Auteursrechterlijke overeenkomst

Opdat de Universiteit Hasselt uw eindverhandeling wereldwijd kan reproduceren, vertalen en distribueren is uw akkoord voor deze overeenkomst noodzakelijk. Gelieve de tijd te nemen om deze overeenkomst door te nemen, de gevraagde informatie in te vullen (en de overeenkomst te ondertekenen en af te geven).

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling met

Titel: Implementation of haptic textures
Richting: 2de masterjaar in de informatica - Human Computer Interaction Jaar: 2009

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Ik ga akkoord,

VEREENOOGHE, Tom

Datum: 14.12.2009

Implementation of haptic textures

Tom Vereenooghe

promotor :
Prof. dr. Chris RAYMAEKERS

Eindverhandeling voorgedragen tot het bekomen van de graad
master in de informatica Human Computer Interaction

universiteit hasselt

transnationale UNIVERSITEIT LIMBURG

Implementation of Haptic Textures

Thesis proposition in order to gain the degree of master in computer science/ICT/knowledge engineering

Academic Year 2007-2009

Author:	Tom Vereenooghe (0422010)
Promotor:	Chris Raymaekers
Co-promotor:	Karin Coninx
Advisor:	Lode Vanacken

Abstract

This thesis provides an insight in haptic textures and their practical use. The thesis is divided in three parts. Part I is a Dutch summary for the Dutch readers. In Part II we will express the importance of computer haptics and situate where haptic textures reside. From a related work study meaningful thoughts about human perception of roughness are put forward. Next, six techniques from different research papers are presented. A conclusion will postulate rising questions and set an aim for the next part of this thesis. In Part III we look into the use of haptic textures in information visualization. We create our own application as a base for a user test. The user test should give us some answers about how people explore textures and the possibility of using them to convey data.

Acknowledgments

I would like to thank my promotor Chris Raymaekers and my advisor Lode Vanacken for their guidance and feedback. I was not always sure in which direction to proceed, but they gave me the insight I needed. Also the many papers that Lode sent me throughout the year were a good help.

Also I would like to thank Erik Vidholm and Ingela Nyström for their quick response and sending me a copy of their publication. A lot of interesting publications online are only available for purchase, so as a student I am glad that these authors are willing to help out.

Finally a word of thanks to my parents, who allowed me to pursue this study in the first place and who support me throughout the year.

Contents

Part I

Nederlandse samenvatting

CHAPTER 1

Nederlandstalige samenvatting

1.1 Inleiding tot computer haptics

De beste manier om haptische textures te beschrijven, is door ze te vergelijken met hun visuele gelijken. In computer graphics worden objecten voorgesteld door een aaneenschakeling van driehoeken, tot een zeker niveau van detail. Het berekenen van al die driehoeken kost CPU-tijd, dus om meer detail te specifiëren zonder de grafische hardware teveel te belasten worden textures gebruikt, afbeeldingen die op de driehoeken gekleefd worden.

In deze thesis is maken we een literaire studie om een algemeen inzicht te krijgen in haptische textures. Deze kennis zal verder gebruikt worden om te bestuderen hoe haptische textures gebruikt kunnen worden in informatievisualisatietoepassingen.

1.1.1 Definitie van haptische feedback

Het woord haptisch vind zijn oorsprong in het Griekse Haphe wat zoveel wil zeggen als 'met betrekking tot het gevoel', of in het woord Hapthesthai, wat 'contact' of 'aanraking' betekent.

Het menselijk gevoel kan opgedeeld worden in twee klassen: het tactiel of cutaan gevoel

en het kinetisch gevoel.

1.1.2 Toepassingen

Haptische systemen worden gebruikt in verschillende domeinen. We komen ze tegen in:

- Telerobotica
- de medische wereld, waar chirurgen d.m.v. haptische systemen operaties kunnen inoefenen
- rehabilitatie, waar patiënten begeleid worden m.b.v. haptische systemen
- vluchtsimulatoren
- de entertainmentindustrie, in game controllers
- CAD (Computer Aided Design), voor het virtueel modelleren van nieuwe producten

1.1.3 Hardware

Er zijn verschillende soorten haptische apparaten op de markt, afhankelijk van de toepassing zien ze er heel verschillend uit. In deze thesis maken we gebruik van de SensAble PHANToM, een van de meest gebruikte haptische apparaten.

De PHANToM is een desktoptoestel bestaande uit een 2-delige beweegbare arm met een stylus aan het einde. De gewrichten van de arm kunnen aangestuurd worden door motoren en zorgen voor de haptische feedback.

1.1.4 Basisprincipes van haptisch renderen

Om een vloeiend gevoel te verkrijgen moet het haptisch systeem een updatefrequentie van 1000Hz aanhouden. In tegenstelling tot de 25 à 30Hz voor grafische systemen.

Elk haptisch systeem moet twee zaken voorzien: detectie van botsingen en genereren van een antwoord op die botsingen. Een virtueel object, de probe genaamd, stelt de cursor voor die de gebruiker kan bewegen d.m.v. het haptisch apparaat. De detectie van botsingen gebeurt door te controleren of er een doorsnede bestaat tussen deze probe en een object uit de scene. Als dit het geval is, dan intersecteert de probe met een

object, en moet een kracht uitgeoefend worden om de probe terug te duwen. Zo wordt het gevoel van een hard oppervlak gegenereerd.

Er zijn twee methodes om dit te doen: penalty-gebaseerd en constraint-gebaseerd. Constraint gebaseerde methodes zijn over het algemeen beter, omdat de pointer hierbij niet door een object kan schieten. Constraint-gebaseerde methodes werken met een HIP of Haptisch InteractiePunt en een IHIP of Ideale HIP. De HIP komt overeen met de positie van de probe. De IHIP volgt deze, maar blijft steeds op het oppervlak van objecten, waar de HIP zich ook in objecten kan bevinden. De pointer wordt uiteraard uitgetekend op de positie van de IHIP. Soms levert dit volgen van de IHIP geen heel correct gevoel op, daarvoor werd force shading toegevoegd.

1.2 Menselijke perceptie

Dit hoofdstuk handelt over het menselijke gevoelssysteem en hoe textuur waargenomen wordt. We geven tevens uitleg bij de invloed van het JND of Just Noticeable Difference (= net merkbaar verschil). Tenslotte bespreken we hoe voelen met je vingers verschilt van voelen door een probe.

1.2.1 Het gevoelssysteem

Het gevoelssysteem staat naast het uitwendig gevoel van aanraking en proprioceptie, dat is het vermogen om de stand en positie van ledematen waar te nemen, ook in voor spierbewegingen, gezichtsuitdrukkingen en het inwendig gevoel, waartoe buikpijn behoort. Aanraking en proprioceptie zijn voor ons het belangrijkst omdat zij rechtstreeks beïnvloed worden door haptische feedback. We kunnen ze onderverdelen in tactiel en kinesthetisch gevoel. Tactiel slaat op wat we voelen door onze huid. Kinesthetisch slaat op het gevoel van spierbeweging en gewrichtspositie.

Er zijn vijf schalen om tactiele oppervlakken onder te classeren: ruw vs. glad, zacht vs. hard, kleverig vs. slipperig, koud vs. warm en elastisch vs. kneedbaar. Ruw/glad en zacht/hard hebben de grootste bijdrage aan het gevoel van de meeste mensen.

Het JND of net merkbaar verschil staat voor de het kleinste verschil waarmee een stimulus moet worden veranderd om een merkbaar verschil in gevoel te verkrijgen. De JND's voor onze armgewrichten zijn experimenteel bepaald door Tan et al.

- Vinger: 2,5°

- Pols: 2°
- Elleboog: 2°
- Schouder: 0,8°

1.2.2 Voelen met een stylus

De huid in onze vingers heeft vele sensoren om gevoel op te vangen. Wanneer men voelt met een stylus, dan is er maar één raakpunt waarlangs het gevoel overgebracht moet worden. Meestal wordt dit raakpunt, het eindpunt van de stylus, als een bol geïmplementeerd. In dit geval spelen er enkele elementen mee in het resulterende gevoel. De grootte van de bol bepaalt hoe gemakkelijk je over grote en kleine bultjes beweegt. De grootte van de bultjes draagt hier eveneens toe bij. De ruimte tussen de bultjes en de snelheid waarmee je beweegt hebben ook invloed op het gevoel van ruwheid of weerstand.

1.3 Haptische textures

Dit hoofdstuk beschrijft verschillende technieken om oppervlaktetextuur te simuleren. Om ze met elkaar te kunnen vergelijken wordt een classificatie opgesteld op basis van eigenschappen die haptische textures kunnen bezitten.

1.3.1 Classificatie

Een eerste onderscheid wordt gemaakt tussen vibratie- en krachtterugkoppeling. Een tweede onderscheid werkt volgens de richting waarin de krachten uitgeoefend worden. We onderscheiden tangentiële en normale krachten. Een derde onderscheid betreft statische of afbeeldingsgebaseerde textures tegenover dynamische of procedurele textures. Statische textures zijn voorgedefinieerd en niet aanpasbaar in real time. Dynamische textures worden wel in real time gegenereerd, met het voordeel dat objecten van vorm mogen veranderen terwijl de textures nog steeds voelbaar blijven. Tenslotte maken we onderscheid tussen realistische en niet-realistische textures op basis van het gevoel.

1.3.2 Minsky's Sandpaper system

Frequent beschreven als de eerste echte implementatie van haptic textures. Maakt gebruik van laterale veerkrachten die de pointer naar een bepaald punt toe trekken. Geeft een beetje het gevoel van neerwaartse krachten. De veerkracht wordt berekend op basis van een hoogtemap van het oppervlak.

1.3.3 Stick-slip frictie model

Een veerkracht loodrecht op het oppervlak wordt gebruikt om een hard oppervlak te modelleren. Daarnaast worden tangentiële krachten gebruikt om een gevoel van frictie te genereren. Het oppervlak wordt beschouwd als vlak met putjes, waarbij de pointer steeds in de putjes blijft haperen, wat de tangentiële frictie genereert. Deze heeft zowel een statische als een dynamische component en is afhankelijk van de diepte van de putjes alsook het aantal en de spreiding.

1.3.4 Force mapping

Force mapping is gebaseerd op de bumpmapping techniek die gebruikt wordt in computer graphics. Normaalvectoren worden gespecificeerd voor elke pixel van een oppervlak. Op elk punt wordt de uit te oefenen kracht dan berekend op basis van deze normaalvectoren.

1.3.5 Height field rendering

Height fields zijn in principe hetzelfde als force maps, maar waarbij de vectoren steeds loodrecht op het draagvlak van de texture staan.

1.3.6 Haptic shading

Haptic shading berust op procedurele berekening van de uit te oefenen kracht op basis van een botsingsmodel dat contactinformatie voorziet en materiaaleigenschappen zoals frictie en stijfheid. De haptische shader voert zijn berekeningen uit na de botsing en telt de berekende krachten daarbij op. Enkel kleine wijzigingen zijn hierdoor mogelijk.

1.3.7 Tactiele feedback

Voor tactiele feedback is aparte hardware vereist. Tactiel vereist immers dat meer sensoren in de huid geprikkeld worden. Dit wordt bekomen met een plaatje gelijkaardig aan een brailleregel, waarbij verschillende staafjes naar boven kunnen komen om tegen de vinger te drukken en zo een gevoel van textuur te bekomen.

1.4 Conclusie

In dit eerste deel hebben we gemotiveerd waar haptic textures nuttig kunnen zijn in moderne toepassingen. We gaven een introductie tot computer haptics en hebben de verschillende methoden om haptische textures te genereren besproken.

1.5 Haptische textures in informatievisualisatie

Er bestaan vele manieren om informatie grafisch voor te stellen, maar waarom zouden we ons beperken tot ons visueel vermogen? Toevoeging van haptics als manier om data te interpreteren heeft twee voordelen. In de eerste plaats kunnen we zo toegang verschaffen tot de data aan visueel gehandicapten. In de tweede plaats kunnen we met haptics een extra informatielaag toevoegen aan visuele data, ter ondersteuning of ter uitbreiding.

De manier waarop we haptische data verwerken verschilt van de manier waarop we visuele informatie verwerken. Visueel kunnen we veel informatie in één oogopslag verwerken. In tegenstelling tot deze parallelle perceptie, is gevoel sequentieel, we kunnen maar op één plek te gelijk voelen. Dit gegeven speelt ook op bij het gebruik van legendes bij data. Visueel kunnen we daar makkelijk naar refereren, maar gevoelsmatig moeten we telkens opnieuw gaan voelen aan de verschillende oppervlakken. Het gebruik van haptics voor informatie met een sequentiële aard kan hier in tegemoetkomen.

1.5.1 Height field textures voor data representative

We focussen verder op het gebruik van height field textures en we stellen enkele onderzoeksvragen voorop.

Kunnen gebruikers data associëren met textuur en kunnen ze de juiste verbanden leggen

om de relaties in de data te begrijpen?

Als we grid-texturen gebruiken, hoe moet de spatiëring tussen de lijnen dan zijn om zo goed mogelijk herkenbaar te zijn?

Kunnen we willekeurige textures gebruiken om kwalitatieve data, zoals iconen, voor te stellen of halen we voordeel uit het samenstellen van een aangepaste reeks textures om de herkenbaarheid te bevorderen?

Op welke manier verkennen gebruikers haptische data?

Gebaseerd op deze vragen maken we een eigen applicatie om haptische gegevens te renderen. We houden tevens een gebruikerstest om enkele antwoorden te achterhalen.

1.6 Het gebruikersexperiment

Dit hoofdstuk beschrijft de applicatie die we geschreven hebben om scenes met haptische data weer te geven. En daarnaast bespreken we ook de uitgevoerde gebruikerstest met de bekomen resultaten.

1.6.1 De applicatie

De applicatie is opgebouwd om twee soorten data te tonen: iconen en kaarten. Iconen in de zin van vierkante oppervlakjes ter representatie van kwalitatieve data, en kaarten worden gebruikt om alle soorten data op weer te geven. Wij beperken ons tot dichtheden.

De applicatie is geschreven in C++ en gebaseerd op H3DAPI 2.0, een open source framework dat OpenGL combineert met X3D. Het zorgt voor de afhandeling van haptische events en is apparaatonafhankelijk. X3D is een bestandsformaat dat gebruikt wordt om 3D scenes en objecten te representeren. Python scripts kunnen gebruikt worden om interactie met de scenegraph mogelijk te maken. Verder maken we gebruik van Qt v4.2.2 voor de user interface.

De applicatie moet naast het tonen van de 3D scene in een OpenGL window, ook 2 scenario's kunnen aflopen. In het eerste scenario tonen we de gebruiker telkens 3 iconen met een bijhorend woord. De gebruiker moet deze combinaties leren en in de volgende stap de iconen weer bij het juiste woord plaatsen. De iconen worden daarvoor in willekeurig volgorde opnieuw getoond. We doen dit met een aantal scenes

met willekeurige haptische textures op de iconen, en een aantal met eigen ontworpen textures, waarvan we vermoeden dat ze beter herkenbaar zijn. In het tweede scenario tonen we de gebruiker telkens een kaart met gekleurde regio's. De regio's hebben weerom een haptische textuur en de gebruiker wordt gevraagd om de regio's te ordenen op gevoelde dichtheid.

1.6.2 De gebruikerstest

Voor de gebruikerstest maken we gebruik van onze geschreven applicatie die draait op een desktop PC verbonden met een PHANToM haptisch apparaat.

De deelnemers zijn vrijwilligers uit het EDM onderzoekscentrum aan de Universiteit.

De test startte met een korte introductie en instructies voor de test. De test zelf bestond uit de twee scenario's zoals we ze net beschreven hebben. Tussendoor en erna volgde nog een korte vragenlijst.

1.6.3 Resultaten

De feedback van onze test komt van drie kanten: de resultaten van de gebruiker op de test die naar een bestand weggeschreven zijn, de antwoorden van de gebruikers op een vragenlijst en onze observaties van de test.

