Haptic Interaction with Depth Video Media

Jongeun Cha[1], Seung-man Kim[2], Ian Oakley[1] Jeha Ryu[1],, and Kwan H. Lee[2]

[1] Human-Machine-Computer Interface Lab., Dept. of Mechatronics, Gwangju Institute of Science and Technology, 1 Oryong-dong, Buk-gu, Gwangju 500-712 Republic of Korea, {gaecha, ian, ryu}@gist.ac.kr, WWW home page: http://dyconlab.gist.ac.kr

[2] Intelligent Design & Graphics Lab., Dept. of Mechatronics, {sman, lee}@kyebek.gist.ac.kr, WWW home page: http://kyebek9.gist.ac.kr

Abstract. In this paper we propose a touch enabled video player system. A conventional video player only allows viewers to passively experience visual and audio media. In virtual environment, touch or haptic interaction has been shown to convey a powerful illusion of the tangible nature - the reality - of the displayed environments and we feel the same benefits may be conferred to a broadcast, viewing domain. To this end, this paper describes a system that uses a video representation based on depth images to add a haptic component to an audio-visual stream. We generate this stream through the combination of a regular RGB image and a synchronized depth image composed of per-pixel depth-from-camera information. The depth video, a unified stream of the color and depth images, can be synthesized from a computer graphics animation by rendering with commercial packages or captured from a real environment by using a active depth camera such as the Zcam^{TM}. In order to provide a haptic representation of this data, we propose a modified proxy graph algorithm for depth video streams. The modified proxy graph algorithm can (i) detect collisions between a moving virtual proxy and time-varying video scenes, (ii) generates smooth touch sensation by handling the implications of the radically different display update rates required by visual (30Hz) and haptic systems (in the order of 1000Hz), (iii) avoid sudden change of contact forces. A sample experiment shows the effectiveness of the proposed system.

1 Introduction

The rapid development of telecommunication technologies such as enhanced CPU speed and power, low cost memory, and ultra fast communication networks has led to the digital multimedia age, where viewers can enjoy and be immersed in high quality video and audio media. Typically, some interaction such as selecting interactive icons with a wireless keyboard or a remote control is also allowed. That interaction is generally a menu-based dialogue where

the viewer selects an option from several ones provided on screen. However, recently viewers' demands for a more realistic and interactive experience with video media are growing and there is interest in exploring the possibilities of an interactive experience with the video media beyond passive watching and listening, i.e. touching and manipulating the video media content directly. Such an interaction style, known as haptic interaction, has the potential to create a truly immersive experience for the viewers by providing a powerful illusion of the tangible nature of the displayed environments.

Traditionally, haptic interaction has been integrated with fully synthesized virtual reality (VR) worlds where several users can share the virtual contents simultaneously. Recently, O'Modhrain and Oakley [1] discussed the potential role that haptic feedback might play in supporting a greater sense of immersion in broadcast content. In addition, they explored two potential program scenarios: the creation of authored haptic effects for children's cartoon and the automatic capture of impact data to be streamed and displayed in the context of a live sport broadcast. However, since the media is based on 2-dimensional information, the possible haptic interaction is limited to 2-dimensions. If the media includes 3-dimensional information, more useful interactions may become possible. For example, it will become possible to touch and explore an object-of-interest such as a peculiar shaped object or the face of a famous actor's face [2]. When an actor is touching his lover on the face in a scene, viewers may want to touch her also to increase immersion in the scene and feel as if they had become the actor.

Typically, in order to represent a 3-dimensional scene, polygonal 3D meshes are used. But, they are not appropriate for 3D video media because of the redundancy of connectivity information, complex level-of-detail, compression, and progressive transmission [3]. To bridge the gap between simple 2D texture mapping and full 3D modeling of a video object, a depth image based representation was proposed [4]. In this system, the 3D video media is a combination of regular video (general RGB image) and synchronized per-pixel depth information (depth image), see Fig. 1. Currently, the European IST project ATTEST [5] has created a novel 3D-TV system by means of a depth image-based representation. They created 3D content by capturing real dynamic environments with an active range camera, and compressed and transmitted them with MPEG codecs.

