Nelson Zagalo & Rui Prada (eds.)
Actas da Conferência ZON | Digital Games 2008
www.lasics.uminho.pt/ojs/index.php/zondgames08/
Centro de Estudos de Comunicação e Sociedade
Instituto de Ciências Sociais
Universidade do Minho
ISBN: 978-989-95500-2-5

Balloons, an Augmented Virtuality computer game

Pedro Lourenço1, Miguel Almeida1, João Madeiras Pereira2
1YDreams: Edifício YDreams, Madan Parque Sul, Caparica,
(pedro.lourenco, miguel.almeida)@ydreams.com
2Portugal, INESC-ID: R. Alves Redol, nº 9 Lisboa, Portugal
jap@inesc.pt

Abstract: This paper summarizes the implementation of a mixed reality game using computer vision techniques that enable players to interact using body motion in a rich and dynamic environment, dubbed: "Balloons". Players have to safely maneuver a balloon through 3D mazes full of perilous obstacles that will affect its behavior. Developed on the Virtools platform, this application takes advantage of single camera tracking techniques, such as background removal and silhouette tracking, allowing the use of limited hardware requirements. Through a physics engine, real life behavior of all elements is emulated, leading to an immersive user experience. To support fast creation of new game levels, a user friendly 3D visual game level editor was built, providing on the fly editing and testing of new content.

Keywords: Augmented Virtuality, optical tracking, motion detection, computer games.

1 INTRODUCTION

Due to recent technological advances it is now possible to interact with computers using methods other than the mouse and keyboard [1]. Beyond Virtual Reality – in which a 100% artificial world is created to interact with a user, there is now a new developing solution: Augmented Reality, achieved by mixing both reality and artificial graphics in order to endow the user with additional information and supporting a more immersive experience. One of the main techniques used in order to create these environments is optical tracking: in this technique one or more cameras are used in order to capture the user's movements.

Tracking can be done with or without the use of markers, and by infrared, high speed or even regular cameras.

2 THEORY

A brief description of the work done will follow, but it's scope could be easily compared to the already popular EyeToy [2] that uses nothing but a web cam to transport the player into the set in which the game takes place, from all the solutions available so far to create augmented/mixed reality games this is probably the simplest one and also the one with the lowest production costs [3, 4, 5]. This approach how ever takes things a step further than most games developed for the Eyetoy, because it not only sets the focus on a simulated physics environment but also enables users to create their own levels.

2.1 Balloons, the game

Balloons is a game of dexterity and mind thinking, the player's shadow is projected in screen and a frame of obstacles is superimposed on him, which is called the game scenario. The player will need to take a light and sensitive balloon from the starting point to the exit point without hitting any dangerous obstacle just by bumping on the balloon with his body.

The scenario is filled with dangerous obstacles and objects, like glue that will get stuck in the balloon and increase the mass for a while, spikes that will burst the balloon, flames or blades that will create difficulties to the player and even fans that will push the balloons in other directions.

The player has to use his body to tap the balloon forward, using any part of his body; a sudden movement can bring doom to the entire game so the player must be gentle most of the time. Of course wind currents and fans will require not so gentle strokes.

Two different playing modes were created for this game. Firstly a full 3D mode, where the action takes place in a large space and the player is able to maneuver the balloon freely through roomlike levels. In this mode the player can push the balloon up/down/left/right but every time he does so he also pushes it forward eventually leading it to safety. A second mode was alternatively created, where the level is set so that the game becomes a side scrolling adventure, where the same room like structure was used, but became much narrower, to restrict movement in depth; instead of a third person view of the balloon, the player has a side view like most side scrolling games. In this mode the player can also direct the balloon up/down and now also forward and backwards. In Figure 1 one can see the structure used for the levels, where on the top right hand corner the narrow walls used on the side scrolling mode is observable.