Uit de resultaten van de gebruikers halen we de volgende informatie.
Gebruikers maakten lichtjes minder fouten bij de zelf ontworpen iconen, dus deze lijken beter herkenbaar. Voor de kaarten met dichtheden, zijn grid-texturen het best onderscheidbaar, en wel deze waar de spatiëring tussen de lijnen het kleinst is. Een statistische analyse werd uitgevoerd met de bijhorende ANOVA tests en geeft echter aan dat de verschillen niet significant zijn. Een uitgebreidere user test moet uitsluitsel brengen.

Uit de vragenlijsten trekken we de volgende conclusies.
8 van de 10 gebruikers vond de tweede serie iconen, zijnde de zelf ontworpen iconen, beter herkenbaar dan de eerste reeks. 7 van de 10 gebruikers vond ook dat er meer logica zat in de tweede reeks. Wat betreft de kaarten vonden 4 van de 10 gebruikers de eerste kaart (grid patroon met grote onderlinge verschillen) het makkelijkst om de dichtheden te onderscheiden. Nog eens 4 gebruikers vonden de tweede kaart (grid patroon met kleine onderline verschillen) makkelijker. En 2 gebruikers vonden ze beide even gemakkelijk of moeilijk.

Uit onze eigen observaties trekken we de volgende conclusies.
Alle gebruikers maken horizontale en verticale bewegingen over de textuur. Daarnaast worden ook diagonale en cirkelvormige bewegingen gebruikt. Eén gebruiker gaf aan zijn manier van voelen aan te passen aan wat hij voelde. Over het algemeen kiezen gebruikers een bepaalde voelstrategie en blijven deze gebruiken doorheen de taken. Ze zijn niet snel geneigd om van strategie te veranderen. Aanvankelijk lijken gebruikers te zoeken naar regelmaat of frequentie van het textuurpatroon om de iconen te identificeren, er wordt niet meteen gezocht naar vormen. Een aantal gebruikers probeert zich een mentaal beeld te vormen van de gevoelde textuur. Hier wordt enige tijd voor genomen. Bij de tweede reeks, de zelf ontworpen iconen, hadden 4 gebruikers snel door dat er een vorm te voelen was. Zij vonden ook de link tussen de textuur en het daarmee geassocieerde woord. Bij lijnpatronen gaan de meeste gebruikers de lijnen kruisen. Slechts één gebruiker ging langs de lijn voelen. Het is mogelijk dat als users vooraf een visueel beeld krijgen van de texturen, dat ze dan een andere strategie zouden gebruiken om de iconen te identificeren. Met voorkennis weten ze immers wat ze kunnen verwachten en waarnaar ze moeten zoeken ter identificatie. Dit zou een snellere en nauwkeurigere identificatie mogelijk kunnen maken. Bij de kaarten deden de gebruikers er significant langer over om de stippatronen te onderscheiden. De gridpatronen vormden dan weer geen probleem. De algemeen gebruikte strategie was hier om eerst een algemeen gevoel te verkrijgen van alle vlakken, en dan 2 aan 2 te gaan vergelijken om een rangorde te kunnen opstellen.

Part II

Literary study

CHAPTER 2

Introduction

With the introduction of haptic technology came the possibility to actually touch virtual objects and surfaces that are presented on a computer screen. Following vision and sound, touch quickly earns its place within modern multimedia. Various haptic devices have been developed along with various algorithms to generate a force feedback response. Research has been and is still being done in the fields of virtual reality, telerobotics, medical simulations, computer aided design (CAD) and entertainment. Computer haptics has many applications and with the development of increasingly realistic haptic simulations and a broader range of haptic devices in different price ranges and for different purposes, a broader audience can be reached and the haptic research domain can only benefit.

Recently more work is being done in the field of haptic textures. Textures provide surface detail and make it possible to identify surfaces. Because different surfaces have different characteristics, they should therefore feel different and preferably as realistic as possible. In this thesis we will conduct a literary study to provide a general insight in haptic textures. This will constitute Part II of the thesis. In Part III we will take a look at how haptic textures can be applied in the domain of information visualization. We create our own sample information *haptization* application and use that as a base to do a user test. The results should provide us with a deeper insight in the way people explore a haptic environment in order to extract information.

We start this work with an introduction to computer haptics itself in Chapter 3. A commonly excepted definition of haptic feedback is given, followed by an overview of current applications of computer haptics and a short description of the necessary hardware. The introduction is concluded with an explanation of the basic haptic rendering techniques.

In Chapter 4 the human sensory system is studied to gain insight in our perception of touch and our perception of surface textures. We make a distinction between tactile and kinaesthetic sense and discuss the properties of surface textures. This is followed by an explanation of Just Noticeable Difference (JND). Finally we examine how feeling with our bare hands differs from feeling through a probe.

Chapter 5 describes different techniques that can be used to simulate haptic textures. A classification is given to categorize the techniques here discussed and the chapter is concluded with a comparison of these techniques based on the given classification.

Chapter 6 closes Part II with a conclusion of our acquired knowledge.

In Chapter 7 we discuss the purpose of using haptic textures for information 'visualization'. We study the differences between visual and haptic perception and set some questions on using height field textures to represent data. These questions are the starting point for the user experiment in Chapter 8.

Chapter 8 describes the application we implemented and the set-up of the user test. It continues with a description of the test itself and finishes with an explanation of the results.

CHAPTER 3

Introduction to computer haptics

Force feedback can add realism to the experience of interacting with a virtual environment. Shapes can be felt and object properties can be distinguished because the user can *touch* the virtual objects. Also grabbing objects and manipulating them using your hands directly whilst receiving haptic cues is more intuitive than doing so through a device without force feedback. Force feedback is first and foremost concerned with simulating forces caused by touching objects. Feeling materials and surfaces of objects is typically restricted to hardness and friction coefficients. It is however possible to generate haptic textures that represent more elaborate surface properties.

Probably the best way to describe haptic textures is to compare them with their visual counterpart. In computer graphics, objects are generally represented by triangle meshes up to a certain level of detail. Rendering these triangles takes up computation time, so the more triangles there are in a scene, the more time the system needs to render it. To specify further visual detail without increasing the load on the graphics hardware too much, textures are used. Visual textures are simply images drawn on top of the triangles, thus the object surfaces. To some extent haptics work the same way. Objects in the scene are represented by geometric surfaces which the haptic device cannot penetrate. This allows the user to feel the shape of the object, but not the roughness or texture of the surfaces. To render further detail without having to increase the number of geometric surfaces used, haptic textures are introduced.

The purpose of this thesis is to conduct a literary study to provide a general insight in haptic textures. This knowledge will then be used to explore how haptic textures can be used to represent data in information visualization applications.

In this introduction the terms *haptics* and *haptic textures* have been used without explaining them. Therefore the next section gives a definition of *haptic feedback*. In order to show why haptic textures are of interest for the industry, section 3.2 gives an overview of past and current applications of machine haptics. Haptic machines enable the user to touch and feel virtual objects and manipulate them in a natural way. The specific hardware that is needed for this purpose is explained in section 3.3, while section 3.4 gives a general introduction to haptic rendering.

3.1 Definition of haptic feedback

The word haptic originates from the Greek, either from the word *Haphe* ($\alpha\phi\eta$) which means 'pertaining to the sense of touch', or possibly from the word *Hapthesthai* ($\alpha\pi\tau\epsilon\sigma\theta\alpha\iota$) which means 'contact' or 'touch' [wik08c]. Touch itself is actually a combined term for several senses. Pressure, shape, softness, temperature and pain, including tickle and itch, are all sensed through touch. Touch is part of the human somatosensory system. Besides touch, the somatosensory system is concerned with proprioception, which is the sensation of muscle movement and joint position that allows us to know where e.g. our hand is located, even when we close our eyes. The somatosensory system is also concerned with movement, facial expressions and visceral senses, which have to do with the senses of the inner body such as stomach aches [wik08d].

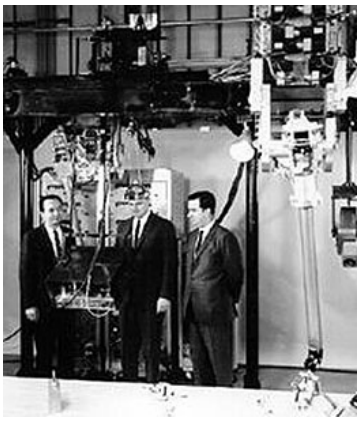
Another classification of human sense divides sense in two classes: *tactile* or *cutaneous* and *kinetic* or *kinaesthetic* sense [McG02]. Tactile sense is perceived through the skin and is responsible for the feeling of touch as just described. Kinetic sense corresponds to proprioception.

In Human Computer Interaction haptic devices are used to augment virtual environments with a sense of touch. These haptic devices both pass on the actions of the user to the computer system and generate a force feedback response when the user touches an object. With the appropriate feedback responses the user can distinguish between hard and soft surfaces, shapes and surface properties and weights when grabbing an object. The haptic feedback is combined with visual feedback to enhance the immersion in the virtual environment.

3.2 Applications

At present time haptic systems are applied in several domains. Often people relate the term haptics to virtual environments, but actually the earliest systems that had integrated haptics were developed before virtual reality even existed. We are talking about telerobotics, developed in the 1950s and 1960s [Bur96].

In telerobotics a person operates some kind of robotic arm, called the *slave device*, through manipulation of a typically multi-degree-of-freedom input device, called the *master device*. The master may look like the robotic arm that is being manipulated through it, but it may just as well be a regular joystick. In early systems the slave was mechanically connected to the master, and there was no mention of haptic feedback. But this telerobotics approach clearly was a point of interest in high risk environments when working with dangerous chemicals as done at Argonne National Laboratory [Figure 3.1(a)] and in nuclear reactors. In 1954 a system using electrical servo actuators was developed by Goertz [GT54]. This set-up could generate feedback forces on the master device allowing the user to feel as if he was operating the slave device directly. Telerobotics are still used when users need to execute operations in dangerous environments or involving dangerous substances. An example in another domain is given by Canadarm [Dea06], the robotic arm on the Space Shuttle, which is used for assembly and repair operations in space [Figure 3.1(b)]. Along with providing a general,



(a) Master-slave manipulators. Photo courtesy of [arg49]

(b) Canadarm. Photo courtesy of [Dea06]

Figure 3.1: Telerobotics

abstracted input interface to the teleoperated slave device and force feedback to the master, comes the possibility to scale the physical movements that the user performs. This allows for manipulation of nanomaterials [Bur96].

A second haptic domain concerns medical applications. This is a domain where haptics can really prove their value. Medical surgery is a very precise job, to say the least, and lots of things can go wrong. Evidently people are working hard to reduce the risks as much as possible. Students are prepared thoroughly before they get out on the workfield, and even then, as surgeons, they prepare for and train the procedures they will need to perform in critical operations. Haptic computer simulations provide a great support when determining a diagnosis of a patient's illness on 3D models. Furthermore it is possible to simulate a complete surgery in which the user can see what is happening on a screen and feel through haptic feedback the material he is touching or cutting. This allows for decent training in a safe environment before operating on real persons. In addition Bethea [BOK+04] describes a robotic surgical system where the surgeon can perform surgery by operating the haptic input devices of the robot. The robot itself operates on the patient, performing the actions of the surgeon.
Another application within the medical domain is found in dental care. It is basically the same as for the surgical field. Haptic feedback is used in analysing 3D models and in training procedures [KHPH05][Por07].
In rehabilitation it is not the doctor, but the patient who profits from the use of a haptic interface. Haptic devices can for instance be used in the rehabilitation process for patients with MS, who often suffer from muscle weakness and loss of coordination, and patients recovering from a stroke [FAG+08]. For this purpose, a force feedback device is used in combination with a computer program that lets the patient execute simple tasks like following the path of a line. At first the haptic device exerts force on the hand of the patient, guiding it along the way. Along time the patient hopefully progresses and the haptic device will be set to exert less force guiding the patients' hand. This way the patient learns to control his/her muscles again.
A third domain where force feedback is used is in flight simulators, shipping simulators and military simulators, where pneumatic/hydraulic actuators are used to simulate realistic movements of an artificial control cabin [Dav05].
A fourth and widely familiar domain is in the entertainment industry. Game controllers nowadays use force feedback to enhance the playing experience. Input devices in arcade halls are designed to resemble the real world objects, providing realistic input control and force feedback. This is particularly done for games in the racing genre. Novint [nov08] is a company that develops its own haptic device, the Novint Falcon, and offers a range of computer games that go with it.
To conclude we present the applications in the CAD [wik08b] (Computer Aided Design) domain. In CAD, computer technology is used to aid in product development and prototyping of new product designs. RJ Studios Inc. [rjS07] is an example of a product

development service company. One part of their job is to sculpt designs in 3D for their clients. Quoting [ST01]:

> It takes you eighty hours to sculpt a model. Your client approves but now you need a production version that is exactly 7% larger than the original. You could start sculpting from scratch... or you could do what RJ Studios did.

With a CAD solution the model can be automatically scaled by the computer software and adjusted in a 3D environment using a haptic input device. Thus the power of haptic technology lies in providing touch feedback, enabling the user to manipulate digital 3D models in a way they are used to in real life.

3.3 Hardware

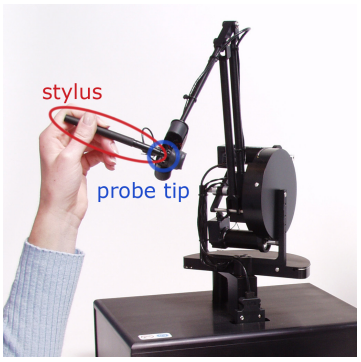
Different application domains use different kinds of haptic devices. The ones used in medical training systems are different from the ones used for desktop applications. As an illustration some different devices are shown in figure 3.2. In the remainder of this thesis we will use the SensAble PHANToM haptic device [ST08a], one of the widest used haptic devices on the market, hence a short introduction is in it's place.

The PHANToM device range is developed from the point of view that a useful desktop device must satisfy 3 requirements [Che99]:

- it must allow the user to move his or her hand freely in a reasonably unrestricted working volume in translation and in rotation on the desktop;
- it must provide a sense of touch;
- it must possess six active degrees of freedom (6 DOF).

To this purpose the PHANToM consists of a two limb movable arm with a stylus attached at the end [Figure 3.2(b)]. The first limb is attached to the base by a revolute joint. Between this joint and the base is another rotational section that allows the arm to rotate around a vertical axis. At the other end the first limb is attached to the second limb by another revolute joint. At the end of the second limb, a stylus is attached some kind of ball-socket joint to assure a comfortable way of holding the stylus in your hand. You can compare it to inverse kinematics [Lan98] in the computer animation domain, where the stylus tip is the end effector.

(a) Novint Falcon. Photo courtesy of [nov08]



(b) SensAble PHANToM. Photo courtesy of [ST08a]

(c) Force feedback steering wheel

(d) Flight simulator

(e) Xitact Instrument haptic port for surgery training. Photo courtesy of [men08]

(f) Immersion CyberGrasp Exoskeleton. Photo courtesy of [imm08]

Figure 3.2: Haptic devices

The PHANToM exists in three models: a Premium version, a Desktop version and an Omni version. The difference between the three is that the Premium version delivers higher fidelity, higher precision, stronger forces and lower friction than the Desktop version, and the same comparison applies for the Desktop and the Omni version [ST08b]. Furthermore the Premium model comes in three classes: 1.0, 1.5 and 3.0. They all provide 3 degrees of freedom (3DOF), but the 1.5 and 3.0 are also available in 6DOF. The difference between these three classes lies in the range of motion. The 1.0 class supports hand movement, pivoting at the wrist, the 1.5 class supports lower arm movement, pivoting at the elbow, and the 3.0 class supports full arm movement, pivoting at the shoulder [ST06].
For more technical specifications, check out the SensAble product overview [ST08a].

