Ghost in the Cave

An interactive collaborative game using non-verbal communication

Presented at the Interactive Evening and pub at the SMAC03 conference, Stockholm

Authors: Sofia Dahl, Anders Friberg KTH-TMH, Marie-Louise Rinman KTH-CID, Bendik Bendiksen Octaga, Damien Cirotteau Hugh McCarthy, DEI, Barbara Mazzarino, DIST, Erik Lindström, UUP
Date: October 6, 2003

General information

Site: Maskinhallen KTH, August 6, 2003.
Context: A part of the interactive evening at SMAC03, Stockholm Music Acoustic Conference, open to all participants at the conference
Involved partners: KTH-TMH, KTH-CID, DEI, TNR/Octaga, DIST
Game Developers: Marie-Louise Rinman (design), Anders Friberg (audio input), Barbara Mazzarino, Sofia Dahl, (video input), Bendik Bendiksen, Ivar Kjellmo (3D world), Damien Cirotteau, Hugh McCarthy (music generation)

Aims

- To investigate how expressive cues in movements and music can be used within a game setting
- To assess audience feedback (see separate report)

The aim of the game was to test the applicability and intelligibility of expressive gesture in a new setting. First, could expressive gesture, e.g. music and dance, be used as a basis for (audience) collaboration and participation? And second, could it be used as an input control / means of communication in an interactive game environment?

Main Features

- Non-verbal communication
  - both the interaction with the computer and between audience and players
  - expression in vocal sounds and body movements
- Collaboration
  - Team work and audience participation
- Mixed-reality environment
- Two playing teams
- Each team has a fish avatar in a 3D underwater world
- Team player is controlling its fish with expressive sounds and expressive movements
- The task is to navigate in the world and to find three caves.

The game involves participants in an activity using non-verbal emotional expressions. Two teams use expressive gestures in either voice or body movements to compete. Each team has an avatar controlled either by singing into a microphone or by moving in front of a video camera. Players control their avatars by using acoustical or motion cues. The avatar is navigated / moving around in a 3D distributed virtual environment using the Octagon server and player system. The voice input is processed using a musical cue analysis module yielding performance variables such as tempo, sound

level and articulation as well as an emotional prediction. Similarly, movements captured from the video camera are analysed in terms of different movement cues.

Relation with MEGA

Most of the features of the game were exploiting results derived directly from MEGA work packages. The video and audio analysis of the players each used cue extraction and an expressive mappers resulting from work within WP3 and WP4. The music output was an application in WP6 and the virtual 3D environment an application in WP5.

Concept

The interactive game was based on the idea of non-verbal communication as a means of controlling an avatar in a virtual environment (an underwater sea world). Two teams use either voice or body movement to navigate their avatar and compete through expressive gestures. There are two types of expressive input: voice or movement. An avatar can be controlled by either voice or gesture input, using expressive features in each domain.

Using the voice input the player can sing, talk or make any sounds and these will be analysed in the system to control the avatar's direction and speed. The sound level in the left or right microphone controls the direction and the number of note onsets influences the speed. For the emotional expression a musical cue analysis is made.

For the movement input a web cam captures the movement of the player. The players arm movements will determine the direction of the avatar and the amount of movement (Quantity of Motion) influences the speed. For the emotional expression a cue analysis similar to that of audio is made utilizing the EyesWeb motion analysis software.

The rest of the team is controlling the music background by moving more or less in front of a web cam. One team is controlling the drums and the other team is controlling the accompaniment. In the navigation phase, the more the team moves - the faster their fish is moving.

The task for each team is to navigate in the underwater world finding three caves. In each cave appears a ghost with a certain expression. The task within the cave is to move or sing corresponding to the ghost's emotion. The avatar responds to the player expression by changing appearance and colour.

