Journal of Korea Multimedia Society Vol. 13, No. 12, December 2010(pp. 1832-1842)

A Study on Contents Manufacturing System for Massive Contents Production

Su Mi Ji[†], Jeong Joong Lee[†], Sangpill Kwon[†], Jin Guk Kim[†], ChangMan Yu[†], Jeong-Gyu Lee[†], Sejong Jeon[†], Taewan Jeong[†], Dongwann Kang[††], Sang Il Park[†], Oh-young Song[†], Jong Weon Lee[†], Kyung-Hyun Yoon[††], Chang-wan Han[†††], Sung Wook Baik[†]

ABSTRACT

This paper introduces a new automatic processing system: "Contents Factory" for the mass production of contents. Through the contents factory, we provide an authoring environment to improve the usability and the efficiency in producing contents. The contents factory integrates recycling techniques for contents resources, contents development engines, authoring tools, and interfaces into a total processing system. Since it is multi-platform based including mobile devices as well as PCs, one can easily produce complete PC and mobile contents from raw resources. We produced an example, "Sejong square" via the contents factory in order to demonstrate its effectiveness and usability

Key words: Contents Manufacturing System, 3D Resource Recycling Technique, Contents Authoring, Contents Factory

1. INTRODUCTION

3D Contents such as computer games and animations have rapidly evolved from simple graphics. In the past, 3D contents were created by game developers and animation producers in conjunction with specialists. Recently, individual developers and producers have been interested in contents production for their own purposes. Thus, more convenient contents development environments are required for them. In particular, as 3D contents for smart phones such as Apple iPhone and Android OS based products have recently become more popular, 3D contents developers have become more competitive in creating higher quality contents products. There are few researches to provide convenient development environments for contents producers, even though some researches [1-4] have been conducted to provide development environments for simple 3D presentations.

An autonomous contents manufacturing system has been developed in order to improve the efficiency of contents production in this study. Many resources for 3D graphic rendering are provided within this system to avoid redundant production of contents resources such as 3D characters, objects, motions and so on. This contents-authoring environment with the system is called Contents Factory.

※ Corresponding Author : Sung Wook Baik, Address : (143-747) College of Electronic and Information Engineering, Sejong University, 98 Gunja-dong, Gwangjin-gu, Seoul, Korea, TEL : +82-2-3408-3797, FAX : +82-2-3408-4339, E-mail : sbaik@sejong.ac.kr
Receipt date : Nov. 30, 2010, Revision date : Dec. 13, 2010
Approval date : Dec. 28, 2010
[†] Dept of Digital Contents, Sejong University
[††] Dept of Computer Science and Engineering, Chung-Ang University
[†††] Dept of Cartoon and Animation, Sejong University
※ This work is supported by the Seoul R & BD program (10557), Korea Creative Content Agency(KOCCA) in the Culture Technology(CT) Research & Developement Program 2010 and Ministry of Culture(1-10-7602-002-10005-00-005), Sports and Tourism(MCST) and Korea Culture Content Agency(KOCCA) in the Culture Technology(CT) Research & Developement Program 2009 and Ministry of Culture, Sports and Tourism(MCST) (21076020021098505003).

2. RELATED WORKS

Havok is physics engine software developed by Havok.com [5]. The version 1.0 of Havok was presented in the Game Developers Conference 2000 (GDC 2000). Recently, version 7.1 was published. The engine supports almost all platforms (MS Windows, XBOX, Nintendo Wii, Sony Playstation, MAC OSX, and Linux). PhyX is a real-time physics engine SDK developed by Ageia, NVIDIA [6]. This SDK supports Geforce GPU acceleration functions using CUDA. The SDK is open to MS windows and Linux developers. Unreal Engine 3 is the third generation of 3D game engines, developed by Epic, USA [7]. Recently, Unreal Engine 3 for iPhone was presented at GDC 2010.

The paper [8] presented an improved corner finding algorithm for sketch-based interfaces. The algorithm efficiently removes incorrectly-detected corners. Thorne et al. [9] proposed a novel system for sketching the motion of a simple human character. This sketch system supports 18 different types of motions.