3.4 Haptic Rendering

In this section, some general concepts of haptic rendering will be explained. First the requirements for creating a natural haptic experience will be given, followed by an explanation of how the force feedback is computed.

3.4.1 Description of the haptic set-up

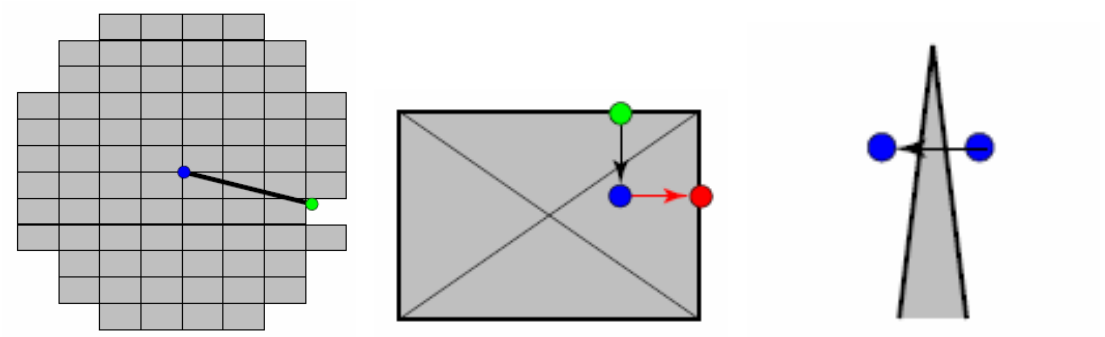
The set-up needed to render a haptic environment consists of a regular desktop PC with a connected haptic device, like the PHANToM discussed in section 3.3. On the software side a graphical framework, such as OpenGL [SWND05] or OpenSG [RVB02], is used to render the visual scene and a haptic library, such as GHOST [ST00] or HAL [hal], is used to address the haptic feedback device.

To simulate a fluent, interactive virtual environment, the current graphic systems render images at an update rate of typically 25 or 30Hz. The haptic component requires a much higher update rate. In order to have a satisfying experience of touch and feel and to achieve stability, an update rate of at least 1000Hz should be maintained [HBS99]. For this reason the system is split up into two asynchronously running components: the *graphics loop* and the *haptic loop*. This decoupling allows both loops to run on a separate dedicated computer, communicating via a network connection [MRF+96]. It makes the system configuration more flexible and results in a better performance.

3.4.2 Haptic interaction basics

Every haptic system needs to address two main issues: *collision detection* and *collision response* [HBS99]. The system needs to detect when a collision occurs between the probe, the virtual object representing the cursor which the user manipulates through the PHANToM, and the objects in the scene. The probe can have any shape you like, but the most common shapes used in literature are a sphere or a stylus with a (spherical) tip. A stylus makes for a better mapping between the virtual and the real world because it also gives visual feedback on how you are holding the real stylus. With a stylus shaped probe, collisions between the tip as well as the side of the probe can be simulated, generating multiple contact points. This is called a *ray-based* technique and is further described in [BHS97]. The other technique, where only the probe tip is used in collisions, is called *point-based*. In the remainder of this section, only point-based techniques will be addressed.

After detecting collision, a response force is generated proportional to the depth of penetration of the probe. An easy way of calculating this force is using Hooke's Law: $F = kx$ where x is the penetration depth and k is the spring constant. The penetration depth is simply the shortest distance between the penetrated probe tip and the object's surface. The method just described is what we call a *penalty-based method*.



(a) Uncertainty of exterior surface association

(b) Force discontinuities

(c) Pop-through thin objects

Figure 3.3: Drawbacks of penalty-based methods. Images courtesy of [RKK97]

3.4.3 From penalty-based to constraint-based methods

Penalty-based methods, being the simplest approach for generating collision response, work well on simple geometries like planes or spheres, but they have a number of drawbacks when it comes to more complex objects composed of a lot of polygons

— usually triangles are used as a building block for 3D models. When an object is composed of smaller primitives of which some are completely surrounded by others, it isn't clear which exterior surface should be associated with the internal volumes. This is illustrated in Figure 3.3(a). A second deficit occurs when the probe tip penetrates the object far enough that it is now closer to another surface than the surface it entered. The resultant force pushes towards this second closer surface, instead of the correct entry surface, as seen in Figure 3.3(b). A third drawback is seen with thin objects. Here the probe is pushed through the object either before a force could be generated or either the previous situation occurs where the probe penetrates so far that it is now closer to the opposite surface, causing it to jump through the object, as illustrated in Figure 3.3(c).

To overcome the limitations of penalty-based methods, *constraint-based methods* were introduced by Zilles and Salisbury [ZS95]. Again interaction with objects happens through the endpoint of a probe which we will now call the *HIP* for *Haptic Interface Point*. Note that in penalty-based methods the HIP actually penetrates the object before a force is generated to expel it. Assuming the HIP is visually represented by a massless sphere to show the user where his cursor is located, this yields an undesirable effect, for the sphere will penetrate what is supposed to be a solid object. This contradicts with the expectations of the user, diminishing the intuitiveness of the interaction and possibly evoking frustration with the user. Therefore a second point is introduced to represent the visual cursor. In literature this point is referred to as the proxy, the god-object point, the surface contact point (SCP), the Ideal Haptic Interface Point (IHIP). Possibly more names exist, but we will stick to *IHIP* as it is used by Ho et al. [HBS99] When no collisions occur the IHIP is identical to the HIP, but when the HIP penetrates an object, the IHIP is constrained to the surface, as shown in Figure 3.4.

The IHIP updates its position by moving directly towards the HIP in a straight line. This is only possible when the HIP is not inside an object. Otherwise a surface is blocking the IHIP's path and the IHIP is constrained to this surface. But it may still be able to reduce the distance to the HIP by moving along the surface. It does so until no further decrease in distance is possible. Notice that this effectively solves the exterior surface, the force discontinuity and pop-through problems, because the IHIP remains on the correct surface of the object.
However, a discontinuity problem arises when crossing from one edge to a neighbouring edge. Figure 3.4 shows this discontinuity in moving the HIP at time t+1 to its new location at time t+2. It's a small displacement for the HIP, but it crosses an edge which causes the IHIP to make a jump to its new position. This discontinuity is also

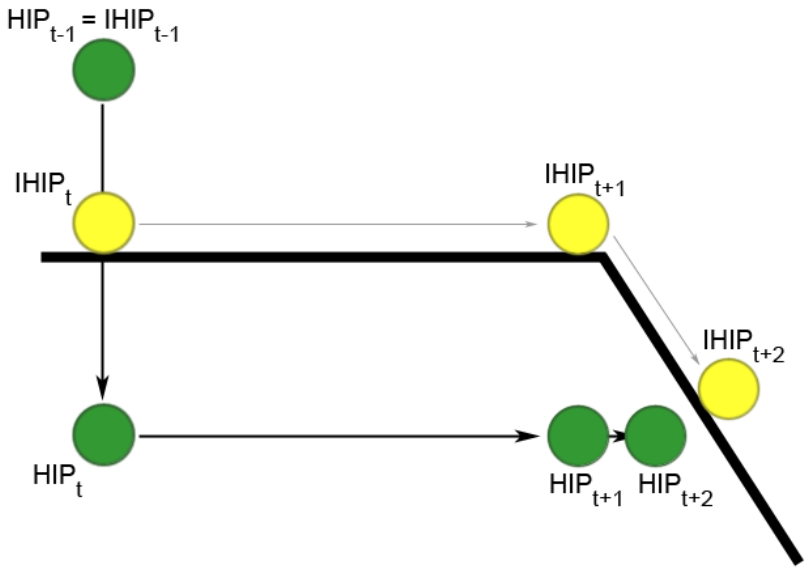


Figure 3.4: Correlation between HIP and IHIP. When the HIP enters the object, the IHIP is constrained to the surface.

illustrated in Figure 3.5 on the left. Shown on the right is the solution to this problem: *Force shading* [RKK97].

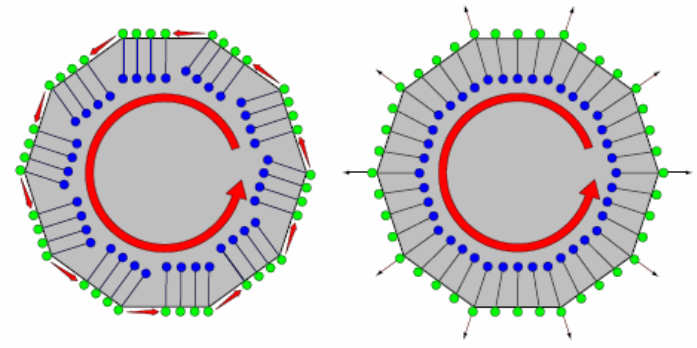
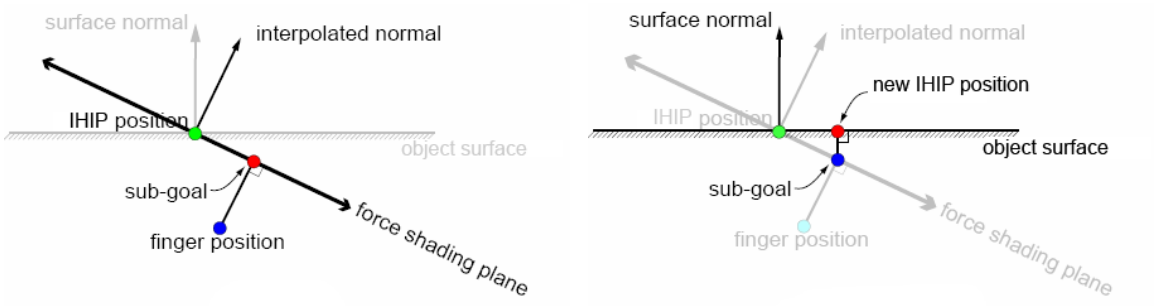


Figure 3.5: Flat (left) vs. Force shaded (right) surface. Image courtesy of [RKK97]

3.4.4 Force shading

Shown in Figure 3.5 on the left, the surface normals are used to determine the position of the IHIP. In contrast, the force shading approach, shown on the right, uses local interpolated normals, calculated from the vertices of the polygon to determine the location of the IHIP.

The update process consists of two steps. In the first step, see Figure 3.6(a), a force



(a) Pass 1

(b) Pass 2

Figure 3.6: Two pass force shading. Image adapted from [RKK97]

shading plane is constructed perpendicular to the interpolated normal at the current position of the IHIP. The finger position or HIP is now projected onto this plane as a subgoal. In the next step, see Figure 3.6(b), the subgoal is projected onto the original object surface and marks the new position for the IHIP corresponding to the current finger position. If the subgoal is above the object surface after the first step, then it is first projected back onto the nearest object surface. The effect of force shading is shown in Figure 3.5 on the right, where the inner dots represent the actual finger positions or HIPs and the outer dots represent the corresponding IHIPs.

3.5 Conclusion

In this chapter an introduction to computer haptics was given. We defined haptics as 'pertaining to the sense of touch' and clarified why haptics and more specifically haptic textures are a valuable addition for the industry. To illustrate this, examples were given of existing applications. Next some hardware that is able to generate haptic feedback response was presented and particular attention was given to the SensAble PHANToM [ST08a] as it is used in many research projects including this thesis. Before going into haptic textures in the remainder of this work, we needed to know something about haptic rendering itself and thus penalty-based methods were explained as the most basic haptic rendering method. We saw that these methods had a number of drawbacks and showed how constraint-based methods in combination with force shading were introduced to overcome these drawbacks.

The hardware and the corresponding implementation of collision response and haptic feedback alone do not define a realistic perception, they just generate forces. In order to know how we can make these forces feel like realistic objects and surfaces, we need to

take a look at the way we humans perceive those forces and textures. To that purpose, the next chapter studies the human sensory system and the way touch as we know it differs from touch through a probe as is the case with the PHANToM [ST08a] device.

CHAPTER 4

Human perception

When studying haptic textures it is important to consider how the human tactile system works and how textures are perceived. In this chapter we will take a look at the major parts of the human sensory system: the *tactile* and the *kinaesthetic* sense. We discuss the properties that are attributed to surfaces and objects and distinguish five perceptual scales. Then the influence of JND or Just Noticeable Difference is explained. Finally we discuss how feeling with our bare fingers differs from feeling through a probe, which is the case when using a PHANToM [ST08a] force feedback device. This will help in our understanding of how much simulated haptic textures resemble real textures.

4.1 The human sensory system

4.1.1 Tactile and kinaesthetic sense

In section 3.1 we already gave a definition of touch and the somatosensory system, which besides touch is concerned with proprioception, muscle movement, facial expressions and the visceral senses. When it comes to computer haptics the most important senses are touch and proprioception, since they are the ones we can directly influence through haptic feedback. We can divide them in two major classes: *cutaneous* or *tactile* sense and *kinaesthetic* sense respectively [McG02].

Tactile sense corresponds to everything we can feel through our skin. The skin is populated with cutaneous sensory receptors that are responsible for feeling touch, pressure and texture. Their distribution is particularly dense in the hands and fingers, which are most commonly used in touching and feeling various things, and also in the lips and tongue [Joh01]. The skin is also responsible for experiencing temperature and pain, but these are the responsibility of other sensory receptors. To simulate this kind of feedback, tactile feedback devices can be used. The most commonly used haptic devices however are force feedback devices which stimulate the kinaesthetic sense. In this case, vibrations can be used to produce some form of tactile feedback.

The kinaesthetic sense involves the sensation of muscle movement and joint position [wik08d]. It is a self-consciousness that tells us where our limbs are at every single moment and allows us to feel forces by pressure that not only pushes the skin away, but also exerts force on a whole limb, so that the muscles have to put in some effort to counter the force.

4.1.2 Five perceptual scales

The human nervous system is so refined that numerous textures can be distinguished just by feeling them with a bare finger. When asked about their perception of a given surface people use terms such as soft, hard, elastic, rough, smooth, ribbed, hairy, woody, plastic, metallic, stonelike, powdery, liquid and much more. Thus stimuli can differ in a large number of physical properties. Hollins et al. [HBKY00] used five perceptual scales to classify tactile surfaces:

- Rough vs. smooth
- Soft vs. hard
- Sticky vs. slippery
- Cool vs. warm
- Springy or elastic vs. mouldable: requires that deformation of objects is allowed whereby springy objects jump back to their original state and mouldable objects stay deformed after manipulation.

They found that, when asked about their overall impression of a stimulus, the rough/smooth and soft/hard scales are the ones that have the largest contribution to the perception of most users. These are also the easiest properties to model in a haptic environment

and along with friction, which takes care of the stickiness, these are the properties that are most discussed in related work.

Temperature feedback is not commonly used because it requires another adaptation to the haptic hardware and the temperature of an output device can not be changed fast enough to create a realistic real-time simulation. But then again, temperature is not really a texture. Similar are the perception of liquids, hair and powder or sandlike surfaces or volumes. They all have a characteristic feeling, but we do not classify them under the term textures.

4.1.3 Just Noticeable Difference (JND)

> 'The Just Noticeable Difference (JND) or Difference Threshold is the minimum amount by which stimulus intensity must be changed in order to produce a noticeable variation in sensory experience [Lab08].'

The JND is used to determine whether an exerted force will be perceivable by the user. If for example two surfaces are meant to have a different roughness, then we need to make sure that the difference in roughness between the two surfaces is greater than the JND for roughness.

The JND can be applied to the visual and auditory domain as well, but we are obviously interested in the JNDs for the human haptic system. Different people have different thresholds when it comes to sensing; therefore the results we cover here are averages that are experimentally determined by Tan et al. [TCES94] As we already saw in section 3.1 and 4.1.1, touch has many forms and each of them has its own difference threshold.

