Touchable 3D Video System

JONGEUN CHA, MOHAMAD EID and ABDULMOTALEB EL SADDIK
Multimedia Communications Research Laboratory (MCRLab)
University of Ottawa

Multimedia technologies are reaching the limits of providing audio-visual media that viewers consume passively. An important factor, that will ultimately enhance the user's experience in terms of impressiveness and immersion, is interaction. Among daily life interactions, haptic interaction plays a prominent role in enhancing the quality of experience of users, and in promoting physical and emotional development. Therefore, a critical step in multimedia research is expected to bring the sense of touch, or haptics, into multimedia systems and applications. This paper proposes a touchable 3D video system where viewers can actively touch a video scene through a force-feedback device, and presents the underlying technologies in three functional components: (1) contents generation, (2) contents transmission, and (3) viewing and interaction. First of all, we introduce a depth image-based haptic representation (DIBHR) method that adds haptic and heightmap images, in addition to the traditional depth image-based representation (DIBR), to encode the haptic surface properties of the video media. In this representation, the haptic image contains the stiffness, static friction, and dynamic friction whereas the heightmap image contains roughness of the video contents. Based on this representation method, we discuss how to generate the synthetic and natural (real) video media through a 3D modeling tool and a depth camera, respectively. Next, we introduce a transmission mechanism based on the MPEG-4 framework where new MPEG-4 BIFS nodes are designed to describe the haptic scene. Finally, a haptic rendering algorithm to compute the interaction force between the scene and the viewer is described. As a result, the performance of the haptic rendering algorithm is evaluated in terms of computational time and smooth contact force. It operates marginally within 1 kHz update rate that is required to provide stable interaction force and provide smoother contact force with the depth image that has high frequency geometrical noise using a median filter.

Categories and Subject Descriptors: H.5.1 [**Multimedia Information Systems**]: Video; I.4.10 [**Image Representation**]: Multidimensional; H.5.2 [**User Interfaces**]: Haptic I/O

General Terms: Design, Algorithms

Additional Key Words and Phrases: Haptic surface properties, haptic rendering algorithm, video representation

1. INTRODUCTION

Recent advances in multimedia contents generation and distribution have led to the creation and widespread deployment of more realistic and immersive display technologies. A central theme of these advances is the eagerness of consumers to

Author's address: Multimedia Communications Research Lab., School of Information Technology and Engineering, University of Ottawa, 800 King Edward, Ottawa, CA, K1N 6N5

Start of a second footnote ...

experience engrossing contents capable of blurring the boundaries between the synthetic contents and reality; they actively seek an engaging feeling of “being there”, usually referred to as presence [Riva et al. 2003]. For instance, wide-screen displays adopting High-Definition (HD) video offer a wide field of view that prevents the viewers from being disturbed by the real environment. Furthermore, three-dimensional television (3D-TV) supports a natural viewing experience such that viewers are able to perceive objects in true dimensions and in natural colors [Matusik and Pfister 2004]. Additionally, 3D sound systems provide directional audio, further helping to increase the sense of presence in the experienced scene.

One important but largely overlooked aspect of presence in the domain of multimedia is interaction. When viewers have the ability to interact naturally with an environment, or are able to affect and be affected by environmental stimuli, they tend to become more immersed and engaged in that environment [Witmer and Singer 1998]. One multimedia approach that provides rich interaction is virtual environments where users, represented as avatars (3D virtual embodiments), are able to interact with computer-generated objects and other avatars. The virtual environment contents can be represented using VRML[1], X3D[2] or MPEG-4 BIFS[3] and delivered and consumed over the Internet. However, in order to produce photo-realistic and immersive contents, the data volume becomes considerably large which makes it hard to deliver and consume due to network resource limitations. That is why video media is widely used in television, movie, and user-created contents such as YouTube[4].

A number of interactive technologies have been applied to multimedia delivery scenarios of video media. For example, in interactive television, viewers receive on-demand information, interactive feedback and online transaction opportunities, which provide them with a more intimate relationship with advertisers, networks and their favorite programs [Bukowska 2001; Chorianopoulos and Lekakos 2007]. However, these services are abstract, information-orientated, and context-dependent and are unlikely to raise the level of presence. This paper argues that presence can only be boosted by more engaging, direct interaction paradigms. Recently, to overcome this deficiency, some researchers have tried to supply direct interaction to explore and navigate audio-visual scenes by freely choosing the viewpoint and viewing direction [Smolic and Kauff 2005].

Several psychological studies have confirmed that the haptic modality can increase the sense of presence and co-presence in virtual reality simulations [Hale and Stanney 2004][Sallnas et al. 2000]. It has been argued that as the human haptic system uniquely encompasses both perception and action, touch interaction has the potential to create a truly immersive experience [Reiner 2004]. However, the domain of multimedia has received scant attention in haptics literature. In addition, bringing the sense of touch into multimedia applications is considered the next

[1] ISO/IEC 14772-1:1997 and ISO/IEC 14772-2:2004 — Virtual Reality Modeling Language (VRML)

[2] ISO/IEC 19775:2004, Extensible 3D (X3D)

[3] ISO/IEC 14496-11, Coding of audio-visual objects – Part 11: Scene description and Application engine (BIFS, XMT, MPEG-J)

[4] www.youtube.com

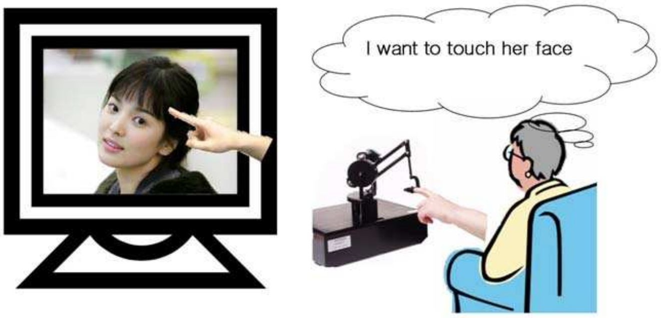


Fig. 1. Active haptic interaction scenario in a multimedia application

critical step in the development of multimedia systems due to their maturity with what can be done with only sight and sound [El Saddik 2007]. Correspondingly, this paper focuses on the incorporation of haptic interaction in multimedia systems.

There have been some endeavors to provide haptic feelings to viewers in movie theaters, such as *Percepto* that employed vibrating devices attached to theater seats in the 1960s [Riva et al. 2003]. Although these systems do not provide various sophisticated feelings synchronized with a viewing scene, a simple vibration cue can help viewers to become more immersed in multimedia contents. In recent years, some researchers, including the authors of this paper, started to investigate the feasibility of applying haptics into multimedia and to propose various haptic interaction scenarios into multimedia applications [Eid et al. 2007][Magnenat-Thalmann and Bonanni 2006][Cha et al. 2004].

To this end, O'Modhrain and Oakley explored how physical interaction, and in particular haptic interaction, might enhance and enrich the experience of broadcast content [O'Modhrain and Oakley 2003]. The authors proposed a scenario for measuring and transmitting accelerations of a vehicle in racing scenarios in order for viewers to feel what the driver feels. Some researchers have been seeking methods of augmenting haptic data related to the motion of objects in an existing video. For example, Gaw et al. proposed a system for recording and annotating haptic information that is time-referenced to a movie, then replaying the recorded haptic information to a user [Gaw et al. 2006]. A 3-DOF motion is manually annotated to a video so that a viewer could trace the motion. Additionally, Yamaguchi et al. proposed a system that generates haptic effects automatically from 2D graphics by relying on metadata that describes the movement characteristics of the contents [Yamaguchi et al. 2006]. Here, viewers can feel the motion of an object through a 2-DOF force-feedback device. On the other hand, Cha et al. provided a tactile stimulus that is synchronized with a video scene. They defined a tactile video to store and transmit the actuation data of a glove-type vibrotactile device [Cha et al. 2007]. However, these haptic interactions are passive so viewers do not actively participate in the scene.

One of the most important haptic interactions in life is to extend hands and arms to actively touch and explore an object [Gibson 1962]. By doing so, we can perceive the shape of the object and examine its surface properties. However, in order

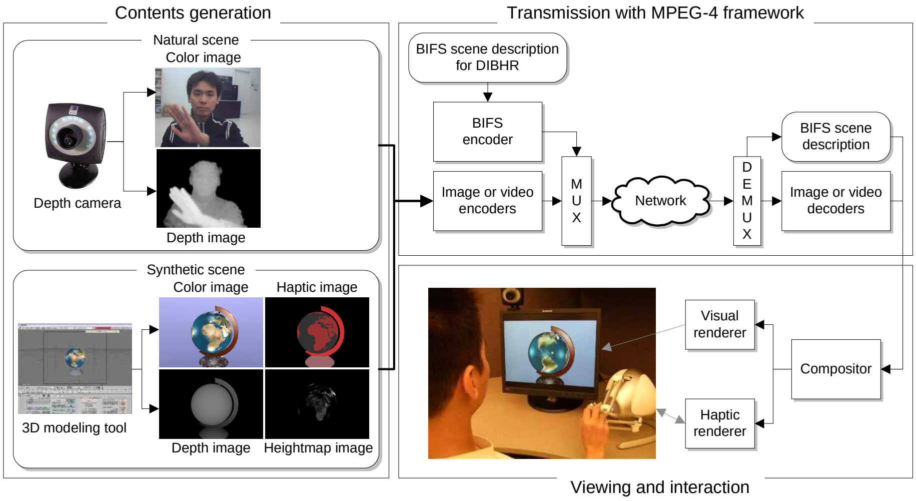


Fig. 2. Block diagram of touchable 3D video system

to touch and explore delivered media in an interaction mode, media data should include 3D geometry information. When the media includes 3D information, more useful interactions become possible. For example, it is possible to touch and explore an object-of-interest such as a peculiar-shaped object, or the face of a famous actor. In the scenario where an actor is touching his lover's face in a scene, viewers may also want to touch her to increase their immersion in the scene and feel as if they have become the actor, as illustrated in Fig. 1. This active interaction scenario with video media was first proposed and implemented by the author in [Cha et al. 2004] and [Cha et al. 2006]. However, in the previous implementation, viewers could not experience various haptic properties such as friction and roughness because the depth image-based representation method contain only geometry but any haptic data and the proposed haptic rendering algorithm did not support friction and roughness rendering.