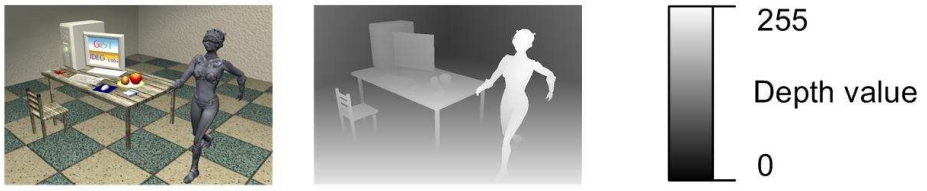

Fig. 1. General RGB immage and synchronized per-pixel depth information

In the 3D-TV system, the viewers could see the 3-dimensional real-world scene stereoscopically.

In order to provide an interaction force, haptic rendering, the process to display the haptic attributes of surface and material properties of virtual objects in real time via a haptic interface, need to be employed. Haptic rendering, however, requires significantly higher update rates (on the order of 1 kHz) than graphic rendering for smooth and stable force calculation. The main bottleneck of the haptic rendering process is the collision detection between the virtual scene and the current haptic interaction point. While a pre-computed "hierarchical bounding boxes" approach has been employed to achieve fast collision detection in the majority of previous works, e.g. [6–9], they are not applicable in the case of depth video that requires the construction of a new bounding box for each frame. Also, an extremely large model, such as depth image, slows down the collision detection considerably (697,430 triangles in 720486 resolution depth image). Walker and Salisbury [10] proposed a proxy graph algorithm for extremely large static topographic maps of over 100 million triangles, which is essentially the same as a single frame depth image. With this technique, they not only reduced the overall number of collision detection operations, but also optimized the collision detection by capitalizing on the special structure of the vertically monotone height field.

In this paper we propose a modified proxy graph algorithm for depth video streams, because the proxy graph algorithm exhibits some problems when applied to depth video streams. First, it fails to detect some collisions between a virtual haptic interaction point and depth video changing its shape frame by frame. Secondly, it produces piecewise-continuous contact forces because of significantly different video and haptic update rates. Finally, it generates sudden large changes in force at scene transitions making the haptic interface unstable.

2 Modified Proxy Graph Algorithm for Depth Video Streams

In order to explain the proposed idea, this section provides a brief review of proxy-based haptic rendering algorithms with detailed summary of the proxy graph algorithm [10]. Then, the problems and solutions involved in applying this algorithm (i.e. the proposed modified proxy graph algorithm) to a stream of depth images are explained in detail.

2.1 Overview of Proxy Graph Algorithm

Haptic rendering algorithms are essentially composed of the coupled processes of collision detection and response between a haptic interaction point (HIP) and a virtual scene. Their objective is to model the forces from this interaction to enable a user to touch, feel and manipulate virtual objects. Typically, the user controls the position of a point in the virtual world, and a force vector is generated based on the distance between this point and what is termed a proxy

point which is constrained to remain on the surface of objects in the scene. This can be easily imagined in the case of a user exploring a virtual plane. Initially, when the user is not in contact with the plane, the proxy and the haptic interaction point are coincident, and no forces are applied. As the user moves onto, and then penetrates the plane, the proxy remains on its surface while the HIP penetrates into the surface, and the user feels forces proportional to the distance between these two points. A typical algorithm renders these forces using a linear spring model.

A key process in a proxy algorithm is the minimisation of the distance between the HIP and the proxy point. As the HIP moves within an object, the proxy must be moved to an appropriate location on its surface in order to provide a realistic force. In the example of the plane mentioned above, this is trivial, but is more complex when considering realistic virtual objects such as polygonal meshes. A typical solution is to generate a list of constraint planes. In each update, this list is initially empty, and the proxy is moved towards the HIP until it is coincident or until collision with a polygon prevents further motion. In the case of collision, the polygon is added to the list of constraints, and a new goal location reflecting this constraint becomes the proxy's optimal destination. It can be shown that in a polygonal mesh, three such iterations will restrict the proxy's motion to a single point, a constrained local minima. Proxy algorithms based on these concepts are commonplace, as they are simple to implement, robust and reliable [9, 12, 13].