The kinaesthetic sense gives us perception of the joint angles that determine the position of our limbs. When it comes to our arms that are primarily used when working with a haptic device, the joints are at the finger, the wrist, the elbow and the shoulder. Table 4.1 shows the just noticeable difference in angles as found by Tan et al. [TCES94] and Bleyen [Ble07].

Joint	**finger**	**wrist**	**elbow**	**shoulder**
JND	2.5°	2.0°	2.0°	0.8°

Table 4.1: The Just Noticeable Difference for joint angles

Another valuable JND for haptic applications is the JND for stiffness. It determines the minimum stiffness required to simulate a rigid object, e.g. a wall. The average

threshold point for stiffness was found to be 242 Newton/cm [TCES94].

Unger et al. [UHK07] made an analysis of the JND for texture roughness perception using a magnetic levitation haptic device. To create surfaces of different roughness, they used a flat surface populated with conical elements as shown in Figure 4.1. They determined JNDs for different probe size at various texture spacings for a particular cone size and shape. Different cone sizes or shapes result in different JNDs so we will not state their numeric results here. We will however go further into roughness perception in the next section. In our discussion we also use a surface with conical elements and a probe as interaction device to reach some conclusions about roughness.

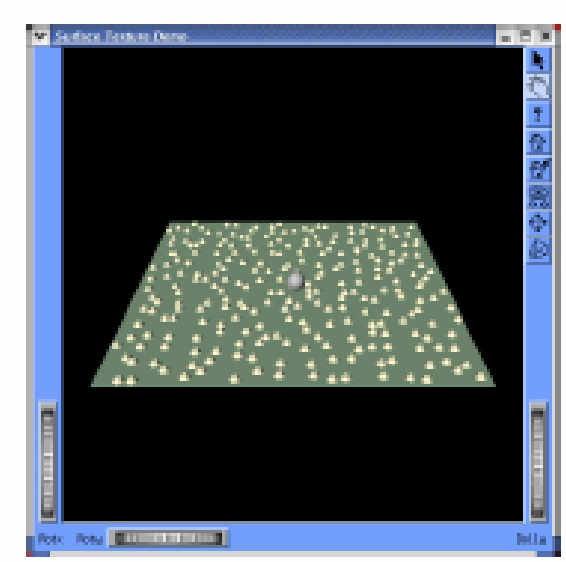


Figure 4.1: A surface with conical elements to simulate roughness. Image courtesy of [UHK07]

4.2 Feeling through a probe

By now we know that it is important when studying haptic textures to consider how the human tactile system works, but there is a great difference between feeling with your bare finger tips and feeling through a probe. When felt through a probe it is harder to distinguish the features of a surface. This definitely applies to computer generated textures. A haptic simulation is bound to certain restrictions. The force that a haptic device can exert is restricted to a certain maximum. As a result surfaces will never be completely impenetrable, thus resulting in a less realistic simulation of the environment. Also the most commonly used haptic devices, such as the PHANToM [ST08a], apply their force to a stylus, restricting the haptic feedback to a single point,

the endpoint of the stylus. There are other haptic devices which are able to provide feedback to multiple fingers of a hand [imm08], but here we will focus on the devices using a single probe.

In his thesis proposal, Unger [Ung05] examines the relationship between virtual probe-texture interaction and real probe-texture interaction. He investigates the influence of probe characteristics such as size and shape on human texture perception. In a more recent publication of his, Unger [UHK08] presents a geometric model to explain how texture size and spacing, probe size and probe speed influence our perception of roughness. In the remainder of the section we will summarize his findings, because of their importance to the perception of texture in general.

Consider a surface populated by conical elements that represent the texture, as already shown in Figure 4.1, and a spherical probe used to explore the surface with. The conical elements are cones with the top cut off and all have the same size. A cross-section of those conical elements is shown in Figure 4.2. They are uniformly distributed across the surface with a given spacing between them. Now we will explain how perception of roughness is influenced by varying the size of the probe, the size of the cones, the spacing between the cones and the speed with which the probe travels across the surface.

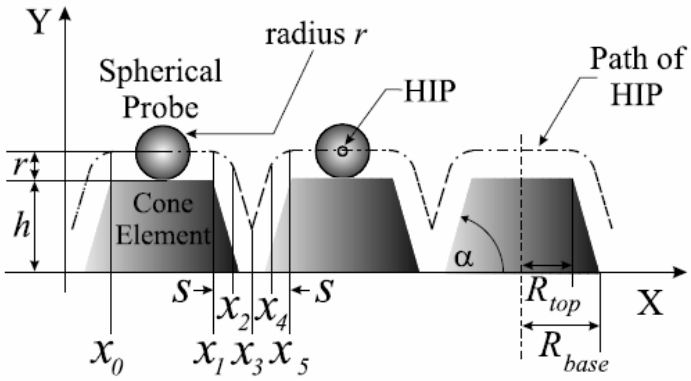


Figure 4.2: Cross-section of conical elements. Image courtesy of [UHK07]

4.2.1 Varying probe size

Consider two different sizes of probe: a large one and a small one which is a quarter the size of the large one. The large probe is larger than the cones, that is, the radius of the probe is larger than the height of the cones. The small probe is smaller than the cones and fits neatly between them. The movement of the larger probe will now be smoother than the movement for the small probe. This can have two reasons: either the cones

are spaced wide apart, at a distance so that both the small and the large probe fit in between the cones as in Figure 4.3, or the probes are spaced closer together so that the large probe doesn't fit in between them, but the small probe does.
In the first case, where the cones are spaced wide apart, both probes will touch the surface in between cones. The difference is made when the probe touches the next cone, where the large probe will need to make a smaller displacement than the small probe to get over the cone. The displacement of the large probe can be compared with the one in Figure 4.3(a) and the displacement of the small probe can be compared with the one in Figure 4.3(b).
In the second case, where the cones are spaced close together, the large probe never touches the surface in between the cones and consequently the displacement never equals the full cone height. The small probe however sinks in completely between the cones and has to make a full vertical displacement to get across the cones.

4.2.2 Varying cone size

For the same probe size, a larger cone size results in a larger vertical displacement of the probe and is thus perceived rougher than a surface with a smaller cone size.

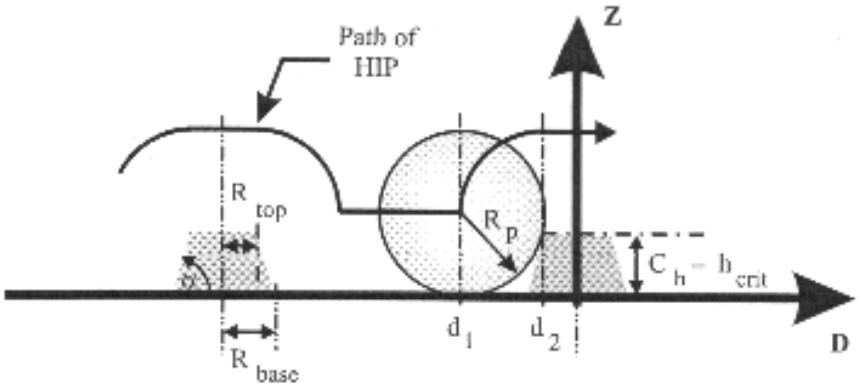
4.2.3 Varying cone spacing

Consider cones that are placed against each other so that the probe can not sink in between them, but glides on top of them. This is perceived as a smooth surface. Now when the spacing between the cones is initially increased, the probe will be able to get deeper and deeper in between the cones until it reaches the surface between the cones. During this phase the surface will be perceived as increasingly rough. This increase eventually reaches a limit for certain cone spacing. When spacing is further increased, the intercone space will become so large that the probe will be in contact with the bare surface most of the time and the surface will be perceived as increasingly smooth with increasing cone spacing. The point of maximum roughness lies somewhere near the point at which the probe can completely touch the surface between cones.

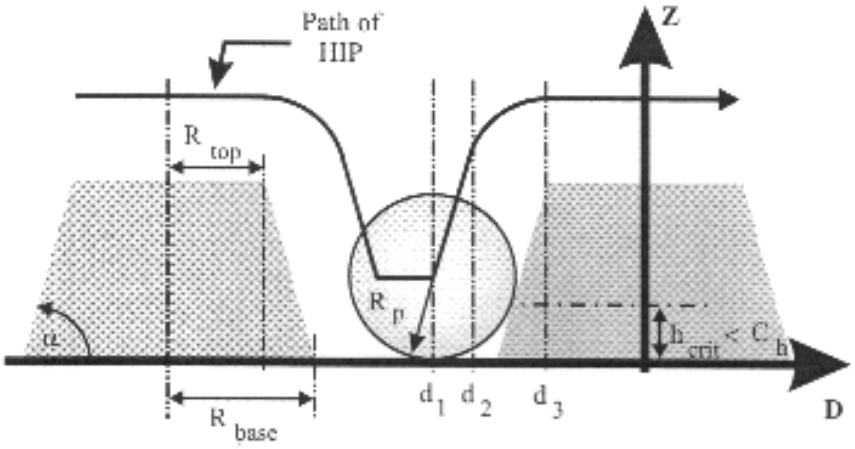
4.2.4 Varying probe speed

Finally speed of motion plays a role. Consider cones being spaced at the point of maximum roughness for a particular speed. Increasing speed will cause the probe to

not sink in completely between cones anymore. It takes a greater horizontal distance to reach the intercone surface. Consequently the probe has a smaller vertical displacement and the texture will be experienced smoother. For higher probe speeds, the point of maximum roughness will lie at higher intercone spacing.



(a) small cones



(b) large cones

Figure 4.3: Effect of cone size on roughness perception [UHK08]

4.3 Conclusion

In this chapter we have seen that the human sensory system consists of two major parts: the tactile and the kinaesthetic sense. The kinaesthetic sense is influenced by muscle movement and joint position, and can be affected by force feedback devices. The tactile sense on the other hand relies on the many cutaneous receptors in the skin,

and is not directly affected by force feedback devices. When we touch a surface directly with the hand and fingers, the surface excites many cutaneous receptors directly. If we would touch the same surface through a probe, the feedback we get comes from only one point, the probe tip. The cutaneous receptors in the hand are excited by the touch of the stylus, not the surface. In this case, surface features are felt through force feedback of the device, and thus when feeling through a probe, we rely mostly on our kinaesthetic sense instead of the cutaneous sense.

Furthermore we saw that the displacement that a texture causes must be greater than the Just Noticeable Difference, otherwise it will not be felt at all. JNDs differ for different applications, different properties and different persons, so there is no real standard of JND values that can be used.

Finally we conclude that when feeling textures through a probe, the perceived roughness is influenced by probe size, height of the texture, spacing of the texture and speed of motion of the probe.

We have now discussed how the human sensory system works and how feeling through a probe differs from feeling with our bare fingers. In the next section, some actual techniques to simulate surface textures are presented. All but one of them rely on the kinaesthetic sense.

CHAPTER 5

Haptic textures

In computer graphics, objects are generally represented by polygon meshes. These polygons form the surfaces of the model and they define the shape of the object. Increasing the detail of the model, and thus the number of polygons, results in an increase in computation time. To increase detail without resulting in too much additional computation time, further surface detail is represented by textures. Visual textures are simply 2D images that are adhered to the polygons' surfaces. Similarly in haptic environments, objects are modelled using polygonal surfaces that can not be penetrated and define the shape of the object. Again increasing the number of polygons requires more computation time, so textures are preferred to represent surface detail. They represent the feel of the surfaces, which can be smooth, rough, bumpy or some other property.

In this chapter we will explain several techniques for simulating surface texture and describe how they work. To be able to compare them with each other, a simple classification is given based on multiple properties that haptic textures can possess. Different techniques employ a different approach to simulating textures, but it is difficult to tell which approach generates the best or most realistic results. Therefore we will discuss the distinctions between the different techniques corresponding to our classification.

5.1 Classification

Based on the properties of the method used to generate the textures and the kind of force feedback that is generated, we can classify haptic texturing methods in several twofold classes.

A first distinction is made between *vibratory* and *force* feedback. Weisenberger and Krier [WK97] committed research to the roles of vibration and force feedback separately in transmitting textural cues. They found that when using vibration to represent surface texture, users could distinguish between surfaces with different vibratory frequency, intensity and spatial density. With force feedback, different surfaces could be felt by varying spatial frequency and amplitude. Both vibratory and force feedback can thus be used equally well in generating distinguishable surface textures. The difference between them lies in the way they generate feedback. When forces are produced according to bumps on the surface, we are speaking of force feedback. The feedback depends on the location of those bumps and their relative placement across the surface. Vibratory feedback on the other hand, is when vibrations are produced that have a certain frequency and intensity. They have a more temporal nature where force feedback has a more spatial nature.

A second distinction is made based on the direction of the exerted forces. We distinguish *tangential* and *normal* forces. Tangential forces are used in simulating friction for example. The user who moves the stylus across the surface experiences forces in the same plane that he moves his hand. The direction of the force either boosts the movement of the hand or slows it down by pulling it towards another destination. Normal forces are used in simulating height differences and bumps that make the surface rougher or simulate ridges. They push the user's hand away from the surface [HBS99].

A third distinction is made between *static or image-based* and *dynamic or procedural* haptic textures. Static means that the texture is predefined, e.g. using texture maps. A texture map is like a 2D image for a visual texture, but in the haptic case every pixel contains a height value that is used to calculate the force to apply. A texture map is specified for each surface and it is not possible to modify the objects at runtime. Dynamic textures on the other hand or procedurally generated at runtime. This has the advantage that textured objects can be modified. When one would cut a piece of a cube for example, the newly generated surfaces will be able to bear a texture, whereas with static textures the cut off edge would be smooth because no texture is specified for it [SO06].
McGee [McG02] writes about a structured and a stochastic approach. In this case the

structured approach uses a spatial structure of texture and a texture is composed of a primitive pattern that is repeated throughout the texture. In the stochastic approach, a random spatial texture distribution is used, in which each texture element is calculated statistically according to the surrounding texture elements.

Finally we can distinguish between realistic and non-realistic textures. Realism is not necessary for differentiation of surfaces. This means that for haptic visualizations where one has to be able to distinguish different surfaces from one another, it is not necessary to use a method that simulates a perfect wooden or other texture, but a simple difference in e.g. roughness will do. In contrast for 3D modelling of objects in the Computer Aided Design (CAD) domain realistic textures are preferred. After all, the purpose is to become a realistic model of the future product. Likewise, realistic models are desired in the medical domain to create simulations that are as realistic as possible.

5.2 Minsky's Sandpaper system

Minsky's Sandpaper system [Min95] is often referenced as the first real implementation of haptic textures. In her thesis, Minsky starts from the observation that lateral spring forces can feel very much the same like downward forces. A possible explanation for this is that the human haptic system experiences a similar feeling when one's hand is pulled down as when one's hand is pulled toward a laterally displaced goal point. This led to the use of lateral forces to simulate the feeling of a wavy texture using a 2D joystick. The rendered scene consists of a surface with hills and valleys. Lateral spring forces are generated proportional to the slopes so that the user needs to exert more force to move the cursor in an uphill direction than when moving downhill.

