Tangible Images: Runtime Generation of Haptic Textures From Images

Hari Vasudevan*
Haptics Lab, Indian Institute of Technology Madras, India.

Manivannan.M†
Haptics Lab, Indian Institute of Technology Madras, India.

Abstract

This paper describes an algorithm for generation of forces from two dimensional bitmapped digital images, the forces allow the user to feel the contours of the image with sufficient realism using haptic devices. The algorithm can also be used to generate textures for virtual surfaces in a simple and efficient manner. The paper describes the design of the mask to be used to achieve the above two objectives and discusses the factors affecting the performance of the algorithm. Although the main focus of this paper pertains to haptic interaction with static images, the algorithm developed could also be used to haptically interact with videos.

keywords: Haptics, Textures, Image Processing

1 Introduction

Generating and rendering of textures is widely known in computer graphics and haptics literatures. In graphics, textures are to provide realistic look for a surface, which is accomplished mainly through various lighting models. Haptics textures serve a similar purpose, they are used to provide realistic feeling of haptic interactions. Most of the techniques for haptic textures need 3D models in order to generate the underlying force models for the surfaces, which inherently need preprocessing. These methods limit the haptic interaction only with static objects, run time generation of haptic textures is difficult. Enabling haptic sensation in bit mapped images can lead to very exciting developments, as we believe this can pave the way toward *movie haptics*, that is an algorithm that allows users to interact with dynamic objects such as video sequences in a sufficiently realistic manner.

In this paper we suggest a common method to achieve two very different goals in haptic rendering, one is to enable haptic sensation in two dimensional images, the other is to generate haptic textures in a simple and efficient manner. We have used well known edge detection algorithms from image processing literatures for achieving these goals.

2 Previous Work

Haptic textures have been studied for quite some time now and a variety of techniques have been used to synthesis textures. Minsky et.al. [6] present a 'Sandpaper' system of displaying haptic texture where a gradient technique is used to display a 3D texture in terms of 2D forces. Ho et.al [4] present a modification of 'bump maps' borrowed from computer graphics to create texture in haptic environments. This method involves the perturbation of surface normals to create the illusion of texture. Saira and Pai [10] present a stochastic approach to haptic textures aimed at reducing the computational complexity of texturing methods. Fritz and Barner [3] follow this up by presenting two stochastic models to generate haptic textures. In a wide ranging paper D.K.Pai et.al [9] describe methods to recreate textures, roughness and sounds that result from an interaction with a virtual object. Otaduy and Lin [8] present the use of a 6-DOF haptic device to render device textures and base their texture model on psychophysical experiments. Otaduy et.al. [7] continue their work on textures by exploring the interaction between textured models. Aysal and Barner [1] explore different methods to map haptic data to virtual environments to create haptic textures most suited to help visually impaired persons. Most of these techniques need 3D model of the objects in the virtual environment for generating the haptic textures.

Of particular interest is technical report [2], this report describes haptic interaction with animation sequences. The paper proposes methods to edit animations using vector fields which are output as forces through the haptic device. The key objective of the paper is to make animation editing more intuitive to the artist. While the paper succeeds in interactive editing of animation between sequences, such as controlling the speed repetitive animation sequences. The algorithm described in the paper may not help in interaction with individual frames within the animation. It is here that we believe that our proposed algorithm will contribute.

While the above approaches seek to produce haptic textures from bump-maps or using on the fly synthesis of textures, we seek an entirely new and simpler method of generating textures. We employ simple image processing techniques to compute textures at runtime and render the same to the user. This method avoids the need to store large amounts of data as is the case with texture maps and avoid time consuming computations involved in stochastic estimation of textures. The key difference between our proposed algorithm and those traditionally used for haptic rendering is that our algorithm operates on static two dimensional images while other algorithms for rendering haptic textures work with three dimensional models. For example implementation of bump maps requires a knowledge of the surface normals of every vertex or triangle in the given three dimensional model. This information is not available in two dimensional images making the implementation of bump maps difficult. We used standard image processing techniques to generate textures. True embossing seems to present an alternative to our algorithm, however the process of calculation the embossed image requires us to precompute the derivatives of the image. It is seen that for most real world images any derivative action results in a noisy haptic image.

Image based techniques have already been used in tactile displays [5], however our work is different in a way that the haptic information need to be computed only for the Haptic Interface Point (HIP), not for the entire image. This method presents a simple and efficient method of generating haptic textures which to the best knowledge of the authors has not been explored so far.