The most well-known algorithms to implement soft shadows are the Heckbert and Herf algorithm, the Haines algorithm, and the Gooch algorithm. Heckbert and Herf [10] created soft shadows using multiple light sources. Haines [11] created soft shadows using an object's silhouette edges, which were not accurate but realistic. Gooch [12] proposed a method that was especially suited for technical illustration. The same shadow mask was projected multiple times and translated to create soft shadows.

The cartoon rendering for 3D objects was proposed in [13] for the first time. Decaudin got a cartoon effect using shading which divides a dot product value between the light vector and normal vector into several steps. [14] suggested the method that extracts edges from 3D object using GPU. They regard the non-continuous part of normal and depth buffer as edges.

Humans interact with each other without any device in their hands so it is natural for humans to interact with computers with their bare hands. For bare hand interaction, hands have to be detected. Many researchers including C. Hardenberg[15] detected hands using skin colors, but this approach failed whenever a background contained areas with color similar to skin colors. K. Oka[16] and other researchers detected hands based on body temperature captured by an infrared camera, which was quite expensive.

3. AUTOMATIC CONTENTS MANUFACTURING SYSTEM

The autonomous contents manufacturing system mainly consists of a contents-development engine and a contents authoring tool for 3D games and animations. (Figure 1)

3.1 Contents Development Engine (G3 Engine)

The Contents Development Engine is composed of a rendering engine, a physics engine, a special effects engine, an artificial intelligence engine, and a cartoon rendering engine. The engine scan creates realistic scenes from various effects created by its components.

3.1.1 Rendering Engine

In addition to creating realistic scenes, flexibility is taken into consideration when the rendering engine was developed. Contents developed for desktop computers can be easily ported to mobile devices with minor modifications. Multiple characters can share one animation, and the shapes of characters can be easily converted into other ones. Two tools are also developed to facilitate the content developing process. The data conversion tool can convert existing data so they can be used in the rendering engine, and the viewing tool can show data in scene graphs.

Shadows are needed for realistic scenes, but it

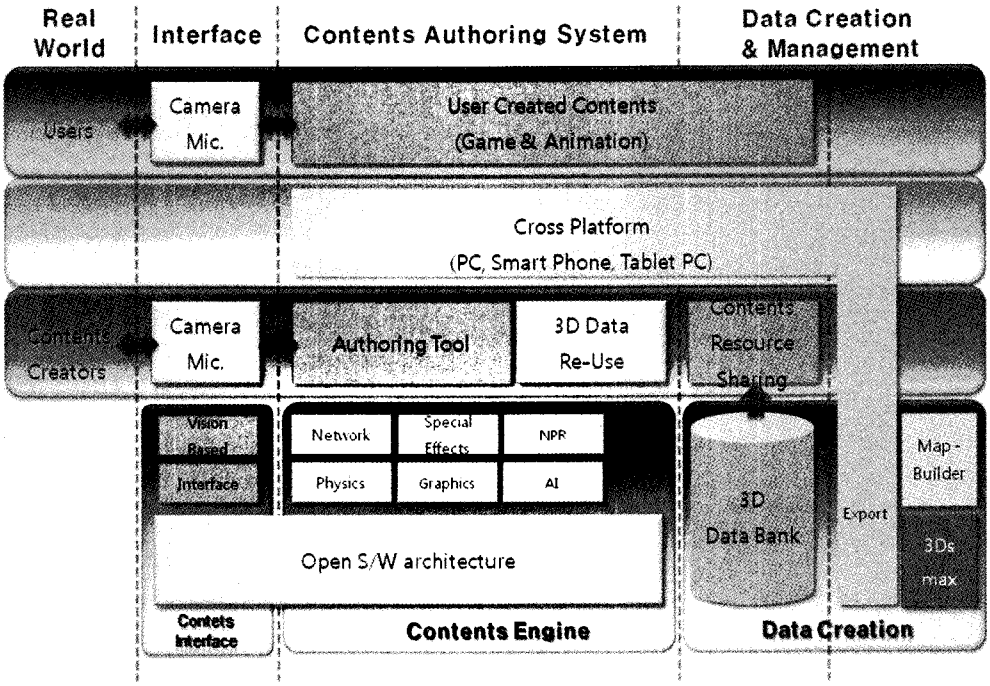
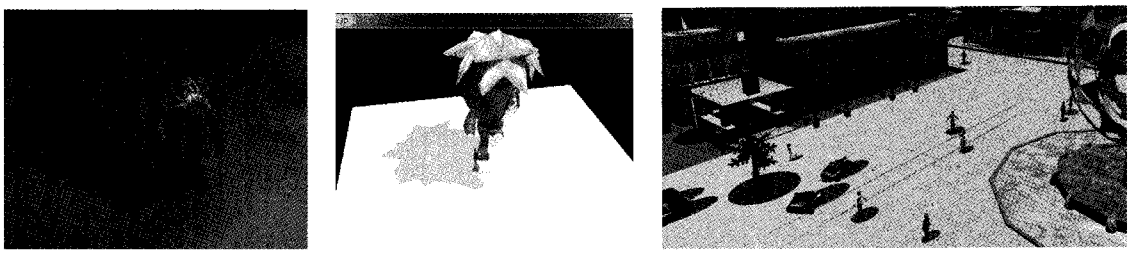