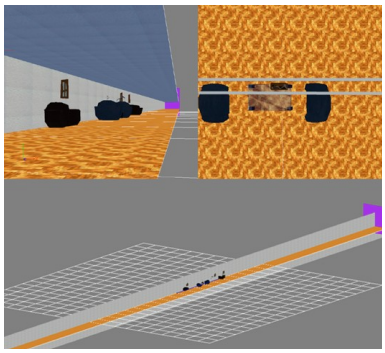

Fig. 1. Game level structure

In both these modes the levels are packed with 3D elements that interact with the balloon, among them are objects such as fans that create wind drafts that blow the balloon to unexpected places, heat sources that warm up the balloon making it lighter and float higher and eventually bursting when it gets to hot, pointy objects that burst the balloon on contact, teleports that transport the balloon to a different point of the level, etc. To each element created there is a behavior component and a graphical component, so that in different levels it is possible to have the same behavior attached to different graphical objects that better fit the environment.

2.2 The Game Level Editor

Along with the game the user has access to a level editor that allows the creation of endless different combinations of sets to be played on. Every level of the game is read via a XML

description file, the same files that the game application uses. One can edit already existing levels or create new ones from scratch; as mentioned before the graphical resources used can be different for the same behavioral element in various different levels, although some previous assembly of each of these elements is required.

The editor allows the creation of both types of levels (side scrolling or not), and a drag and drop style interaction with each element, with additional tools to rotate, move and delete elements.

This editor was created in full 3D so the person editing the levels has all the possible freedom to "walk" around and place elements where he sees fit. Navigational keys allow simple movements like panning and rotating around the set and additionally a small set of extra cameras were created to support a better viewing of the level.

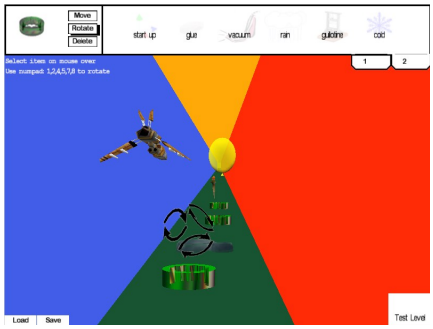

Fig. 2. Game Level Editor screenshot

The editor features a Live Test mode, where the game designer has the capability of testing the level he just created in a non augmented version of the game; here he can control the balloon with the mouse, thus being able to check the layout of his recently edited level.

2.3 Requirements and resources used

This game is intended to run in a regular household computer, using only a regular web cam as support, it was developed with the Virtools platform and making use of several different OpenCV functions.

2.3.1 Virtools [6]

Virtools is a Dassault Systemes platform, which enables high speed prototyping of applications rich in multimedia contents. It supports the creation of very complex applications that make full use of physics, Artificial Intelligence and Virtual Reality engines.

It supports a wide range of standards as OpenGL, DirectX and 3DXML, as well as many different types of media files like 3d objects/animations (3ds Max®, Maya®, Lightwave®, etc.) or image and sound standards (JPG, PNG, TIFF, TGA, BMP, PCX, MP3, WMA, WAV, MIDI) making it a very easy to work with all kinds of different resources.

It's a real time engine in what regards processing issues, enabling the creation of highly interactive applications, where behaviors accurately respond to every stimulus including the user's interactions. It supports development in a scripting environment, with which a developer can create a very wide range of behaviors making use of the number of building blocks provided by the platform, but one can also develop any kid of application using the

software development kit (SDK) available with Virtools to either create new building blocks or plugins for external hard/software; it's in this ability that Virtools excels, giving the developer a lot of freedom and control.

2.3.2 OpenCV [7]

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real time computer vision. It's multi-platform, although optimized for Intel processors, and highly useful for applications that require real time shape/motion recognition or the development of human/machine interfaces.

3 GAME DEVELOPMENT

In order to get the details right, the game was first developed as a plain mouse controlled game. A simple room with all the interacting elements was created and each one was tested thoroughly; only after every element was ready the augmented version began to be implemented.