3 Motivation

Our work is inspired by the Sandpaper system described by Minsky et.al.[6] which first described haptic texture algorithm in 1990. They argue that the feeling of textures could be generated by creating tangential forces as a haptic device moves over features in the virtual environment. Figure 1, reproduced from [6], shows the forces rendered while traversing on textures with the haptic device. However generating haptic textures need geometric, topological, and haptic information of the objects in the virtual scene. Since the Sandpaper system all the haptic texture algorithms developed so far [3, 9, 8, 7, 1] need geometric and other information. In this

*e-mail: hvasudevan@iitm.ac.in
†e-mail:mani@iitm.ac.in

Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems 2008
13-14 March, Reno, Nevada, USA
978-1-4244-2005-6/08/$25.00 ©2008 IEEE

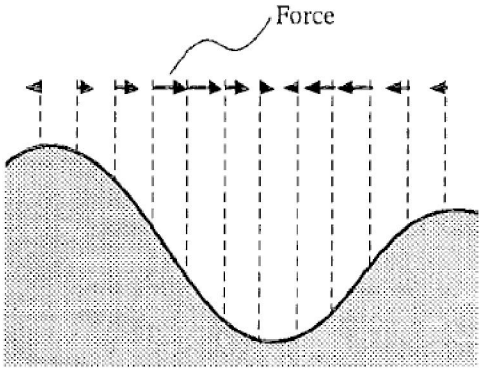


Figure 1: Gradient Technique : Enlarged cross section of surface depth map

paper we seek to generate similar haptic textures, without geometric information, from digital bitmapped images.

We exploit the fact that generally in an image convex surfaces are illuminated brighter compared to concave surfaces. Hence a 'mask' designed to generate forces that attract the user toward the darker(concave) regions of an image and repels the user from brighter(convex) regions will perform the same function as the 'Sandpaper' system.

While this method can be used to generate haptic textures, it can also be used in another context. The mask when run on images of objects or people, allows the user to 'feel' the image. In case of user interfaces it can be used to allow user to feel icons, buttons, scroll bars etc. The generation of haptic texture from tangential forces is a fairly established idea. However the generation of haptic images from real world images is an application of this concept that merits further investigation.

4 Tangible Image Algorithm

This section describes the algorithm and provides a description of the mask employed. A generalized description of the mask is as shown in the figure 2. The mask is an $m \times m$ matrix. The element of matrix colored red indicates the position of the haptic interface point. To illustrate the algorithm consider a 5×5 mask as shown in the figures 5 and 4. The algorithm used to generate haptic textures follows the numbered steps shown below.

-mi+mj	-(m-1)i+mj	....i+....j	0i+mj	....i+....j	(m-1)i+mj	mi+mj
-mi+(m-1)j	-(m-1)i+(m-1)j	....i+....j	0i+(m-1)j	....i+....j	(m-1)i+(m-1)j	mi+(m-1)j
....i+....j	....i+....j	....i+....j	0i+....j	....i+....j	....i+....j	....i+....j
-mi+0j	-(m-1)i+0j	....i+0j	0i+0j	....i+0j	(m-1)i+0j	mi+0j
....i+....j	....i+....j	....i+....j	0i+....j	....i+....j	....i+....j	....i+....j
-mi-(m-1)j	-(m-1)i-(m-1)j	....i+....j	0i-(m-1)j	....i+....j	(m-1)i-(m-1)j	mi-(m-1)j
-mi-mj	-(m-1)i-mj	....i+....j	0i-mj	....i+....j	(m-1)i-mj	mi-mj

Figure 2: A generalised description of the Mask

1. If a color image is intended to be used as the image to generate textures and is being displayed, it's grayscale[1] equivalent is stored in memory.
2. An area equal to the size of the mask and centered on the haptic interface point is selected from the image.
3. An inversion operation is performed on this selected area to obtain the negative of the image. For example if an 8bit image is being displayed. The operation defined by 255-(value of pixel) is performed.
4. An element wise scalar multiplication is performed on the mask and the selected portion of the image.
5. The vectors resulting out of this multiplication are summed up to obtain a resultant vector.