Fig. 1. Content Factory Environment.

is difficult to create shadows in real-time, especially in mobile devices. Since flexibility is one of the main concerns for developing the rendering engine, soft shadows, hard shadows, and simple oval shadows are implemented as shown in Figure 2 so an appropriate shadow can be chosen for all contents. Soft shadows can be implemented by several methods. A shadow map [17] is used as the base of the rendering engine.

3.1.2 Physics engine

A physics engine plays an essential role in producing physical movements of the natural phenomena in the contents. In the PC environment, some commercial engines such as Havok [5] and PhysX [6] are widely used to develop games. Recently, several game engines which include simulation modules for physical phenomena have been published [18]. "Epic Citadel" [19], which was published via the iPhone App store in 2010, utilizes a simulation module of Unreal Engine 3 [18] to implement the physical motion of a flag. However, the conventional mobile-based physics engines are not mature yet. Moreover, physics engines that



(a) (b) (c)

Fig. 2. Types of shadows (a) soft shadows (b) hard shadows (c) simple oval shadows.

support a multi-platform environment including mobile devices are not common. So, we developed the cross-platform physics engine to work on PCs and mobile devices, and then integrated it into the G3 engine. Our physics engine enables one to simulate a box-shaped rigid body including collision handling in real-time. For adapting to a mobile environment, the engine focuses on minimization of computational cost rather than correctness of the simulation. Also, all modules of the G3 physics engine are implemented using standard C++ for maximizing the portability between PCs and mobiles. The current version of the physics engine is specialized to polygon-based primitive objects but the next version will support various geometry types.

3.1.3 Special effects engine

The special effects are used to improve realism and immersiveness of the contents as giving secondary effects. We utilize the special effects mainly in animating subjects such as characters. Our special effects engine is based on the particle system, which is computationally efficient and is scalable in any device - PCs or mobile devices. The special effects engine can easily generate natural phenomena (smoke, fire, snow, rain, etc.) with tweaking control parameters of the procedural functions. It also gives shading parameters to adjust transparency and textures. When rendering the particle effects, we use a billboard method which can set an arbitrary axis as well as the fixed Y-axis. Also, the rendering module supports (1) two-pass rendering for harmonizing the effects with background objects and (2) sprite-based animation for generating various effects only with several textures. Besides the particle module, we integrate a "shallow water simulation" module into the special effects engine for producing height field based water wave motion. To render water surfaces, a GLSL shader of OpenGL 2.0 was developed.

3.1.4 AI Engine

AI modules for behavior determination have been developed to make the NPC (NonPlayer Character) adapt intelligently to given situations in computer games. NPCs can autonomously behave with behavior pattern recognition based on FSM (Finite State Machine). The genetic algorithm, fuzzy theory, neural networks, and machine learning will be applied to behavior determination of 1) an NPC's status such as its stamina, fatigue, and hunger, and 2) an NPC's characteristics such as age and personality.