In this paper, we present a touchable 3D video system that enables users to physically explore the video content and feel various haptic properties as shown in Fig. 2. The contribution of this work is to present the underlying technologies in constructing the system spanning from the process of contents generation through contents transmission and finally to viewing and interaction. More specifically, first of all, we propose a depth image-based haptic representation (DIBHR) method that represents a touchable 3D video scene and describes how to generate the synthetic and natural (real) video media through a 3D modeling tool and a depth camera, respectively. Second, a communication mechanism is introduced using the MPEG-4 framework[5] for the contents delivery. Third, a haptic rendering algorithm to compute the interaction force between the viewer and the 3D video scene is presented. In DIBHR, haptic information is encoded as a haptic image and a heightmap image that contain the stiffness, static and dynamic friction, and roughness of a video

[5] ISO/IEC 14496-1:2001, Coding of audio-visual objects – Part 1: Systems

scene. The proposed representation is implemented based on the MPEG-4 framework that supports streaming data for various media objects so that incorporating haptic data becomes straightforward. In order to incorporate the touchable 3D video contents into the MPEG-4 System, new MPEG-4 BIFS nodes are designed based on DIBHR. The haptic rendering algorithm based on our previous work [Cha et al. 2008; Cha et al. 2006] is described by adding the mapping between the depth image coordinate and the 3D world coordinate and the median filter for smoothing the force rendering.

The remainder of this paper is organized as follows: section 2 proposes the depth image-based haptic representation by highlighting the previous representation methods for the haptic scene. Next, in section 3, the architecture of the 3D video system and its comprising components are proposed and implemented based on Fig. 2. Section 4 presents the performance evaluation of the haptic rendering algorithm. Finally, Section 5 summarizes this paper and provides future perspectives for haptic multimedia.

2. 3D VIDEO REPRESENTATION WITH HAPTIC INFORMATION

In haptic literatures, the 3D haptic scene is represented by classical 3D polygon-based or voxel-based modeling with haptic surface properties. However, in order to provide an efficient haptic exploration service, the creation of 3D scenes and object modeling are complex and time consuming, and become even more complex if a dynamically changing scene simulating real life is being created. Moreover, it is not appropriate for 3D video media because of the redundancy of connectivity information, the complex level-of-detail, compression, and progressive transmission [Ignatenko and Konushin 2003]. Therefore, another method is needed to represent the 3D video that encodes the photorealistic and dynamically changing scene. This section gives an overview of previous haptic scene representation method, introduces a depth image-based representation as an alternative and proposes a depth image-based haptic representation for touchable 3D video.

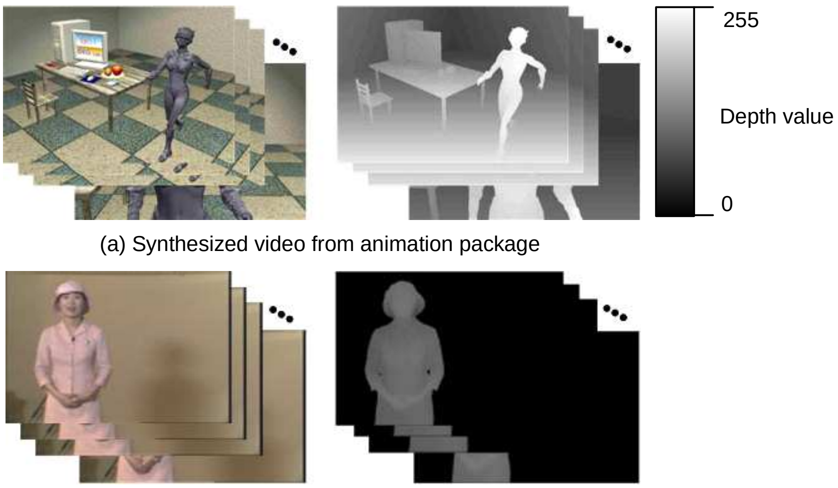
2.1 Related Work: Representation of the Haptic Scene

In order to enhance the realism of haptic feedback, the haptic surface properties such as stiffness, friction, and roughness should be incorporated in the geometry model. There are several representation models for haptic properties that can roughly be classified according to the surface representation they use: polygon-based [Zilles and Salisbury 1995; Ruspini et al. 1997; Ho et al. 1999], volume-based [Avila and Sobierajski 1996; McNeely et al. 1999; Ikits et al. 2003], implicit-based [Salisbury and Tarr 1997; Kim et al. 2004], and image-based [Cha et al. 2006]. In the following, we provide an overview of researches using each representation method.

The simplest model in haptic applications is the polygonal representation. The basic object used in polygonal modeling is the vertex, which is simply a point in the 3D space. Multiple vertices connected together in one plane form a polygon or a face. A group of polygons that are connected to each other by sharing vertices is generally referred to as an element or a mesh. Polygonal models are widely used due to their simplicity for calculating intersections and thus interaction force. Haptic properties such as stiffness and friction (static or dynamic) are represented as scalar values and imposed onto selected parts of the polygonal model. On the other hand, roughness is parameterized by a mathematical formula [SensAble Technologies, Inc. 2005] or a gray-scale heightmap image (or bumpmap) with texture coordinates [Ruspini et al. 1997; Ho et al. 1999; Reachin AB 2003; SenseGraphics AB 2006]. Despite the clear advantages, polygonal meshes are not well suited for the representation method of multimedia with photorealistic or animated contents due to intolerable resource consumption in terms of transmission and memory overload.

Polygon-based representation does not provide any information about the internal structure of the model. This knowledge is very important in medical simulations, for instance, when simulating an interactive cutting operation for a human organ. In such cases, volumetric representations (such as voxels) can be used. A voxels-based object is represented as a 3D rectilinear array of volume elements - voxels - each specifying a large number of physical properties such as density, stiffness, and viscosity. Although each voxel does not have surface property information, users can feel a gradient force that is obtained from the intensity values through mathematic functions [Salisbury and Tarr 1997; Avila and Sobierajski 1996]. In recent research, a proxy-based algorithm is introduced to enable the user to touch the surface of the voxel model [Ikits et al. 2003]. In the context of data representation in voxel-based models, the haptic information is contained in each voxel with the intensity values and a govern function. In some researches, each voxel has its own surface properties directly beside the intensity values [McNeely et al. 1999]. More details about volumetric representation can be found in the references [Kim et al. 2004; Lawrence et al. 2000]. Volumetric representation is challenged by significant degradation in haptic rendering accuracy and memory efficiency [Alatan et al. 2007].

Implicit representation uses geometric primitives (such as spheres, cones, cylinders, etc.) that are defined through mathematical expressions and wrapped around the geometrical models for force rendering [Kim et al. 2004], where the haptic properties are implicitly assigned to the implicit surfaces. Implicit representation provides several advantages [Alatan et al. 2007]. First, it enables faster and easier collision detection since a simple point inclusion function can be used to calculate collisions between objects and points in space. Second, the tangent to a surface can be easily calculated in order to display surface properties. Finally, several arithmetic operations (such as addition, subtraction, and concatenation) can be applied to make more complex objects. More details about implicit surface representations for haptic rendering can be found in [Salisbury and Tarr 1997]. With all these advantages, there is still the issue of finding which point on the surface should be used to model the interaction force in case of a collision. In real-time scenarios where quick-and-dirty rendering is required, representation methodologies such as the Non-Uniform Rational B-Spline (NURBS) and bezier patch have been widely used [Thompson et al. 1997]. NURBS surfaces, typically used in graphics of CAD environments, have the advantages of compactness, embedded smoothness, and exact computation of surface tangents and normals [Thompson and Cohen 1999]. The NURBS representation for haptic properties is the same as for implicit surfaces. Due to its efficiency of computation, these representation methods are widely used



(a) Synthesized video from animation package

(b) Captured video from depth camera

Fig. 3. Depth image-based representation with color and synchronized depth images

in deformable body simulation such as sculpturing and surgery training [Kim et al. 2004; Gao and Gibson 2005]. However, these methods have difficulty to describe sharp edges compared to the polygon-based method.

Implicit and NURBS representations are not widely used because of their limitation to describe complex objects. The most common representation methods are polygon-based for general purposes and voxel-based for specific medical simulations. However, these representation methods are not proper for multimedia systems, as we will see in the next section. Consequently, the depth image-based representation (DIBR) is an emerging methodology that has been originally proposed in [Ignatenko and Konushin 2003]. DIBR uses two images for each video frame: the RGB image and the depth image. Several advantages of using DIBR have been pinpointed in [Levkovich-Maslyuk et al. 2004]. First, existing methods of image processing and compression can also be applied to DIBR due to its simple and regular structure. Second, real world objects and scenes can be rendered without the need of millions of models (polygons) and expensive computations. Finally, the rendering time remains constant and independent from the complexity of the scene since it is proportional to the number of pixels of the captured images. Cha et al. adopted this representation to enable viewers to touch a 3D video scene [Cha et al. 2006]. However, since DIBR does not contain any haptic information, it could not provide a rich interaction force feeling. In addition, the haptic rendering algorithm did not support haptic texture rendering such as friction and roughness. In this paper, we adopt their approach to model a 3D scene and incorporate haptic surface properties into DIBR as well as introduce a haptic rendering algorithm that supports haptic texture rendering.

2.2 Depth Image-Based Representation (DIBR)

In order to bridge the gap between the simple conventional 2D rectangular video and full 3D modeling, a depth image-based representation was proposed [Kauff et al. 2001]. In this representation, 3D video media are the combination of general color images and synchronized gray-scale depth images containing per-pixel depth information, as shown in Fig. 3. The gray-level of each pixel in the depth image indicates the distance from a camera. The higher (whiter) the level is, the closer the distance to the camera. Since DIBR uses images for modeling a scene, a natural video, that captures a real moving scene as shown in Fig. 3(b), can be easily generated using stereo matching algorithms or a depth camera, such as ZCam$^{\text{TM}}$[6], while it is very hard to model a real moving scene with polygons or voxels.