[1]Though the conversion from RGB image to greyscale has no defined formula, we employ the commonly used formula Y = 0.3×R + 0.59×G + 0.11×B

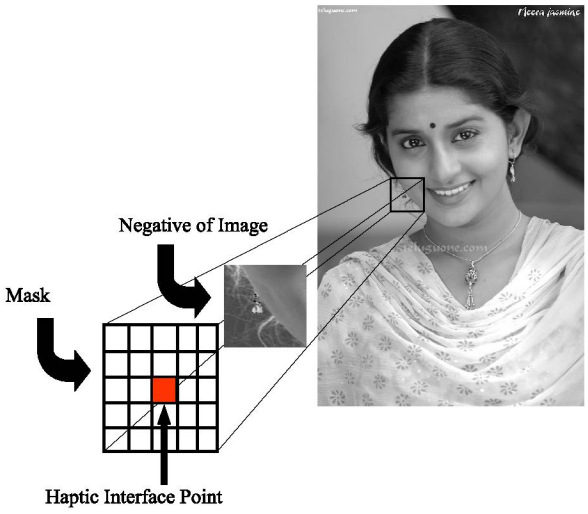


Figure 3: A mask overlaid on an image

An illustration of the process of application of the mask is shown in figure 3. The resultant vector is commanded as a force to the haptic device (The Phantom Omni from SensAble Technologies) after suitably scaling the forces to keep with the permissible range of the haptic device. The result of the above mentioned algorithm generates a vector pointing to the darker regions of the image. This effect is shown in figure 4. However if the user is exploring a region of the image with little or no features, the result of the mask operation will produce a zero force. This is illustrated in figure 5.

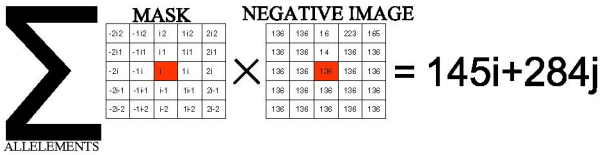


Figure 4: Result of Mask Multiplication over an area with a finite gradient

Dealing with the edges of images is a challenging problem, this is because the mask of size 5×5 cannot be overlaid on the pixel at location (1,1) without a portion of the mask extending beyond the image. We over come this problem suspending the algorithm when the user is exploring pixels that cause the mask to fall outside the boundaries of the image. It is seen that this does not affect the performance of the algorithm as the users tend to stay to the center of the image during a majority of the time spent exploring the object.

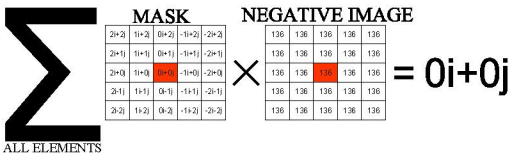


Figure 5: Result of Mask Multiplication over an area with zero gradient

5 Factors Affecting Performance

The performance of our method to generate textures is affected by three factors, namely mask size, image size and image contrast.In this section we attempt to elaborate on this.

5.1 Effect of Mask Size

The performance of the algorithm depends on the size of the mask. Here we can see two conflicting requirements for the design of the mask. Ideally we should generate forces for every feature on the image so as to render a much more immersive environments. To do this we would like to use a mask that is as small as possible. Following the nyquist sampling theorem for a feature of a width of 'p' pixels, any mask should have a maximum width of 'p/2' pixels.

The other factor which affects the mask size is the noise present in the image. Stochastic noise present in the image generates high frequency vibrations in the force rendered to the user. To generate a truly immersive experience, it is desired to keep the noise to a minimum. The size of the mask directly contributes to the amount of noise perceived by the user. Assuming a uniform distribution of noise in the image that is used, we can state that due to the nature of the mask, which contains larger multipliers on the edges, the distribution of noise power in the area operated on by the mask is as shown in figure 6. Since the multiplication operation is followed by an addition operation, the noise gets canceled out on successive additions. We can state that in the figure 6 the noise in row one mostly gets canceled by the power in row five. Similarly the noise in column one mostly gets canceled by column five. Hence the number of columns and rows determine the noise performance of the mask. For an $m \times m$ matrix, the signal to noise ratio can be expressed as.

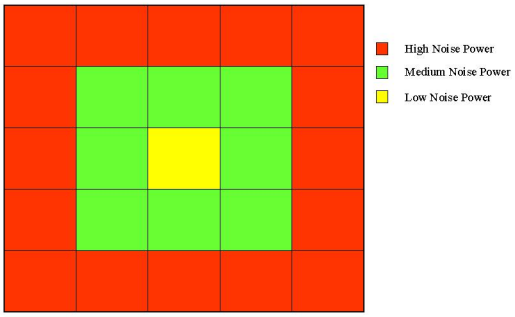
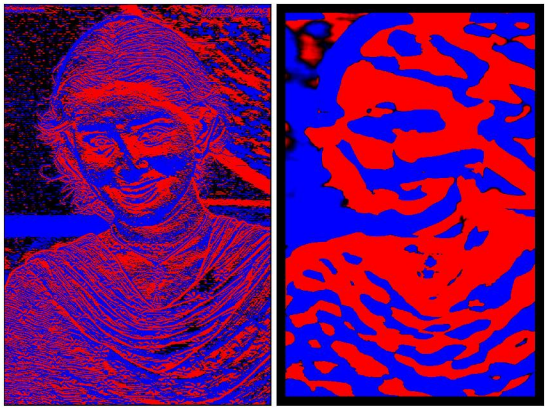