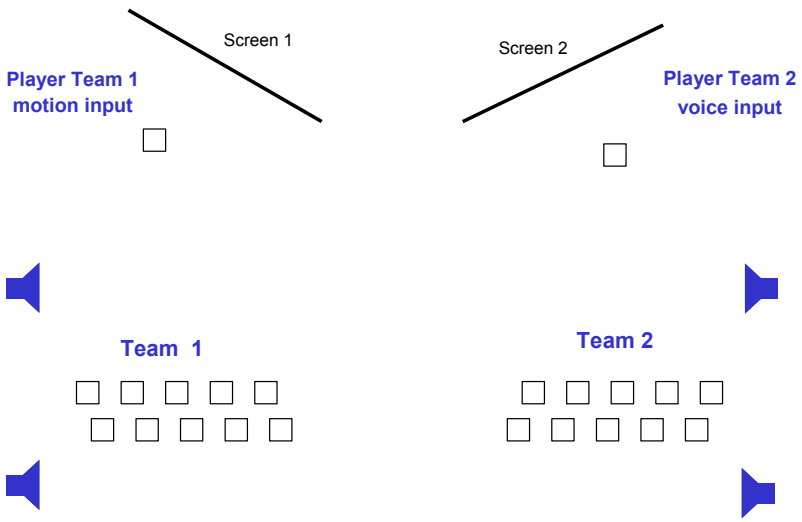


Figure 1. Overview of the game setup from above

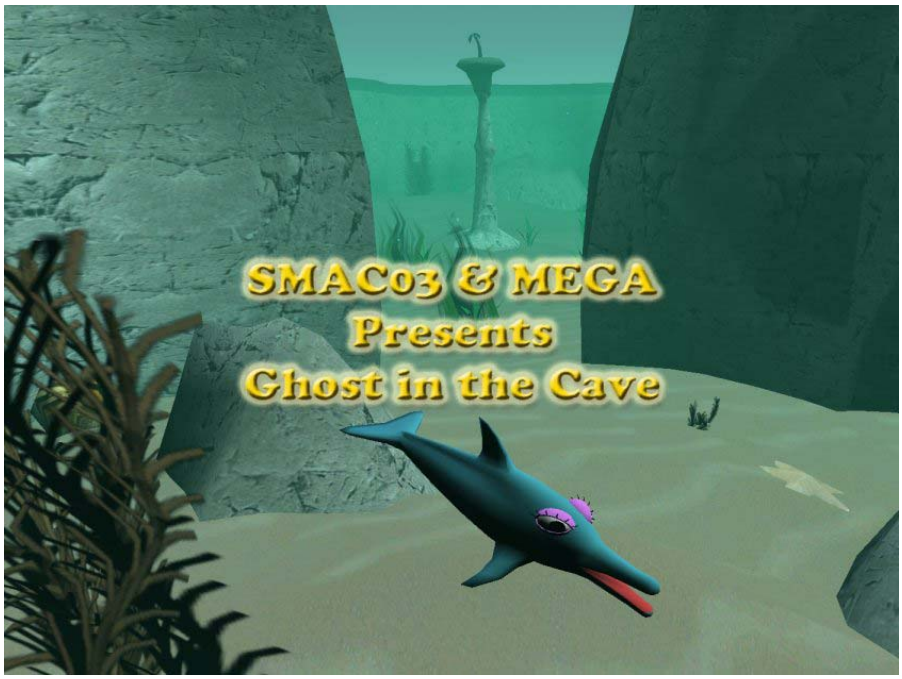


Figure 2. Startup screen for the Dolphin team.