However, a weakness of this constraint resolution process is that it restricts the number of polygons a user can traverse in a given update cycle to one. This is essentially due to the fact that constraints are not released when a user moves to an adjacent polygon. Given the 1000 Hz refresh rates required for smooth haptic display, this limitation only becomes evident when considering very large or dense meshes, in which a user can move across multiple triangles in a single millisecond. In these situations, the constraint model yields an undesirable feeling of viscosity, due to the proxy point lagging behind the user's actual position. While this can be remedied by simply modifying the basic algorithm to release constraints when new polygons are encountered, this entails considerable additional processing, and is currently an unrealistic solution. Walker and Salisbury [10] describe the proxy graph algorithm, a alternative constraint resolution method to address this issue. Essentially, they restrict proxy motion to the vertices of the mesh, and choose among vertices simply by selecting the gradient that slopes towards the target most rapidly. Although, this approach can result in erroneous proxy positioning, these errors are usually minor, and allow a large number of polygons to be traversed efficiently within a single update.

Depth images are simply height maps. They consist of an evenly spaced two dimensional array of elevations. A triangle based surface can be derived from them by the simple precedent of adding horizontal and diagonal lines between each element. They are also vertically monotone, with each horizontal point possessing only a single point above or below it. In the context of their proxy graph algorithm, Walker and Salisbury [10] discuss several optimisations to col-

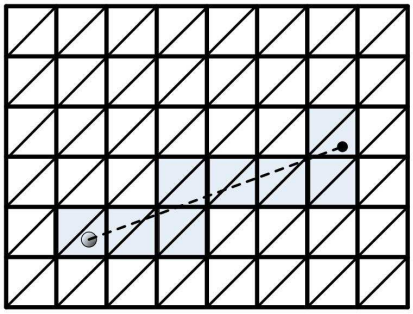

Fig. 2. Local triangulation for collision detection

lision detection algorithms that can be used with this type of data structure. Firstly, the HIP path can be projected onto the two-dimensional representation of the height map in order to generate a list of candidate polygons which should be checked for collisions (Fig. 2). Secondly, cells within this candidate list that possesses elevation values below that of the HIP path can simply be discarded. These optimisations, in conjunction with those related to constraint resolution described above, yield an algorithm which executes extremely rapidly when applied to large data-sets.

2.2 Collision Detection Correction with Depth Video Streams

The proxy graph algorithm was developed for haptic interaction with the large static topographic maps. In contrast, the surface triangulated from the depth video is dynamic - it changes its shape frame by frame. Consequently, when the current surface transforms into the next surface, the collision detection fails in some cases and the haptic device can go through the surface as shown in Fig. 3(a, b).

These kinds of problems occur in haptic interaction with a moving rigid-body object, which can pass through the previous proxy. Ruspini [14] corrected this problem by moving the previous proxy position with respect to the movement of the object in the same local frame of the object as shown in Fig. 3(c). However, the same method cannot be applied to the depth video streams because the surface moves regardless of the local coordinates. To remedy this situation, the surface depth is stored by projecting the proxy position on to the surface. When the surface moves upward, the proxy is also moved by that amount, as shown in Fig.4. This ensures that the surface cannot pass through the proxy.

2.3 Local Surface Interpolation

Humans can sense force vibrations well in excess of 300Hz. For smooth and stable force display, generally, a high haptic update rate (>1000Hz) is required.