Figure 6: Result of Mask Multiplication over an area with zero gradient

$$SNR_{mask} = \sqrt{m}\, SNR_{original} \tag{1}$$

From equation 1 we can see that the amount of noise in the processing of the image is directly linked to the size of the mask. Though after a mask size of fifty the noise reduction achieved will only be 0.14 for a unit increase in mask size and rate of improvement in the SNR continues to fall.

To illustrate the effect of change in mask size we present the figures in 7 (Courtesy: Teluguone.com http://www.teluguone.com/) . The figures depict the forces generated by the haptic device as the user scans over the image. The red lines represent the forces produced in the positive 'y' direction and the blue lines represent the forces produced in the negative 'y' direction. The intensity of color shows the strength of the force produced. The figures only show the 'y' component of the forces generated. The original greyscale image is shown in figure 3.



(a) 5×5 (b) 65×65

Figure 7: Effect of mask size

At a mask size of 5×5 (figure 7(a)) we see that forces are produced as the user scans over every minute detail in the original picture. We also note the presence of noise in this picture, the upper left section of the image is grainy and shows that in this region the noise will cause serious deterioration of the haptic experience. The noise performance improves with a mask size of 17×17, however now some of the features in the picture are not perceivable. Finally with a mask size of 65×65 the noise performance is excellent, however the features in the image are indistinguishable and generating textures with a mask of this size is not advisable.

5.2 Effect of Image resolution

Unlike the effect of mask size, the effect of image size or resolution is more straightforward. This section will reason that larger the image size/resolution the better the performance of the algorithm. To reason this we consider photographs of the same object x_1, x_2, x_3 with equal aspect ratios and resolutions of $p1 \times q1$, $p2 \times q2$, $p3 \times q3$ such that $p1 > p2 > p3$ and $q1 > q2 > q3$. We assume noise power in an image 'x' is 't_x' watts and because x_1, x_2 and x_3 are images of the same object all three images will contain the same power. We now define a parameter r_x which defines the average noise per pixel. The expression for r_x now becomes:

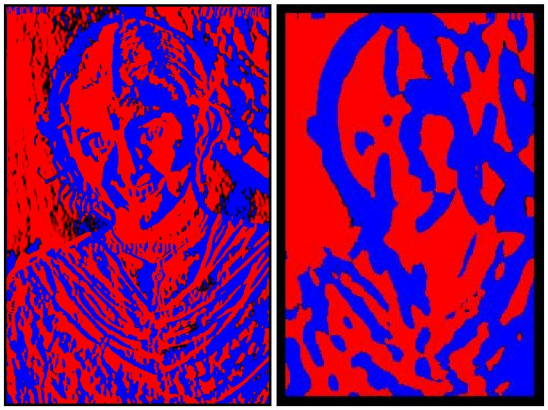
$$r_x = \frac{t_x}{p_x \times q_x} \tag{2}$$

The parameter r_x for images x_1, x_2, x_3 will follow the order $r_1 < r_2 < r_3$. We can see from the above explanation that the performance of the algorithm will be best in the image with highest resolution because it will have the minimum noise per pixel.

Another factor that arises out of the resolution of the image is the smallest feature in the image that can be felt. Section 5.1 describes

the effect size of the mask to the smallest feature that can be felt. The effect of resolution of the image is explained with the help of the following example. If there exists a feature in images x_1, x_2 and x_3 with a pixel width of l_1, l_2, l_3, the pixel width of the feature will follow the order $l_1 > l_2 > l_3$ due to the difference in resolution. Hence the size of the mask $m_x \times m_x$ that can be used to feel this feature in images x_1, x_2 and x_3 will follow the order $m_1 \times m_1 > m_2 \times m_2 > m_3 \times m_3$. This shows that in an image with a higher resolution, a larger mask can be used to feel the same feature as compared to an image with lower resolution. The larger mask size directly translates into improved performance since the noise immunity of the algorithm depends on the size of the mask.

To illustrate the change in performance of the algorithm with the image resolution we present images of three different resolutions in figure 8. The figures show the 'x' component of forces generated, with the blue color indicating forces in the negative 'x' direction and the red indicating forces in the positive 'x' direction. The intensity of the colors represent the magnitude of forces generated. The mask that was used to generate the forces from these images was of the size 17×17. It can clearly be seen that as the image resolution decreases the features that can be felt in the images decrease.