3.1.5 Cartoon Rendering Engine

Cartoon rendering is a method that represents a 3D object using simple colors like a cartoon. This eliminates the super reality of a 3D object which is rendered by computer and flattens the color of an object[13]. As it makes a fancy image, cartoon rendering is being used widely in computer games.

In our system, we use a step function that converts the smooth intensity obtained using the Lambertian lighting into several steps. This method is similar to conventional cartoon shading that uses a 1D texture which consists of several steps of intensity. We get the color of an object c_t using the equation below in programable shaders.

$$c_t = \begin{cases} k_1 \times c, & \max(\bar{l}\cdot\bar{n}, 0) \le t_1 \\ k_2 \times c, & t_1 < \max(\bar{l}\cdot\bar{n}, 0) \le t_2 \\ \vdots & \\ k_{m-1} \times c, & t_{m-2} < \max(\bar{l}\cdot\bar{n}, 0) \le t_{m-1} \\ c, & \max(\bar{l}\cdot\bar{n}, 0) > t_{m-1} \\ & (0 \le k_1 \le k_2 \le \cdots \le k_{m-1} \le 1) \\ & (0 \le t_1 \le t_2 \le \cdots \le t_{m-1} \le 1) \end{cases}$$

c is the color of an object, $\bar{l}$ is the light vector and $\bar{n}$ is the normal vector of object surface. This equation makes intensities which consist of m steps. In this study, we use 3 steps and values. $k_1 = 0.3, k_2 = 0.6, t_1 = 0.3, t_2 = 0.6$.

For more effective cartoon rendering, some methods which draw an outline are occasionally used [13,14]. We detect pixels on outline and render them using below equation in programable shaders.

$$g(x) = \sum \frac{d(N(x)) - d(x)}{8}$$

$$e(x) = \begin{cases} 1, & \text{if } g(x) \geq s \\ 0, & \text{else} \end{cases}$$

$d(x)$ is the depth on pixel x, $N(x)$ is neighbor pixels of pixel x, $\bar{l}$ is the light vector, and $\bar{n}$ is normal vector on pixel x. This method detects the pixels of which depth variation is bigger than threshold value, s into outline.

As this method is very simple and effectively makes cartoon-like scenes, it shows good performance in the mobile environment which has many limitations compared to the PC. Therefore this method is suitable for our system on the cross platform.

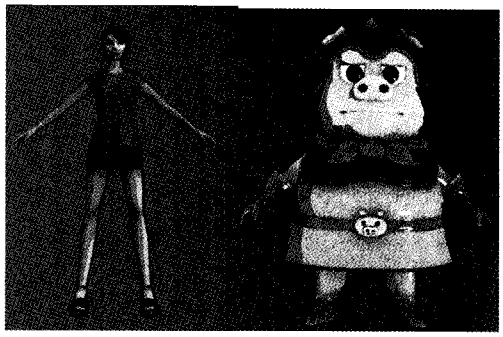


Fig. 3. The result of our cartoon rendering engine.

3.2 Resource-reuse Method

We developed a method for sharing resources in developing different contents. Resources such as characters, motion, and accessories are usually designed for a specific content, and require much effort if adapting them into developing a new content. Our method aims to reduce the effort. The method addresses two main issues: one is for retargeting the resources, and the other is for building a virtual directory for sharing.

3.2.1 Retargeting the Resources

Our system mainly focuses on the reuse of the motion database. Motion data are generally actor-specific, which means adapting to a new character is not trivial; a simple mapping of the motion to a new character will cause artifacts in the resulting animation such as a foot sliding and penetrating into the ground. To provide an effective and efficient tool for novice users for handling the motion data, we developed an automatic rigging method, which enables the users to assign the skinning weight values by selecting a few key features in a character.

Given a surface model, the user picks 8 feature points on the model including the forehead, right/left wrists, right/left shoulders, right/left ankles, and pelvis. In practice, selecting these 8 feature points was not hard even to a novice user because of our interactive interface. Based on this annotation, we fit a pre-defined skeleton structure inside the given model by adjusting the position of the joints and the length of the links. Based on the proximity between a link of the skeleton and the given surface, we compute the skinning weight values of each vertex of the surface. Finally, inspired by the method of [20], the weight values are smoothed over the surface in order to avoid any undesired discontinuity in the resulting deformation when applying any motion data. Figure 4 illustrates the process of our automatic rigging method.

