TEI CRETE – DEPT. OF MUSIC TECHNOLOGY AND ACOUSTICS

DESIGN AND DEVELOPMENT OF AN INTERACTION SURFACE (AUDIO TABLET) FOR MUSICAL EXPRESSION AND PERFORMANCE

Romaios Aristeidis

FINAL PROJECT DISSERTATION, JUNE 2008

Supervisor: Paschalidou Stella

Εισαγωγικό Κείμενο

Στις μέρες μας ένας υπολογιστής βρίσκεται σχεδόν σε κάθε σπίτι. Η επεξεργαστική ισχύς προσφέρει τη δυνατότητα να επεξεργαζόμαστε πολλές πληροφορίες ταυτοχρόνως, με αποτέλεσμα πολλοί πλέον μουσικοί αλλά και τεχνικοί ήχου, να αντικαθιστούν τα παλαιότερα ακριβά μουσικά όργανα και μηχανήματα, με virtual pc based. Το βασικό πρόβλημα που υπάρχει στο χειρισμό αυτών, είναι πως ο χρήστης δεν μπορεί να αφεθεί μόνο στο ηχητικό αποτέλεσμα, όπως και θα έπρεπε, αλλά πρέπει να είναι μπροστά σε μία οθόνη, ή ακόμα και σε ένα πλήθος από knobs ή faders, και να πρέπει να τα χειριστεί όλα αυτά μόνο με τα δύο του χέρια. Άρα δεν μπορεί να διαχειριστεί ταυτόχρονα πολλές λειτουργίες για το επιθυμητό αποτέλεσμα. Όλη αυτή η διαδικασία λοιπόν, αποσπά τον χρήστη από το βασικό του στόχο, τον ήχο. Ένα ακόμα μειονέκτημα είναι αυτό της επικοινωνίας ανθρώπου και υπολογιστή. Δυστυχώς δεν υπάρχει αμφίδρομη επικοινωνία μεταξύ τους. Για να φτάσει σε ένα αποτέλεσμα ο χρήστης πρέπει να περιπλανηθεί σε αντίστοιχα menus και στη συνέχεια για να αντιληφθεί το αποτέλεσμα των πράξεων του, πρέπει να ακούσει (τον ήχο) και να δει (την οθόνη). Αυτή την «τρύπα» στον τρόπο επικοινωνίας ανθρώπου – υπολογιστή, έρχεται να καλύψει η αναζήτηση και η δημιουργία νέων μέσων αλληλεπίδρασης.

Στη δική μας εργασία έχουμε ως στόχο την ανάπτυξη και το σχεδιασμό μιας επιφάνειας αλληλεπίδρασης ως εναλλακτικό μέσο ελέγχου παραμέτρων ηχητικών δειγμάτων σε πραγματικό χρόνο. Στοχεύουμε στη δημιουργία ενός μέσου ελέγχου του ήχου, αλλά και παραμέτρων αυτού, από έναν ή περισσότερους χρήστες που έχουν το ρόλο του εκτελεστή, αλλά μπορούν και να παρέμβουν στα ηχητικά δείγματα που χρησιμοποιούν χωρίς να κρίνεται αναγκαία η γνώση ηχοληψίας. Το πρακτικό μέρος της εργασίας μας περιλαμβάνει την κατασκευή μιας επιφάνειας εργασίας, στην οποία ο χρήστης θα μετακινεί μικρούς κύβους, οι οποίοι θα αντιστοιχούν σε αυτόνομα αρχεία ήχου. Η μετακίνησή τους πάνω

στην επιφάνεια θα αντιστοιχεί σε αλλαγή κάποιων ηχητικών παραμέτρων και θα ανιχνεύεται με τη χρήση μιας απλής web κάμερας. Πιθανή εφαρμογή αυτής της ιδέας θα αφορά στην αλλαγή της χωροτοποθέτησης του ήχου και στην επιλογή των audio samples που αντιστοιχούν στους κύβους, καθώς και σε άλλες παραμέτρους τους, όπως η αλλαγή της έντασης κάθε ηχητικού δείγματος ξεχωριστά ή η προσθήκη reverb για καλύτερη αίσθηση του χώρου.

Abstract

The aim of my project will be to develop an alternative interface for musical expression and performance, which will be intended for non-expert users, i.e. with no sound engineering or music background. In specific, one or more users (the audience) that will act as the performer/s will be able to select audio samples and control their parameters in real-time. In this way, we encourage the individual- or multiple-users' music creation. I believe that the haptic element for the control of the audio parameters and the acoustic and visual feedback are able to create an effective, for this purpose, interactive system.

The practical part of my project will include the creation of an interaction surface (audio tablet). On the surface area the user will be able to move small cubes. Each cube will be connected to a single sampler and the user will have the opportunity to load each cube with different audio samples. Their movement on the surface area will correspond to the modification of certain sound parameters and will be detected with the use of one simple web camera. A possible application of this basic idea may concern the samples' diffusion in space, as well as the control of other parameters, such as volume, or reverb for a better representation of the space.

The aim of this project is to develop a low-cost system, which will be easy to use also by non-expert people. The possible use of this system by children and people with special needs (kinetic or mental limitations) remains an open issue.