3.1 Augmenting the game

The game was fully developed in 3D, and the step to augment it led to several different possibilities. Since tracking was done using only a web cam the players body couldn't be recognized in full 3D, at least not with a common household computer, since it would require tremendous amounts of calculus such as complex geometry like homography [8] and even a precise camera calibration, so the solution was to implement a simple algorithm that would perform the tracking and calculate the interaction in a 2D coordinate system instead of a 3D one, thus reducing the processing required.

For this approach all level elements were taken into account only as outlines that would interact with the users own outlines, captured from the web cam.

In order to convert the player into an outline, one must first remove the background on the image so all that is left is whatever is moving, in this case, the player's body.

After removing the background all that is left is a binary image with every background pixel painted black and the moving pixels in white thus appearing the silhouette of the player as a white shade in the image. The second step is to calculate interception between the player and whatever exists in the level that can interact with him. This was done with the help of OpenCV, which was used to develop a Virtools building block, dubbed "Interaction".

3.1.1 Interaction

The Interaction building block developed for Balloons is able to deliver a count of all intercepting pixels of the silhouette and a determined region of interest in the image (where the interacting objects are placed). It was optimized to support circular objects as well as rectangular ones, thus minimizing the number of pixels in error.

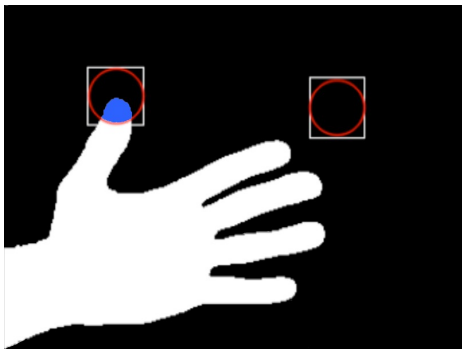

Fig. 3. Interaction building block, showing the accounted pixels in blue and two interacting round targets

Once the number of intercepting pixels is known, interaction with simple types of elements such as buttons is supported, but motion must also be taken into account if one wants to support interaction with more dynamic objects.

In order to do this, one other building block was developed - Motion Detect - which as the name certainly indicates provides a method for motion detecting. Paired with the Interaction building block it is possible to not only determine “when” interaction occurs but also “how”.

3.1.2 Motion Detect

Using motion history image (MHI), which enables a leveled approach to motion detecting, by recording several different steps of a movement in a single image, which can be analyzed in order to figure the motion orientation and magnitude of whatever moving objects are captured in it.

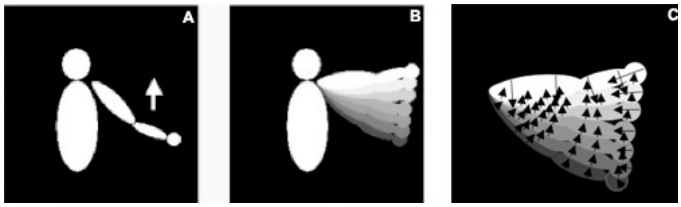

Fig. 4. Motion History Image example (a) the motion itself (b) the MHI and (c) the resulting motion vectors

The Motion Detect building block takes into account only a region of interest of the whole image itself just like the Interaction building block, thus reducing the processing time, taking advantage of the fact that only places in the image where actual interactive objects exist must be analyzed, and every other pixel is just non-interactive. So the building block returns a magnitude and orientation for any movement detected inside this determined region of interest of the image, providing not only an orientation component but a magnitude one as well.

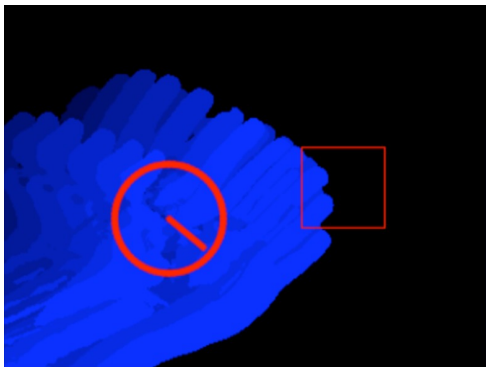

Fig. 5. Motion Detect building block in action

3.2 Final result

With both the developed building blocks working, it is now possible to create dynamic interaction between the player and the balloon. Being that both blocks run once before every frame is created, one can establish a relation between time and the impulse or thrust that is to be applied on the balloon. So for every pixel that the Interaction building block counts as positive, the thrust applied must grow in accordance to the magnitude value provided by the Motion Detect building block, being that the number of intercepting pixels grows in direct proportion to the velocity of the motion performed.

