A MULTI-CAMERA FRAMEWORK FOR INTERACTIVE VIDEO GAMES

Tom Cuypers, Cedric Vanaken, Yannick Francken, Frank Van Reeth and Philippe Bekaert
Hasselt University - Expertise Centre for Digital Media, Transnationale Universiteit Limburg
Wetenschapspark 2, 3590 Diepenbeek, Belgium

Keywords: Interactive Video Games, Video-Based Rendering, Human Computer Interaction.

Abstract: We present a framework that allows for a straightforward development of multi-camera controlled interactive video games. Compared to traditional gaming input devices, cameras provide players with many degrees of freedom and a natural kind of interaction. The use of cameras can even obsolete the need for special clothing or other tracking devices. This partly accounted for the success of the currently popular single-camera video games like the Sony Eyetoy. However, these games are fairly limited in the use of 3D scene information. Using multi-camera setups, this limitation can be surpassed, but typically many different image processing and computer vision techniques are involved. Our framework divides multi-camera based games into basic algorithms that are easily combinable into several sequentially executed stages. Therefore the amount of effort to develop new games can significantly be reduced. The capabilities of our framework are demonstrated with a number of conceptual games, proving that multi-camera controlled video games can be created with off-the-shelf hardware.

1 INTRODUCTION

Video game companies have recently started broadening their range of traditional input devices - mouse, keyboard, joystick - with alternative devices such as motion sensors and cameras. Sony successfully released camera based *Eyetoy* games for the Playstation 2 (Eyetoy Sony, 2003) and furthermore created best-sellers that are controlled with other non traditional input devices like microphones (Singstar Sony, 2004) and guitars (Guitar Hero Sony, 2005). Last year, Nintento released the Wii console (Nintendo Wii, 2006) of which the user-input is mainly based on the tracked motion of the Wii-Remote. The success of these games shows that a consumer demand exists for alternative input devices to control video games.

We choose to work with cameras because they provide multiple degrees of freedom as well as a natural kind of interaction for the gamer. Current image processing and tracking techniques eliminate the need for wearing special clothing or tracking devices. The live-captured images in camera controlled games furthermore allow for creating an immersive augmented reality.

Many commercial camera controlled video games and webcam applications have already been released, but the Playstation Eyetoy has set the standard for most of them. Albeit that these products do have some limitations. The use of a single camera restricts the developer to only work with 2D scene information. Without depth information, the gameplay will often only feel natural when the user performs movements in a plane parallel to the camera's image plane. Many 3D movements are undetectable due to the 2D projection on the camera's image plane, substantially restricting the range of interaction possibilities. The resulting interaction with the game will often be unintuitive and unsatisfying.

When working with multiple synchronized cameras that are viewing a scene from different positions and/or angles, 3D information of the scene can be reconstructed. This reconstructed information allows for creating games that are controlled by 3D human motions, providing players with a natural way of controlling a virtual avatar.

In this paper we present a framework that allows for easily creating video games that are controlled with several cameras. The framework is built keeping a highly modular paradigm in mind. The video games are created by implementing up to five different programming stages that are executed sequentially. Traditional or new image/video processing or computer

Cuypers T., Vanaken C., Francken Y., Van Reeth F. and Bekaert P. (2008).
A MULTI-CAMERA FRAMEWORK FOR INTERACTIVE VIDEO GAMES.
In *Proceedings of the Third International Conference on Computer Graphics Theory and Applications*, pages 443-449
DOI: 10.5220/0001096904430449

vision techniques can be individually supplied to each stage.

Our framework consists of the following five stages:

Input: capturing the scene and player using a number of calibrated and synchronized cameras

Video Processing: transforming the captured video frames into a format better suited for the efficient extraction of relevant data in subsequent steps

Depth Extraction: combining 2D information of the images to reconstruct 3D information

Post-processing: using the reconstructed 3D information to control the game

Output: composing visual output

An in-depth overview of these stages will be discussed later, as well as suggestions for algorithms that can be used in each stage. Some examples of video games that were created using our framework will be presented and discussed in the results section.