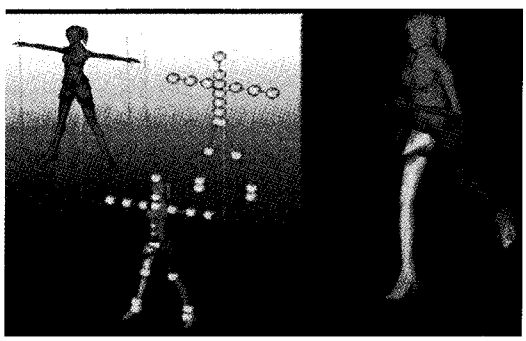


Fig. 4. Process of automatic rigging method.

3.3 Contents Authoring Tool

This contents authoring tool provides novices with the intuitive contents production concept.

This tool consists of essential functions, special effect production, and human motion generation for contents authoring.

3.3.1 The authoring tool for special effects

The authoring tool is essential to efficiently produce phenomena that have very high degrees of freedom like special effects. We designed the authoring tool for special effects by making reference to the user interface of the Unreal 3 editor [20]. The Unreal 3 editor can generate plenty of effects. However, it has too many controlling options and is too complicated even for professional artists. Our authoring tool for special effects (FX authoring tool) has only the essential functions to efficiently produce fire- and water-related effects. The first step for producing effects is to create a particle system object. The second step is to choose a billboard option according to the type of effects and to tweak the degree of transparency. The third step is to enable or disable sprite animation effects with controlling texture-coordinate options. The final step is to tweak the parameters related to time, color and position of effects. The generated effects can be shared with Contents Factoryby using scripts and the reference UI. Using script modules of the G3 engine, the special effects can be showed in the mobile environment, which has no FX authoring tool. Figure 5 shows a screen shot of our FX authoring tool based on GUI the 3D view for FX (upper left), the particle system (upper right) and the attribute control (below).

Fig. 5. Authoring tool for special effects.

Besides the GUI-based authoring tool, we provide a sketch interface tool for non-specialists. The sketch tool consists of the sketch recognition module and the effect manipulation module. In stage of sketch recognition, the "Improved Shortstraw" algorithm [8] is employed to detect corners of a sketch and the tokenization technique, which is proposed in "Motion Doodles" [9], is used to decide the type of effects - fire, smoke, explosion, or mushroom detonation. In stage of effects manipulation, sketch interaction is interpreted as manipulation of the characteristic curve of effects (Figure 6).

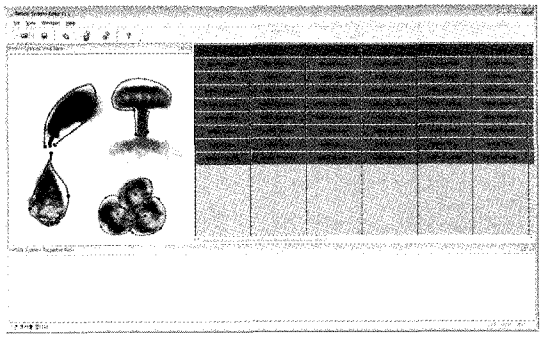


Fig. 6. sketch interaction for special effects.

3.3.2 Tool for generating human motion (motion retargetting technique)

We provide an interactive tool for generating natural-looking human walking motion sequences. Given a trajectory of a character by using a mouse as an input device, a walking motion is generated on-the-fly which changes its speed and turning angle along with the given constraints. The key idea of this technique is to blend a set of motion clips that are different in their speed and trajectories from each other. Then, the motion generation is formulated as a problem of finding the blending weight values of the clips in order to satisfy the given goals. We employed the scattered data interpolation technique of [21] for computing the blending weights, in which both the example clips and the goal constraints are treated as the sample

points in the parameter space, and the weight values are computed based on the spatial relationships among them. This motion blending method enables us to achieve realism in the resulting motion as well as the efficiency in computing.