To calculate the force, the change in potential energy of the joystick is used as it moves over the slopes [Min95]. Assume that the joystick telescopes in order to keep the hand of the user at the same height and create a lateral motion. Figure 5.1 shows a slope with two instances of the joystick: one at location x_1 and one at location x_2, with d the distance between the two. The length of the joysticks is given by l_1 and l_2 respectively and the slope is given by θ. The only external force that is present is gravity, so the potential energy of the joystick at both points is given by:

$$\Delta PE = F \cdot d \tag{5.1}$$

where F is the force in the lateral x direction. To calculate the force to be exerted

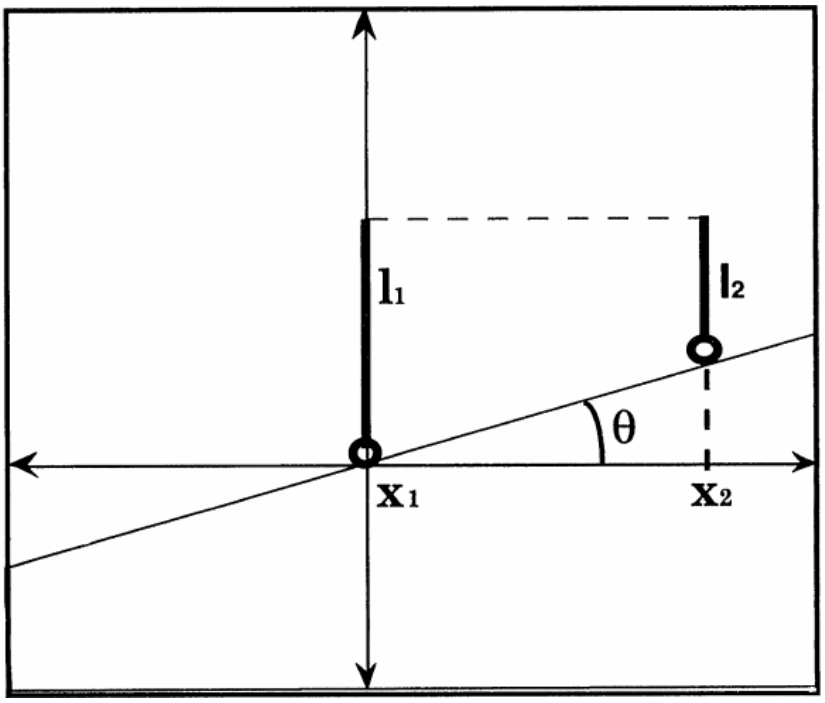


Figure 5.1: The basic principle behind Minsky's Sandpaper force calculation. Image courtesy of [Min95]

on the joystick when moving from x_1 to x_2, we need to take the difference in potential energy:

$$\Delta PE = PE_2 - PE1 = ((l_1 - l_2)mg) - 0 = (l_1 - l_2)mg \tag{5.2}$$

where PE_1 is the potential energy at x_1 and PE_2 at x_2, g is the acceleration due to gravity and m is the combined mass of the joystick and the hand of the user.
From Figure 5.1 we have:

$$\frac{(l_1 - l_2)}{(x_2 - x_1)} = \tan\theta \tag{5.3}$$

Substituting in Equation 5.1, we get:

$$(l_1 - l_2)mg = F \cdot d = F \cdot (x_2 - x_1) \tag{5.4}$$

$$(x_2 - x_1)\tan\theta \cdot mg = F \cdot (x_2 - x_1) \tag{5.5}$$

$$F = mg\tan\theta \tag{5.6}$$

Thus the resulting force is directly proportional to the incline of the slope.
The same calculation is independently done for the y-direction, since both the x and y axes are orthogonal and thus independent.

The algoritm implemented to realize force feedback in two dimensions is called the lateral-force gradient algorithm. It uses height maps which attribute a height value to every surface point. From these heights the forces in the x and y direction can be independently calculated using the following equations:

$$f_x = k(h_{x+1} - h_{x-1})$$
$$f_y = k(h_{y+1} - h_{y-1})$$

where k is a spring constant and h gives the height at the specified coordinate.

By varying the hill and valley width, the hill height, the force amplitude and the location of the hills, different feeling textures can be generated.

5.3 Stick-slip friction model

A solid surface should be impenetrable for a user. Thus at the moment of contact, a sufficiently large force must be applied to stop the motion of the user's hand. To realize this, a spring force is generated perpendicular to the surface.

$$F = kx \tag{5.7}$$

where k is the spring constant, x is the depth of penetration and F is the restoring force.
Salisbury et al. [SBM+95] found that adding a viscous damping term to this equation enhanced the user's perception of a hard surface.

$$F = kx + F_d \quad \text{with}$$
$$F_d = cv$$

where c is the damping coefficient and v is the velocity of the probe.

With only a force perpendicular to the surface, one would experience the surface as slippery or frictionless. Not many surfaces are indeed completely frictionless, thus Salisbury et al. [SBM+95] applied tangential forces on the probe when users would stroke the probe across a surface. A distinction is made between *static* and *dynamic* friction, also called *stiction* and *Coulomb friction*. The model has two states: either the probe *sticks* to the spot and we have stiction or static friction, or the probe *slides* across the surface and we experience dynamic friction. When a user initiates motion

on a stationary probe, a tangential force is applied that restores the probe to its initial starting point, the 'stiction point'. If the user eventually exerts enough force to move the probe, the probe gets loose and starts sliding. When this transition occurs, the displacement from the stiction point is used to determine the direction of movement. To generate friction when moving the probe, a tangential force in the opposite direction is applied. The magnitude of this force is determined by the dynamic friction coefficient.

Continuing from the static and dynamic friction method, Mark et al. [MRF+96] created their own friction model. A surface is again modelled by a spring force perpendicular to the surface and proportional to the depth of penetration of the probe. They observed that the probe had a tendency to slip into concave areas and off of convex ones, leading them to an adapted stick-slip friction model. Figure 5.2 shows how it works.

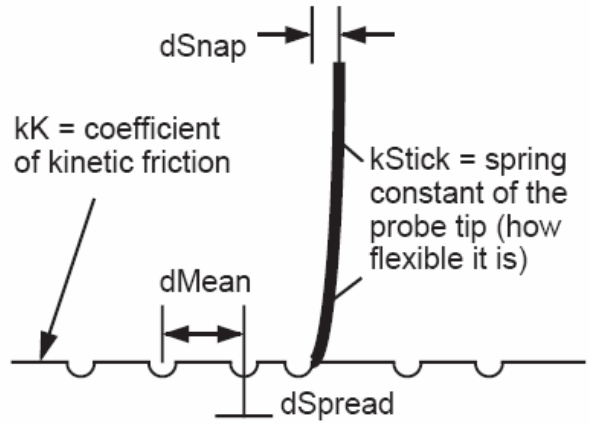


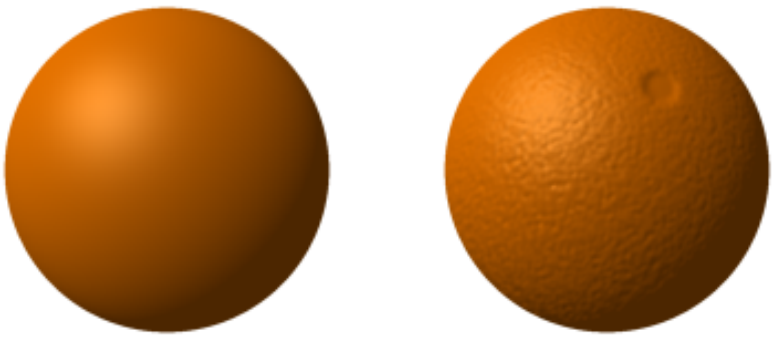
Figure 5.2: Stick-slip friction model[MRF+96]

The surface is populated with snags that tend to hold the probe in place. A surface with a higher snag density will consequently be experienced as a surface with more friction. The snags are distributed across the surface with a mean distance between them of *dMean* and a spreading of *dSpread*. Thus a snag will lie somewhere between a distance of *dMean-dSpread/2* and *dMean+dSpread/2* from a neighboring snag. The probe is implemented as having a flexible tip. When sliding across the surface without encountering any snags it is opposed only by a friction force proportional to the normal force with *kK* as coefficient of kinetic friction. When the probe runs into a snag, it slips in the concave area. A tangential force proportional to the distance from the probe tip to the snag center, and proportional to the normal force, pulls the snag towards the center. The spring constant of the tangential spring force is the one of the flexible tip *kStick*. The probe tip sticks in the snag until it is moved a distance *dSnap* away from its center, then it snaps free. During this stick phase the user experiences a force pulling the probe back to the center of the snag.

Higher densities of snags in the surface produce more friction or 'station keeping'. Deeper or wider snags will also effect in more friction, because the generated forces will be greater. Finally speed also plays a role as described in chapter 4.

5.4 Force mapping

Force mapping is based on the bump mapping technique used in computer graphics. In computer graphics normals are specified for every surface to allow for a more natural illumination model. But by basing the light reflection on these surface normals alone, surfaces all seem smooth, as shown in Figure 5.3(a). To create a richer and more detailed surface representation that better represents the real world objects, bump mapping is used. This technique specifies a height map for every surface of the object. The height maps here are simply 2D images containing a perturbation value at every pixel. The surface normal at a certain point of the object is then perturbed using the value at the corresponding pixel in the height map. The perturbed normals cause the light to reflect in a different way for different positions on the surface, causing a more realistic surface texture. Figure 5.3(b) shows the same image as in 5.3(a), but with a bump map applied changing its appearance so it looks like an orange [wik08a].



(a) Sphere without bump mapping. Photo courtesy of [wik08a]

(b) Sphere with bump mapping. Photo courtesy of [wik08a]

Figure 5.3: Bump mapping

Analogous to bump mapping in computer graphics, force mapping is used in the simulation of haptic textures [TFNM05]. This time a force map contains a normal vector for every pixel in the map, with the normal taken from the textured surface one wishes to become, not from the underlying surface which takes care of the collision response. Figure 5.4 shows a lateral cut of such a surface with the corresponding force map. At every point the force direction and magnitude are computed from the normal vectors

and are applied to the haptic device. For small distortions this technique gives a correct perception of surface roughness, just like bump mapping did for the visual surface.

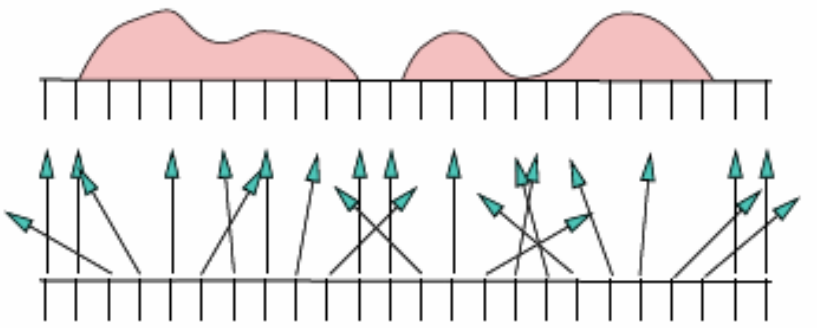


Figure 5.4: Force mapping. Image courtesy of [TFNM05]

5.5 Height field rendering

Height fields [TFNM05] also consist of force maps, but in this approach the force vectors are always normal to the polygon surface instead of the texture surface as seen in figure 5.5. It is simpler in the way that force maps only have to specify a vertical height value for every point in the map and not a direction.

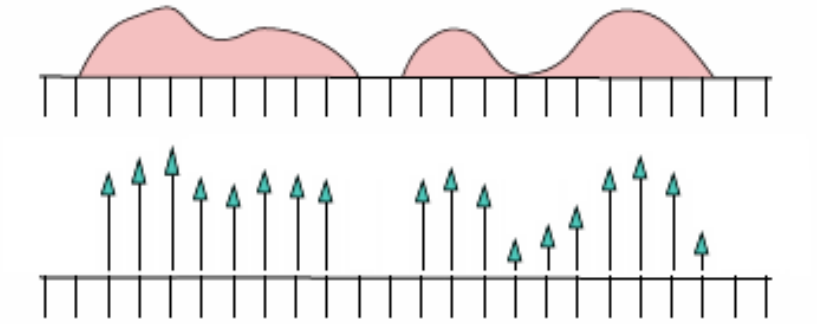


Figure 5.5: Height fields. Image adapted from [TFNM05]

The main difference compared to the force mapping technique is that collisions are not detected against the triangle mesh, but against a prism consisting of the triangle itself, the same triangle displaced over a certain maximum distance above the original one and the sides connecting the two triangles. When a collision is detected against this prism, the HIP is inside the prism, then the HIP position is projected on the triangle mesh, the corresponding value of the height field map is returned and used to calculate the upward normal force to be applied to the haptic device. On the edges of the triangles the height values are forced to zero to avoid force discontinuities.

Compared to the force mapping technique height fields actually give the user the feeling of height differences whilst with force mapping one only perceives the resistance of going up and a jump going down.

5.6 Haptic shading

Shaders have been around in computer graphics for years. They make up the part of the rendering system that is responsible for calculating the colour of the pixels on screen. Nowadays these calculations are executed in parallel in the graphics hardware and supported by software libraries such as OpenGL and DirectX. In their work Shopf and Olano [SO06] have shown how shading can be similarly done in computer haptics. Haptic shading even has a major advantage compared to visual shading. Where visual shaders have to run at least once per pixel per frame, resulting in millions of executions per second, a haptic shader only has to run once per haptic interaction point, which is only one for a PHANToM [ST08a] device.

A haptic shader takes two sources of input wherefrom the output is derived. The first source is the collision model which provides contact information like surface normal, surface position, movement direction, velocity and acceleration, collision force, static and dynamic friction. The second source consists of the specified material properties like static and dynamic friction, persistent texture data, stiffness, damping, and surface parameters. The data flow is illustrated in figure 5.6.
The haptic texture shader takes these input values and computes corresponding output forces. Different shaders use different procedures for this computation and thus produce different feeling textures. Different material properties also yield different results. Note that the haptic shaders execute after collision itself, therefore only small justifications can be made to the output force. Larger displacements will cause inconsistent collision responses.

5.7 Tactile feedback

To simulate tactile feedback a whole other approach is needed, even the hardware differs. So far we have discussed techniques that work with a PHANToM [ST08a]device. The problem is that it can only generate force feedback and some vibratory feedback by vibrating the stylus the user holds, but it is not capable of producing tactile feedback. Tactile feedback corresponds to the more delicate perception of surface features through

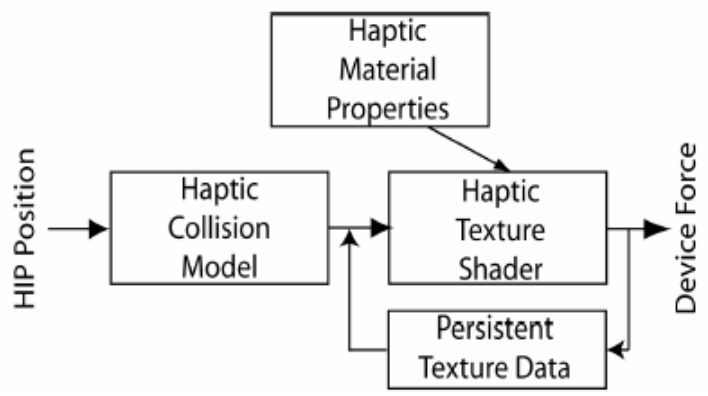


Figure 5.6: Data flow of the haptic shader. Image courtesy of [SO06]

the skin as we explained in section 4.1.1. To address our tactile sense a tactile display is used which typically consists of a dense array of skin contactors [HCH00]. These skin contactors are small rods that can move in a vertical direction, coming up to exert pressure on the skin of a finger in an attempt to display surface features through its spatial configuration, as shown in Figure 5.7. The device can be compared with a Braille rule, with that difference that it needs a larger array of contactors and it must be able to update its configuration at much higher update rates.

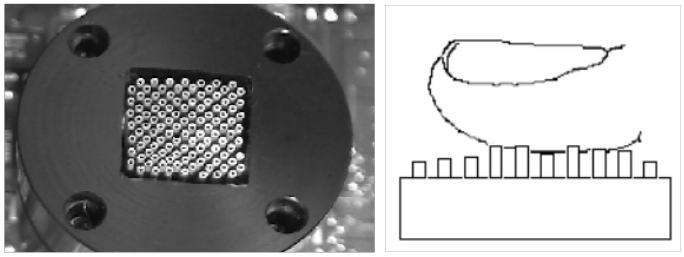


Figure 5.7: Tactile display. Image courtesy of [HCH00]

Another way this tactile display can produce surface features is by vibrotactile stimulation. With this approach the skin is stimulated by vibrating the contactors at a fixed frequency. A frequency of 200-300Hz is used to maximize the loudness of the sensation.