DIBR is considered a 2.5D representation in the sense that the depth image has incomplete 3D geometrical information describing the scene from the camera view, and thus viewers can touch what they see. This means that the interaction capability is reduced compared to a full 3D scene and thus viewers cannot touch invisible parts of the scene. However, the purpose of this representation in the context of touch interaction is to enhance the immersion into the multimedia content where viewers are interested in touching what they see. In other words, the visible scene is intended, by a producer, for viewers to touch.

The depth image-based representation that enables haptic interaction with a still image and/or video contents was initially proposed in [Cha et al. 2006]. In order to calculate the contact force between the content and the navigator, the authors adopted the Proxy Graph Algorithm (PGA) [Walker and Salisbury 2003] for haptic rendering and modified it to overcome problems related to the direct application of PGA to a sequence of depth images. However, since the representation was limited to geometry (depth image) and visual appearance (color image) with no haptic information, they could not render rich haptic attributes of the scene, such as friction or roughness. Furthermore, their haptic rendering algorithm based on PGA did not support the haptic surface property rendering. In a proof-of-concept application, the viewers could only explore the shape of a scene. In contrast, we propose a depth image-based haptic representation method that contains haptic attributes of the scene as well as geometry and visual appearance in order to provide rich haptic interaction.

2.3 Depth Image-Based Haptic Representation (DIBHR)

In DIBR, a depth image is treated as a single object. In other words, although there exist many objects in a scene, the depth image captures and merges the objects into a single image object. Therefore, when haptic properties are assigned as a scalar value like in a polygonal model, just one haptic property can be assigned and thus applied to all the objects in the depth image. In order to avoid this deficiency, the haptic properties need to be assigned to each pixel in a similar way that the RGB color values are given. Consequently, in this paper, the haptic surface property information may be seen as a texture or a haptic image. Moreover, in the context of multimedia, images will be appropriate for compression and streaming

[6] http://www.3dvsystems.com

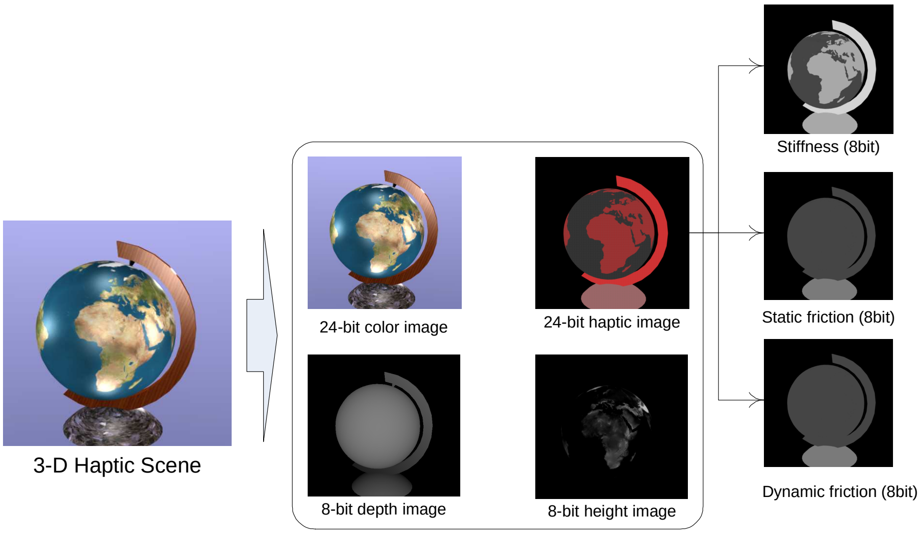


Fig. 4. Depth image-based haptic representation

because the transmission technology for images has matured enough. When the depth image is replaced with depth video to model a dynamically changing scene, the haptic image should be replaced correspondingly with a haptic video.

However, if the haptic image represents only one haptic property such as stiffness, friction, or roughness, many haptic images will be needed to represent one haptic scene. This in turn will significantly increase the media size to the extent that it becomes impractical for storage and transmission. Therefore, in this paper, we use three 8-bit channels to represent stiffness, static friction, and dynamic friction, respectively; in a similar way of using red, green and blue colors in a general RGB image. These three haptic properties are sufficient to express many key haptic experiences and are used in most haptic applications [SensAble Technologies, Inc. 2005; Reachin AB 2003; SenseGraphics AB 2006; Conti et al. 2005]. In addition, the roughness of the depth image surface is represented with a heightmap that is commonly used in computer graphics and many haptic property representations to express richer and more detailed surface geometry. The pixel values of the heightmap represent fine elevation changes of the surface relative to the original surface that is represented by the depth image. Although the depth image and the heightmap image have a same format of a gray image and stand for a distance toward a camera, range of the depth image is from a far plane to a near plane of the camera and that of the heightmap image is from the depth image surface to a predefined maximum height value. These ranges are stored as metadata with the images. In other words, the macro geometry shape is represented by the depth image whereas the micro geometry roughness is represented by the heightmap image. By having two different images with two different ranges for geometry representation, we can save the number of bits while keeping fine roughness representation.

Therefore, each frame of a scene is represented using four images: a 24-bit color image, an 8-bit depth image, a 24-bit haptic image, and an 8-bit heightmap image. We refer to this representation method as *Depth Image-Based Haptic Representation (DIBHR)*, as shown in Fig. 4, and is considered complimentary to our previous work in image-based haptic rendering presented in [Cha et al. 2008]. The 24-bit color image contains the three color components (red, green, and blue). The 8-bit depth image describes the global depth variations of various objects in the video contents (based on the camera parameters). The haptic image has three 8-bit channels for stiffness, static friction and dynamic friction, respectively. Finally, the 8-bit heightmap image contains the fine height details for the object surface. By bearing haptic information as images, we maintain compatibility with traditional image representations for video and take advantage of well developed image compression methods. However, the haptic property values are usually expressed with a floating-point variable to express a wide range of surface feelings. With an 8-bit channel, we can only express 256 levels of property values. In order to overcome this defect, we set a meta-data that contains two floating-point scalar values that represent minimum and maximum haptic property values for each channel. Therefore, when a pixel has an intensity value from 0 to 255, p, in one channel, the resultant property value, P, will be set following Equation 1.

$$P = MIN_{property} + (MAX_{property} - MIN_{property})\,\frac{p}{255} \tag{1}$$

where, $MIN_{property}$ and $MAX_{property}$ are the minimum and maximum values of haptic properties for each channel of stiffness and static/dynamic frictions. For example, if the stiffness channel has 0.01 N/mm and 1.0 N/mm as a meta-data, the pixel value of 70 indicates $(0.01 + (1.0 - 0.01) \times 70/255 = 0.28176)$ N/mm. As for the heightmap image, since the minimum value over zero means the height elevation of a whole depth image, only the maximum value is set. In addition, in order to reconstruct a 3D scene from the depth image in the world coordinate where haptic interaction is occurring, intrinsic camera parameters need to be considered such as the focal length of a camera that captures the depth image, physical pixel width, and the near and far depth clipping planes. These parameters are stored as meta-data as well.

3. TOUCHABLE 3D VIDEO SYSTEM

Fig. 2 shows the functional components of the touchable 3D video system. The system comprises three main components: the contents generation component, the transmission component, and the viewing and interaction component. The contents generation component has two different types of contents: synthetic and natural. The synthetic images are created by exporting the synthetic 3D scene as images using different rendering pipelines in 3D modeling tools. The natural images of real scene are captured by a 2.5D depth camera that generates depth images in addition to RGB images. The generated contents are fed to the transmission component that takes care of communicating the multimedia contents to the other side of the network. This is realized using the MPEG-4 framework and through the definition of BIFS haptic nodes that will include the haptic properties attached to the contents. Finally, in the viewing and interaction component, the video player receives and renders the contents visually and haptically to the viewer. The viewers

