Developing games with Magic Playground: a gesture-based game engine

Carolina Cabral
ADETTI
Av. das Forças Armadas,
Edifício ISCTE 1600-082 Lisboa
(+351) 21 782 64 80, Portugal
carolinacabral@netcabo.pt

Juana Dehanov
ADETTI
Av. das Forças Armadas,
Edifício ISCTE 1600-082 Lisboa
(+351) 21 782 64 80, Portugal
juana@netcabo.pt

José Miguel Salles Dias
ADETTI
Av. das Forças Armadas,
Edifício ISCTE 1600-082 Lisboa
(+351) 21 782 64 80, Portugal
Miguel.Dias@adetti.iscte.pt

Rafael Bastos
ADETTI
Av. das Forças Armadas,
Edifício ISCTE 1600-082 Lisboa
(+351) 21 782 64 80, Portugal
rafael.bastos@sapo.pt

ABSTRACT

This paper presents Magic Playground, a game engine that enables the development of entertainment applications with real-time gesture-based Human-Computer Interaction (HCI). We describe the main architectural elements of our system and provide a guideline on how to program the engine in order to create games. Finally, we present usability evaluation results of a game, which emulates the known Tetris game[1].

Categories and Subject Descriptors

D.3.3 I.5.4 I.4.8 [Image Processing and Computer vision]: Scene Analysis -- *Motion, Tracking*; J.7 [Computer in other systems]: consumer products.

General Terms

Algorithms, Measurement, Experimentation, Human Factors.

Keywords

Game Engine, Gesture based Human-Computer Interaction, Usability Evaluation.

1. BACKGROUND

Gesture recognition systems based on computer vision are subject of widely diffused research works. In [1] we find gesture recognition systems applied to useful, Sign Language interpretation. Other systems, like in [2], are more directed for human-computer interface control. Recognition techniques, either image or vector based, all tend to acquire hand contour as a starting point for gesture perception and then its motion, being somewhat influenced by the human visual system process. Background scenario complexity is not frequently addressed, requiring gestures to be executed against a homogenous prepared scene. Related works from the computer vision domain, like advanced background subtraction [3], can be useful for gesture recognition. The EyesWeb toolkit [4] has provided a framework for the detection of human body motion, specially oriented to performing arts applications. Recent developments in game technology by Sony, namely the EyeToy[2] for Playstation, brought new possibilities for the interaction between the player and the game. The technique uses a camera that captures the player hands and body motion and places him/her in the game, allowing him/her to interact with virtual elements, possibly replacing the real background with a virtual one (fixed or moving) and increasing his/her sense of being part of the game. The Magic Playground exploits this new concept for PC gaming, adding new possibilities for the interaction and providing a programmable SDK for the game engine.

2. SYSTEM ARCHITECTURE AND FUNCTIONS

The Magic Playground engine intends to support he creation human hands and body motion-based entertainment applications and games. The engine is divided in five main modules depicted in Figure 1: Video Capture, Statistical Image Processing Unit,

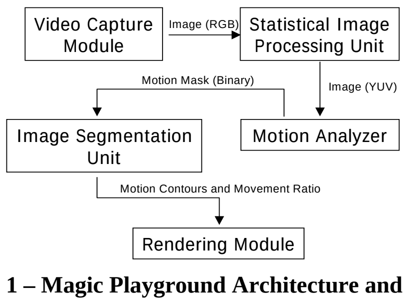

Figure 1 – Magic Playground Architecture and System Flow Diagram

Motion Analyzer, Image Segmentation and Rendering. The Video Capture Module is responsible for acquiring the image data from the input video device (Web Camera, Firewire DV Cam, Video In

[1] TETRIS was invented by Alexey Pazhitnov, in 1985, in an Electronica 60 computer at the Moscow Academy of Science's Computer Center.

[2] EyeToy and Playstation are registered trademarks of Sony Computer Entertainment Inc.

or other) and for delivering it to the Statistical Image Processing Unit. This unit main goal is to create a statistical model of the input real scene, by analyzing each pixel luminous energy and by computing color thresholds that will be delivered to the Motion Analyzer. The Motion Analyzer performs an image transformation into the YUV subspace and then, a convulsion with a Gaussian Filter, in order to reduce image noise and to perform contour smoothing. The previous computed (in the Statistical Image Processing Unit) color thresholds are then used to compose a motion mask binary image, where 0 is considered to be a pixel classified as a part of the real scene background and 1, a pixel classified as a foreground moving object (which can be, for example, the user playing the game). The next stage is Image Segmentation, where all foreground contours are retrieved and the movement ratio of all foreground image blobs is computed. This final data is then delivered to the Rendering Module, which is responsible for blending fixed or moving virtual backgrounds with the foreground segmented image, so that background substitution can be performed.