Table of Contents

1. Introduction – The Control Problem

In recent years the drop in personal computers' price and the rise of processing power has given us extended possibilities. The evolution of digital technology has resulted in personal computers that are fast enough to process a number of applications in parallel. The revolution of digital technology has influenced the sound engineering domain as well. In specific, we are nowadays able to handle a vast number of parameters simultaneously in real-time via a wide range of parameters. This led musicians to create new pc-based systems for music creation and a large variety of programs like DAWs (Digital Audio Workstations), samplers, virtual instruments, virtual effects and many more. Thus, any user can now have a computer-based home studio without the need of expensive hardware or an expensive recording mixer.

Nevertheless, things have changed very little in the way we control these. Thus, limitations are introduced to the possibilities of current interactive systems due to the use of the classical input devices and methods, such as keyboard and mouse. J. A. Moorer foresees that the main problem digital audio engineers will be facing in a few years will not be how to perform a particular manipulation of sound, but how the amount of power that will be available can possibly be controlled [MOO00].

In the sound engineering domain, we still have the classical analog or digital consoles (with a large number of rotary knobs and sliders) for the control of the audio samples and their parameters (Multi-Parameter Controllers for Audio Mixing). Taking into account that humans only have two hands, this is a big restriction in the number of parameters that we can handle simultaneously.

To my point of view there seems to be a lack of expressiveness in the mixing procedure due to the control methods still used. It

is very difficult to concentrate on a computer monitor and to use only a computer mouse and/or keyboard (or knobs and sliders) for everything we need to do, as this often may distract the user's attention from the ultimate goal, which is the actual sound. [VER94]

There is no 'dialogue' between user and computer, instead the computer responds instantly to the user's hand movements. It is then up to the users to interpret the sound produced and to use this as feedback to whatever process they are currently involved in. The computer does not set the agenda or dictate the conversation or insist the user selects from a set of predefined options, but instead provides an environment for creative exploration. This mode of operation is very different to the accepted means of communication with a computer. Traditionally the software is there to gather data, and often does so by dominating the interaction. Even in those situations where the user is fully in charge of the interaction, it usually takes place at a certain level of language ability (for example, the need to read, interpret and take action on hierarchically-arranged menus). [HUN03]

The need for establishing a more intuitive interaction method between user and computer for mixing (and other) purposes leads to the development of new controllers and controlling metaphors.

As music creation and expressivity in live performance situations will be in the target of our project, the user can be imagined as "a new kind of performer". So, in our concept there will be a shift of the user's role from the strict definition of a "sound engineer" to a new kind of musician/performer. The system we are aiming at will involve the development of an alternative controller for the modification of the computer-based audio parameters. The concept shows similarities to a "new musical instrument" in the sense that there is a separation of the controlling surface (interface) and the sound-producing device.

Our approach in this perspective will involve the use of interactive multi-modal systems that make use of sound, vision and touch. A first step in this direction will be attempted in this project, by which we will try to develop an alternative system for diffusing audio samples in a more intuitive manner than the analog or digital consoles.

2. Human Computer Interaction and Music Expression

The field of study known as Human Computer Interaction (HCI) emerged as a subject area in its own right during the 1980's. The ACM Special Interest Group on Computer-Human Interaction recently defined it as:
"... a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them." [HBC96]

A comparison of this definition with one just a decade earlier demonstrates the rapid change of emphasis in the subject area. Tom Moran described the 'Man-Machine Interface (MMI)', as it was then known, as:
'Those aspects of the system that the user comes into contact with" [MOR81], which rather implies a fait accompli where users just have to put up with whatever the engineers have designed. [HUN03]

According to 'Wikipedia', "Human–computer interaction (HCI) -- alternatively man-machine interaction (MMI) or computer–human interaction (CHI)-- is the study of interaction between people (users) and computers". [Wikipedia, "Human-computer interaction"]

According to 'EARS: ElectroAcoustic Resource Site', "Interactivity refers broadly to human-computer interaction, or human-human musical interaction that is mediated through a computer, or possibly a series of networked computers that are also interacting with each other. The means and manner by which a human may interact with a machine for the purpose of music making has proven an area of immense research and activity since the days of early electronic instruments. The term 'interactive instrument' is used without any great consistency, in

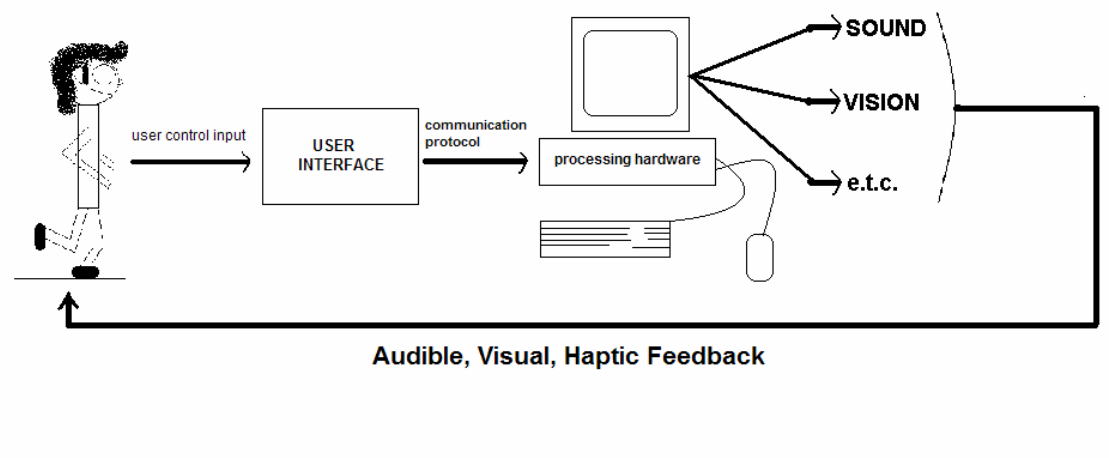
part due to the immense scale of current research and activity in the field." [Ears, "Interactivity"]