2 RELATED WORK

Cameras have been previously used as input hardware for various human computer interaction and virtual reality applications as well as for computer games. The quality of these applications is highly dependent on the hardware and computer vision software used. Several suitable vision algorithms along with possible interaction applications are discussed by Crowley et al. (Crowley, 1997; Crowley et al., 2000).

Camera controlled human computer interaction applications are also being used by persons with physical disabilities who are not able to conveniently use a mouse or a keyboard. These kind of interactions can be achieved by filming and analyzing certain body features (Betke et al., 2002) like eg. a person's eyes (Chen et al., 2001) or nose (Gorodnichy et al., 2002).

Balcisoy and Thalmann (Balcisoy and Thalmann, 1997) describe a technique that allows to record a real actor and place him into a virtual reality world. The actor can then interact with virtual characters, where the virtual characters' actions are chosen from prerecorded key-actions by a human operator. De Decker et al. (De Decker et al., 2007) propose a technique for interactive collision detection and response between real objects/people and virtual objects using multiple cameras.

Mitsubishi Electronics made a considerable contribution to the area of camera controlled interactive video games. To answer the need for low cost hardware and high speed processing power, they invented the Artificial Retina Chip (Kyuma et al., 1994; Freeman et al., 1996; Freeman et al., 1998), this by analogy with biological retinas that combine the functions of detection and processing. The chip has a built-in detector for mereley 32×32 to 256×256 resolution images and was used by Sega Saturn (Sega Saturn, 1994) and Nintendo Gameboy (Nintendo Gameboy, 1998). Current days, the most famous camera controlled games are those of Sony Playstation's Eyetoy (Eyetoy Sony, 2003).

Camera controlled applications receive their primary input from filmed objects and/or actors. These individual subjects can be located and tracked by using object tracking techniques (Metaxas and Terzopoulos, 1993; Wren et al., 1997).

Most traditional camera based applications are restricted to using only one 2D video input stream. With a number of calibrated cameras this input stream can be extended to obtain 3D information using vision techniques such as dense depth estimation (Scharstein and Szeliski, 2002) and visual hull reconstruction (Laurentini, 1994), each having its (dis)advantages. Depth estimation is especially useful when two cameras are used in a small baseline setup. Scharstein and Szeliski (Scharstein and Szeliski, 2002) have published an overview of different modern stereo depth estimation algorithms. The purpose of these algorithms is to create a disparity map from two small baseline cameras for one of the reference images. Alternatively, visual hull reconstructions represent an estimated 3D model of an actor or object. This 3D model is acquired by using silhouettes extracted from different video streams (Laurentini, 1994). The accuracy of visual hull reconstructions is higher when two or more cameras are placed in a wide baseline setup.

3 MULTI-CAMERA INTERACTION FRAMEWORK

In this section a more detailed description of our framework will be given. We subdivide the production process of camera controlled games into five high level stages, namely *input*, *video processing*, *depth extraction*, *post-processing* and *output*. For each stage, several appropriate algorithms are proposed. The combination of these stages allows to create interactive video games as well as general purpose applications. However, we mainly focus on real-time game interaction.

Although we present five different stages, not all of them need to be executed. For instance, by ignoring the *depth extraction* stage, regular single cam-

era games (Eyetoy Sony, 2003; Freeman et al., 1996; Freeman et al., 1998) can be created. For real-time user-interaction, one will most likely need to make a tradeoff between speed and quality, sometimes even resulting in eliminating or combining different stages. The implementation of each stage will furthermore depend highly on the kind of video game that needs to be developed, combined with the used camera setup, processing hardware etc.

3.1 Input

Typically two or three digital cameras are used to capture a real world actor that controls the virtual world. Nevertheless, other setups are possible, eg. remote network cameras, infrared cameras, prerecorded videos or alternative input hardware.

3.2 Video Processing

In this stage the video frames received from the input devices are transformed into a format better suited for efficient extraction of relevant data in subsequent stages.