(a) 1000×1504 (b) 250×376

Figure 8: Effect of Image Resolution

5.3 Effect of Image Contrast

The contrast of an image serves to control the magnitude of forces are generated by the algorithm. Our algorithm responds to the average gradient present in the area of the mask. An increase in the contrast of the image increase the average gradient and hence the force calculated by the algorithm.

6 Discussion and conclusion

The algorithm presented in this paper describes a method of generating *tangible images* - haptic textures from images, specifically for haptic devices using point based interaction techniques. The paper also discusses the factors affecting the design of a mask for generating touchable images which include the mask size and the image size. The change in performance of the system for various mask and image resolutions is discussed in detail. This method presents a simple and efficient method of generating haptic textures which to the best knowledge of the authors has not been explored so far.

The selection of mask size strongly dependent on the application for which the algorithm is used. For example, while inspecting a small region of an image it is useful to use smaller masks as they will help in the perception of finer features. While for the perception a larger scene, like that of a scenery with mountains or lakes, larger masks will be better. In the case of user interfaces, it is useful to use smaller masks as these are capable of capturing the edges and slopes of widgets and icons.

Though we used Prewitt masks, other masks such as sobel, Roberts, and other first-derivative methods could very well suit the purpose. Since second-derivative methods are extremely sensitive to noise, it could be combined with smoothing operators such as gaussian and then used. Image embossing algorithms can also be employed to generate image which contain features that can be haptically rendered.

The noise present in images presents a major challenge to the efficient haptic rendering of forces. Future work will focus on the design of suitable filters to reduce the noise content of the rendered forces effectively increasing the realism of the technique. In digitally created images the noise performance of the algorithm is not a significant problem since such images contain very little noise.

Although we presented a simple and efficient algorithm for tangible images, the algorithm is yet to be evaluated compared to other existing similar techniques. It is to be investigated how different edge detection algorithms will allow the user to perceive the details in an image.

A logical extension of this work would be to create tangible video sequences. Video clips are often composed of multiple scenes from different camera locations. The discontinuity in switching from one scene to the next is a major problem, another problem is the fact that modern video compression results in the video content being very noisy, the algorithm fails to work adequately under these conditions.

References

[1] T. C. Aysal and K. E. Barner. Stochastic and deterministic models for haptic pseudo-textures. In *HAPTICS '06: Proc. of the Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS'06)*, page 71, Washington, DC, USA, 2006. IEEE Computer Society.

[2] B. R. Donald and F. Henle. Using haptic vector fields for animation motion control. In *Robotics and Automation, 2000.*, pages 3435–3443, Washington, DC, USA, 2000. IEEE Computer Society.

[3] J. P. Fritz and K. E. Barner. Stochastic models for haptic texture. *Proc. SPIE Int. Sym. on Intelligent Systems and Advanced Manufacturing, Telemanipulator and Telepresence Technologies III Conference*, November 1996.

[4] C.-H. Ho, C. Basdogan, and M. A. Srinivasan. Efficient point-based rendering techniques for haptic display of virtual objects. *Presence*, 8(5):477491, October 1999.

[5] Y. Ikei, W. K., and S. Fukuda. Texture presentation by vibratory tactile display-image based presentation of a tactile texture. In *Virtual Reality Annual International Symposium*, pages 199–205, 1997.

[6] M. Minsky, O. young Ming, O. Steele, J. Frederick P. Brooks, and M. Behensky. Feeling and seeing: issues in force display. *SI3D '90: Proc. of the 1990 symp. on Interactive 3D graphics*, pages 235–241, 1990.

[7] M. A. Otaduy, N. Jain, A. Sud, and M. C. Lin. Haptic display of interaction between textured models. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Courses*, page 133, New York, NY, USA, 2005. ACM Press.

[8] M. A. Otaduy and M. C. Lin. A perceptually-inspired force model for haptic texture rendering. In *APGV '04: Proc. of the 1st Symp. on Applied perception in graphics and visualization*, pages 123–126, New York, NY, USA, 2004. ACM Press.

[9] D. Pai, K. van den Doel, D. James, J. Lang, J. E. Lloyd, J. L. Richmond, and S. H. Yau. Scanning physical interaction behavior of 3d objects. In *Proc. of ACM SIGGRAPH*, pages 87–96, August 2001.

[10] J. Siira and D. K. Pai. Haptic texturing - a stochastic approach. *Proc. of the 1996 IEEE, Int. Conf on Robotics and Automation*, April 1996.