The problem of HCI can be viewed as two powerful information processors (human and computer) attempting to communicate with each other via a narrow-bandwidth, highly constrained interface [TUF89]. Acoustic instruments are ready to produce sound when we act on them (hit a string, press a key on the piano, e.t.c.), because their playing interface (interaction surface) is inseparable from their sound source. In contrary, in interactive computer-based audio systems (electronic & computer instruments) the interface (hardware input device, with which the user physically interacts) is a completely separated piece of equipment from the sound producing source [HWP02].

Andy Hunt also mentions, that "there is no 'dialogue' between user and computer, instead the computer responds instantly to the user's hand movements. It is then up to the users to interpret the sound produced and to use this as feedback to whatever process they are currently involved in. The computer does not set the agenda or dictate the conversation or insist the user selects from a set of predefined options, but instead provides an environment for creative exploration". [HUN03]

A classic pc-based system for producing and controlling musical parameters in real time is composed of:

- The user's control input data to the computer based system.
- The process of data by the computer.
- The processed data parameters output to audio, visual or other type of feedback.



{IMAGE 1}: “Structure and separation of the components in interactive musical systems”

Image 1 describes the structure and separation of the components in interactive musical systems. The communication between these parts takes place at two levels; the ***physical*** and the ***logical level***.

At the “physical level” (electrical signal) of the communication, we need to use a hardware device that will establish the physical connection between the human gestures and the computer. The hardware device with which the user directly interacts (through different possible modalities) is called “Input Device” or “User Interface”. According to ‘Ears’, in general an interface is a “software or hardware-based device or protocol, which serves as an intermediary between a computer and a peripheral device or between two different systems which permits the exchange of information between them” [Ears, “Interface”].

On the “logical level” of the connection, the transfer of the gesture data information into the computer is based on a communication protocol (such as MIDI, OSC, e.t.c.). On top of this, a conceptual process of translating the gesture data information to any kind of output –such as sound, vision, force feedback or smell— is involved, which is called “mapping” .

3. Input Devices

A mechanical input device encodes motion into a signal that can be read by the computer [JSM94].

Any technology that can transduce human gesture or movement into electrical signal is available to build sensors. Regardless of sensor type, it is important to recognize not only what is being measured, but how it is being measured: resolution and sampling rate are fundamental aspects to take into account. The commonly used technologies include infrared, ultrasonic, Hall Effect, electromagnetic and video [SAP00].

3.1. Computer borrowed input devices – questioning

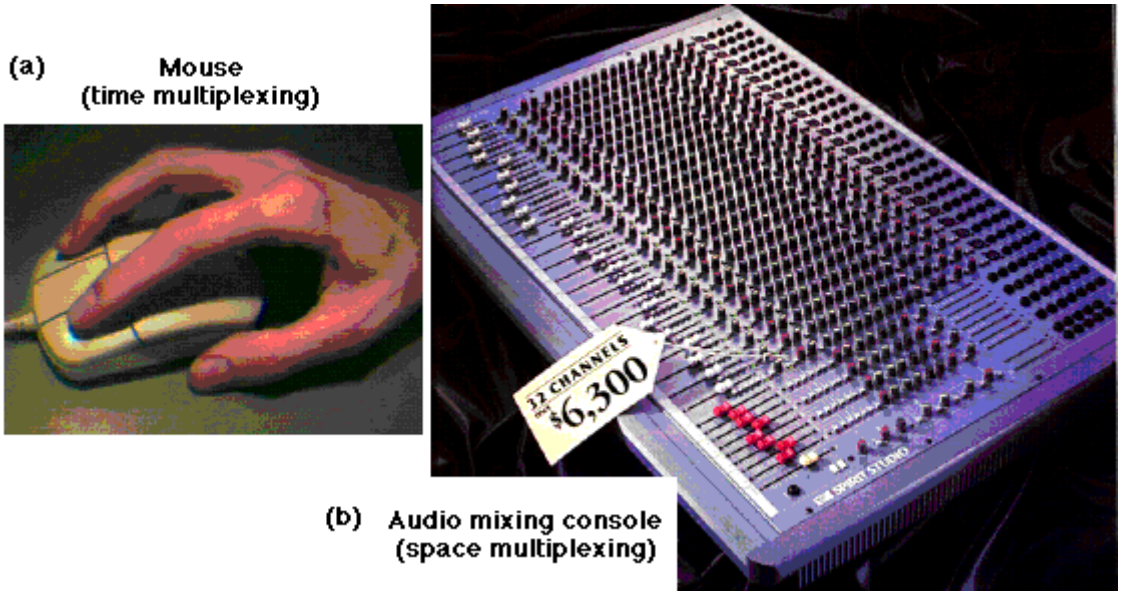
Keyboard, switch, pushbutton, slider, joystick, mouse, graphic tablet, touch-sensitive pad, etc. belong to the computer hardware. However, although such devices have been intensively used as gestural input devices for musical purpose, they hardly fit musical gestuality requirements [SAP00].