One commonly used video processing operation is camera noise reduction. The necessity of this operation depends on the quality of the digital cameras as well as the lighting conditions in the scene. Input video frames can be denoised in the spatial, frequency and temporal domain (Dubois and Sabri, 1984; Gonzalez and Woods, 1992), typically having a loss of high frequency detail as a side effect.

Another commonly used technique in this stage is background subtraction. Camera controlled applications are mostly interested in foreground objects or actors, therefore background subtraction is inevitable (eg. shape-from-silhouette, see section 3.3). Interactive applications require the background to be eliminated in real-time. Blue or green screen matting techniques (Mishima, 1993; Smith and Blinn, 1996) fulfill this requirement, as well as (less efficient) techniques that handle more natural backgrounds (Cheung and Kamath, 2004; McIvor, 2000).

Many other image or video processing filters like motion detection (Spacek, 1986), skin detection (Vezhnevets et al., 2003), etc. also fit perfectly into this stage.

3.3 Depth Extraction

Given the filtered and annotated video frames, the required 3D information can be extracted.

When reconstructing the body of the actor without focussing on specific features, standard 3D reconstruction techniques such as dense depth estimation (Scharstein and Szeliski, 2002; Mühlmann et al., 2002) and visual hull modeling (Laurentini, 1994; Erol et al., 2005) are appropriate. Certain local dense depth estimation algorithms (Woetzel and Koch, 2004; Yang and Pollefeys, 2003) can get a significant speed boost when parallelized on multiprocessor systems such as those featured on current graphics hardware.

Visual hull methods are best suited for setups that contain at least three cameras, especially when novel views need to be synthesized. Although visual hull reconstruction is practically independent of image resolution, full model reconstruction has to be avoided whenever possible because of the high computational load. For example, in situations where the visual hull is only needed to determine physical collisions and responses, calculations can be performed directly in image space (De Decker et al., 2007).

Depending on the type of game, the full actor's motion or a selection of his body parts needs to be tracked. When it suffices to focus on particular body parts such as hands and head, standard object tracking (Kalman, 1960; Metaxas and Terzopoulos, 1993; Wren et al., 1997; Isard and Blake, 1998; Li et al., 2003) can be applied.

3.4 Post-processing

Up until this stage the necessary 2D/3D information has been gathered and processed to provide interaction between the real scene and the virtual scene. Using a standard physics engine, this can be achieved by applying collision detection and response between the player and virtual objects.

Also alternative avatar approximations and interactions are possible. A very plausible method to create an avatar for the player can consist of using kinematics. The player is represented by a skeleton and its pose is defined by the degrees of freedom of the joints. This pose can be obtained with forward kinematics by tracking the joints separately (Rubio et al., 2006), or with inverse kinematics (Welman, 1993; Parent, 2001) by tracking only the end-effectors and estimating the intermediate joints (Jaume et al., 2006). This skeleton provides many interaction possibilities from simple collision detection to 3D motion recognition etc.

3.5 Output

During the final stage, the visual output for the game has to be composed. Depending on the specifications, 2D and/or 3D data is rendered to a display. Since real

input frames were obtained at the input stage, a virtual but realistic approximation of the avatar can be integrated into the output using video-based rendering methods (Magnor, 2005). This virtual/augmented reality data can be displayed on a TV or computer screen, but can also be rendered to alternative output devices like 3D glasses.

4 RESULTS

In this section we demonstrate the power and the ease of use of our framework by discussing how the five stages are implemented for a number of conceptual interactive games. The optimal setup to capture 3D scene information is a blue/green screen environment that contains many cameras. Unfortunately, creating and setting up this kind of construction is often unpractical and using it may be computationally very expensive. Therefore we discuss several different games that make use of alternative setups. The presented games run at a framerate between 10 and 15 fps on a Pentium Mobile 1.5 GHz laptop, which is sufficiently accurate and usable for real-time interaction.

