvv.org/tiki-index.php?page=freeframedetectobject* (last visit: March 4 2009)

[4] WebCam Splitter Pro: *http://www.verysoft.com/en/wcspro* (last visit: March 4 2009)