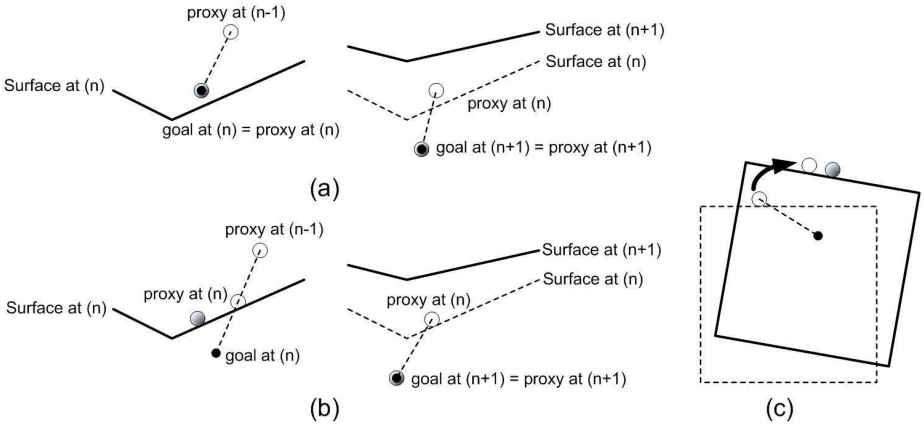

Fig. 3. Illustration of the collision detection failure of the proxy graph algorithm with depth video: The general proxy algorithm checks the collision between the surface and the line segment connecting the current goal and the previous proxy. (a) When the surface goes up with the device position in the vicinity of the surface, the surface can pass through the line segment in frame (n+1). (b) Even though the first collision is detected and the proxy position is determined at frame (n), the algorithm regards proxy is inside of the frame (n+1) when the surface goes up. (c) The previous proxy contacting on the rigid-body surface is moving in the same local coordinate and maintains the same contact point.

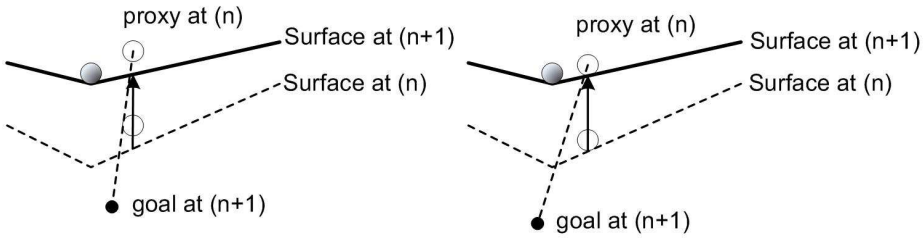

Fig. 4. Correction of the previous proxy position out of or on the surface by the amount of the surface movement.

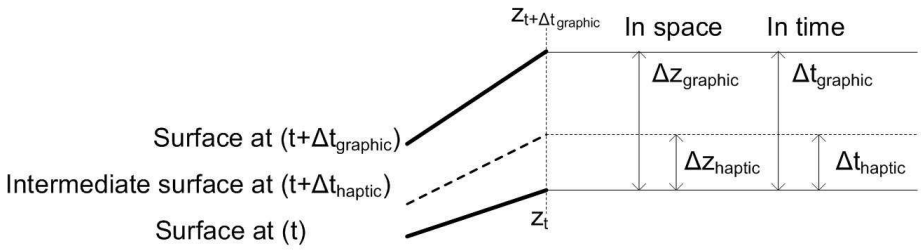

Fig. 5. Intermediate surface interpolated between the current surface and the next buffered surface.

The depth video refresh rate (30Hz) is much slower than the haptic update rate. This performance gap may cause the contact force to be piecewise continuous, that is to say, a viewer will perceive the discrete change of the depth video surface. Ruspini [14] proposed a proxy blending method in haptic interaction with deformable objects, which interpolates an intermediate proxy between the old proxy constrained to the old surface and the current proxy constrained to the current surface. This method requires additional time to calculate two proxy positions during the blending period and resultant force is delayed by about one frame.

In this paper as shown in Fig. 5, we interpolate the local intermediate surface needed to be used in collision detection between the current and the next depth video local surface using Eq.(1), since the next depth image can be obtained at the current time by buffering.

$$Z_{intermediate} = Z_t + \Delta Z_{graphic} \frac{\Delta t_{haptic}}{\Delta t_{graphic}} = Z_t + (Z_{t+\Delta t_{graphic}} - Z_t) \frac{\Delta t_{haptic}}{\Delta t_{graphic}} \quad (1)$$

where, Z_t and $Z_{t+\Delta t_{graphic}}$ are the depth values of the surface at time (t) and $(t + \Delta t_{graphic})$, respectively, $\Delta t_{graphic}$ is the graphic update time (about 30ms), and Δt_{haptic} is the elapsed time from time (t) to current haptic update time. Δt_{haptic} can be obtained by counting the clocks from (t). Since, our system is running on the MS Windows which is not a real time OS, $\Delta t_{graphic}$ is not an exact constant. However, subjective evaluation by users confirms that this interpolation process is sufficient to give the apparent continuous force.