4.1 Frogger

This game is a camera controlled version of the classic Frogger game, introduced by Konami (Konami, 1981). The goal of Frogger is to cross the street without getting hit by cars. The player is filmed head-on and can move in any direction to navigate the avatar in the game. The setup consists of three small baseline Point Grey Research Flea2 (Point Grey Research, 2007) cameras and a diffuse green background. Navigation is achieved by locating the player in the input images using green screen matting (Smith and Blinn, 1996), applying local dense depth map estimation (Yang and Pollefeys, 2003) and calculating the center of mass. Because of the green screen matting, almost no specularities are visible, allowing us to make use of a local depth estimation method to robustly detect the center of mass. A structural scheme is shown in Figure 1.

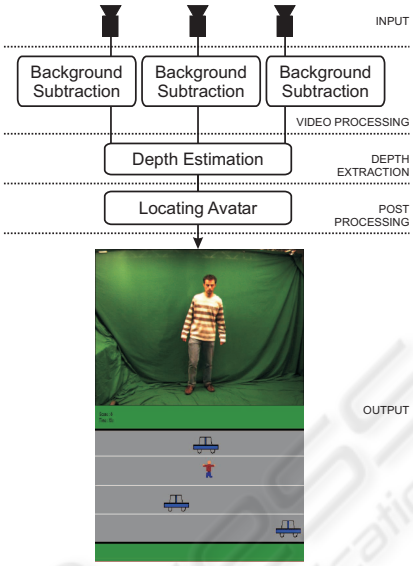

Figure 1: Scheme of Frogger.

4.2 Depth based Drawing

The goal of this game is to paint a given sequence of letters or symbols on the screen. The setup consists of three small baseline Point Grey Research Flea2 (Point Grey Research, 2007) cameras that film the player head-on. To make sure that moving persons in the background do not disrupt the gameplay, background changes behind a certain depth threshold are discarded. Skin detection (Vezhnevets et al., 2003) is performed on one image and a local depth estimation technique (Yang and Pollefeys, 2003) is executed on the skin-classified pixels. Due to the skin detection, possible specular highlights are automatically ignored, which makes a local depth estimation sufficient. Unfortunately, fast player movements are often undetected because of an excess of motion blur. The depth-thresholded skin-pixels are then fed into a letter recognition algorithm that we have built upon the LibStroke library (Willey, 1997). An overview is depicted in Figure 2.

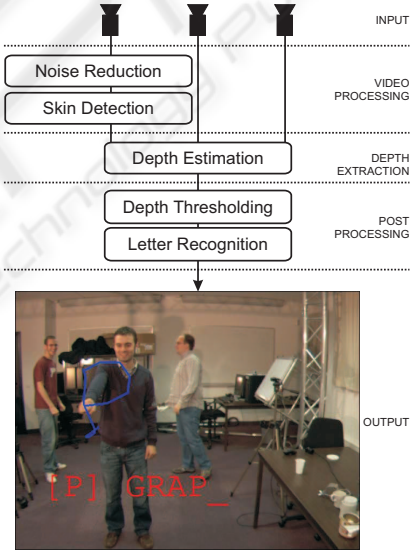

Figure 2: Scheme of the depth based drawing game.

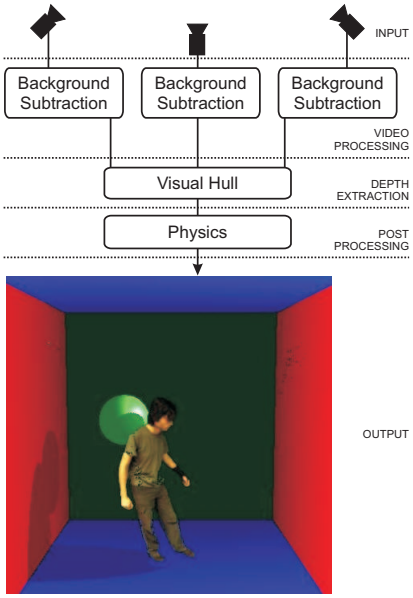

Figure 3: Scheme of Dodge-it.