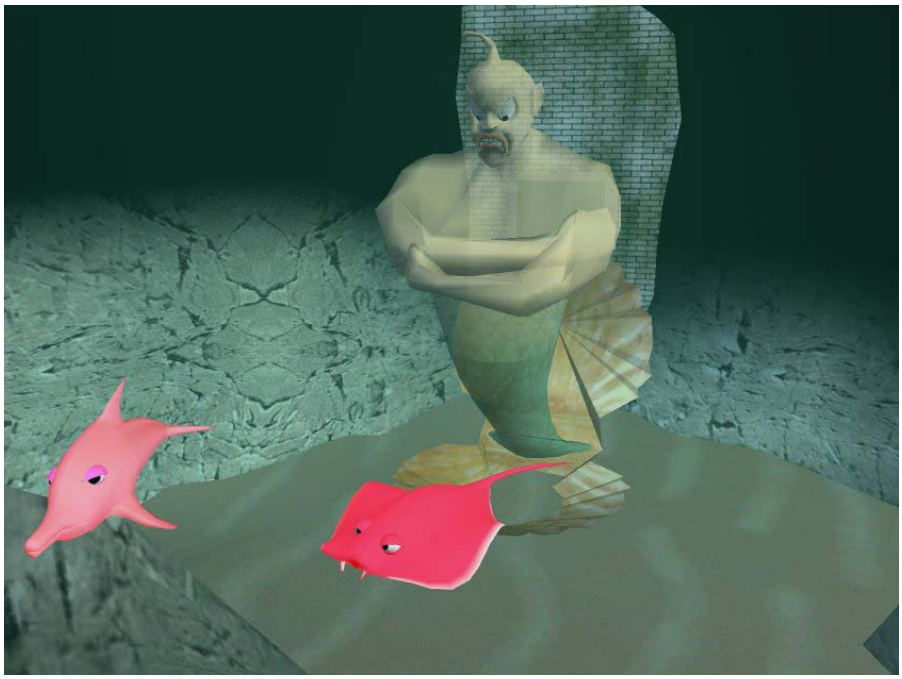


Figure 3. In the "angry" cave. Both players have managed to change their avatar into an angry "red" appearance.

SMAC event

The game was presented at the Interactive evening event included in the SMAC03 conference. The event was taken place in an industrial large hall at KTH. In the entrance each visitor were given a team number ticket. The first game run was a demonstration and instruction as to how the game worked. Two of the authors were controlling the avatars while parts of the public were engaged as team members. One author was continuously commenting in a microphone what was going on. After this introduction, the teams were asked to come up and play. In total, three complete games were played thus including 6 different teams. There were approximately 10-20 persons in each team. Figure 3 is a

photo from the first test run at the interactive evening with the motion input to the left and the voice input player to the right. Figure 4 shows team 1 helping their motion player. The event was evaluated both by giving the audience a questionnaire by analysing video recordings. The questionnaire results are described in a separate document by Erik Lindström et al. in MEGA Delivery 16. A general impression was the most of the involved participants found it exiting and fun while the passive audience found it of medium interest.

Figure 3. Picture from the SMAC event. The players' avatars have entered the "happy" cave. The motion input player can be seen to the left and the voice input player to the right.

Figure 4. Picture from the SMAC event. Team 1 is helping the movement player to reach the gate of the cave seen on the screen to the left

Technical description

The implementation was mainly in form of patches within the software environments EyesWeb, Octagon and pd. The software modules are shown in figure 5. The upper left part in the figure are for the motion input player including a movement navigation and emotion mapper. The upper right part is the corresponding voice input. The lower part of the picture is the music generation by the team groups.

Software:
Octagon
EyesWeb
Pure Data
(+ proxy)