2.4 Sudden Change of the Proxy Position

During the stream of the depth video, there can be dramatic changes in depth values between individual frames when a scene changes or when objects in the scene move as shown in Fig. 6(a, b). If this occurs, the proxy may move with significant force very rapidly, potentially damaging the device and injuring the user. Therefore, this situation should be avoided to provide stable interaction to

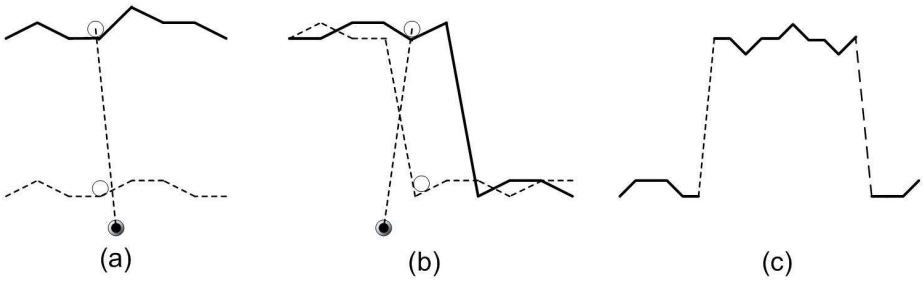

Fig. 6. Sudden change of the proxy position: (a) While the scene changes to another, the previous proxy on the old surface abruptly goes up on the higher new surface. (b) The proxy around the object edge goes up to another higher moving object. (c) The viewer is prevented from touching the object's edge surface that comes from an occlusion.

the viewer. In order to solve this problem, the penetration depth is constantly monitored and the variation of that value is checked. If a certain threshold is exceeded, we make the proxy collocated with the device position to make the force zero for safety. In other words, when the force changes abruptly, the proxy is allowed to pass through the surface. Consequently, at the moment of scene change the viewer will not experience a dangerous force increase. Additionally, the viewer is prevented from touching an object's edge surface (the dotted surface in Fig. 6(c)) that comes from an occlusion at the camera view.

3 Example and Results

A sample experiment had been performed to verify the effectiveness of the proposed algorithm as shown in Fig. 7. The depth video can be easily synthesized

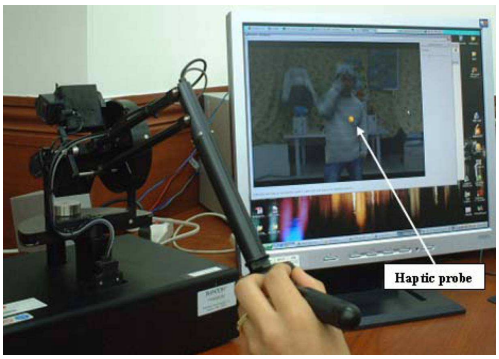

Fig. 7. Haptically enhanced depth video player

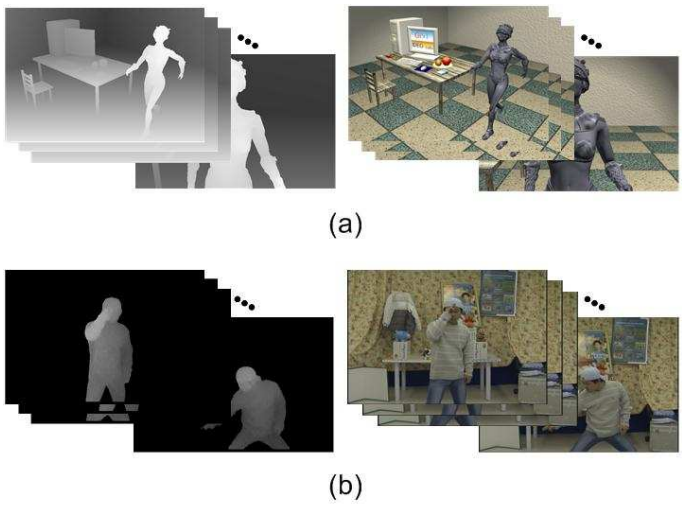

Fig. 8. Depth videos synthesized from computer graphics animation and (b) captured and smoothed from Zcam^{TM}.