3.4 Vision-based Interfaces

Vision-based interfaces are developed to provide natural interactions with contents. Artificial markers are used for the interface since they can be recognized clearly from input images. An infrared camera is used for hand recognition-based interfaces since it is robust to illumination changes.

3.4.1 Marker recognition-based Interface

Aninterface based on the artificial marker recognition has been developed to provide more intuitive interaction to users. Artificial Markers have been designed so that they can be matched well with relevant characters and objects. Figure 7 presents artificial marker samples which have been designed similarly to real objects. Thanks to such similarity, users can easily select any markers that they want. Figure 8 presents an animation contents authoring using an interface based marker recognition. Several markers perceived from several web-cams can be interpreted by an internal recognition module.

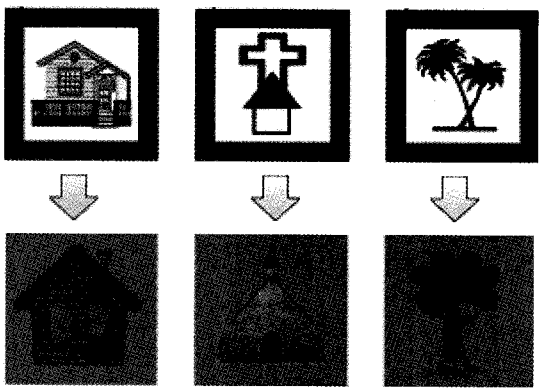


Fig. 7. Artificial Marker Samples.

3.4.2 Hand recognition-based Interface

An interface based on hand recognition is developed to provide intuitive interaction to users. Infrared cameras can robustly recognize rays coming from monitors, so infrared cameras are used for the hand recognition-based interface. The motions of the user's hands on the monitor are recognized, and corresponding operations are executed. Two hands can be recognized in real time, and users can rotate, zoom, and select objects easily. The fluorescent lamp does not emit infrared radiation so the presented interface can be used in an indoor environment. However, the sun emits infrared radiation so a user has to block the sunlight coming from a window.

4. CONTENTS PRODUCTION RESULTS

Game contents called "Sejong (Univ.) Square" was produced in the contents factory including a main authoring environment with extra facilities. The game was developed by novices with 3D resources including 3D characters, backgrounds, and motions for 3D characters. Buildings, streets and

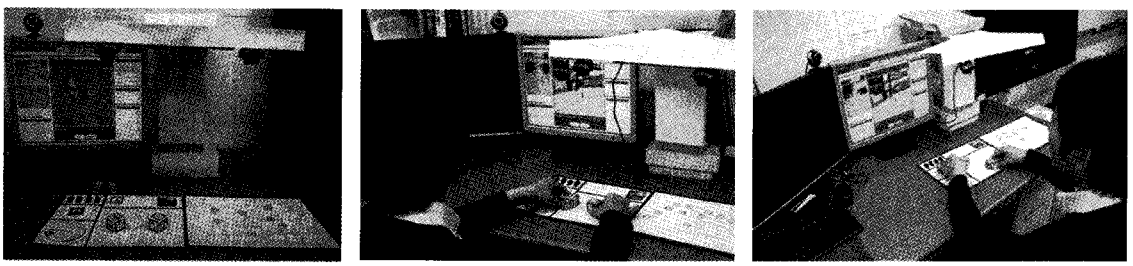


Fig. 8. Animation Contents Authoring using an Interface based on Marker Recognition.

street lamps could be easily deployed on the square by using maker recognition based interfaces. By using the tool for generating human motion, non-player characters (NPCs) can get extra motion data from the main character for which most motions had been already produced. Due to the special effects engine, water, fire, lights of street lamps and fireworks are displayed.

A main character can be controlled so that it can go around the square to get a variety of situational experiences. Also, NPCs can adapt to unknown situations since they are endowed with artificial intelligence. Eventually, the game contents could be stored with a script format. Thus, the contents can be played on multi-platforms. Figure 9 presents a screen shot of the developed game contents.

Fig. 9. User created contents.