Both vibrational frequency and spatial configuration combined can form patterns that depend both on space and time. These patterns are called tactile images or tactile movies, since the image changes in real-time. In order for the resulting pattern to resolve into one single continuous image, the contactors should be spaced about one millimetre from each other [HCH00].

5.8 Comparison

To conclude this chapter we will give a comparison of the five techniques here described according to the classification given in section 5.1. A summary is given in Table 5.1.

Minsky's Sandpaper system [Min95] applies tangential forces on a 2D joystick to simulate the feeling of slopes. The position of the slopes is determined by height fields that are specified in advance, thus it is a static technique. This technique is often mentioned as the first real attempt at simulating surface texture. It is very limited and does not generate realistically feeling textures.

The stick-slip friction model [MRF+96] is also a tangential force feedback technique. The grid containing the snags defines the amount of friction perceived. The size of the snags determines the exerted forces while the density and position define where on the surface friction is felt. Two approaches can be used here. One approach defines the snag grids in advance and thus allows specifying surfaces that have areas with friction as well as areas without friction, by the distribution of the snags. The second approach generates these friction grids automatically based on some randomization function to distribute the snags across the surface. In this approach either a whole surface contains friction or it has no friction at all. The stick-slip friction model is thus either static or dynamic depending on the chosen implementation.

Force mapping [TFNM05] uses normal forces to simulate surface texture in a realistic way. However, when the distortions become too great, it may not give a correct perception of surface roughness anymore. Force mapping is a static technique in that the direction and magnitude of the forces need to be specified in advance for every pixel in the texture map.

Height fields [TFNM05] are analogous to force maps. They generate realistic textures using normal forces. The difference between them is that with height fields only a height value needs to be specified for every pixel in the texture map. Unlike force maps, which give a perception of going up and down, height fields also give a realistic perception of height difference to the surface texture. The exerted force depends on the penetration of the probe and on the height value for that location.

Haptic shading [SO06] uses a mathematical calculation to determine the force to be applied at a given location on the surface. This makes it a dynamic texture, because it calculates these values at real-time and doesn't need a predefined texture map. Haptic shading works separately from collision detection, making it a second step in calculating the force to exert to the user. This also limits the force output to small justifications,

to not distort the collision response. Haptic shading allows the simulation of realistic surface textures, given that the specified material properties are accurate enough.

Tactile feedback [HCH00] is the only tactile technique we have discussed. It uses a combination of forces exerted on a whole array of small rods and vibrations of these rods. The forces are not used to displace the finger but they put pressure on the finger in a certain pattern. Vibrations of the rods can generate a buzzing or ticking experience. The forces are always normal, since that is the only direction in which the rods can move. Simulated textures can be either static or dynamic, where static textures would use a texture mapping technique and dynamic textures would be mathematically generated. Dynamic textures cover the time domain, while static textures cover the spatial domain. Realistic textures are possible.

Property	**Minsky's Sandpaper**	**Stick-slip**	**Force mapping**	**Height fields**	**Haptic shading**	**Tactile feedback**
Vibratory feedback						X
Force feedback	X	X	X	X	X	X
Tangential forces	X	X				
Normal forces		X	X	X	X	X
Static textures	X	X	X	X		X
Dynamic textures		X			X	X
Realistic textures			X	X	X	X
Non-realistic textures	X	X				X

Table 5.1: Comparison of haptic texture rendering techniques

5.9 Conclusion

Several different techniques for simulating surface textures have been described in this chapter. Several more that are not discussed here certainly exist. Some techniques may be preferred for medical simulations, while others will be preferred in CAD applications. Some domains require a high level of realism, while others are satisfied with some difference in roughness. Creating the perfect technique that incorporates all possibilities is hard; it may not even exist. All current techniques have their own advantages and

disadvantages and more work is being done on perfecting and extending these methods to achieve better results with the hardware and software possibilities existing today.

CHAPTER 6

Conclusion

In this first part, we have motivated why haptics, and haptic textures in particular, are useful in modern applications. We have given an introduction to computer haptics in general as a foundation for studying haptic textures. We described the characteristics of perception that are relevant for the perception of textures through a probe, because feeling through a single probe endpoint is far from the same as feeling with our bare hands. Next we presented six techniques for simulating surface roughness or texture. Minsky has laid the foundation with her Sandpaper system [Min95]. After her, more research has been done towards haptic textures.

Two classes of techniques can be distinguished based on the way textures are generated and presented: static and dynamic. Static means that the texture is predefined, e.g. using texture maps, and it is not possible to modify it at runtime. Dynamic textures on the other hand are procedurally generated at runtime. This has the advantage that textured objects can be modified. When one would cut a piece of a cube for example, the newly generated surfaces will be able to bear a texture. Intuitively one can conclude that the latter class has more potential. Haptic shading belongs to this class of techniques.

To find out which techniques result in the best texture perception would need testing them all with some different users, human perception is a subjective notion after all. Perhaps a combination of methods works best. Testing this is not the aim of this thesis,

so we will not elaborate on it.

We did at first want to find out whether and how we can generate haptic textures automatically from their visual counterparts. This would facilitate the work of modelling a visual and haptic 3D virtual environment. As we have seen, the height field technique requires height maps to be specified in advance. A height value needs to be specified for every pixel of the surface. To realize this, simple 2D greyscale images can be used where the color of each pixel defines the height value at that point. Defining black as 0 and white as 1, we can map the value of every pixel to a corresponding height, rescaling this interval to the maximum height we want to become. Because we only need to specify these 2D images in advance this technique seems particularly interesting to use for automatically generating haptic textures from visual ones. We already need to specify visual textures in advance, so why not use them to generate haptic feedback as well. The main problem to overcome is that in computer graphics, the textures used are usually in color, and not simple greyscale images. Perhaps converting the color values to their corresponding greyscale value will already yield an acceptable result. But this would always make the lighter regions of the visual texture the higher ones in our haptic texture. That may not always be desirable. This problem of automatic haptic texture generation would be the main focus in the next part of this thesis, where we were going to implement it with the HAL haptic library [hal], wasn't it for a coincidence that pointed us to H3D [h3d04], another haptic library that has already implemented this. Therefore a change of course is at hand.

6.1 Introduction to Part III

Instead of implementing height field rendering, which was the first plan, we have to find a new purpose for this thesis. We find this purpose in the field of information visualization. In Part III we will take a look into the possibilities of using these height field textures to represent data.

Part III

Practical research

CHAPTER 7

Using haptic textures in information visualization

A lot of ways exist to visually represent data, from simple static tables to dynamic graphs and interactive maps. But why should we focus on just our visual sense, when we can make use of all of our senses? That is where our hearing came in and audio was added. It was the next logical step. Any system with speakers can generate sound, and speakers are since long well known and used. More research and alternative hardware were needed to use touch as a way of both input and output in a computer system. Force feedback systems were invented, and can now provide an alternative way of representing data. In this chapter we will investigate how haptic surface textures can be used to represent data.

The purpose is twofold. In the first place, blind or visually handicapped people can benefit from making information haptically 'visible'. A solution exists to make 2D graphs touchable by printing them like Braille on a paper sheet. However this does not allow for dynamic manipulation of variables through which changes in other variables may be studied. A computer haptics application can provide this dynamics. Also, for visually handicapped people, computer haptics provide much more possibilities for displaying data. The computer system can provide additional auditory clues, we are not restricted to 2 dimensions and we can let the haptics device guide the user to the data so that he doesn't need to search the whole information space. The second purpose of using haptics in information visualization is that we can use it to increase the amount

of perceivable data in one image for seeing people. Visual data can be combined with touchable surfaces so that two data properties can be perceived at the same time, much like using layers of visual data, but without the side effect of cluttering the display.

The haptic extension can perhaps equally well support the user in discovering relationships in the combined data. However the way we can perceive haptic data differs from the way we perceive visual data, so it is important to look at this difference to create an understanding of how we can incorporate haptics to represent data.

7.1 Visual vs. haptic perception

Obviously there is a difference in perception through touch and through vision. In this case we are both limited by the human body itself as by the restrictions of the computer system.

Visually, we can perceive a lot of information in one eye sight. Properties like color, overall texture, the kind of information (text, image, plain surface,...) and their place in the scene are all clear to us in just one look. From a haptic 'point of view' we only have one point of information per haptic device used, at least when working with a PHANToM device like we do. A haptic feedback glove on the other hand can have multiple. This means that the amount of data to be perceived at once is always limited.

Another difference shows in using legends or reference scales as an explanation for symbols used on a map or a chart. Looking up the meaning of a symbol visually is very quick and easy, thanks to parallel perception of visual data. Should we do it by touch, it is a whole other story. Perception through touch is sequential by nature, so a quick overview of the legend is unfeasible. We would have to feel every item in the legend until we found one that corresponds to the data. And possibly we would need to go back to the data to compare if it is the same thing, because our memory can let us down. Unless we can have a visual legend for haptic data also, this would not work out.

These are obstacles for haptic information representation applications which have to be overcome. A possible way to cope with the sequential nature of touch is to use haptics to represent sequential data. That way the mapping will also more likely be accepted.

7.1.1 Surface properties

Textures are surface properties, which means that we can use them to represent data on the surface of objects. As long as it has a 2D area, we can cover it with a haptic texture. This means dots in a graph for example cannot be covered with a texture. The graph itself however does have an area and the collection of dots in this area can be modeled as a texture.

7.2 Height field textures for data representation

As seen in Chapter 5, haptic textures can be generated by various techniques. To narrow our scope, we will focus on the use of height field textures and set a couple of research questions to answer.

Can users associate data with texture and can they make the right connections to understand existing relations in the data? For example, visually we can map temperatures to colors using blue for cold and red for warm. We can then ask users to sort colored regions from warm to cold. This is a commonly used mapping and many people will therefore easily be able to do this. We would like to find out if users can draw the same correct conclusions, but now based on touch instead of vision. Can they make the right connections and draw the right conclusions when using haptic textures instead of colors?

When using textures with a repeating element like dots, lines and grids, how should the spacing between elements be to be optimally recognizable? When we want to make a distinction between two surfaces using two textures with a different spacing, how small or how large may the difference in spacing be?

For representing qualitative data, for example in the form of icons, can we use random textures? Will users be able to recognize these random textures or should we define a custom set of images that are found to be well recognizable? What aspects should we hold into account? Does size have an influence?

When exploring haptic data, what method do users use to recognize things? What is their behavior when interpreting texture? Do they look for patterns, differences, shapes, something else?

Based on these questions we build an application to render sample data haptically using height field textures. We will then set up a preliminary user test that should give us

some answers. The application and the user experiment are explained in the following chapter.

CHAPTER 8

The user experiment

In this chapter we will discuss the application we wrote to 'haptically visualize' a scene with sample data. With the haptic scenes we create, we try to find an answer to the questions posed in the previous chapter. A preliminary user test is set up to gain some feedback of and insight in actual user behavior.

8.1 The application

We build our application to show two kinds of data: icons and maps. Icons are used to represent qualitative data on maps, in diagrams, wherever one likes to use them. They are also frequently used in user interfaces to represent commands or additional info. We will use square surfaces as icons and provide them with a haptic texture, representing the data. What we want to find out here is whether users are able to associate data with the icons. We do this by attaching a different word to each icon. The words are of course visible to the user. Then we let the users feel the icons with the words, and see if they can afterwards still place the right word with the right icon. What we also want to find out is whether there is an effect between the kind of textures used and the users being able to recognize the icons well.

Maps are used to display all kinds of data. We want to see if we can use haptic textures

to represent data as an additional layer of information on a map, besides the visual. Too much visual information can clutter the display, and possibly we can benefit from an additional haptic layer. By integrating touch we can also support visually impaired people to work with the same data. More specific we will look at grids as a way to represent density on a map. We are interested in the effect of spacing between grid lines. How small can the difference in spacing be to still be able to discriminate between different textures? For this we create a sample map with regions that are visually covered with a color, and haptically with a height field grid texture.

8.1.1 Building blocks

The application is based on H3DAPI 2.0 [h3d04]. H3DAPI is an open source haptics framework combining OpenGL [SWND05] and X3D [x3d09]. H3DAPI manages an OpenGL scene graph to visually display the scene. It is also responsible for handling the haptics events, in a cross platform, and haptic device independent manner. X3D is the file format used to represent 3D scenes and objects. It is an XML based ISO standard. Scenes written in X3D are loaded by H3DAPI into nodes and fields of the scene graph. Written in C++ it offers the possibility to create custom nodes or extend existing ones and to interact with the scene graph. Python scripts can also be used to implement interaction with the scenegraph.

The surfaces covered with haptic textures in our application are implemented as a H3D `IndexedTriangleSet` with a `DepthMapSurface` node. The `DepthMapSurface` node contains an `ImageTexture` pointing to the image file containing the texture. The `DepthMapSurface` represents a height field, using the given image to extract the height values from. For greyvalue images we set black to be the highest point and white as the lowest, so a black dot on a white surface will be felt as a bump sticking out of the surface. For color images the value of the red channel of the RGB value of a pixel is used to determine the height at the corresponding location.

H3D has a well supported website providing useful documentation in the form of a manual, a wiki with examples and Doxygen pages. Besides these they also have an active user/developer community with an active forum providing lots of answers and an easy environment to ask questions. The admins there are also very helpful and try to give a quick response to all questions concerning H3D.

Further we used Qt v4.2.2 [qt09] to make a user interface to display the OpenGL window and to facilitate user interaction outside of the 3D scene.

8.1.2 Functionality

The application should be able to do the following things:

- Load a scene from an X3D file and display it in an OpenGL window.
- Write X3D files to create scenes with icons.
- Write texture files to use for the haptic height fields.
- Run a scenario showing a number of scenes with icons and words, each time after scrambling the icons asking the user to put the right icon with right word again. Repeat for a number of scenes using different types of textures for the icons.
- Run a scenario showing a map with colored regions and ask the user to sort the regions by density perceived through touch. Repeat with different maps and different textures.
- Save user input for later analysis.

8.1.3 The icon scenario

For the first scenario we create a number of scenes with 3 icons per scene. Every icon has an associated word that is displayed next to it. The icon itself is shown as a plain pale blue square with a texture that is only perceivable by touch. In the test, the users will be asked to explore the icons using the PHANToM haptic device, and to memorize the words for each feeling. At first we wanted to use 5 icons per scene, but this would definitely be too much to remember, so we used 3 instead. There is no time limit to explore and memorize the icons, because that could result in stress and be harmful for the results. As soon as the user indicates he is ready, the scene is reloaded with the icons randomly reordered and without the words showing next to them. The user is then asked to touch the icons and to put the right word with each icon again.

We do this for four scenes using random visual textures to generate the height field haptic textures from. You can see a visual representation of the textures used in Figure 8.1. The words have no obvious relation with the textures, and are chosen from short commands, commonly used in user interfaces. E.g. new, save, exit, undo, etc.