by rendering a computer graphics animation in off-line rendering packages (e.g. Discreet 3DS MAX or Alias Maya), where the depth image can be saved by reading Z-buffer as shown in Fig. 8(a). In recent years, with the technological advancement of active depth sensors such as the ZCam^{TM} depth camera [11], depth video can be directly captured in real time as shown in Fig. 8(b). However, depth video usually contains quantization errors and optical noises, mainly due to the reflectivity or color variation of the objects being filmed. When the raw depth map is applied to haptic interactions, the high frequency geometric errors produce distort feeling such as a jagged texture and a tremor.

To enhance depth map, we applied a median-filtering technique to reduce noise. Then, we adaptively sampled feature points using 1st-gradient analysis, since depth variation greatly affects the quality of a reconstructed surface. The Delaunay triangulation technique situated the feature points in 2D space. Although a 3D surface is reconstructed by projecting a 2D triangular mesh into a 3D space based on the filtered depth value, it still contains local noise that produce jagged surfaces. For that reason we applied Gaussian smoothing to the 3D mesh to enhance the smoothness of the surface. Finally we rendered the 3D surface with a commercial animation package (Maya) to generate a smooth depth map from a reconstructed 3D surface. Fig. 9 shows the zoomed part of a raw, median-filtered and Gaussian smoothed set of depth images. We experimented with SD (standard definition, 720x486 pixels) depth videos. In order to display these two depth videos with OpenGL, we assigned column index, row index and depth value to x, y and z positions, respectively. Also, each point was set as the color value corresponding to the captured RGB image and then all points are tri-

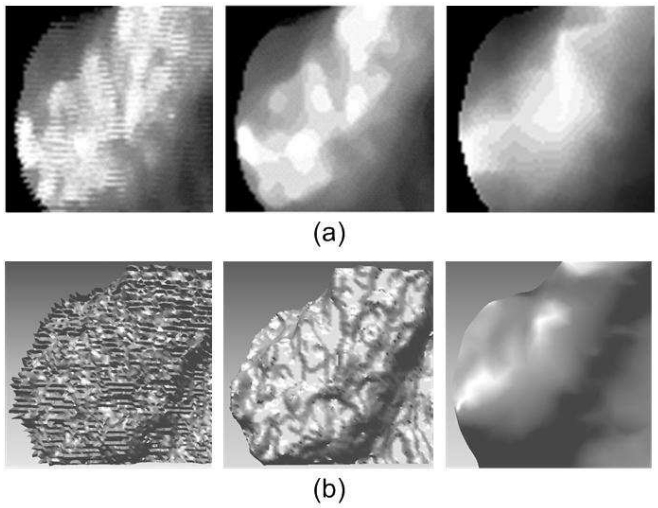

Fig. 9. Depth video enhancement captured from Zcam^{TM}: (a) The raw, median-filtered and Gaussian smoothed depth images. (b) Corresponding mesh models to the depth images.

angulated by adding a diagonal edge. However, for faster graphic rendering, we reduced the resolution to 360x243 pixels. We did not, however, reduce the resolution in the haptic rendering. Moreover, for more immersion in the depth video, viewers wore CrystalEye shutter-glasses which provide a stereoscopic view.

This application was implemented to run on an Intel based PC (Dual 3.0Ghz Pentium IV Xeon, 1GB DDRRAM, nVidia QuadroFX 1300 PCI-Express) under Microsoft Windows XP using PHANToM premium 1.5/6 DOF made by SensAble Technologies. PHANToM haptic interfaces provide high-performance 3D positioning and force feedback plus a 3 degree-of-freedom orientation sensing gimbal. The haptic rendering algorithm was implemented using PHANToM Device Drivers Version 4.0 and HDAPI. The HDAPI is a low-level foundational layer for haptics and provides the functions to acquire 3D positions and set the 3D forces at a near realtime 1Khz servo rate.

Table 1 shows the results of the depth video haptic rendering performance tests. The haptic computation time for each haptic update was measured using a high-resolution timer provided in Windows. The minimum computation time is obtained with the haptic interface away from the surface and the maximum

Table 1. Haptic computation time during each update.