Both building blocks were created with a debug option that enables seeing what information is being processed.

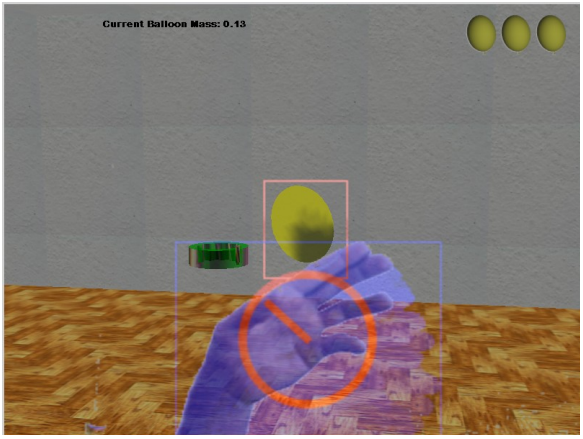

Fig. 6. In game example of use with debug prints

4 CONCLUSIONS

We were able to create interaction in augmented Virtuality ambient for a computer games that requires minimal hardware and software capabilities. In this example the interaction is controlled mainly by two building blocks developed with the use of OpenCV functions. There is also a physics engine ruling all behaviors of the various game elements, creating motion and interaction that can be easily perceived as a real physics experience, enabling the player to actually to touch and influence the behavior of virtual 3D elements that compose the game. Some minor inconsistencies occur with the movements performed either due to miscalculations on part of the MHI, or because the boundaries of the considered objects aren't tight enough (this mainly to save on processing times, since it is much easier to use

rectangular shapes than any other). Light also plays a big part on this kind of applications and is responsible for most errors accounted for, since any variation might result in pixels being accounted for as moving objects when in fact they are just variations in light, colour and shades. This can be worked around using dynamic thresholds in order to maintain an adaptive background notion that supports changes without compromising behavior.

The platform enabled a short creation period, allowing the developer to focus on issues related to tracking as well as attending to problems related with creating a large number of interacting elements through complex physics capabilities (e.g. light would heat the balloons air, making it lighter and eventually bursting it).

REFERENCES

[1] Azuma, R. “Recent Advances in Augmented Reality”, 2001

[2] Sony Computer Entertainment Inc.: Sony EyeToy, Official Site: www.eyetoy.com, 2003.

[3] Santos, P., et al 2003. 3D Interactive Augmented Reality in Early Stages of Product Design. In Proceedings of the 10th International Conference on Human-Computer Interaction (HCI International). 1203-1207.

[4] Lourenço, P. S.: Balloons - An Augmented Virtuality Computer Game, Master Thesis, Instituto Superior Técnico, Technical University of Lisbon 2007.

[5] Welch, G., Foxlin, E. 2002. Motion Tracking: No Silver Bullet, but a Respectable Arsenal. In: IEEE Computer Graphics and Applications, special issue on “Tracking”, November/December 2002, 22(6). 24-38.

[6] Dassault Systemes, Virtools developer, official site: http://www.3ds.com/home/, Official Virtools site: http://www.virtools.com/

[7] ARToolKit, Optical Tracking Software (M. Billinghurst, H. Kato), Official site: http://www.hitl.washington.edu/artoolkit/

[8] Sukthankar R., Stockton R. G., Mullin M. D., Smarter Presentations: Exploiting Homography in Camera-Projector Systems, in: Proceedings of International Conference on Computer Vision, 2001