According to George W. Fitzmaurice and William Buxton [FB97], the primary principle behind Graspable UIs is to adopt a space-multiplexed input design. Input devices can be classified as being *space-multiplexed* or *time-multiplexed*.

With space-multiplexed input, each function to be controlled has a dedicated transducer, each occupying its own space. For example, an automobile has a brake, clutch, throttle, steering wheel, and gear shift which are distinct, dedicated transducers controlling a single specific task [FB97]. A classical example of a space-multiplexed input device is the QWERTY keyboard, i.e. the keyboard all personal computers have. The name “QWERTY” comes from the first six letters in the top alphabet row (the one

just below the numbers). The keyboard arrangement was considered important enough because its' aim was to speed up typing. This means, that all the parameters are physically spaced out on its surface and the user can reach all the hardware functionality, although the subjects are limited to the operation of only some of them at the same time (due to 10 fingers). [FB97]

In contrast, time-multiplexing input uses one device to control different functions at different points in time. For instance, the mouse uses time multiplexing as it controls functions as diverse as menu selection, navigation using the scroll widgets, pointing, and activating buttons [FB97].



{IMAGE 2}: "(a) Time multiplexed – (b) Space multiplexed" [FB97]

This means, that with the computer mouse we choose each time one specific task (we click on it). This is helpful for office tasks, but not for music systems. According to [OZS06], "Imagine a virtual keyboard where the mouse is used to click each letter. The process of writing a sentence would take much longer than using the keyboard". This simple example demonstrates the weaknesses of the mouse.

Consider the task of text entry with a space- versus time-multiplex input style. The time-multiplex input scheme could, for example, employ a Morse code button where a single key is used to encode each alpha numeric character. In contrast, the space-multiplex input scheme could use a QWERTY keyboard where there is one key per character. The Graspable UI argues for shifting interactions to a more space-multiplexed design [FB97].

3.2. Music dedicated input devices

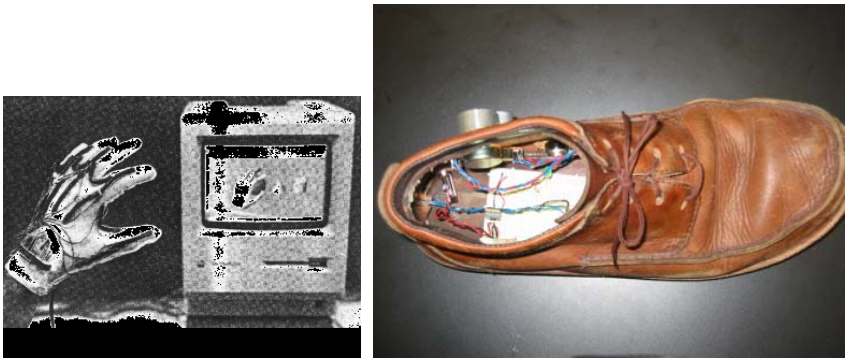
On the other hand, musical keyboards, foot pedals, drum pads, ribbon controllers, breath controllers and other instrument like controllers belong to the MIDI world. They have been exclusively developed for musical aim and they capture musical gesture on the scheme of the MIDI protocol, but they are sometimes used for other purposes, lighting controls for example [SAP00].

According to [JSM94], designing effective input devices requires that the motion clearly conveys the intention of the user and complements his or her physical capabilities. The design of current input devices and their interaction techniques have been driven more by what is technologically feasible than from an understanding of human performance. Their success relies, in part, on the well-documented human ability to adapt. To design and select more effective input devices and interaction techniques, we need to use a deeper understanding of task, device, and the interrelationship between task and device from the perspective of the user. Such understanding may come from the intuition and judgment of designers and, perhaps, from empirical studies of specific new devices.

There seems to be a necessity for developing task-specific input devices, which means that a different user interface needs to be devised for each new music interaction system; we haven't reached yet the point of developing a generic user interface and, in fact, the question arises of whether this will ever be the case.

A wide range of task-specific input devices for interactive audio applications has been devised by integrating such sensors. We will present a few examples by organizing them in the following non-exclusive categories:

3.2.1. Wearable Input Devices



{IMAGE 3}: “Glove and shoes input devices” [IMG3]

Wearable Input devices are clothes, shoes, data-gloves provided with types of sensors. The user-artist wears them and through his/her movements sends gesture data information to a computer.