4.3 Dodge-it

The goal of this game is to dodge the virtual objects that fall into the scene. A wide baseline setup is used that consists of three Point Grey Research Flea2 (Point Grey Research, 2007) cameras that film the player head-on and sideways. More cameras can be used in this setup, filming the player from an other side of the room, or from in-between angles etc., however note the trade-off between accuracy and speed. Green screen background subtraction (Smith and Blinn, 1996) is first performed on every image, and visual hulls (Laurentini, 1994) are calculated for the foreground objects and actors. Note that visual hulls are used because dense depth maps only allow for limited physical collisions. A voxel based visual hull is created of the foreground objects (Martin and Aggarwal, 1983). This model is used to check for collision detection and response with virtual objects, handled by ODE (Open Dynamics Engine, 2001). The output is a combination of a virtual rendered scene and a billboard of the player. Shadows have been added to provide an extra depth cue. An overview is given in Figure 3.

4.4 Spiral

Spiral is a digital version of an old traditional folk game where one must navigate a metal ring around a curved tube while avoiding collisions between the ring and the tube. Herefore an object, containing three different colors, is tracked using a wide baseline setup of two (or three) Point Grey Research Flea2 (Point Grey Research, 2007) cameras. We use the Condensation algorithm (Isard and Blake, 1998) - a popular variant of the particle filter - to track this object in the *depth extraction* stage. The calculated location and orientation is then mapped into the virtual world. Figure 4 shows this game where the player holds the tracked object that controls the ring in the game. Collision detection between the ring and the spiral is performed to define the player's score. The output is a pure virtual world.

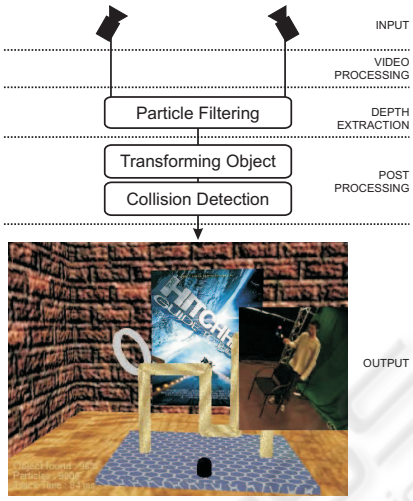

Figure 4: Scheme of spiral.

4.5 Roboshoot

The purpose of Roboshoot is to shoot moving virtual 3D objects in the scene. The player is filmed head-on by a Point Grey Research Bumblebee (Point Grey Research, 2007) camera and can move his arms in the direction he wants to aim his virtual guns at. First, skin detection (Vezhnevets et al., 2003) is performed to locate the hands and face of the player in one of the input images. This 2D location is expanded to a 3D location in the *depth extraction* stage by using the depth map which is automatically created by the Bumblebee camera. The *transposed Jacobian method* (Welman, 1993) for inverse kinematics is then used with these locations to estimate the upper-body pose of the player, defining the movements of the avatar. Although these virtual movements are not always exactly corresponding to the real movements, this proves to be a very efficient and plausible pose approximation method. However, problems may occur when bodyparts other than face and hands are exposed. Figure 5 contains a synopsis of the game.

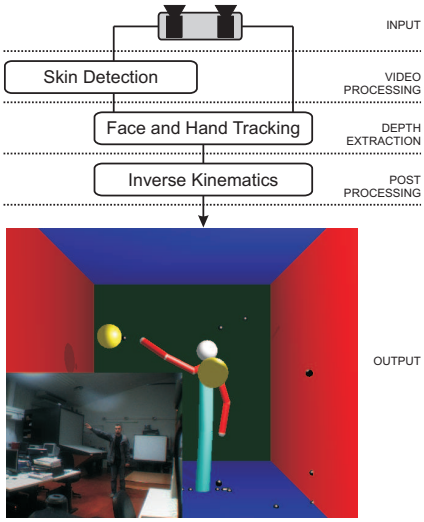

Figure 5: Scheme of Roboshoot.