Next we do the same with three more scenes. This time using custom made textures that we believe should be more easily recognizable compared to the random ones. A visual representation of these textures are shown in Figure 8.2. Why do we believe

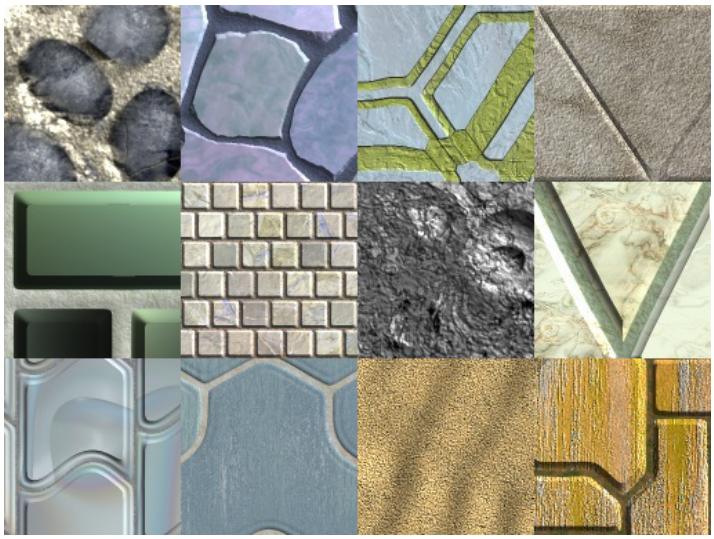


Figure 8.1: Random textures used in our experiment. The textures are courtesy of [spi09]

that these icons are more easily recognizable? This time we try to fit the texture with the associated word. A mapping that is logical will make it easier for the user to learn and remember. Also with the rounded shapes we want to become that the icon is recognizable no matter in which way the user moves the pointer over it, and with as few strokes as possible. We assume that most people will move the pointer horizontally or vertically over the icons. So when moving the pointer from top to bottom on the icon with the 'left' texture (Figure 8.2(a)), they will suddenly feel that the pointer is being pushed to the right because of the diagonal line of the half disc. When moving the pointer from left to right over the same icon, they would feel a drop as soon as they cross the border between black and white. The lines should also be easily recognizable since the user can quickly feel a number of bumps in either the horizontal or vertical direction, provided the number of lines is limited and well spaced apart.

8.1.4 The map scenario

In the second scenario we create three scenes, each showing a map with 10 colored regions in 5 different colors. With each color we associate a density, represented by a height field texture. We then ask the user for each scene to use their touch to sort the colors from highest to lowest density.

The first map uses grid textures with horizontal and vertical lines as shown in Figure 8.3. The spacing between the lines is 10 pixels for the grid depicting the highest density. Then 20, 30, 40 and 50 pixels respectively for increasingly lower densities. In the second

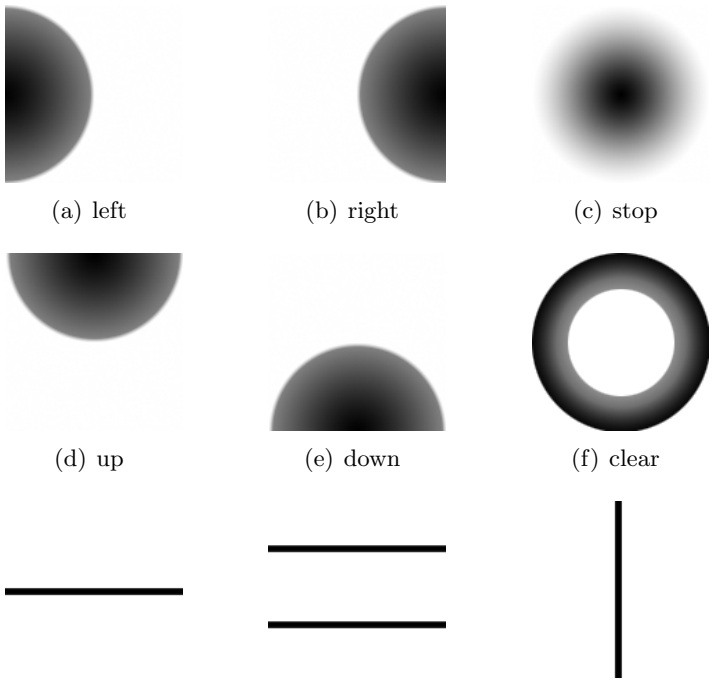


Figure 8.2: Custom made textures along with the corresponding words.

map we also use the same type of grid textures, but this time with a smaller difference between the different densities. For the spacing values we take 5 pixels for the highest density, and 10, 15, 20 and 25 pixels respectively for increasingly lower densities.

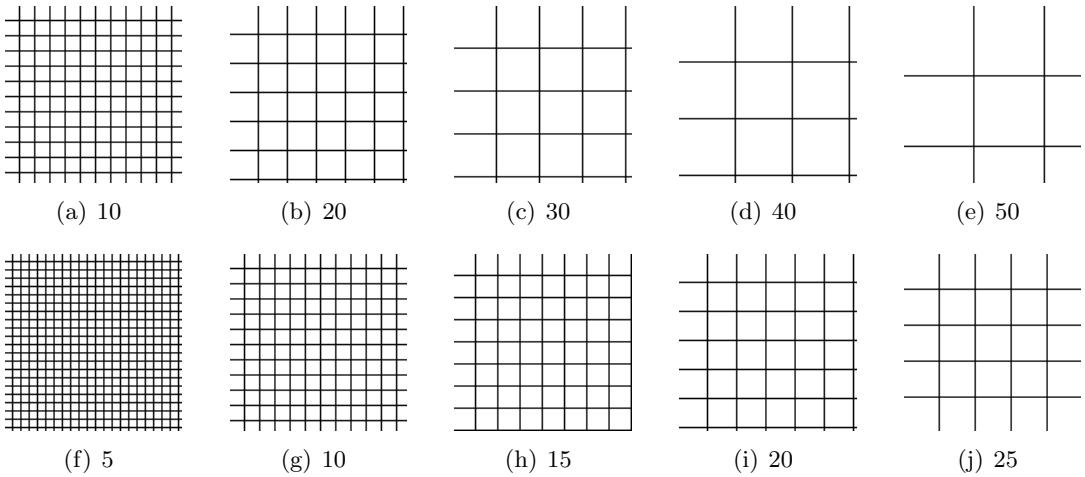


Figure 8.3: Grid textures for the first map (above) and for the second map (below)

Using these two first set-ups we want to find out if users can still equally well distinguish the densities when the difference between them becomes smaller.

In the third map a dotted texture, see Figure 8.4, is used to compare its effectivity against the grid textures. We assume that dots are far more difficult to recognize and distinguish.

(a) 5 (b) 10 (c) 15 (d) 20 (e) 25

Figure 8.4: Dot textures for the third map

8.2 The user test

8.2.1 Set-up

For the test we used a desktop computer with a SensAble PHANToM Premium Haptic Device [ST08a] attached. Running on the computer was our own application based on the H3D framework as described in Section 8.1.

8.2.2 Participants

10 volunteers have been recruited from the research centre EDM, Expertise Centre for Digital Media, at the University campus. Among them were researchers as well as students. 6 participants were male and 4 were female. All were aged between 20 and 40 years old. Nine of them had Dutch as their native language, one was an external PhD communicating in English. The test instructions were written in Dutch, but were given orally in English to accommodate for the PhD student. All of them used the PHANToM right-handedly. Three of them had never used a PHANToM before.

8.2.3 Procedure

The user test started with a short introduction in which the participants were asked to read the instructions for the tasks ahead. They were allowed to ask questions now and during the test should anything be unclear to them. Before starting the actual

test, participants were asked if they were familiar with the haptic device, and given the chance to a short practice.

The test consisted of two separate scenarios as described in Sections 8.1.3 and 8.1.4. The participants were asked to fill in a questionnaire halfway scenario 1, at the end of the first scenario and at the end of the second scenario. The answers the user gives during the test scenarios are written to file for later analysis. Together with the questionnaires they form the results of the test.

8.3 The results

The feedback from our user experiment comes from three sources: the users' answers from the scenarios that were saved to file, their answers on the questionnaire, and the observations made during the test. Thus from our user experiment, however limited in scale, we could draw the following conclusions.

8.3.1 Users' test results

From the users' answers that were saved to a file, we can distract the following information.

On the first series of random icons (Figure 8.1) every icon was mapped to a wrong word on average 3.17 times with a standard deviation of 0.89. On the second series of custom made icons (Figure 8.2) every icon was mapped to a wrong word on average 2.33 times with a standard deviation of 0.74. This lightly suggests that users make less mistakes with the custom made icons and thus these icons are better recognizable. A test with more users could provide clearer results.

For the first map (grids with difference of 10 pixels between consecutive regions) the participants made an average of 1.5 mistakes with a standard deviation of 1.2. Three out of ten participants made no mistakes at all. One participant had placed all regions in reverse order, and did this also on the other two maps, indicating that he had a different perception of high vs. low density. The reverse order was correct though, making a total of 4 participants with no mistakes at all on the first map.

For the second map (grids with smaller difference between consecutive regions) the participants made an average of 1.2 mistakes with a standard deviation of 1.2. Five participants made no mistakes at all, the one in reverse order included.

For the third map (dots) the participants made an average of 1.6 mistakes with a standard deviation of 0.64. Only two participants made no mistakes at all.

8.3.2 Statistical analysis of the test results

To check the significance of our results we make a statistical analysis using the R statistical software package [r09]. Also installed was the xlsReadWrite package to read data from an Excel spreadsheet.

The independent variables in the test were:

- IconType: the type of icons used for a specific task, either random or custom made
- MapType: the type of grid density texture used for a specific task, either the larger spacing difference or the smaller spacing difference

The dependent variable in each case was:

- Errors: the number of errors the user made on a specific task

For both test scenarios a within subjects design was used. Since the number of participants was already limited, all participants performed all tasks.

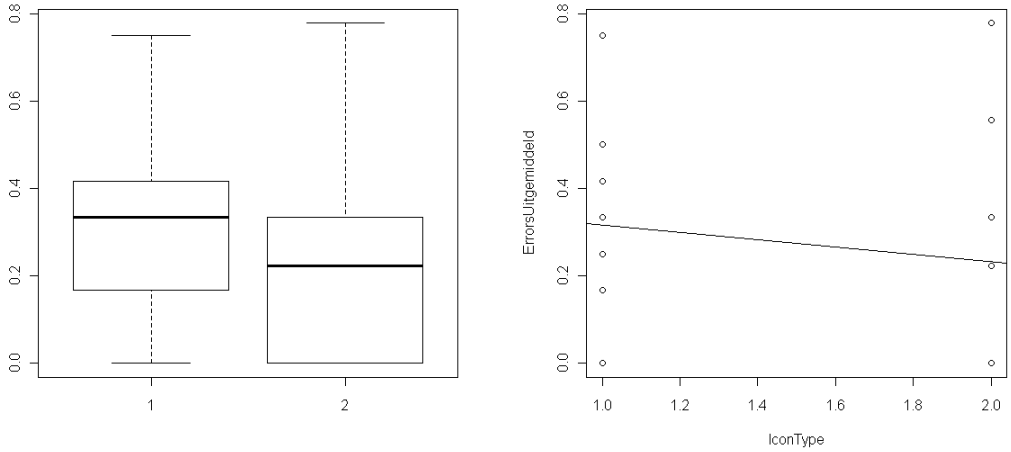
Icons

We believe that using custom texture icons, users make less errors than using random icons. Thus for the null hypothesis we state:

> H_0: mean number of errors for *random* icons = mean number of errors for *custom* icons
> H_1: mean number of errors for *random* icons != mean number of errors for *custom* icons

We first make some plots to see what our data looks like. These plots are shown in Figure 8.5.

IconType 1 depicts the icons with random textures.



(a) Boxplot with nr. Errors in function of IconType

(b) Linear plot

Figure 8.5: Average number of errors for IconType 1 and 2

IconType 2 depicts the icons with custom made textures.

In the box plot we see that the average number of errors is lower for IconType 2, being the icons with custom made textures. The linear regression shows us the same trend; the number of errors is larger for the random icons than it is for the custom made icons.

Now we want to check whether the observed difference in mean number of errors is significant for different icon types. We do this through a null hypothesis stating that the mean number of errors is equal for both icon types. Therefore we perform an Analysis of Variance using an ANOVA test on the factor IconType. The output of R is:

```
> icontype<-as.factor(icons$IconType)
> aov.data = aov(Errors~icontype, data=icons)
> summary(aov.data)
            Df  Sum Sq Mean Sq F value Pr(>F)
icontype     1 0.03472 0.03472  0.5693 0.4603
Residuals   18 1.09784 0.06099
```

The p-value is 0.4603, this means that for a significance level of 5% we conclude that there is no significant difference between the mean number of errors for both icon types. Very likely this is due to the limited number of test users. Repeating the test with a

larger test group should bring clarification.

Maps

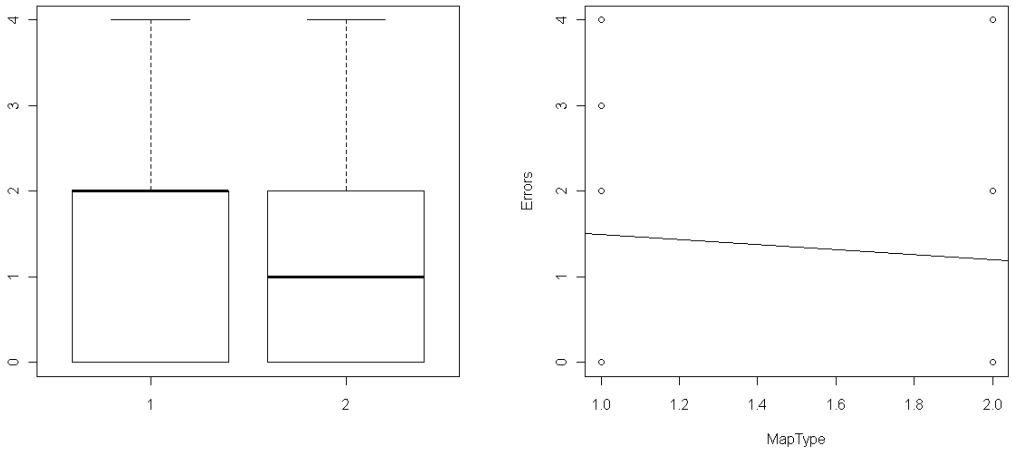
Here we will only compare the two grid maps, since the dotted map actually presents a different category. We believe that grid textures with a larger difference in spacing are more easily distinguished and thus less errors are made compared to grid textures with a smaller difference in spacing. Thus for the null hypothesis we state:

> H_0: mean number of errors for *large spacing difference* = mean number of errors for *small spacing difference*
> H_1: mean number of errors for *large spacing difference* != mean number of errors for *small spacing difference*

We start by making some plots again to see what our data looks like. These plots are shown in Figure 8.6.

MapType 1 depicts the grid with larger spacing difference.

MapType 2 depicts the grid with smaller spacing difference.



(a) Boxplot with nr. Errors in function of MapType

(b) Linear plot

Figure 8.6: Average number of errors for MapType 1 and 2

In the boxplot we see that the average number of errors is lower for the grid with the smaller spacing difference (MapType 2). The same is shown in the linear regression.

Again we check whether the observed difference in mean number of errors is significant for different map types. We do this with a null hypothesis stating that the mean number of errors for large spacing difference equals the mean number of errors for small spacing difference. We perform an ANOVA test on the factor MapType. R gives us the following results:

```
> maptype<-as.factor(maps$MapType)
> aov.data2 = aov(Errors~maptype, data=maps)
> summary(aov.data2)
            Df Sum Sq Mean Sq F value Pr(>F)
maptype      2  0.867   0.433  0.2753 0.7615
Residuals   27 42.500   1.574
```

The p-value of 0.7615 tells us that there is again no significant difference here. The tests thus tell us that there is no trend. However since this was a small scale test, we suggest enlarging the test population and in this case the number of different spacing differences to test, to gain a better understanding of a possibly existing trend.

8.3.3 Questionnaires

After each part of the test we let the users fill in a short questionnaire. We asked them if they used a certain strategy in exploring the textures. For the icons, we asked them about how recognizable they found both the random and the custom made icons, and in how far they thought they recognized some structure or logic in the icons, icon-word combinations. For each of the maps we inquired in how far they could easily distinguish between the densities felt. This gave us the following results.

8 out of 10 users found the second series of custom made icons more recognizable then the first series of random icons.

7 out of 10 users found that there was more structure or logic in the second series of icons.