A few examples of such projects include:

3.2.1.1. Bodysuit Controller

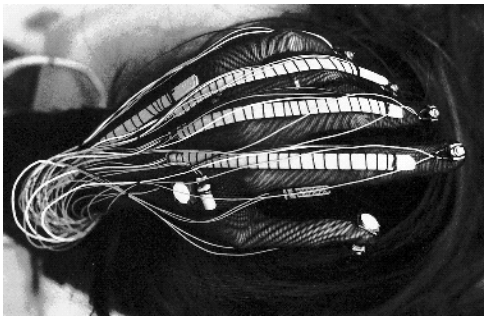
The “BodySuit” is equipped with 12 sensors, which are placed on each joint of the body, such as wrist, elbow, shoulder, arm, ankle, knee, and the beginning of the leg. The bending sensors are placed on the outer sides of the arms and on the front sides of the legs and fixed on a suit. Each sensor is connected with a cable to a box, and then it is and an A/D interface. The performer needs to wear the suit, but doesn’t need to hold it in his hands.

{IMAGE 4}: “The Bodysuit” [GOT06]

Therefore, his gesture doesn't have to be based upon playing an instrument, but could be liberated to become a larger gesture, like a mime. This allows for collaboration with a person in a different field, for instance a dancer or an actor. That is to say it can be well adapted to a performance and musical theater situation. One may consider this as a body instrument. This efficiently works in a percussionist-like gesture. [GOT06]

3.2.1.2. Ladies Glove

Laetitia Sonami's (now at CNMAT, U.C. Berkeley) "Lady's Glove" is an extremely dexterous system, with bend sensors to measure the inclination of both finger joints for the middle 3 fingers, with microswitches at the end of the fingers for tactile control, Hall sensors to measure distance of the fingers from a magnet in the thumb, pressure sensing between index finger and thumb, and sonar ranging to emitters in the belt, shoe, or opposite arm. [SONLG]



{IMAGE 5}: "Ladies Glove" [SONLG]

3.2.1.3. Glove-TalkII

GloveTalkII is a system that translates hand gestures to speech through an adaptive interface. Hand gestures are mapped continuously to ten control parameters of a parallel formant speech synthesizer. The mapping allows the hand to act as an artificial vocal tract that produces speech in real time. This gives an unlimited vocabulary in addition to the direct control of fundamental frequency and volume. Currently, Glove-TalkII uses several input devices (including a Cyberglove, a ContactGlove, a three space tracker, and a foot pedal), a parallel formant speech

synthesizer, and three neural networks. One subject has been trained to speak intelligibly with Glove-TalkII. He speaks slowly with natural sounding pitch variations than a text-to-speech synthesizer. [FELGT]

{IMAGE 6}: “Glove-TalkII graphic” [IMG3]

3.3. Non-intrusive Input Devices

By “non-intrusive” we mean those input devices that are not based on user’s contact, hence they don’t limit the user’s intended movements. These input devices use different methods (Infrared, Bluetooth ...) for receiving the human’s actions and send information to computer based systems. Usually they have high costs and they are used for other purposes, like scientific applications (e.g. medical). On the other hand, their characteristics allows their use for experimentation, e.g. for musical purposes, in computer based science and technology.

A few examples follow:

3.3.1. The Theremin

The Theremin was invented in 1919 by a Russian physicist named Lev Termen (his name was later changed to Leon Theremin).

{IMAGE 7}: “Leon Theremin and the Theremin” [Theremin]

Besides looking like no other instrument, the Theremin is unique in that it is played without being touched. Two antennas protrude

from the Theremin - one controlling pitch, and the other controlling volume. As a hand approaches the vertical antenna, the pitch gets higher. Approaching the horizontal antenna makes the volume softer. Because there is no physical contact with the instrument, playing the Theremin requires precise skill and perfect pitch [Theremin].

The spooky sound of the Theremin was used in several movie soundtracks during the 1950's and 1960's.
During the 60's and 70's, bands such as “Lothar and the Hand People”, the “Bonzo Doo Dah Dog Band”, and “Led Zeppelin” brought the Theremin into the public eye for a short time. Then, the theremin slipped back into obscurity until the recent revival of the 1990s. Today, lots of bands use Theremins, though few in a musical context.

3.3.2. Laser

A laser sensor can work as a precise optical tracker for demanding tracking environments, as it can continuously track and output position and orientation data to a host computer. Examples of its implementation include head and object tracking in simulator, medical and 3D visualization applications.

A consumer product that makes use of this type of laser tracking technology is “Laser BIRD 2” by “Ascension Technology Corporation”. [LBird2]

{IMAGE 8}: “Laser Bird 2” [LBird2]

3.3.3. Inertial and GPS

Inertial and GPS Navigation systems are made for measuring motion, position, orientation and velocity at high speed rate. The RT4000 family products by “OXTS” company are such implementation examples. With a high speed (250Hz), they use Inertial Navigation System (INS) technology and combine this with the Survey Grade GPS receivers. The systems improve the performance over GPS-only ones by providing accurate in urban or tree-covered environments. [OXTS]

{IMAGE 9}: “OXTS RT4000” [OXTS]

3.3.4. Electromagnetic

A technology based on electromagnetic sensors, can be used for the position/orientation sensing requirements of 3D applications and environments and it is ideal for head tracking, biomechanical analysis, computer graphics, cursor control, and stereotaxic localization. “INITION” company produces “Polhemus Patriot”, which makes use of electromagnetic sensors. The *Patriot* is a 3D digitizer and a dual sensor motion tracker. Computing the position and orientation of a small sensor as it moves through space, provides dynamic, real-time measurements of position (X, Y, and Z Cartesian coordinates) and orientation (azimuth, elevation, and roll).