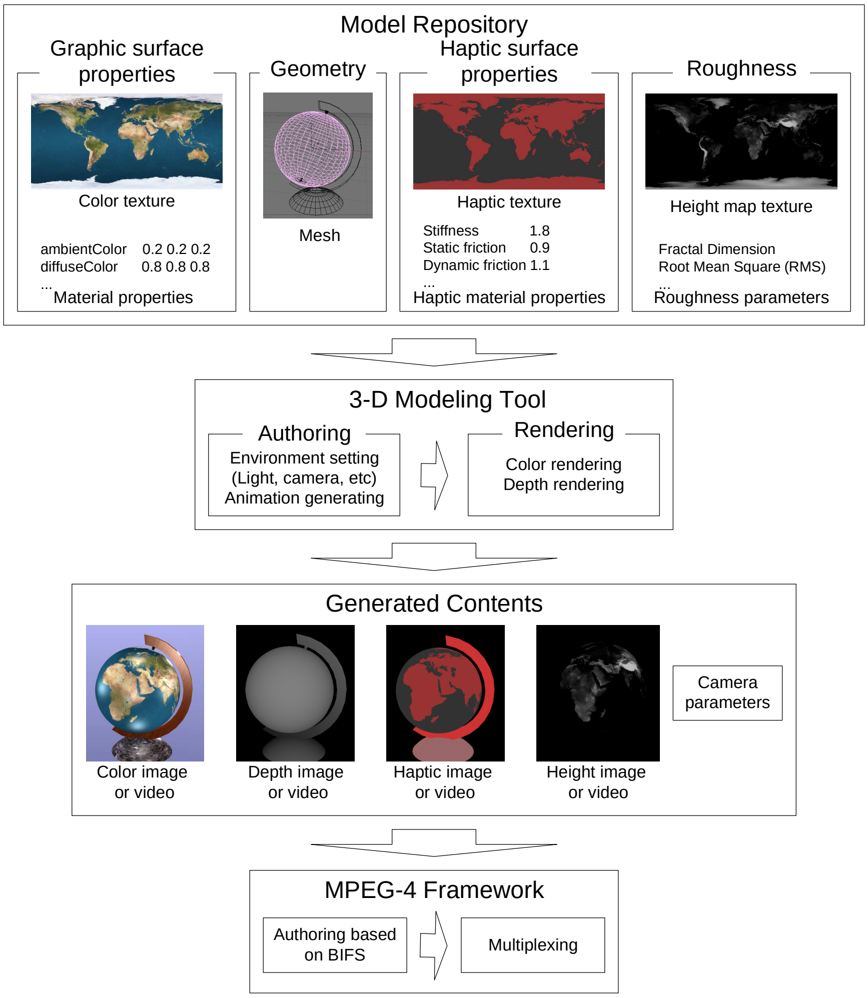


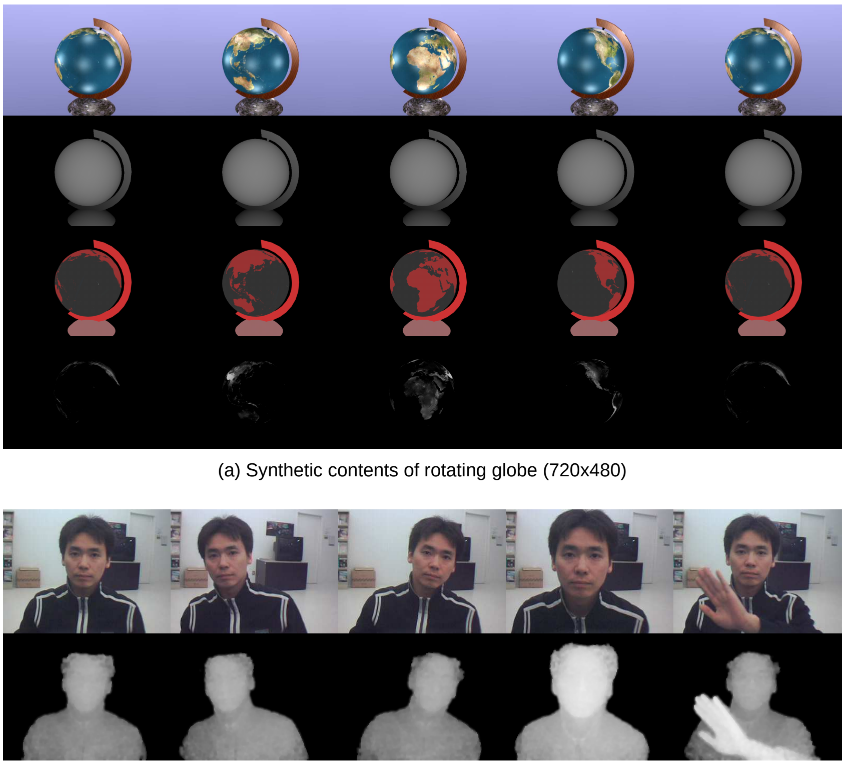
Fig. 5. Synthetic contents generation pipeline

can actively touch a video scene through a force-feedback device.

In the following subsections, we present the components of the system with the underlying technologies according to information flow (that is from contents generation, to communication and delivery, an finally to viewing and interaction).

3.1 Contents Generation

Content generation is performed for either synthetic or natural contents. The synthetic contents generation pipeline is shown in Fig. 5. The pipeline starts from the model repository that stores the 3D graphic data and haptic properties of the virtual objects used to populate the haptic virtual scene. The repository includes the object geometry (for example the mesh model), the color information (such as surface material attributes or color textures), the haptic surface properties (hap-



(a) Synthetic contents of rotating globe (720x480)

(b) Natural contents captured by Zcam (320x240)

Fig. 6. Generated contents

tic surface material attributes or haptic textures), and the roughness information (roughness parameters or heightmap). This information is imported to a conventional 3D modeling tool (Blender in our case) that composes the 3D scene both in the spatial and temporal aspects and renders it into four 2D output images per video frame: the color image, the depth image, the haptic image, and the heightmap image (as shown in the Generated Contents block). Notice that the settings of the virtual environment such as light and camera must be setup properly for each image.

In the rendering process, first, the 2D color image is generated based on the geometry and graphic color inputs (and the heightmap image in some cases to provide more detailed surface representation) in a similar way a 3D animation is produced. The depth image is composed by extracting the Z depth information from the depth buffer during the color image rendering process. And, since conventional 3D modeling tools do not support setting haptic properties, the haptic image is rendered by replacing the color properties (RGB values for the scene) by the haptic surface properties. One possible way is to replace the color texture by the haptic texture that defines the haptic material properties. This haptic texture can either be generated by assigning color values for each object to represent its haptic properties, either manually by the media author or by using an authoring tool that can create haptic textures based on the knowledge of the object's physical properties. Since

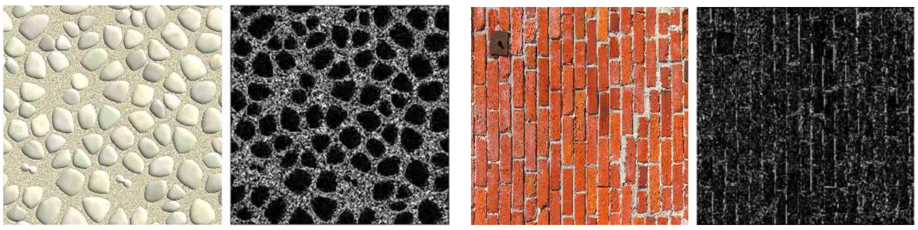


Fig. 7. Example of Laplacian of Gaussian filter

other environmental effects such as light, shadow, etc. can affect colors in the rendered haptic image, we receive the haptic image through the color buffer that only has the color information of the scene without the light effect. The heightmap image is rendered in the same way except for replacing the color texture with the heightmap texture. Fig. 6(a) shows the generated contents of a globe that rotates. The land part of the globe was set with a higher stiffness and friction than the ocean part and the land part was covered with the heightmap that has the height information of mountain ranges.

Finally, the generated contents (four image streams) are synthesized into one scene using MPEG-4 BIFS and encoded and multiplexed into one stream by utilizing the MPEG-4 framework. Eventually, viewers can consume the MPEG-4 contents by touching the displayed scene with a force-feedback device through a haptic-enabled MPEG-4 player, either locally or remotely via networks.

In case of a real natural scene, the depth camera is used to capture the real scene and generate 2D color and depth videos. Fig. 6(b) shows the captured scene with a ZCam^{TM}. However, in this work, we assigned one haptic property to one scene due to the existing sensors' capabilities. Contrary to a camera as a tool to capture visual information, tactile sensors usually capture local haptic properties by physically scanning objects [Lee and Nicholls 1999]. Even the tactile sensors based on optical technique need to be set very close to the objects to acquire the haptic properties. This makes it very difficult to generate natural haptic video. However, in the case of off-line generation, there is one possible approach to generate the haptic and heightmap images from the natural color image. First, we measure locally different haptic properties of objects in a scene. Second, we segment the objects into many parts based on color information by using image-based segmentation algorithms [Lucchese and Mitra 2001]. Finally, we can assign the measured or predefined haptic properties and roughness into each segment manually. In short, each segment is painted with the color that corresponds to the haptic surface properties and roughness. This way, the natural haptic video generation can be semi-automated. In addition, as for the roughness, one potential technique is the Laplacian of Gaussian (LoG) approach to automatically model scene roughness. Since the Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection, this method can extract the heightmap image from the color images that have quite obvious textures like in Fig. 7.

```
Shape {
   exposedField   SFAppearanceNode   appearance   NULL
   exposedField   SFGeometryNode     geometry     NULL
}
```

Fig. 8. **Shape** node in MPEG-4 BIFS

3.2 Transmission of DIBHR Contents Based on an MPEG-4 Framework

There have been several attempts to construct a multimedia framework supporting the easy creation and distribution of haptic contents. For instance, the Reachin API provides a fully extensible programming framework for building haptic interactive contents, based on VRML [Reachin AB 2003]. It allows the creation of virtual worlds featuring a range of media types including video, audio, graphics, and haptics that can be sent over the Internet. H3D supports a similar framework based on X3D [SenseGraphics AB 2006]. However, these frameworks are not appropriate for transmitting DIBHR data as they are based on a simple download-and-play delivery system rather than one which has the ability to stream media data.

To stream media containing haptic data, this paper focuses on the MPEG-4 framework, which not only supports streaming of a wide range of media objects, but also provides flexible interactivity designed for broadcasting specific applications. A key difference in MPEG-4, when compared to prior audio-visual standards, is its object-based audio-visual representation model. For example, an object describing an animated moving head can encode movement using mathematical parameters, while a simultaneously displayed video scene can remain composed of adaptive pixel values. It also supports the harmonious integration of these varied data types, providing a unifying system that can enable feats, such as the seamless interaction between a cartoon character and a real actor in a studio. This flexibility makes MPEG-4 an ideal technology for supporting haptic contents streaming. Each image in DIBHR can be encoded/decoded as independent objects yet easily synthesized and synchronized together or with other audio-visual media. In addition, the MPEG-4 3D audio-visual (3DAV) group investigates new kinds of media that extend the functionalities of available standard technology in the viewpoint of interactivity and 3D vision [Fehn et al. 2003] similar to our concept of haptic interaction.

In MPEG-4, every object in a scene is described and combined with other objects through an MPEG-4 scene description language called BInary Format for Scenes (BIFS), which is suited for online transmission and streaming of 2D and 3D scenes. Technically, images in DIBHR can be described using MPEG-4 BIFS. However, the current MPEG-4 specifications do not consider haptic media, an omission addressed by the work in this paper. Therefore, we have extended the MPEG-4 BIFS to support new types of nodes that incorporate DIBHR. In addition, we briefly discuss the compression issue of the image/video in DIBHR.

3.2.1 *New MPEG-4 Node Specifications for DIBHR.* All visible objects in a scene with MPEG-4 BIFS are described within the **Shape** node that contains the appearance and geometric information as shown in Fig. 8. Therefore, it is reasonable to define DIBHR in the **Shape** node. In the **Shape** node, two fields, namely *appearance* and *geometry*, point to appearance nodes and geometry nodes,

```
Depth {
    field          SFFloat        focalLength   4.7
    field          SFFloat        pixelWidth    0.0112
    field          SFFloat        nearPlane     40
    field          SFFloat        farPlane      180
    exposedField   SFTextureNode  texture       NULL
}
```

Fig. 9. **Depth** node

```
Appearance {
    exposedField   SFMaterialNode          material          NULL
    exposedField   SFTextureNode           texture           NULL
    exposedField   SFTextureTransformNode  textureTransform  NULL
    exposedField   SFHapticSurfaceNode     hapticSurface     NULL
}
```

Fig. 10. **Appearance** node extended by *hapticSurface* field

```
HapticTextureSurface {
    exposedField   SFVec2f        stiffnessRange         0.1 10
    exposedField   SFVec2f        staticFrictionRange    0.1 1.0
    exposedField   SFVec2f        dynamicFrictionRange   0.1 1.0
    exposedField   SFFloat        maxHeight              1.0
    exposedField   SFTextureNode  hapticTexture          NULL
    exposedField   SFTextureNode  heightTexture          NULL
}
```

Fig. 11. **HapticTextureSurface** node for haptic surface properties

respectively. One frame of a scene represented with DIBHR is composed of 4 streams of images as geometry and surface property data and 19 floating-point values as meta-data. From modeling perspective, the depth image has geometry information (and thus described with a geometry node) whereas the color and haptic images have visual and haptic appearance information (described with an appearance node).