For the maps 4 out of 10 users found the densities easiest to distinguish in the first map (grids with difference of 10 pixels between consecutive regions). Another 4 users found it easiest in the second map (grids with smaller difference between consecutive regions), and 2 users found them equally easy or difficult. All but one found the third map (dots) significantly harder to distinguish between regions.

8.3.4 Observations

During the test we observed the users and their behavior. Attention was given to the way users explore the textures. This resulted in the following observations.

All users used horizontal and vertical strokes when moving the pointer over the icons. 3 users also used diagonal movements and 2 users moved in a circular pattern. One user indicated he adapted his movements depending on what he felt.

In general users seem to pick a certain type of movement that suits them. Most users were not likely to change their exploration pattern over the course of different textures, by for example moving faster, slower or in another pattern. Instead they sticked to the same method throughout the different scenes.

At first users seem to look for identification in regularity, in the frequence of the pattern of bumpiness/roughness they are feeling, not a shape of any kind. Some users did try to form a mental picture of the texture. Also they take their time to form this mental image.

We noticed that for the second series of icons, see Figure 8.2, at least 4 users quickly understood that there was some kind of shape underneath. This showed in them suddenly exploring the sides of the shapes instead of further randomly moving the pointer left to right or up and down across the icon, as they did with the first series of random textured icons. However, only one user took noticeably less time to explore the custom textured icons. Perhaps this is because there was no time limit and they wanted to form a good mental image of the texture.

Also with the second series of icons, for example Figure 8.2(a), we noticed that some users spontaneously found the link between the higher, black region being to the left and the word 'left' associated with it. Finding the link they confirmed it for themselves by feeling to the right on the icon associated with the word 'right' and so on. This suggests that users benefit from a logical connection between the data and the feeling, and that it will be easier to remember.

When discovering a line pattern as in Figures 8.2(g), 8.2(h) and 8.2(i), most users used movements crossing the line, only one user was following the side of the lines. This suggests that they (subconsciously) count the number of bumps when crossing the whole icon. By repeatedly crossing the line on different locations, they eventually find that the texture contains a line.

One user suggested that the texture associated with a negative word like 'abort' should

feel tougher.

It is possible that when users are given a visual hint of what the textures look like, thus giving them some prior knowledge, they may handle a different strategy in identifying the icons. We believe that with prior knowledge they will be actively looking for features they have seen instead of randomly exploring something they don't know. They may even more quickly change their exploration pattern if they are not finding what they are looking for. This would result in a quicker identification because the process of forming a mental image can be skipped because they already have a visual image in their mind.

In the second scenario, the one with the maps, users took significantly longer to sort the map where a dotted texture was used. Small dots are easily missed when exploring the surface, while with grids it doesn't matter in which direction you move, you will always encounter a next line, either horizontally or vertically.

A common strategy used to explore the maps was to first get a general feel of all the different regions, and then start comparing regions to each other. When doubting about two regions, users tend to go over them slower, trying to count lines or paying extra attention to the spacing.

One user suggested that for the map with the grid texture and the smaller difference in spacing, as in Figure 8.3 below, it is easier to distinguish between the regions, because there is more haptic feedback from the smaller spacing than with the larger spacing.

8.4 Conclusion

Through our user test, though limited in size, we learnt the following things.

When using haptic icons as a representation of qualitative data, there is a difference between using randomly chosen textures and custom textures prepared to be better recognizable. The custom made textures cause less errors by the users and seem to be recognized faster. This appears from the test results, but the difference was not found significant by statistical analysis. A more extended user test should clarify this. Besides that, the participants themselves indicated this trend in the questionnaire. They perceived the custom textures to be more easily recognized and overall more logically connected to the words, being the data.

When exploring a texture people tend to make a mental image of what they are feeling.

If the textures are unknown, this process takes time. We suggest that when people are given a visual representation of the textures on beforehand, they will know what kind of textures to look for, and what they can expect. We believe that this may result in better and faster recognition and that users will be more confident in drawing their own conclusions. However to proof this would need an additional user test, since it was not tested in the experiment here.

Our test users used different ways of investigating a texture. Most of them started out with straight horizontal and vertical movements across the texture. Some sticked to this method all throughout the task, others also tried other movements when they couldn't find some characterizing feelings this way. When they do not know anything about the 'visual' shape of the texture, users try to get a feel of the overall roughness and bumpiness of the surface. When they discover a shape underneath some change their movement and go on exploring the shape, while others didn't pay attention to the shapes and also didn't change their feeling strategy. The exploration and interpretation strategy used, is different for different people. Some will pay more attention to frequency or bumpiness of the texture, and others will look for shapes. We cannot assume one or the other, but may again be able to guide the user in the process by giving him some more information on what he should expect in advance.

All users showed capable of identifying textures either used in the form of icons as on the surface of regions on a map. Touch is not equally efficient as vision though. Because of its sequential nature it allows for less information absorption in parallel, and it overall takes longer to form a mental image of what is felt. This makes that touch is not very useful for comparing pieces of data. However, touch can be useful to provide additional information on the point of focus, that is the point the user is already actively exploring. Also for visually impaired users, touch is, next to hearing the only alternative and can in this case be of great help.

In the case of the maps, the height field textures showed that they can be used to effectively represent densities. Grid textures are far better suited for this task than dot textures, because the user can not miss the lines when moving either horizontally or vertically. Dots however can be missed and make them harder to identify. When using grids with different spacings, the ones with the smaller difference in spacing amongst each other proved to be better recognizable compared to the grids with a greater difference amongst each other. Observations learned that regions with a larger inter-line spacing were harder to compare to each other, because the spacing provides no feedback, whilst the lines do.

CHAPTER 9

Conclusion

In this part, we have discussed how haptics can contribute to information visualization. We have illustrated this in a sample application giving two different data visualizations/*haptizations*. To limit the scope, only height fields were considered, but they prove to be capable of conveying information in a reliable way. There is a difference in the perception of haptic data vs. visual information, which cannot be denied. Nevertheless haptics can prove their worth for visually handicapped people as well as in adding additional data to other applications.

We have set a few questions to research and answer, and naturally there are lots more still to study. Through the execution of a user test we were able to get some answers and have made some useful observations that can serve as a basis for others to build upon. Our main conclusions were that for the representation of qualitative data using icons or alike, users benefit from a custom developed set of textures that are designed to be easily recognizable. In this process one must take into account the way users explore a texture. Observing the movements they made in our user test shows us that straight horizontal and vertical movements are made by everyone, complemented with other motions depending on what they believe to be feeling. This means that in designing well recognizable textures it is good practice to make them symmetric, both horizontally and vertically. Symmetric icons will produce the same feeling no matter from which side they are crossed with the cursor. Another suggestion we wish to add

is for seeing people not to use a haptic texture only, but rather give them a visual hint of the textures they can expect to feel. Based on their visual interpretation, users can get an idea of what the textures may feel like, and can use a more direct way for their identification strategy.

Lastly we conclude that the symmetry criterium also holds for textures used on maps. Making sure the textures are perceived in both horizontal and vertical direction, will prevent users from missing the texture at all. Additionally the flat spacing regions between texture elements should not be too large. Better place elements closer to eachother and make the difference between different textures smaller instead. Users benefit more from the feedback than from the larger difference in distance between elements.

Bibliography

[arg49] Argonne national laboratory, 1949. `http://www.anl.gov/index.html`, 12/06/2008.

[BHS97] C. Basdogan, CH Ho, and MA Srinivasan. Haptic rendering: Point- and ray-based interactions. 1997.

[Ble07] Geert Bleyen. Evaluation of haptic algorithms. Master's thesis, Universiteit Hasselt, Diepenbeek, Belgium, 2007.

[BOK+04] Brian T. Bethea, Allison M. Okamura, Masaya Kitagawa, Torin P. Fitton, Stephen M. Cattaneo, Vincent L. Gott, William A. Baumgartner, and David D. Yuh. Application of haptic feedback to robotic surgery. 2004.

[Bur96] Grigore C. Burdea. *Force and touch feedback for virtual reality.* John Wiley & Sons, Inc., 1996.

[Che99] E. Chen. Six degree-of-freedom haptic system for desktop virtual prototyping applications. 1999.

[Dav05] Rod David. B747-400 simulator, 2005. `http://www.simlabs.arc.nasa.gov/cvsrf/747_sim.html`, 03/08/2008.

[Dea06] Brandi Dean. Space shuttle canadarm robotic arm marks 25 years in space, 2006. `http://www.nasa.gov/mission_pages/shuttle/behindscenes/rms_anniversary.html`, 12/06/2008.

[FAG+08] P Feys, G Alders, D Gijbels, J De Boeck, T De Weyer, K Coninx, C Raymaekers, J Annegarn, K Meijer, H Savelberg, V Truyens, M Thijs,

N Goyens, and BO Eijnde. Robot-assisted rehabilitation of the upper limb in persons with multiple sclerosis: a pilot study. In *Robotic Helpers: User interaction, interfaces and companions in assistive an therapy robotics.* ACM/IEEE Human-Robot Interaction Conference (HRI08), 2008.

[GT54] R.C. Goertz and W.M. Thompson. Electronically controlled manipulator. 1954.

[h3d04] H3d.org open source haptics, 2004. `http://www.h3dapi.org/`, 19/08/2009.

[hal] Hal haptic library. `http://www2.edm.uhasselt.be/software/hal/`, 21/06/2008.

[HBKY00] M. Hollins, S. Bensmaïa, K. Karlof, and F. Young. Individual differences in perceptual space for tactile textures: Evidence from multidimensional scaling. *Perception & Psychophysics*, 62(8):1534–1544, 2000.

[HBS99] Chih-Hao Ho, Cagatay Basdogan, and Mandayam A. Srinivasan. Efficient point-based rendering techniques for haptic display of virtual objects. *Presence: Teleoper. Virtual Environ.*, 8(5):477–491, 1999.

[HCH00] V. Hayward and M. Cruz-Hernandez. Tactile display device using distributed lateral skin stretch. 2000.

[imm08] Immersion, 2008. `http://www.immersion.com/`, 19/06/2008.

[Joh01] K.O. Johnson. The roles and functions of cutaneous mechanoreceptors. *Current Opinion in Neurobiology*, 11:455–461, 2001.

[KHPH05] L. Kim, Y. Hwang, S.H. Park, and S. Ha. Dental training system using multi-modal interface. *Computer-Aided Design and Applications*, 2(5):591–598, 2005.

[Lab08] USD Internet Sensation & Perception Laboratory. Weber's law of just noticeable differences, 2008. `http://www.usd.edu/psyc301/WebersLaw.htm`, 10/08/2008.

[Lan98] J. Lander. Making kine more flexible. *Game Developer Magazine*, 1:15–22, 1998.

[McG02] M.R. McGee. *Investigating a Multimodal Solution for Improving Force Feedback Generated Textures.* PhD thesis, University of Glasgow, 2002.

[men08] Mentice, 2008. `http://www.mentice.com/`, 19/06/2008.

[Min95] Margaret Diane Rezvan R. Minsky. *Computational haptics: the sandpaper system for synthesizing texture for a force-feedback display.* PhD thesis, Cambridge, MA, USA, 1995.

[MRF+96] W.R. Mark, S.C. Randolph, M. Finch, J.M. Van Verth, and R.M.T. II. Adding force feedback to graphics systems: Issues and solutions. 1996.

[nov08] Novint technologies, inc., 2008. `http://home.novint.com/`, 19/06/2008.

[Por07] Pascal Porta. Extending an haptic api. Master's thesis, Universiteit Hasselt, Diepenbeek, Belgium, 2007.

[qt09] Qt - a cross-platform application and ui framework, 2009. `http://qt.nokia.com/`, 19/08/2009.

[r09] The r project for statistical computing, 2009. `http://www.r-project.org/`, 30/08/2009.

[rjS07] Rj studios inc., 2007. `http://www.rjstudios.com/`, 03/08/2008.

[RKK97] Diego C. Ruspini, Krasimir Kolarov, and Oussama Khatib. The haptic display of complex graphical environments. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 345–352, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[RVB02] D. Reiners, G. Voß, and J. Behr. Opensg: Basic concepts. *Proc. of OpenSG Symposium 2002*, 2002.

[SBM+95] K. Salisbury, D. Brock, T. Massie, N. Swarup, and C. Zilles. Haptic rendering: programming touch interaction with virtual objects. In *SI3D '95: Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 123–130, New York, NY, USA, 1995. ACM.

[SO06] J. Shopf and M. Olano. Procedural haptic texture. *Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 179–186, 2006.

[spi09] Genetica texture packs, 2009. `http://www.spiralgraphics.biz/packs/index.htm`, 19/08/2009.

[ST00] Inc. SensAble Technologies. *GHOST SDK Programmer´s Guide*, 2000. `http://www.inition.com/inition/product.php?URL_=product_software_sensable_ghostsdk&SubCatID_=82`, 03/08/2008.

[ST01] Inc. SensAble Technologies. Customer succes story: Rj studios inc. case study on the SensAble corporate website, 2001. http://www.sensable.com/documents/Galleries/CaseStudies/CaseStudy_RJStudios.pdf, 19/06/2008.

[ST06] Inc. SensAble Technologies. Specifications comparison for the phantom premium 1.0, 1.5, 1.5 high force, and 3.0 haptic devices, 1993–2006. http://www.sensable.com/documents/documents/Premium_Models_Comparison.pdf, 20/06/2008.

[ST08a] Inc. SensAble Technologies. Phantom haptic devices, 1993–2008. http://www.sensable.com/products-haptic-devices.htm, 19/06/2008.

[ST08b] Inc. SensAble Technologies. Specifications for the phantom desktop and phantom omni haptic devices, 1993–2008. http://www.sensable.com/documents/documents/PDesktop_POmni_Specifications.pdf, 20/06/2008.

[SWND05] Dave Shreiner, Mason Woo, Jackie Neider, and Tom Davis. *OpenGL Programming Guide: The Official Guide to Learning OpenGL*. Addison-Wesley Professional, 5th edition edition, 2005.

[TCES94] H.Z. Tan, B. Cheng, B Eberman, and M.A. Srinivasan. Human factors for the design of force-reflecting haptic interfaces. 1994.

[TFNM05] V. Theoktisto, M. Fairen, I. Navazo, and E. Monclus. Rendering detailed haptic textures. *Workshop on Virtual Reality Interaction and Physical Simulation*, 2005.

[UHK07] Bertram Unger, Ralph Hollis, and Roberta Klatzky. Jnd analysis of texture roughness perception using a magnetic levitation haptic device. In *WHC '07: Proceedings of the Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 9–14, Washington, DC, USA, 2007. IEEE Computer Society.

[UHK08] B. Unger, R. Hollis, and R. Klatzky. The geometric model for perceived roughness applies to virtual textures. *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2008. HAPTICS 2008. Symposium on*, pages 3–10, 2008.

[Ung05] B.J. Unger. *Texture Perception with Realistic Probes in Virtual Haptic Environments*. PhD thesis, 2005.

[wik08a] Bump mapping, 2008. http://en.wikipedia.org/wiki/Bump_mapping, 06/08/2008.

[wik08b] Computer-aided design, 2008. http://en.wikipedia.org/wiki/CAD, 03/08/2008.

[wik08c] Haptic technology, 2008. http://en.wikipedia.org/wiki/Haptic, 03/08/2008.

[wik08d] Somatosensory system, 2008. http://www.reference.com/browse/wiki/Somatosensory_system, 03/08/2008.

[WK97] J.M. Weisenberger and M.J. Krier. Haptic perception of simulated surface textures via vibratory and force feedback displays. pages 55–60, 1997.

[x3d09] X3d for developers, 2009. http://www.web3d.org/x3d/, 19/08/2009.

[ZS95] C. B. Zilles and J. K. Salisbury. A constraint-based god-object method for haptic display. In *IROS '95: Proceedings of the International Conference on Intelligent Robots and Systems-Volume 3*, page 3146, Washington, DC, USA, 1995. IEEE Computer Society.