In the motion capture session, the movements of one or more actors are sampled many times per second. High resolution optical motion capture systems can be used to sample body, facial and finger movement at the same time. [Inition]

{IMAGE 10}: “Polhemus Patriot” [Inition]

3.3.5. Vision-based systems

Special reference will be made to these systems in the next chapter, as this is the control method that has been used in this project.

There are plenty of computer based systems (video game consoles, personal computers and others) that make use of video cameras to capture images from whole-body movements in real time and send these datasets (position, orientation etc) as control parameters to several applications, like the ones just mentioned. Video cameras (analog, digital, in IR or not) function as input devices in this case.

3.3.5.1. 2D video motion analysis

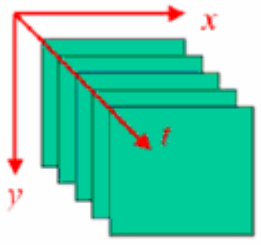
To understand the way a computer represents and processes a digital image, we have to start from scratch and explain the basics of a digital signal. Digital technology breaks information down into discrete pieces and represents those pieces as numbers. Each number represents the voltage level of one short period of time (perhaps 40,000 measurements every second). The number of measurements per second is called the sampling rate. The higher the sampling rate is, the more measurements will be taken each second, and the digitized signal will take up more memory in the computer. But because changes that occur in the signal between samples are lost, the sampling rate must be sufficiently high. On the other end, due to information size and processing power an upper limit is also introduced; after all the

sampling rate must at least be as high as the control rate, the audio parameters dictate.

All information in a computer is stored and managed as binary numbers. The binary number system only has two digits (0 and 1). A single binary digit is called a bit. A single bit can represent two possible items or situations. The bit is either 1 or 0, like an electrical switch that is turned on or off. Two bits taken together can represent four possible items, because there are exactly four combinations of two bits (00, 01, 10, and 11). Similarly, three bits can represent eight unique items, because there are eight combinations of three bits. In general, n bits can represent 2^n unique items.

A video is just a sequence of images over time. The images can be of all the types (e.g. binary images, grayscale images or color images) with different resolutions and bit depths. The more information you have in each picture, the larger the video file will be. But the framerate of the video also influences the size of the video. The more images the video has each second, the more information we will need to store.

An image is described with the function: $f(x,y)$. We can describe a video with the function: $f_t(x,y)$. At each timestep t, we have an image $f(x,y)$. [NBT03]



A number of images over time.

{IMAGE 11}: “A number of images over time” [NBT03]

Some of the image/video processing techniques include:

Threshold

Thresholding is a point processing technique that only considers one pixel in the input image to produce a corresponding pixel in the output image. This is why it is referred to as point processing. The input to a threshold operation is often a grayscale or color image. In the simplest implementation, the output is a binary image, i.e. black-white image. [NBT03]

Blob Tracking

This is a useful method for tracking isolated moving objects within an image (usually a non-changing background that is initially excluded from the picture) and focusing merely on a blob. Blob is short for “Binary Large OBject”. There are different methods to track a blob, both in binary images and in color images. [NBT03]

Binary image Blob tracking

In this case blobs are recognized due to the fact that they are moving in contrary to the still background. Moving objects are represented by white pixels and background (still) areas are represented by black pixels, i.e. images are binarized according to this rule. When applying blob tracking to a binary image, there are four parameters that need to be computed in a special order; first the center of the blob, followed by the variance, then the size of the surrounding box and last the corner positions of the box. [NBT03]

Color image Blob tracking

In this case blobs are recognized according to their colour. As colour comes in a number of (even slight) variations, it is necessary to first define the colour variation region for each color to be tracked, through a *color histogram* or *Gaussian distribution* (in HSV, RGB, etc). [NBT03]

3.3.5.2. Video Tracking Programming Environments based on Image Processing Techniques

An extensive (but not exhaustive) list of such video tracking programming environments follows:

EyesWeb

EyesWeb refers both to the research projects of InfoMus Lab on multimodal interactive systems and expressive gesture, and to the open software platform to support the development of real-time multimodal distributed interactive applications. The EyesWeb project started in 1997, as a natural evolution of the HARP Project (see www.infomus.org).

The current release of the open software platform is EyesWeb XMI (eXtended Multimodal Interaction). The EyesWeb software platform has been developed in EU IST projects in the 5th (MEGA, www.megaproject.org) and 6th Framework Programme (TAI-CHI, Tangible Acoustic Interfaces for Computer Human Interaction). EyesWeb has been adopted in several other EU projects, has been licensed to more than 15,000 individual users, companies, and institutions. EyesWeb is also used in University courses and summer schools (e.g. the New York University Summer Program on "Music, dance and new technologies").

Eyesweb is based on the Intel OpenCV (Open Source Computer Library) and works under Microsoft Windows. The Intel OpenCV is a collection of open source code (C, C++, known algorithms) that is being used and enriched by scientists worldwide.

The EyesWeb research project aims at exploring and developing models of interaction by extending music language toward gesture and visual languages, with a particular focus on the understanding of affect and expressive content in gesture. For example, in EyesWeb the aim is to develop methods able to distinguish between the different expressive content from two

instances of the same movement pattern, e.g., two performances of the same dance fragment.