As for the depth image, we propose a new **Depth** node that can be stored in the *geometry* field. The **Depth** node contains the meta-data for the camera parameters as floating-point values. The actual geometry data, the depth image, is stored as a texture node in the *texture* field, as described in Fig. 9.

In MPEG-4 BIFS, since the appearance of any geometry is expressed in the **Appearance** node, the color image is stored in the *texture* field of the **Appearance** node. However, since the **Appearance** node does not include haptic appearance, we have defined a new field named the *hapticSurface* that can direct any haptic surface nodes as shown in Fig. 10. In order to describe the haptic surface properties, this field directs a **HapticTextureSurface** node that contains the range parameters and two texture fields: *hapticTexture* and *heightTexture*, that stores the haptic image and the heightmap image as shown in Fig. 11. Fig. 12 shows the hierarchical structure of the nodes for DIBHR.

In MPEG-4 BIFS, the texture field can direct texture nodes, such as **ImageTexture** and **MovieTexture** nodes. Each node can store a static image and a

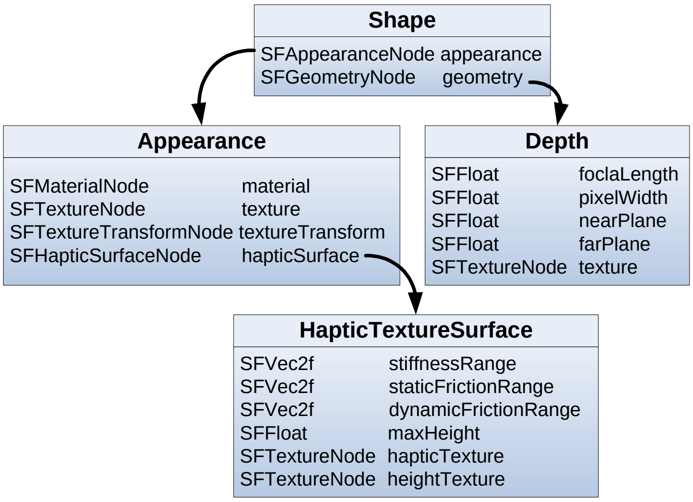


Fig. 12. Structure of newly designed MPEG-4 BIFS nodes

dynamic video, respectively. Therefore, the contents described with newly designed BIFS nodes based on DIBHR can be a static scene or a dynamically changing scene. Fig. 13 shows an example of a static scene described with new BIFS nodes.

3.2.2 *Compression.* As for the color image, high-quality lossy compression can be adopted, which does not degrade the quality of appearance of the rendered 3D scene. Since, the degradation of the depth information may severely distort the geometry of the scene in 3D space and deteriorate the haptic perception, lossless or near-lossless compression needs to be applied. Due to the fact that haptic properties are characterized by local homogeneity, the haptic image is likely to have piecewise homogeneous regions and does not have as many colors as the color image. Therefore, the palette-indexed color format can significantly reduce the haptic image size and the lossless compression is reasonable to maintain the quality of force feeling. As for the heightmap image, since the pixel values represent micro-geometry in the 3D space, lossless or near-lossless compression is used like the depth image.

Therefore, as a practical application, in the case of a static scene, the depth image and the heightmap image are stored in a PNG format for lossless compression whereas the color image is stored in a high-quality JPEG format. The haptic image is converted into an indexed-color mode, and then stored in the PNG format. In the case of the dynamically changing scene, all color, depth, haptic and heightmap videos are compressed using the H.264/MPEG-4 AVC[7]. As for the depth and heightmap video, we tried to minimize the degradation in quality by setting the quantization values as low as possible. Indeed, one future work avenue will be to develop a novel compression method for haptic video, especially for real-time performance.

[7] ISO/IEC 14496-10, Coding of audio-visual objects – Part 10: Advanced Video Coding

```
Shape {
  appearance Appearance {
    texture ImageTexture {
      url “color_image.jpg”
    }
    hapticSurface HapticTextureSurface {
      stiffnessRange 0.1 10
      staticFrictionRange 0.2 0.9
      dynamicFrictionRange 0.2 0.9
      maxHeight 1.0
      hapticTexture ImageTexture {
        url “haptic_image.jpg”
      }
      heightTexture ImageTexture {
        url “height_image.png”
      }
    }
  }
  geometry Depth {
    focalLength 6.983
    pixelWidth 0.00123
    nearPlane 10
    farPlane 200
    texture ImageTexture {
        url “depth_image.png”
    }
  }
}
```

Fig. 13. Example of a scene description with newly designed MPEG-4 BIFS nodes

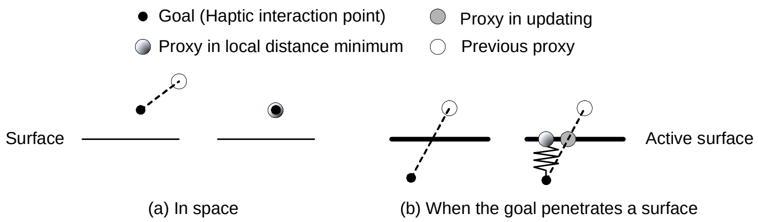

● Goal (Haptic interaction point)
Proxy in updating
Proxy in local distance minimum
○ Previous proxy
Surface
Active surface
(a) In space
(b) When the goal penetrates a surface

Fig. 14. Proxy-based algorithm

3.3 Viewing and Interaction

In the viewing and interaction stage, viewers can enjoy actively touching a video scene through a force-feedback device, as well as more traditional experiences, such as watching. The decoded media, 4 streams of images, are fed to a compositor process which has access to the BIFS scene graph. Traditionally, the compositor scans the scene graph, determines what visual content should be shown and then passes this to the visual renderers that actually handle the display of the media on an entertainment device, such as a TV. The system proposed in this paper extends the compositor process to deal with haptic elements in the scene graph by a similar process of routing them to the appropriate renderer. The visual renderer receives color images and depth images to draw a 3D scene. The haptic renderer receives depth images, haptic images and heightmap images and obtains the viewer's interaction point to compute the interaction force generated from the touchable objects in the scene, and then transfers that value to the force feedback device worn or held by the viewer. The force calculation is performed through a haptic rendering algorithm that will be described in this subsection. The haptic rendering algorithm is the process of computing and generating forces in response to interaction between the haptic device and the virtual environment. 3DOF haptic rendering algorithms restrict the user's avatar to a single point of interaction. Several algorithms were proposed, for instance a proxy based 3DOF algorithm for polygonal meshes was first introduced in [Zilles and Salisbury 1995]. A proxy or god-object, that is an ideal massless point that can not penetrate any surface, is connected to a goal point that represents the position of the haptic device in the virtual environment. The proxy and goal objects are connected through an ideal spring that is zero to infinity in length. In a haptic loop, when the goal is moved, the proxy is updated to a location with a minimum local distance to the goal (because of the ideal spring). In other words, if the goal is in free space or the path of the goal (the line segment from the previous proxy to the goal) does not collide with any object, the proxy coincides with the goal as shown in Fig. 14(a). Also, if the goal object penetrates a surface or the line segment collides with a surface, the collided surface is set as active and the proxy is updated to the closest location to the goal constrained on a plane that contains the active surface as shown in Fig. 14(b). Then, the resultant force is computed using a linear spring model where the spring coefficient refers to the stiffness value of the active surface. In this section, we introduce a haptic rendering algorithm to compute the interaction force between the user and the touchable 3D scene.

3.3.1 *Overview of Haptic Rendering Algorithm.* The core process of a haptic rendering algorithm is to search for a new proxy location that minimizes the distance to the goal object. However, when we compute a friction force, the proxy path during the haptic rendering algorithm as well as the new proxy location are important . Traditional proxy-based haptic rendering algorithms such as the god-object algorithm [Zilles and Salisbury 1995], the neighborhood search algorithm [Ho et al. 1999], and the proxy graph algorithm [Walker and Salisbury 2003] result in distorted force rendering due to the incorrect computation of the proxy path. In our previous work [Cha et al. 2008], we remedied this problem by computing the correct proxy path to minimize the distance between the proxy and the goal in order to render smoother friction forces.