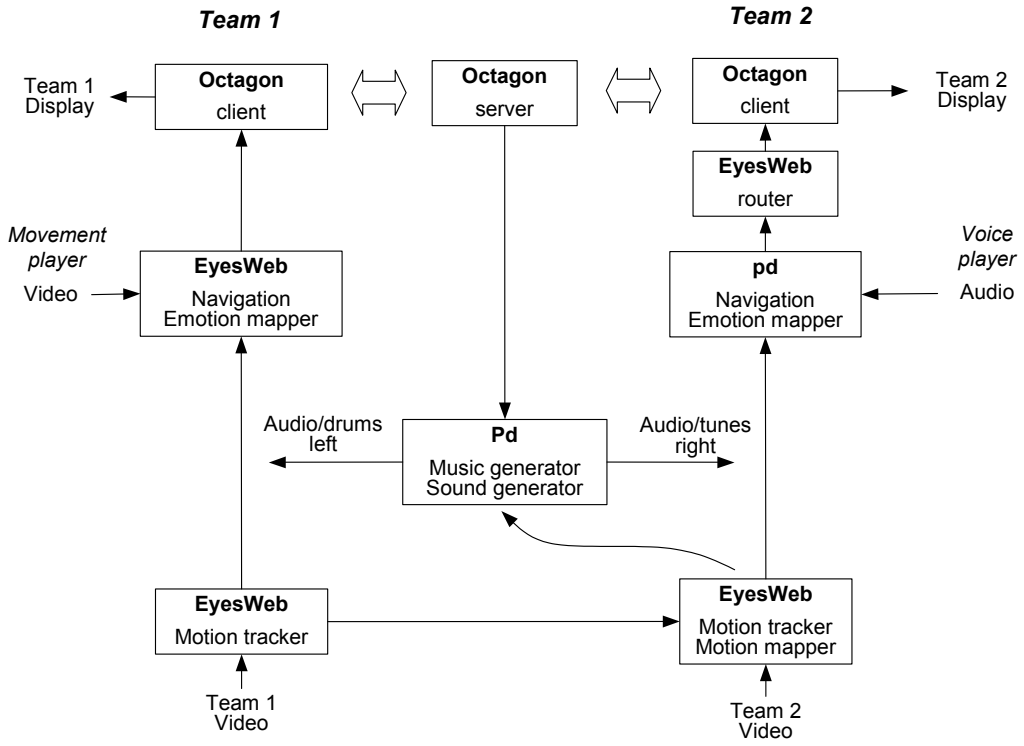


Figure 5. Software setup.

Hardware

The software modules all communicates via TcpIP which means that the physical configuration is flexible. At the SMAC event we used for maximum speed six computers connected in a network, see list of equipment below:

6 computers
with network connections (ca 50m network cable)
2 projectors
two backprojecting screens
3 webcams (3 x 7m USB cables)
2 microphones + stands
4 loudspeakers, amplifiers
mixers
light (3 x 300 W and 1 x 800 W)
black paint and black fabric

Following is a more detailed description of the major sofware modules

3D world

Octagon patch
Avatar parameter input forward (0 to 1), left-right (-1 to 1), three emotions (0-1)
Output screen

The virtual 3D environment is run in the Octagon server and player system. Multiple users are represented in the virtual world by their own figure, or avatar. Here they are present in the same shared virtual environment (an underwater sea world) and can interact with each other. The distribution goes through a network using the MPEG4 protocol.

Audio navigation

pd patch
Input Audio
Parameter input forward (0 to 1), – velocity scaling from team movements
Parameter output forward (0 to 1), left-right (-1 to 1)

The left-right direction is controlled by the left and right channels. Right microphone only - steers right, both microphones used - forward. The velocity of the avatar is given by the number of (detected) note onsets. This velocity can also be scaled by the overall movement of the supporting team members.

Video navigation

EW patch
Input Video
Parameter input forward (0 to 1), – velocity scaling from team movements
Parameter output forward (0 to 1), left-right (-1 to 1)

The left-right direction is controlled by the arm movements as when swimming. Left arm movement only - steers right, both arms moving – forward. The velocity of the avatar is given by the overall movement. As for the audio navigation control this can also be scaled by the overall movement of the supporting team members.

Audio emotion mapper

pd patch
Input Audio
Parameter output Sad (0-1), Happy (0-1), Angry (0-1)

All the audio cues are mapped to the three emotion outputs using a qualitative mapping similar to fuzzy logic. A detailed description is given in MEGA deliverable 13 and 14. We used the cues tempo (tones/s), sound level and articulation. The cues are transformed to standardized form (mean = 0, SD = 1) using data from a calibration phase. In the calibration phase, different expressive examples were sung, in order for the system to compute the mean and standard deviation (SD) for each cue. This was done in advance at the game setup.

Video emotion mapper

EyesWeb patch
Input Video
Parameter output Sad (0-1), Happy (0-1), Angry (0-1)

The video emotion recognizer used the cues Quantity of Motion (QoM), Contraction Index (CI)and Upward Gesture Tempo (UGT) in the mapping to the three emotions.