EyesWeb helps us create interactive-digital-multimedia applications. It gives us, also, the possibility to create sounds and images in real time, through several modalities (vision, touch, etc), although it is mainly focused on visual motion tracking. With EyesWeb, we are able to determine objects, persons and gestures or locate movements.

{IMAGE 12}: “Eyesweb Interface” [InfoMus]

EyesWeb is being used by artists all over the world, for the exploration of the universe of electronic music and the optics. [InfoMus]

Isadora

Isadora is a commercial product designed by composer and media artist Mark Coniglio and it is accessible online [TroikaTronix]; an academic discount is offered.

Isadora is a graphic programming environment for Macintosh, but now has a Windows version in public beta. It provides interactive control over digital media, with special emphasis on the real-time manipulation of digital video.

{IMAGE 13}: “Isadora Interface” [TroikaTronix]

Isadora is a user friendly interface with the following general features:

Realtime Video Processing Modules -- Isadora provides a selection of real-time video processing modules, both those built in to Isadora, and those supplied by *FreeFrame*, an evolving, open-source standard for video-processing plugins. Over forty FreeFrame plugins are available from their homepage, on the internet.

Hardware Accelerated Rendering - Isadora now uses the power of our built-in graphics card to speed rendering to the screen, and allow different types of 3D effects. This new feature also means that there are more flexible ways to composite images on the screen.

G4 Native Video Processing – Yields extremely fast video processing (MacOS version only)

Control Panels – We can create a user interface for our Isadora program. Version 1.1 features several new controls, including Dials, Radio Buttons, and more; and, we can use our own graphics customize the look of sliders, buttons and the background of the Control Panel.

Custom User Actors – We can create our own "actors" by grouping together several Isadora modules. These can be saved for use in other patches, or to be shared with other users.

Snapshot Feature – Store and instantly recall the settings within a Scene.

Isadora SDK – With this feature, we can create our own Isadora modules in C or C++.

Isadora also offers numerous ways to render our video; in real-time to an external monitor or video projector, to a DV Camera (MacOS X only), or to render its output to a movie file. [TroikaTronix]

EyeCon

Eyecon's main use has been to facilitate interactive performances and installations in which the motion of human bodies is used to trigger or control various other media (music, sounds, photos, films, lighting changes, etc.).

{IMAGE 14}: “EyeCon Interface” [EyeCon]

Eyecon does this using a video feed from the performance or installation area (any normal video camera may be used). When the video signal is fed into the computer, the image appears in the main window of the program. Then we are able to draw lines, fields or other elements over the video picture. [EyeCon]

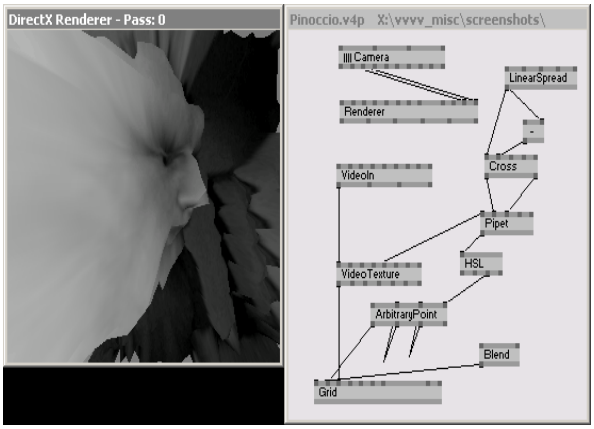
If a person then moves into the video image and some part of their body touches one of the elements we have drawn on, then an event can be triggered, for example a certain sound might be heard. Alternatively, if we have drawn a field, eyecon can measure the amount of motion occurring within that field. [EyeCon]

Additional features let us track the position of persons within the performance area, measure their height, their width, their overall size or the degree of left-right symmetry in their shape (assuming they are facing the camera). These control elements may each be assigned to a different output. Any number of

elements may be used simultaneously in any combination. Multiple cameras may also be used, though not simultaneously (unless you have more than one computer running). Eyecon is only available for PCs and needs at least Directshow 8.1 installed. [EyeCon]

vvvv

"vvvv" is being developed by the vvvv group. "vvvv" is a toolkit for real time video synthesis. It is designed to facilitate the handling of large media environments with physical interfaces, real-time motion graphics, audio and video that can interact with many users simultaneously.



{IMAGE 15}: "vvvv interface" [vvvv]

"vvvv" uses a visual programming interface. Therefore it provides a graphical programming language for easy prototyping and development. [vvvv]

"vvvv" is real time. Where many other languages have distinct modes for building and running programs, vvvv only has one mode-runtime. "vvvv" is free for non-commercial use. [vvvv]

3.3.5.3. 3D motion capturing - reflection based

There are many ways to generate motion capture data using, e.g., mechanical, magnetic, or optical systems, each technology having its own strengths and weaknesses.

We are going to present and discuss about an optical marker-based technology that gives us a very clean and detailed motion capture data. This technique is often used in computer graphics for animating virtual human characters.