In this haptic rendering algorithm, three types of primitives are defined: Triangle, Edge, and Vertex; each contains its geometry information and neighborhood information [Cha et al. 2008]. For instance, a Triangle primitive has three vertices and a normal as geometry information and three edges as neighbors. The algorithm searches for a proxy position that results in minimum distance between the proxy and the haptic interaction point, and eventually finds the shortest path along which the proxy traces to the new proxy location. When a collision is detected between a primitive and the line connecting the goal object and the proxy object, the proxy is

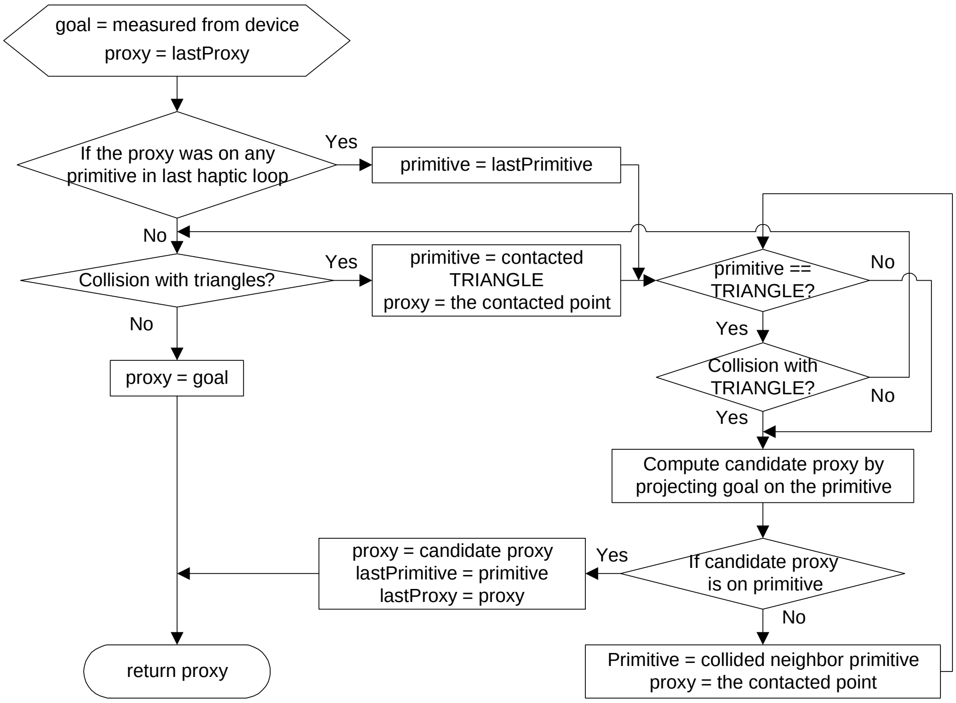

goal = measured from device
proxy = lastProxy
If the proxy was on any primitive in last haptic loop
Yes
primitive = lastPrimitive
No
Collision with triangles?
Yes
primitive = contacted TRIANGLE
proxy = the contacted point
No
proxy = goal
primitive == TRIANGLE?
No
Yes
Collision with TRIANGLE?
No
Yes
Compute candidate proxy by projecting goal on the primitive
proxy = candidate proxy
lastPrimitive = primitive
lastProxy = proxy
Yes
If candidate proxy is on primitive
No
return proxy
Primitive = collided neighbor primitive
proxy = the contacted point

Fig. 15. The complete flow chart for DIBHR algorithm

moved on the obstructing primitive at the collided position and the current primitive is set to active. Next, the neighborhood search algorithm determines whether the proxy will go into space or not. Then the algorithm computes the candidate for the new proxy location. If the candidate location is not on the primitive and goes over any neighbor, the active primitive is updated to the neighbor primitive and the proxy will be located at the collided position. If the candidate location is on the active primitive, it becomes the new proxy location in local minimum. The same process repeats until the proxy location is obtained at local minimum. The complete flow chart of the algorithm is described in Fig. 15

In order to provide richer haptic experiences, we considered friction and roughness properties of the surface. The friction force is computed using the friction cone algorithm [Melder and Harwin 2004]. While traversing the primitives to look for the local minimum, the path of the proxy at each update is checked against the friction cone that restricts the proxy position outside itself and then produces a friction force. In the case of roughness rendering, the haptic rendering algorithm is not responsible. Besides, during the haptic rendering update, the depth values, z, are modulated by the height values, z_{height}, of the corresponding pixels in the heightmap image following Equation 2.

$$z = z_{depth} + MAX_{height} \frac{z_{height}}{255} \tag{2}$$

where, z_{depth} is the depth value of a pixel in the depth image and MAX_{height}

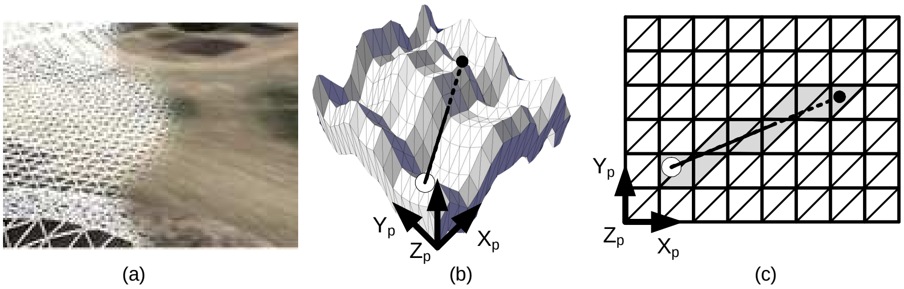


Fig. 16. Triangulation of depth image and optimized collision detection

is the maximum height value described in meta-data when the height value is 255. Since the haptic rendering algorithm is performed locally around the interaction point, this operation can be performed locally in real-time when the haptic rendering algorithm refers to the depth values which does not delay the processing time much.

3.3.2 *Application to Depth Image Sequences.* To facilitate haptic rendering, the depth image needs to be triangulated into a 3D surface. The surface generation is performed as follows: first, the depth information represented as a gray-scale image is mapped into an array of elevations in the pixel coordinates of X_p, Y_p, and Z_p in Fig. 16. Then the tips of these elevation vectors are connected vertically, horizontally, and diagonally to form a continuous surface that describes the 3D scene (as shown in Fig. 16(b)). The derived surface is used to test for collisions in the pixel coordinate. Finally, it is worth mentioning that the triangulation is performed locally wherever needed since the contact occurs in the local proximity of the haptic interaction point.

The first step in the collision detection process is to project the line segment that connects the goal and the previous proxy positions onto the 2D representation of the depth image in order to generate a list of candidate triangles that should be checked for collisions (shaded area in Fig. 16(c)). Then the cells within this candidate list that possess elevations below that of the line segment are discarded. This optimization makes the algorithm execute rapidly when applied to a depth image and independently from the scene complexity [Walker and Salisbury 2003]. Furthermore, the optimized collision detection does not require bounding boxes that are usually used in general collision detection and results in significant precomputation overhead and memory consumption. Therefore, we do not have to store additional data for collision detection. Although the actual haptic rendering algorithm should be performed in the world coordinate not the pixel coordinate, the collision detection and the proxy update processes are performed in the pixel coordinate for simpler and faster computation as explained above. However, the resultant force is computed in the world coordinate by transforming the resultant proxy position from the pixel coordinate to the world coordinate.

When the haptic rendering algorithm is directly applied to a dynamically changing scene comprising of a sequence of depth images, there can be three problems, as depicted in [Cha et al. 2006]. The first problem encountered is that the algo-

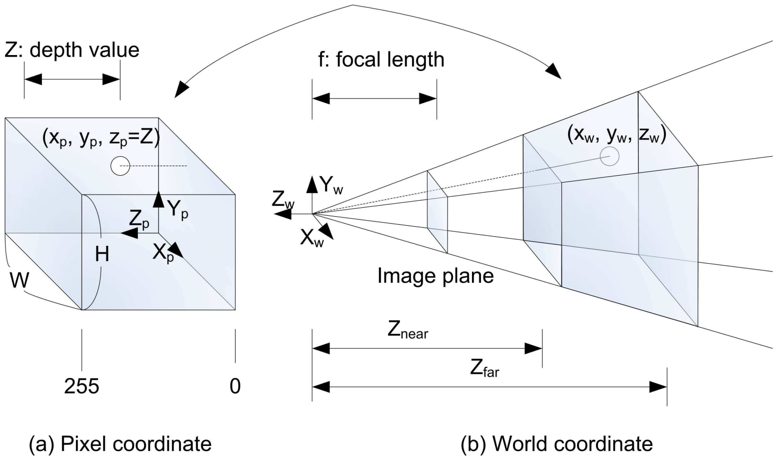


Fig. 17. Coordinate mapping

rithm fails to detect some collisions between the sequences of the depth images and the proxy position. Second, the rendered contact forces are piecewise-continuous due to the significant difference between the video and haptic update rates (30 Hz versus 1000 Hz). Finally, abrupt changes in force rendering due to sudden scene transitions make the haptic device unstable. Therefore, we correct the collision detection by updating the proxy position based on the depth value, interpolate the depth values between visual updates, and monitor the sudden change of the proxy position as described in [Cha et al. 2006].

Finally, since the depth image has an 8-bit channel and has just 256 levels, a smooth surface can be felt like a rugged surface. Furthermore, when captured from a depth camera, such as the ZCam^{TM}, the depth image has high frequency geometrical noise from optical noise, which is mainly due to the reflectivity or color variation of captured objects [Kim et al. 2006]. Therefore, in order to smooth the depth value, a well-known low-pass filter, a median filter, is used. The median filter is the simplest low-pass filter that averages neighbors' values. The size of the area to average depends on how much noise the depth image has. Since the haptic rendering algorithm refers to the depth values in the local area, we apply the median filter locally in real time when the haptic rendering algorithm refers to the depth value.

3.3.3 *Relationship Between Pixel Coordinate and World Coordinate.* This section describes the mapping technique that transforms the pixel coordinate into the world coordinate and then converts the depth image into the 3D scene geometry as shown in Fig. 17. In order to reconstruct the 3D scene geometry, each pixel of the depth image needs to be transformed into the 3D world coordinates. The depth image is represented based on the pixel coordinate. In the pixel coordinate, the X_p, Y_p and Z_p directions represent the width indices, the height indices, and the depth values, respectively, of each pixel. Fig. 17 (a) shows the pixel coordinate centered at the lower-left corner. Each pixel can be transformed into the 3D world coordinate through intrinsic camera parameters that depict the spatial relationship between

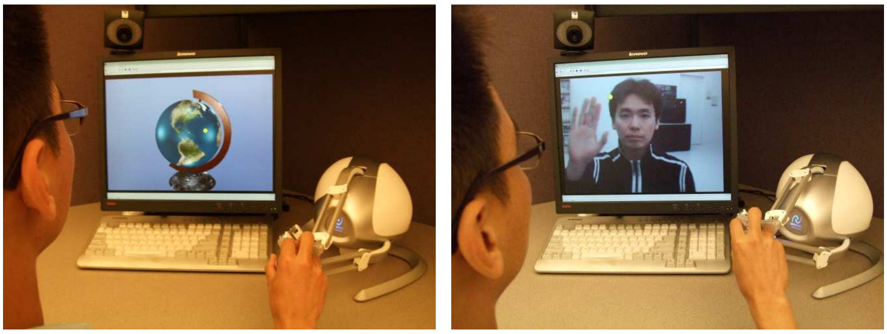


Fig. 18. 3D touchable video player and its consumption of synthetic and natural contents

two coordinates as expressed using Equation 3. By using this equation, the pixel of (x_p, y_p, z_p) in the pixel coordinate is transformed into a point of (x_w, y_w, z_w) in the world coordinate.

$$\begin{aligned}(x, y, z) &= \left(\sigma\left(x_p - \tfrac{W}{2}\right), \sigma\left(y_p - \tfrac{H}{2}\right), -f\right) \\ (x_w, y_w, z_w) &= \left(Z_{near} + (Z_{far} - Z_{near})\tfrac{255 - z_p}{255}\right)\tfrac{(x,y,z)}{\sqrt{x^2+y^2+z^2}}\end{aligned} \tag{3}$$

where,f is the focal length of the camera, σ is the pixel width where square pixels are assumed, W and H are the respective width and height of the depth image, Z_{near} and Z_{far} are the depth clipping planes. Points on the far clipping plane Z_{far} are assigned zero depth values whereas points on the near clipping plane Z_{near} are assigned the full scale value of 255. (x, y, z) indicate the equivalent position of a pixel on the image plane in the world coordinate.