Team video and sound

Input Video
Parameter output
team_1_quantity_of_motion (0-1)
team_2_quantity_of_motion (0-1)
team_1_happy (0-1)
team_1_sad (0-1)
team_1_angry (0-1)
team_2_happy (0-1)
team_2_sad (0-1)
team_2_angry (0-1)
Audio output 4 -channel Audio (left-right, front-rear)

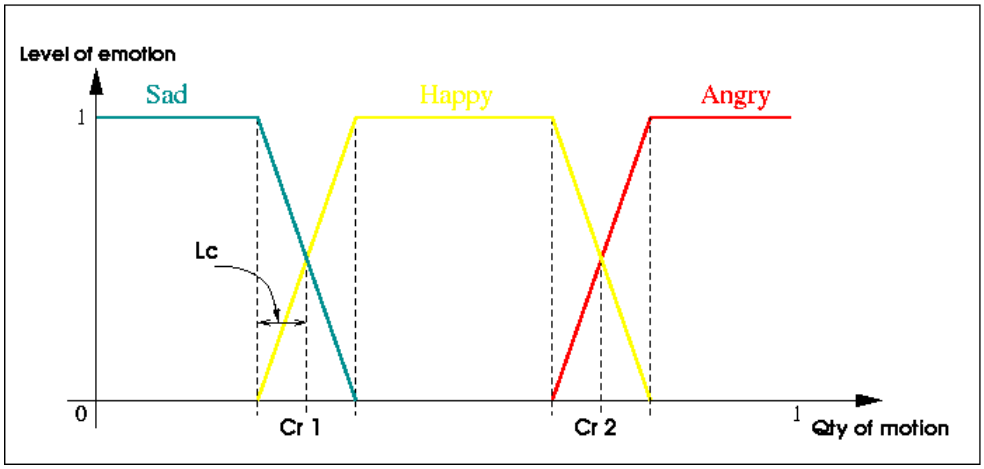
Team Motion Capture
2 web cams, 1 for each of the two teams/audience groups.
3 levels of quantity of motion reflecting sadness, happiness and anger

The audience can interact in two different ways in the game.

- During the navigation part, the are in control of a scaling factor on the velocity of the avatar of their player. Technically, the EyesWeb patches from each team send a coefficient (0 no movement/ 1 max movement) to the EyesWeb and pd patches of the player. The audience feel involved in the game because the need to activate themselves to get their avatar fast.

- During the whole game, there is a mapping between the quantity of motion produce by the team and the music produce such as in figure 5

Team Sound Output
Music backdrop from audio loops like in the groove machine reflecting three emotions (figure 5). All the music is at the same tempo. One team controls the drums, the other team controls the bass and the accompaniment. The music changes all along the different phases of the game to avoid public to be bored. In each cave, the public has to produce the right quantity of motion in order to produce the matching music.The pd patch produces the sonification of the 3D world from the information received from the Octagon server (see figure 4). Each event in the game is coded on an integer `(phase_of_the_game)` and each time the pd patch receive a new event, a new sound is played (for example opening door). The voice from the audio player is amplified on each side (Analog direct connection to mixer).



Parameter communication between EyesWeb and Octagon

Each team computer

From EyesWeb to Octagon client there are 8 float numbers in one package:

```
forward (0-1)
right_left (-1 to 1)
audio_sad (0-1)
audio_happy (0-1)
audio_angry (0-1)
video_sad (0-1)
video_happy (0-1)
video_angry (0-1)
```

Team computers

From EyesWeb to Octagon client there are 2 float numbers:

```
team_1_quantity_of_motion (0-1)
team_2_quantity_of_motion (0-1)
```

From EyesWeb to pd there

```
team_1_quantity_of_motion (0-1)
team_2_quantity_of_motion (0-1)
team_1_sad (0-1)
team_1_happy (0-1)
team_1_angry (0-1)
team_2_sad (0-1)
team_2_happy (0-1)
team_2_angry (0-1)
```

From the octagon server to pd there is one integer

```
phase_of_the_game (0-24)
```