5. CONCLUSION

In this paper, we have proposed a new notion of "content factory" for the automatic manufacture of the digital content. In order to improve the efficiency of the manufacture, our system provided an integrated frame work consisting of the four sub-tools: the content engine, the resource-reuse method, the content-creation method, and the interface method. Each of these tools is specifically and carefully designed to archive the common goal of providing an easy-to-use tool for every user for efficiently creating digital content. To prove the usefulness of our system, we developed several playable game contents by using the system.

REFERENCES

[1] V. Bonifaci, C. Demetrescu, I. Finocchi, and L. Laura, "A Java-based System for Building Animated Presentations over the Web," *Science of Computer Programming*, Vol.53, No.1, pp. 37-49, 2004

[2] C. Elcacho, A. Schfer, R. Drner, and V. Luckas, "Performing 3D Scene and animation Authoring Task Efficiently: An Innovative Approach", *Computer Graphics International*, pp.242-244, 1998

[3] K. Miyazakil, Y. Nagail, and R. Nakatsul, "Architecture of an Authoring System to Support Interactive Contents Creation for Games/E-Learning", *Technologies for E-Learning and Digital Entertainment*, LNCS 3942, pp.70-79, 2006

[4] M. McNaughton, D. Szafron, J. Schaeffer, J. Redford, and D. Parker, "ScriptEase: Generating Scripting Code for Computer Role-Playing Games", *Automated Software Engineering*, pp.386-387, 2004

[5] HAVOK.COM.INC., 2008. Havok physics 5.5, http://www.havok.com/index.php?page=havok-physics.

[6] NVIDIA, 2009. PhysX 9.09.1112

[7] Adam Lake, Carl Marshall, Mark Harris, and Marc Blackstein, "Stylized Rendering Techniques for Scalable Real-Time 3D Animation," *NPAR*, pp.13-20, 2000

[8] Xiong Y., and LaViola J., "Revisiting ShortStraw-Improving Corner Finding in Sketch-

Based Interfaces," *EUROGRAPHICS Symposium on Sketch-Based Interfaces and Modeling*, pp.101-108, 2009.

[9] Matthew T., David B., and Michiel V., "Motion doodles: An interface for sketching character motion," *In SIGGRAPH: ACM Special Interest Group on Computer Graphics and Interactive Techniques*, pp.424-431, 2006.

[10] Paul S. Heckbert, and Michael Herf, "Simulating Soft Shadows with Graphics Hardwaare," Technical Report CMU-CS-97-104, Carnegie Mellon University, 1997.

[11] Eric Hains, "Soft Planar Shadows Using Plateaus," *Journal of Graphics Tools*, Vol.6, No.1, pp.19-27, 2001.

[12] Amy Gooch, Bruce Gooch, Peter Shirley, and Elaine Cohen, "A Non-Photorealistic Lighting Model for Automatic Technical Illustration," *SIGGRAPH 98*, pp.447-452, 1998.

[13] Christian von, Hardenberg and Francois Berard, "Bare-Hand Human-Computer Interaction," *Proceedings of the 2001 workshop on Perceptive user interfaces*, 2001.

[14] Kenji Oka, Yoichi Sato, and Hideki Koike, "Real-Time Fingertip Tracking and Gesture Recognition," *IEEE Computer Graphics and Applications*, 2002.

[15] Real-time Shadow Algorithms and Techniques, developer.nvidia.com/object/doc_shadows.html

[16] Epic Games, 2010. Unreal Engine 3 on Mobile, http://www.epicgames.com/technology//

[17] Epic Games, 2010. Epic Citadel, http://www.epicgames.com/technology/epic-citadel/ *In SIGGRAPH: ACM Special Interest Group on Computer Graphics and Interactive Techniques*, 2006/

[18] Decaudin, P., Cartoon-Looking Rendering of 3D-Scenes, Technical Report 2919, INRIA, 1996.

[19] Nienhaus, M. and Döllner, J. Blueprint Rendering and 'Sketchy Drawings', GPU Gems II: Programming Techniques for High Performance Graphics and General-Purpose Computation, 2004.