As for haptic rendering, the collision detection and the proxy update processes are performed in the pixel coordinate for simpler and faster computation. Therefore, the goal location in the world coordinate is transformed into the pixel coordinate and then the collision detection and the proxy update process is performed. However, since the interaction force needs to be computed in the world coordinate, the updated proxy location is converted back to the 3D world coordinate. Then the friction cone algorithm is applied and the resultant proxy in the 3D world coordinate yields the force applied based on the stiffness value.

3.4 Implementation of a Touchable 3D Video Player

The touchable 3D video player is implemented using GPAC[8], a multimedia framework based on the MPEG-4 systems standard. It supplies basic modules for encoding and decoding and multiplexing and demultiplexing various multimedia including MPEG-4 BIFS and for playing the multimedia audio-visually. In order to enable haptic interaction, a Novint Falcon[9] was used as a force feedback device and the haptic rendering algorithm was included in the player. Fig. 18 shows the displayed

[8] http://gpac.sourceforge.net

[9] http://home.novint.com/

scene through the player and a viewer touching the scene through the force feedback device.

4. RESULTS

In order to show the feasibility of the representation method and evaluate the haptic rendering algorithm, we created consumable contents as shown in Fig. 6 and measured the computation time of the haptic rendering algorithm and the spectral density of the resultant force to verify if the haptic rendering algorithm functions marginally within a 1 kHz update rate in order to produce stable forces and if the applied median filter smoothed the interaction force, respectively. The contents are both synthetically generated by using a conventional 3D modeling tool and naturally captured with a commercially available depth camera, named Z-Cam^{TM}. The haptic rendering algorithm was implemented under Windows XP on an Intel based PC (Pentium®D 3.4GHz, 1 GB RAM, Intel®946GZ Express Chipset). As a force-feedback device, the Novint Falcon was used.

4.1 Haptic Rendering Performance Evaluation

In general, the haptic rendering update rate should be as fast as possible (typically 1 kHz, i.e. haptic computational time needs to be within 1 millisecond) for stable and transparent haptic interaction. In order to estimate the performance of the haptic rendering algorithm for stable force rendering, the computational time for each haptic update was measured for synthetic and natural video contents as shown in Fig. 6 using a high-resolution timer provided by the Windows XP OS. The resolution of the synthetic video was 720×480 which is a Standard Definition(SD) video for general television and that of the natural video was 320×240 that is usually used for a messenger program or internet video. In the first experiment, three users were asked to move the haptic device as fast as possible across the surface while keeping contact with the surface for 30 seconds. This was done in order to measure the average maximum computation time in a haptic loop, which is a similar method to evaluate the performance of a haptic rendering algorithm in [Walker and Salisbury 2003; Cha et al. 2006].

Table I shows the results of the performance test for the computational time and the number of primitives traversed. As for the synthetic contents, the maximum computational time among the three users was 274.5 microseconds on average for 30 seconds and the number of primitives transitioned was 6.712 primitives. The average values of the three users were 253.4 microseconds and 5.549 primitives. As for the natural contents, the maximum values were 91.0 microseconds and 4.174 primitives. The average values of the three users were 80.9 microseconds and 3.498 primitives. The results show that the proposed algorithm operates comfortably within the 1 millisecond range, producing a stable force. Since an average of the maximum computation time for SD video is quite short, we can apply multi-point (multi-finger) haptic rendering that simply repeats the same process for other interaction points. For example, if we have multiple point-based force-feedback device configured to fit each finger like in [Michelitsch et al. 2002], we can enjoy more natural and intuitive interaction. In case of five points for five fingers, the maximum computation time for SD video would be around 1,267ms that is five times of 253.4ms and this will provide stable forces with reasonable stiffness. Note that

Table I. Computation time of Haptic Rendering

	Synthetic contests (720×480)		Natural contents (320×240)	
Users	Computational time (microseconds)	# of primitives traversed	Computational time (microseconds)	# of primitives traversed
User1	274.5	6.712	77.5	3.303
User2	229.3	4.433	74.1	3.017
User3	256.5	5.502	91.0	4.174
Average	253.4	5.549	80.9	3.498

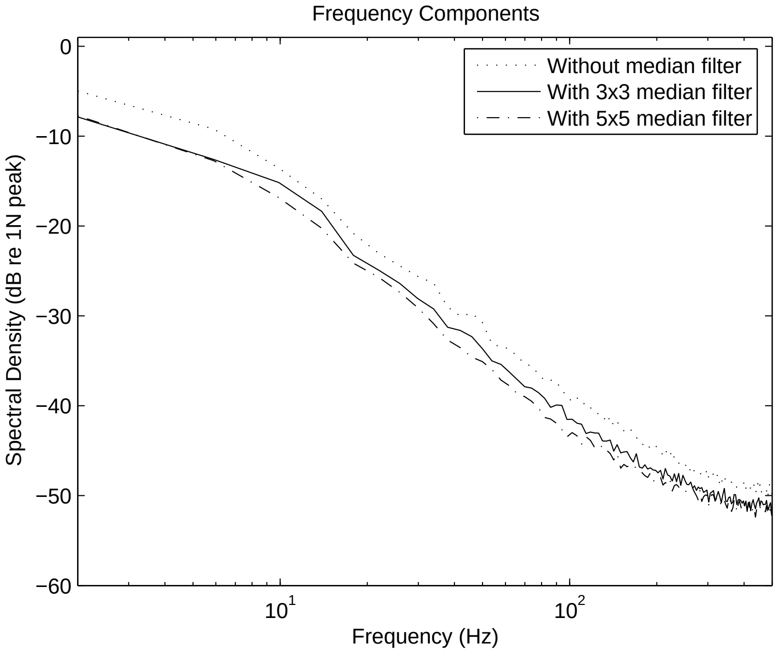


Fig. 19. Power spectral densities of the acquired force magnitudes before and after the median filters.

multi-point haptic rendering is different from the object-to-object haptic rendering. In multi-point haptic interaction, a hand could be modeled as five single points that represent each fingertip.

The second experiment evaluated the improvement in the smoothness of interaction force when utilizing a median filter. This experiment was applied only to the natural content because the natural content has undesirable high frequency geometrical noise. A subject was asked to softly scratch the chest area of the touched person in Fig. 6 keeping the haptic device in contact with the touched body for as long as possible. The contact force magnitudes without the median filter and with the 3x3 and 5x5 median filters were acquired at a 1 millisecond rate for 100 seconds. In order to show the degree of smoothness of the force, time-domain data of the force magnitudes was transferred to the frequency domain and power spectral densities were obtained. Then spectral densities corresponding to the ten 10-second segments of the force magnitude data were computed and averaged for noise reduction. Fig. 19 shows the power spectral densities of the force magnitude for three cases: no median filter, a 3x3 median filter, and a 5x5 median filter. The results show that the spectral density has noticeably decreased after using a smoothing filter. This implies that the user could touch smoother surface with the median filter. The user subjectively reported that he could not explore the surface well without the filter because the haptic interaction point frequently fell into the crevasses of the surface from the high frequency geometrical noise.

4.2 Usability Study

In order to test the feasibility of touchable video systems, we performed a usability test using the rotating globe animation (shown earlier in the paper). Eight participants took part in the experiment (6 male, 2 female; five of them had no previous knowledge of haptics). They were asked to explore the globe animation and experience the haptic properties (land texture, heightmap, stiffness, etc.), and then fill a questionnaire that we designed for this purpose. To minimize the side effects, participants were allowed 'trial' sessions with the haptic device until they feel comfortable before performing the experiment.

All the participants confirmed that haptic feedback was useful, and helped particularly to perceive the fine details of the globe surface (all answered by 'No' when asked if they would like to disable the haptic feedback). Seven out of eight agreed that the haptic feedback was realistic enough to convey the globe surface roughness. Furthermore, six out of eight confirmed that the feeling of the fine details of the globe was realistically conveyed by the height map. On the other hand, two out of eight mentioned that the haptic device was somehow tiring and has caused some fatigue. This test shows the preliminary results of the system but more usability tests about comfort, presence, intermodal conflict, etc. need to be conducted in the near future.

5. CONCLUSION AND FUTURE WORK

This work aims at integrating the haptic modality in audio-visual multimedia systems and applications. We presented a representation method, named DIBHR (Depth Image-Based Haptic Representation), to encode haptic properties in a similar way that color information is represented. Four images are generated per frame: color, depth, haptic, and heightmap. The haptic image contains the stiffness, static friction, and dynamic friction whereas the heightmap image contains roughness of the video object. The proposed representation was implemented based on an MPEG-4 framework where new MPEG-4 BIFS nodes were designed to describe the haptic properties. The realization of contents generation and their corresponding haptic rendering algorithm were also presented. The performance evaluation showed that the haptic rendering of the DIBHR representation operates marginally within a 1 kHz update rate, which is required to provide stable interaction force and to calculate smoother force using the median filter.