	Without contact	With contact
Average time(milliseconds)	0.017	0.070

one is measured by moving the haptic interface rapidly across the surface. The proposed modified proxy graph algorithm operates comfortably within the 1kHz update rate.

This example verified that we were able to stably touch and acquire the shape of the scene. Also, we could interact with a moving object and feel the smooth movement. Even if the scene was changed to another and a closer object showed up abruptly, we could penetrate the surface without feeling sudden force increase.

4 Conclusion

The depth video based haptic rendering algorithm was successfully implemented to allow the viewers to directly touch the 3-dimensional scene captured in a depth video. This is an initial stage to provide the viewers with more realistic and interactive experience by touch. At the present time, this algorithm offers the ability to touch a scene in video media and an application scenario is to touch unusual or interesting on-screen objects or an actor's face to acquire the shape or feeling of the skin. We believe that the synergy of the viewers' demand, the producer's creativity, and this kind of technical capability will make touch enabled media rich and abundant.

Acknowledgements

This work was supported by the Ministry of Information and Communication (MIC) through the Realistic Broadcasting IT Research Center (RBRC) at Gwangju Institute of Science and Technology (GIST).

References

1. O'Modhrain, S., Oakley, I.: Touch TV: Adding Feeling to Broadcast Media, Proc. European Conf. Interactive Television: from Viewers to Actors, Brighton, UK (2003) 41-47
2. Cha, J., Ryu, J., Kim, S., Eom, S., Ahn, B.: Haptic Interaction in Realistic Multimedia Broadcasting, Proc. 5th Pacific-Rim Conf. Multimedia on Advances in Multimedia Information Processing, Part III, Nov./Dec. (2004) 482-490
3. Ignatenko, A., Konushin, A.: A Framework for Depth Image-Based Modeling and Rendering, Proc. Graphicon-2003, Sep. (2003) 169-172
4. Kauff, P., Cooke, E., Fehn, C., Schreer, O.: Advanced Incomplete 3D Representation of Video Objects Using Trilinear Warping for Novel View Synthesis, Proc. PCS'01 (2001) 429-432
5. Redert, A., Op de Beeck, M., Fehn, C., IJsselsteijn, W., Pollefeys, M., Van Gool, L., Ofek, E., Sexton, I., Surman, P.: ATTEST - Advanced Three-Dimensional Television System Technologies, Proc. 1st Int. Symp. 3D Data Processing, Visualization and Transmission, Padova, Italy (2002) 313-319

6. Salisbury, K., Brock, D., Massie, T., Swarup, N., Zilles, C.: Haptic rendering: Programming touch interaction with virtual objects, Proc. 1995 ACM Symp. Interactive 3D Graphics (1995) 123-130
7. Gottschalk, S., Lin, M., Manocha, D.: OBB-Tree: A hierarchical structure for rapid inter-ference detection, Proc. ACM SIGGRAPH 1996 (1996)
8. Inc. SensAble Technologies: GHOST^{TM}: Software developer's toolkit, Programmer's Guide (1997)
9. Ruspini, D., Kolarov, K., Khatib, O.: The haptic display of complex graphical environments, Proc. ACM SIGGRAPH 1997 (1997) 345-352
10. Walker, S., Salisbury, K.: Large Haptic Topographic Maps: MarsView and the Proxy Graph Algorithm, Proc. ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics (2003) 83-92
11. http://www.3dvsystems.com/
12. Zilles, C., Salisbury, K.: A Constraint Based God-Object Method For Haptic Display, Proc. IEE/RSJ Int. Conf. on Intelligent Robots and Systems, Human Robot Interaction, and Cooperative Robots, Vol 3 (1995) 146-151
13. Ho, C., Basdogan, C., Srinivasan, M.: Efficient point-based rendering techniques for haptic display of virtual objects, Presence: Teleoperators and Virtual Environments, Vol.8, no. 5 (1999) 477-491
14. Ruspini, D., Khatib, O.: Dynamic Models for Haptic Rendering Systems, Proc. Advances in Robot Kinematics: ARK'98, Strobl/Salzburg, Austria (1998) 523-532