[20] Ilya Baran and Jovan Popović, "Automatic Rigging and Animation of 3D Characters," *ACM Transactions on Graphics*, Vol.26, No.3, 2007.

[21] S. Park, H. Shin, and S. Shin, "On-line Locomotion Generation Based on Motion Blending," *In Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp.105-111, 2002.

Su Mi Ji

is a doctoral student in the Department of Digital Contents at Sejong University. Her research interests include serious game and Contents authoring tool. she has a ME in the Department of Education game from Kwangwoon University.

Jeong Joong Lee

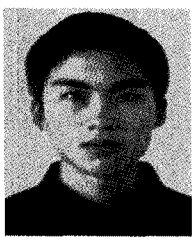
is a doctoral student in the Department of Digital Contents at Sejong University. His research interests include AI and contents authoring tool. he has a ME in the Department of e-Learning Contents from Kwangwoon University.

Sangpill Kwon

is a master's student in the Department of Digital Contents at Sejong University. His research interests include Computer graphics and 3D character animation. He has a BS in the Department of Digital Contents from Sejong University.

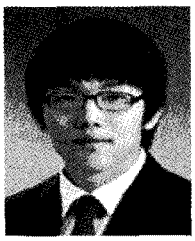
Jin Guk Kim

is a master's student in Digital Contents at Sejong University. His research interests include Mobile AR, Mobile Interaction and HCI. He has a BS in the Department of Digital Contents from Sejong University.



ChangMan Yu

is a master's student in the Department of Digital Contents at Sejong University. His research interests include Physics-based animation, Particle systems and UI. He has a BS in the Department of Digital Contents from Sejong University.

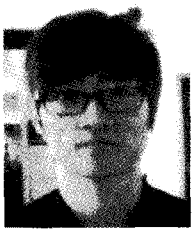


Jeong-Gyu Lee

is a engineer at Samsung Electronics. His research interests include Augmented Reality, HCI and Computer Graphics. He has a MS in the Department of Digital Contents from Sejong University.

Sejong Jeon

is a master's student in Digital Contents at Sejong University. His research interests include Physics-based animation and Fluids simulation. He has a BS in the Department of Digital Contents from Sejong University.



Taewan Jeong

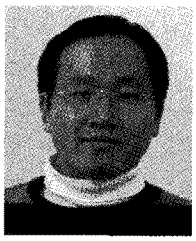
is a master's student in the Department of Digital Contents at Sejong University. His research interests include Computer graphics and 3D character animation. He has a BS in the Department of Digital Contents from Sejong University.

Dongwann Kang

is a MD-PhD combined course student in the Department of Computer Science and Engineering at Chung-Ang Univ. His research interests include Computer graphics, NPR and Computational aesthetics. He has a BS in Computer science from Chung-Ang University.

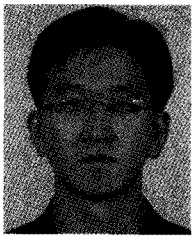
Sang Il Park

is a professor of the Department of Digital Contents at Sejong University. His research interests include computer graphics and 3D character animation. He has a PhD in Computer Science from Korea Advance Institute of Science and Technology.



Oh-young Song

is a professor in the Department of Digital Contents at Sejong University. His research interests include Physics-based animation and Character animation. He has a PhD in the School of Electrical Engineering and Computer Science from Seoul National University.



Jong Weon Lee

is a professor in the Department of Digital Contents at Sejong University. His research interests include Augmented Reality, HCI and Computer Vision. He has a PhD in Computer Science from University of Southern California.

Kyung-Hyun Yoon

is a professor in the Department of Computer Science and Engineering at Chung-Ang University. His research interests include Computer graphics, NPR and Computational aesthetics. He has a PhD in Computer science from Connecticut University.

Chang-wan Han

is a professor in the Department of Cartoon & Animation at Sejong University. His research interests include Animation, Game, Comics, Character Industry Analysis. He has a PhD in Media Economics from Sejong Universtiy.

Sung Wook Baik

is a professor in the Department of Digital Contents at Sejong University. His research interests include Computer vision, Pattern recognition, Computer game and AI. He has a PhD in Information Technology and Engineering from George Mason University.