As for future work, we are planning to investigate three major issues related to the current work. Content authoring is currently performed manually in a traditional 3D modeling tool. First, we plan to develop an authoring tool that enables content authors to generate haptic and depth/heightmap images by adding haptic property editor and haptic image renderer that produce haptic and heightmap image based on the adjusted haptic properties. This will be done by extending HAMLAT (HAML-based Authoring Tool) [Eid et al. 2008] that we have developed based on Blender, an open source 3D modeling tool, to generate a haptic scene. However, this becomes even more challenging for natural video applications where the haptic and heightmap images need be generated automatically or semi-automatically. Therefore, secondly, a method for assigning haptic textures to the natural video, for instance, using segmentation algorithms, will be the subject of our future work. In addition, an automatic feature extraction is needed to generate heightmap images from the color image by using such as LoG. Finally, since currently four images per frame are needed for the DIBHR method, a real-time compression algorithm is needed for efficient real-time communication of DIBHR data. This is another direction in the development of the current approach.

REFERENCES

Alatan, A. A., Yemez, Y., Güdükbay, U., Zabulis, X., Müller, K., Ç. E. Erdem, Weigel, C., and Smolic, A. 2007. Scene representation technologies for 3DTV—A survey. *IEEE Trans. Circuits and Systems for Video Technology 17,* 11 (Nov.), 1587–1605.

Avila, R. S. and Sobierajski, L. M. 1996. A haptic interaction method for volume visualization. In *Proceedings of IEEE Visualization Conf.* 197–204.

Bukowska, M. 2001. Winky Dink half a century later. Interaction with broadcast content: Concept development based on an interactive storytelling application for children. Tech. rep., Media Interaction Group, Philips Research Laboratories, Eindhoven, The Netherland. Aug.

Cha, J., Eid, M., and El Saddik, A. 2008. DIBHR: Depth image-based haptic rendering. In *Proceedings of EuroHaptics.* 640–650.

Cha, J., Kim, S., Ho, Y., and Ryu, J. 2006. 3D video player system with haptic interaction based on depth image-based representation. *IEEE Trans. Consumer Electronics 52,* 2 (May), 477–484.

Cha, J., Ryu, J., Kim, S., Eom, S., and Ahn, B. 2004. Haptic interaction in realistic multimedia broadcasting. In *Proceedings of 5th Pacific Rim Conference on Multimedia, Advances in Multimedia Information Processing.* 482–490.

Cha, J., Seo, Y., Kim, Y., and Ryu, J. 2007. An authoring/editing framework for haptic broadcasting: Passive haptic interactions using MPEG-4 BIFS. In *Proceedings of Joint EuroHaptics Conf. and Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems.* 274–279.

Chorianopoulos, K. and Lekakos, G. 2007. Learn and play with interactive TV. *ACM Computers in Entertainment 5,* 2.

Conti, F., Barbagli, F., Morris, D., and Sewell, C. 2005. CHAI: An open-source library for the rapid development of haptic scenes. In *Proceedings of Joint EuroHaptics Conf. and Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems.* Demo paper.

Eid, M., Andrews, S., Alamri, A., and El Saddik, A. 2008. HAMLAT: A HAML-based authoring tool for haptic application development. In *Proceedings of EuroHaptics.* Vol. 5024. 857–866.

Eid, M., Orozco, M., and El Saddik, A. 2007. A guided tour in haptic audio visual environments and applications. *International Journal of Advanced Media and Communication 1,* 3, 265–297.

El Saddik, A. 2007. The potential of haptics technologies. *IEEE Instrumentation & Measurement Magazine 10,* 1 (Feb.), 10–17.

Fehn, C., Schüür, K., Kauff, P., and Smolic, A. 2003. Meta-data requirements for EE4 in MPEG 3DAV. Pattaya, Thailand. ISO/IEC JTC1/SC29/WG11, Document M9559.

Gao, Z. and Gibson, I. 2005. Haptic B-spline surface sculpting with a shaped tool of implicit surface. *Computer-Aided Design & Applications 2,* 1-4, 263–272.

GAW, D., MORRIS, D., AND SALISBURY, K. 2006. Haptically annotated movies: Reaching out and touching the silver screen. In *Proceedings of Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*. 287–288.

GIBSON, J. J. 1962. Observations on active touch. *Psychological Review 69*, 477–491.

HALE, K. S. AND STANNEY, K. M. 2004. Deriving haptic design guidelines from human physiological, psychophysical, and neurological foundations. *IEEE Computer Graphics and Applications 24,* 2, 33–39.

HO, C.-H., BASDOGAN, C., AND SRINIVASAN, M. A. 1999. Efficient point-based rendering techniques for haptic display of virtual objects. *Presence: Teleoperators and Virtual Environments 8,* 5 (Oct.), 477–491.

IGNATENKO, A. AND KONUSHIN, A. 2003. A framework for depth image-based modeling and rendering. In *Proceedings of Graphicon-2003*. 169–172.

IKITS, M., BREDERSON, J. D., HANSEN, C. D., AND JOHNSON, C. R. 2003. A constraint-based technique for haptic volume exploration. In *Proceedings of IEEE Visualization Conf.* 263–269.

KAUFF, P., FEHN, C., COOKE, E., AND SCHREER, O. 2001. Advanced incomplete 3D representation of video objects using trilinear warping for novel view synthesis. In *Proceedings of Picture Coding Symp.* 429–432.

KIM, L., SUKHATME, G. S., AND DESBRUN, M. 2004. A haptic-rendering technique based on hybrid surface representation. *IEEE Tran. Computer Graphics and Applications 24,* 2 (Mar.), 66–75.

KIM, S.-M., CHA, J., RYU, J., AND LEE, K. H. 2006. Depth video enhancement for haptic interaction using a smooth surface reconstruction. *IEICE Trans. Information and Systems E89-D,* 1 (Jan.), 37–44.

LAWRENCE, D. A., LEE, C. D., PAO, L. Y., AND NOVOSELOV, R. Y. 2000. Shock and vortex visualization using a combined visual/haptic interface. In *Proceedings of IEEE Visualization Conf.* 131–137.

LEE, M. H. AND NICHOLLS, H. R. 1999. Tactile sensing for mechatronics–A state of the art survey. *Mechatronics 9,* 1 (Jan.), 1–31.

LEVKOVICH-MASLYUK, L., IGNATENKO, A., ZHIRKOV, A., KONUSHIN, A., PARK, I. K., HAN, M., AND BAYAKOVSKI, Y. 2004. Depth image-based representation and compression for static and animated 3-D objects. *IEEE Trans. Circuits and Systems for Video Technology 14,* 7 (July), 1032–1045.

LUCCHESE, L. AND MITRA, S. K. 2001. Color image segmentation: A state-of-the-art survey. In *Proceedings of Indian National Science Academy (INSA-A)*. Vol. 67. 207–221.

MAGNENAT-THALMANN, N. AND BONANNI, U. 2006. Haptics in virtual reality and multimedia. *IEEE Multimedia 13,* 3 (July), 6–11.

MATUSIK, W. AND PFISTER, H. 2004. 3D TV: A scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. *ACM Transactions on Graphics 23,* 3 (Aug.), 814–824.

MCNEELY, W. A., PUTERBAUGH, K. D., AND TROY, J. J. 1999. Six degree-of-freedom haptic rendering using voxel sampling. In *Proceedings of ACM SIGGRAPH*. 401–408.

MELDER, N. AND HARWIN, W. S. 2004. Extending the friction cone algorithm for arbitrary polygon based haptic objects. In *Proceedings of Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*. 234–241.

MICHELITSCH, G., RUF, A., VAN VEEN, H., AND VAN ERP, J. 2002. Multi-finger haptic interaction within the MIAMM project. In *Proceedings of EuroHaptics*.

O'MODHRAIN, S. AND OAKLEY, I. 2003. Touch TV: Adding feeling to broadcast media. In *Proceedings of European Conf. Interactive Television: from Viewers to Actors*. 41–47.

Reachin AB 2003. *Reachin API 3.2 - Programmer's Guide*. Reachin AB.

REINER, M. 2004. The role of haptics in immersive telecommunication environments. *IEEE Trans. Circuits and Systems for Video Technology 14,* 3 (Mar.), 392–401.

RIVA, G., DAVIDE, F., AND IJSSELSTEIJN, W. A. 2003. *Being There: Concepts, Effects and Measurements of User Presence in Synthetic Environments.* Amsterdam, The Netherlands, Chapter 2.

RUSPINI, D. C., KOLAROV, K., AND KHATIB, O. 1997. The haptic display of complex graphical environments. In *Proceedings of ACM SIGGRAPH*. ACM, New York, NY, USA, 345–352.

SALISBURY, J. K. AND TARR, C. 1997. Haptic rendering of surfaces defined by implicit functions. In *Proceedings of Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*. 61–68.

SALLNAS, E.-L., RASSMUS-GROHN, K., AND SJOSTROM, C. 2000. Supporting presence in collaborative environments by haptic force feedback. *ACM Trans. Computer-Human Interaction 7,* 4 (Dec.), 461–476.

SensAble Technologies, Inc. 2005. *OpenHaptics Toolkit Version 2.0 - Programmer's Guide*. SensAble Technologies, Inc.

SenseGraphics AB 2006. *H3D API Manual for Version 1.5*. SenseGraphics AB.

SMOLIC, A. AND KAUFF, P. 2005. Interactive 3-D video representation and coding technologies. *Proceedings of the IEEE 93,* 1 (Jan.), 98–110.

THOMPSON, T. V. AND COHEN, E. 1999. Direct haptic rendering of complex trimmed NURBS models. In *Proceedings of Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*. 89–96.

THOMPSON, T. V., NELSON, D. D., COHEN, E., AND HOLLERBACH, J. 1997. Maneuverable NURBS models within a haptic virtual environment. In *Proceedings of Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems*. 37–44.

WALKER, S. P. AND SALISBURY, J. K. 2003. Large haptic topographic maps: MarsView and the proxy graph algorithm. In *Proceedings of ACM SIGGRAPH*. ACM, New York, NY, USA, 83–92.

WITMER, B. G. AND SINGER, M. J. 1998. Measuring presence in virtual environments: A presence questionnaire. *Presence: Teleoperators and Virtual Environments 7,* 3 (June), 225–240.

YAMAGUCHI, T., AKABANE, A., MURAYAMA, J., AND SATO, M. 2006. Automatic generation of haptic effect into published 2D graphics. In *Proceedings of EuroHaptics*.

ZILLES, C. B. AND SALISBURY, J. K. 1995. A constraint based god-object method for haptic display. In *Proceedings of IEEE/RSJ Int. Conf. Intelligent Robots and Systems*. 146–151.

Received February 1986; November 1993; accepted January 1996