2.1 How to create a game with Magic Playground

Our engine provides a set of utilities and classes, supporting gesture-based HCI. To develop entertainment applications using this engine, we have to create a class that extends XMotionApp. This class allows dynamic loading of XMotionSystem based applications and has three main methods that will be called sequentially when the application runs: GetInstance(), called only when cloning the application object at initialization; PreRender(), called before rendering and PostRender(), called after rendering. PreRender() and PostRender() are called until the application ends. The PreRender() is where the programmer can modify the motion mask computed by the main system and overlay the areas that fits the needs of the game. He/she can overlay the background with fixed or moving pictures, by blending the silhouette of the player captured by the camera and the image(s) selected for the background. In the PostRender() function, the programmer can define and manipulate objects that provide 2D custom graphical rendering as CDXAnimatedSprite, CDXBall, CDXButton, CDXImageSprite, CDXImageSpriteInterpolatorEx, etc. These classes allow the programmer to create an attractive user interface that responds to the motion performed by the player, when performed in the areas where they are located, and/or compute the collision between the user silhouette and virtual objects in the viewport. The class CDXAnimatedStripe is useful when rendering animated objects, such as backgrounds, objects with motion (AVI movies), etc. The CDXImageSprite performs 2D graphical rendering of 2D objects, supporting various image formats. It provides also other functionalities, such as a spatial interpolator. The CDXBall class contains a gravity-enabled physical ball model, which is responsible for all motion computation before the rendering stage, so that a more enhanced and realistic movement effect can be perceived.

3. EVALUATION AND DISCUSSION

We have evaluated the engine with a game - MagicTetris, which emulates the well known Tetris game, where the user manipulates the motion and rotation of the pieces by producing moving hand gestures in the appropriated screen control areas, as it can be seen in Figure 2. The experiment was run on a sample of 10 unpaid users in one run. The subjects were students with an age range between 20 and 24 from both sexes, selected randomly from undergraduate students at our institute in Lisbon.

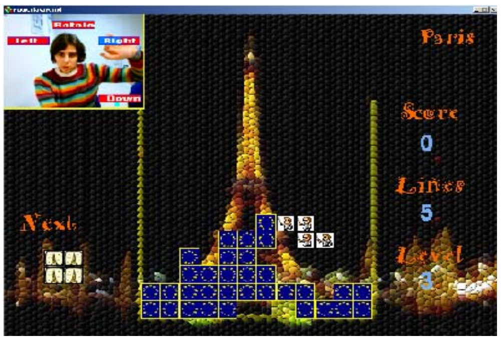

Figure 2 – MagicTetris game

Regarding the evaluation methodology, we began by performing the first part of a questionnaire to each subject, aiming at understanding the subject profile, in particular if the subject had already been in contact with this kind of gesture-based HCI, with the Tetris game and if he/she knew how a Webcam works. Afterwards the facilitator gave a briefly explanation on how to operate the game. The subject was given approximately two minutes to gain acquaintance with the game, after which the game testing begun. The subject had to play MagicTetris the maximum time possible and reaching the higher game level possible (passing a level meant to complete 5 lines). At that point, experimental data was retrieved by observing the subject playing: such as the number of lines completed, the high score obtained, the time duration and the accomplished level, among others. Finally, the second part of the questionnaire was performed upon completion of the game, were we asked each subject to give his/her opinion, regarding the game's usability/understanding and the easiness to work with it, and the observations and comments obtained were:

- Only approximately half of the group had already been in contact with this type of HCI, which took some time to adapt to;
- Since the HCI is accomplished through the subject's motion, some physical tiredness has occurred with some subjects;
- The subject's willingness to experience this kind of HCI was high;
- The HCI, the usability and the enjoyment that the game provided, were considered "good";
- Subjects recommended a re-organisation of the buttons layout and a better response of the gesture HCI modality.

4. REFERENCES

[1] Yoshinori Niwa, et. al., "Robust face detection and Japanese Sign Language hand posture recognition for human-computer interaction in an "intelligent" room". VI'2002, Faculty of Engineering, Gifu University, 2002.

[2] Mario Aguilar, et. al.. "Facilitating user interaction with complex systems via hand gesture recognition". ACMSE'03.Knowledge Systems Laboratory, Jacksonville State University, 2003.

[3] Larry S., et. al., "A statistical approach for real-time robust background subtraction and shadow detection". Technical report, Computer Vision Laboratory University of Maryland, 1999.

[4] Barbara Mazzarino, et. al., "Analysis of expressive gesture: The eyesweb expressive gesture processing library". Proc. of Gesture Workshop 2003, Genova, 2003.