5 CONCLUSIONS

We have presented a framework targeted at creating video games that are controlled by input from multiple cameras. Our framework embodies five modular stages from which each serves a different role in the execution of the game. While the purpose of each stage is fixed during the lifetime of a video game, novel algorithms can easily be plugged in to replace or complement already used algorithms. A proof of concept is provided by discussing several interactive video games that were implemented using the framework. These multi-camera controlled games not only show the usability of the framework, but also prove to be an asset to standard game interaction techniques. We believe that our framework is applicable to various other forms of human computer interaction.

6 FUTURE WORK

Currently our framework mainly focuses on single setup vision-based interaction methods. Integrating a specific network layer could allow for creating online camera-based multiplayer video games. Providing multi-modal interaction possibilities such as combinations with standard game controllers, speech recognition etc. can furthermore strengthen of this framework.

Although we believe that off-the-shelf technology is ready for multi-camera computer games for home users, user friendly setup calibration methods still need further investigation. While fixed pre-calibrated cameras such as the Bumblebee (Point Grey Research, 2007) are already available, wide baseline setups are location dependent and need to be calibrated on the spot. In general, adaptive scene dependent estimation of the setup parameters such as camera calibration data, image resolutions, exposure times, enabling or disabling denoising, etc. is an interesting future direction.

Experimenting with other hardware such as an infrared projector-camera combination might also improve our system, however this hardware is less commonly used.

ACKNOWLEDGEMENTS

We gratefully express our gratitude to the European Regional Development Fund (ERDF), the Flemish Government and the Flemish Interdisciplinary institute for Broadband Technology (IBBT), which are kindly funding part of the research at the Expertise Centre for Digital Media. Part of the work is also funded by the European research project IST-2-511316-IP : IP-RACINE (Integrated Project Research Area CINE). We also like to thank our colleagues, especially Maarten Dumont and Steven Maesen, for their time and effort in this work.

REFERENCES

Balcisoy, S. and Thalmann, D. (1997). Interaction between real and virtual humans in augmented reality. In *CA '97: Proceedings of the Computer Animation*, page 31, Washington, DC, USA. IEEE Computer Society.

Betke, M., Gips, J., and Fleming, P. (2002). The Camera Mouse: visual tracking of body features to provide computer access for people with severe disabilities. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on [see also IEEE Trans. on Rehabilitation Engineering]*, 10(1):1–10.

Chen, Y., Su, C., Chen, J., Chen, C., Hung, Y., and Fuh, C. (2001). Video-based eye tracking for autostereoscopic displays. *Optical Engineering*, 40:2726.

Cheung, S. S. and Kamath, C. (2004). Robust techniques for background subtraction in urban traffic video. *Proceedings of Video Communications and Image Processing*, 5308:881–892.

Crowley, J. L. (1997). Vision for man machine interaction. *Robotics and Autonomous Systems*, 19(3-4):347–359.

Crowley, J. L., Coutaz, J., and Brard, F. (2000). Things that see: Machine perception for human computer interaction. *ICommunications of the A.C.M.*, 43(3):55–64.

De Decker, B., Mertens, T., and Bekaert, P. (2007). Interactive collision detection for free-viewpoint video. In *GRAPP 2007: Proceedings of the Second International Conference on Computer Graphics Theory and Applications*, pages 114–120.

Dubois, E. and Sabri, S. (1984). Noise Reduction in Image Sequences Using Motion-Compensated Temporal Filtering. *Communications, IEEE Transactions on [legacy, pre-1988]*, 32(7):826–831.

Erol, A., Bebis, G., Boyle, R. D., and Nicolescu, M. (2005). Visual hull construction using adaptive sampling. In *WACV-MOTION '05: Proceedings of the Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTION'05) - Volume 1*, pages 234–241, Washington, DC, USA. IEEE Computer Society.

Eyetoy Sony (2003). http://www.eyetoy.com.

Freeman, W. T., Anderson, D. B., Beardsley, P. A., Dodge, C. N., Roth, M., Weissman, C. D., Yerazunis, W. S., Kage, H., Kyuma, K., Miyake, Y., and Tanaka, K. (1998). Computer vision for interactive computer graphics. *IEEE Comput. Graph. Appl.*, 18(3):42–53.

Freeman, W. T., Tanaka, K., Ohta, J., and Kyuma, K. (1996). Computer vision for computer games. In *FG '96: Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition (FG '96)*, page 100, Washington, DC, USA. IEEE Computer Society.

Gonzalez, R. C. and Woods, R. E. (1992). *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Gorodnichy, D., Malik, S., and Roth, G. (2002). Nouse Use your nose as a mouse-a new technology for hands-free games and interfaces. *Proceedings of International Conference on Vision Interface (VI2002)*, pages 354–361.

Guitar Hero Sony (2005). http://www.guitarherogame.com.

Isard, M. and Blake, A. (1998). Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28.

Jaume, A., Varona, J., Gonzalez, M., Mas, R., and Perales, F. J. (2006). Automatic human body modeling for vision-based moion capture. *WSCG*.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45.

Konami (1981). http://www.konami.com.

Kyuma, K., Lange, E., Ohta, J., Hermanns, A., Banish, B., and Oita, M. (1994). Artificial retinas-fast, versatile image processors. *Nature*, 372(6502):197–198.

Laurentini, A. (1994). The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162.

Li, P., Zhang, T., and Pece, A. (2003). Visual contour tracking based on particle filters. *Image Vision Comput.*, 21(1):111–123.

Magnor, M. A. (2005). *Video-Based Rendering*. AK Peters Ltd.

Martin, W. and Aggarwal, J. K. (1983). Volumetric descriptions of objects from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):150–158.

McIvor, A. M. (2000). Background subtraction techniques.

Metaxas, D. and Terzopoulos, D. (1993). Shape and nonrigid motion estimation through physics-based synthesis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(6):580–591.

Mishima, Y. (1993). Soft edge chroma-key generation based upon hexoctahedral color space. *U.S. Patent 5,355,174*.

Mühlmann, K., Maier, D., Hesser, J., and Männer, R. (2002). Calculating dense disparity maps from color stereo images, an efficient implementation. *Int. J. Comput. Vision*, 47(1-3):79–88.

Nintendo Gameboy (1998). http://www.gameboy.com.

Nintendo Wii (2006). http://wii.com.

Open Dynamics Engine (2001). http://www.ode.org/.

Parent, R. (2001). *Computer animation: algorithms and techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Point Grey Research (2007). http://www.ptgrey.com.

Rubio, J. M. B., López, F. J. P., Hidalgo, M. G., and Varona, X. (2006). Upper body tracking for interactive applications. *AMDO*.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*, 47(1-3):7–42.

Sega Saturn (1994). http://www.sega.com.

Singstar Sony (2004). http://www.singstargame.com.

Smith, A. R. and Blinn, J. F. (1996). Blue screen matting. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 259–268, New York, NY, USA. ACM Press.

Spacek, L. A. (1986). Edge detection and motion detection. *Image Vision Computing*, 4(1):43–56.

Vezhnevets, V., Sazonov, V., and Andreeva, A. (2003). A survey on pixel-based skin color detection techniques. *Graphics and Media Laboratory, Faculty of Computational Mathematics and Cybernetics, Moscow State University, Moscow, Russia*.

Welman, C. (1993). Inverse kinematics and geometric constraints for articulated figure manipulation. Master's thesis, B.Sc. Simon Fraser University.

Willey, M. (1997). Design and implementation of a stroke interface library.

Woetzel, J. and Koch, R. (2004). Real-time multi-stereo depth estimation on gpu with approximative discontinuity. *1st European Conference on Visual Media Production (CVMP 2004)*, pages 245–254.

Wren, C. R., Azarbayejani, A., Darrell, T., and Pentland, A. P. (1997). Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):780–785.

Yang, R. and Pollefeys, M. (2003). Multi-resolution real-time stereo on commodity graphics hardware. *cvpr*, 01:211.