The first thing we need is a human (an actor for example), wearing a special designed uniform equipped with a set of 40-50 retro-reflective markers attached on it. These markers are tracked by an array of 6-12 calibrated high-resolution cameras at a frame rate up to 240 Hz. From the recorded 2D images of the marker positions, the system can then reconstruct the 3D marker positions with high precision (present systems have a resolution of less than a millimeter). Then the data are cleaned with the aid of semi-automatic gap filling algorithms exploiting kinematic constraints. Cleaning is necessary for missing and defective data, where the defects are due to marker occlusions and tracking errors. In many applications, the 3D marker positions can be directly used for further processing. [MUL07]

There are plenty of systems that use this kind of tracking. Most of them are very expensive and are developed for professional use. VICON systems are going to be briefly presented here.

Vicon works by tracking a number of infrared (IR) reflective markers using cameras that emit IR light and work in high frame rates (up to 2000 frames/sec), thus allowing capturing under many different environments and ambient light conditions.

{IMAGE 16}: “Vicon Camera” [IMG16]

Vicon works by tracking a number of infrared (IR) reflective markers using cameras that emit IR light. Essentially each camera image is processed in hardware by a component called the MX Ultranet. It computes multiple camera views into a list of 3D marker coordinates. These coordinates can also be mapped onto predefined rigid bodies providing matching 3D coordinates to marker names (e.g. wand tip). This information is then sent to the Vicon system via a TCP network protocol and can be accessed in real time using many different programming languages (e.g. C++, C#).

3.4. Haptic Input Devices

Haptic, from the Greek αφή (Haphe), means pertaining to the sense of touch. Touching is not limited to a feeling, but it allows interactivity in real-time with virtual objects [Wikipedia, “Haptic”].

Computer users have been exposed to Haptic Input devices through the most common input device, “the joystick”. Most joysticks are two-dimensional, having two axes of movement (similar to a mouse), but three-dimensional joysticks do exist. A joystick is generally configured so that moving the stick left or right signals movement along the X axis, and moving it forward (up) or back (down) movement signals are sent along the Y axis. In joysticks that are configured for three-dimensional movement, twisting the stick left (counter-clockwise) or right (clockwise) movement signals are sent along the Z axis. These three axes - X Y and Z - are, in relation to an aircraft, roll, pitch, and yaw.

{IMAGE 17}: “Joystick input device” [IMG17]

An “analog joystick” is a joystick which has continuous states, i.e. returns an angle measure of the movement in any direction in the plane or the space (usually utilizing potentiometers) and a “digital joystick” gives only on/off signals for four different directions, and mechanically possible combinations (such as up-right, down-left, etc.). Digital joysticks were very common as game controllers for the video game consoles, arcade machines, and home computers of the 1980s.

Additionally joysticks often have one or more fire buttons, used to trigger some kind of action. These are simple on/off switches. Some joysticks have “force feedback” capability. These are thus active devices, not just simple input devices. The computer can return a signal to the joystick that causes it to resist the movement with a returning force or make the joystick vibrate.

Most I/O interface cards for PCs have a joystick (game control) port. Modern joysticks (as of 2007) mostly use a USB interface for connection to the PC.

Our project, as already mentioned, will concentrate only on a specific type of controller, known as “Tablet Controller”. Tablets can be regarded as haptic input devices, or even in fact as “graspable” or “tangible” input devices.

A few musical applications of such devices follow:

3.4.1. Lemur

The Lemur is a portable controller for live performance computer music applications featuring a 12” LCD display and a touch screen interface that can simultaneously track multiple fingers.

{IMAGE 18}: “Jazzmutant – Lemur” [jazzmutant]

The Lemur communicates with a host computer over 100-baseT Ethernet using the Open Sound Control (OSC) protocol developed at UC Berkeley's Center for New Music and Audio Technologies (CNMAT). [jazzmutant]

Users configure the Lemur using an editor application that runs on Mac OS X, Windows XP and Linux. Users drag and drop graphical control objects such as faders, buttons, two-dimensional area controllers and status monitors to create interfaces. After a collection of interfaces is uploaded to the Lemur, the device sends data to parameters in a sound-generating application when a user touches objects in the display. Performers can flip between interfaces on the Lemur using buttons located above the touch screen. The Lemur's interface objects can be customized with JazzMuttant's "physical" properties that include friction, smoothing, fade in and fade out. For example, faders with decreased friction "glide" across the screen and can even "bounce" after hitting zero. Objects can also transmit data based on complex floating-point mathematical formulas, moving far beyond the 0-127 limitations of typical MIDI controllers. [jazzmutant]

3.4.2. Buchla Thunder

Thunder is a midi controller that senses various aspects of the touch of hands on its playing surface, and transmits the resultant gestural information via MIDI (Musical Instrument Digital Interface) to responsive electronic instrumentation.

It is an alternative controller. Making no attempt to emulate the appearance or playing techniques of existing acoustic instruments, Thunder introduces new concepts for defining musically interesting relationships between performance gesture and modern electronic vocabularies. Thunder's multi-faceted playing surface is organized to complement the shape and reach of the human hand; its 36 individual elements ought to sense a performer's slightest touch.

All keys respond to pressure; some sense position as well; others incorporate light emitting diodes used to indicate key status or currently selected options.

{IMAGE 19}: “Thunder” [Thunder]

Called STORM, Thunder's built-in operating language performs the essential function of defining the potential interaction between a musician and his instrument. It is designed for use by musicians (programming experience not required). Instrument setups ("configurations") can be stored in internal memory or on plug-in data cards. [Thunder]

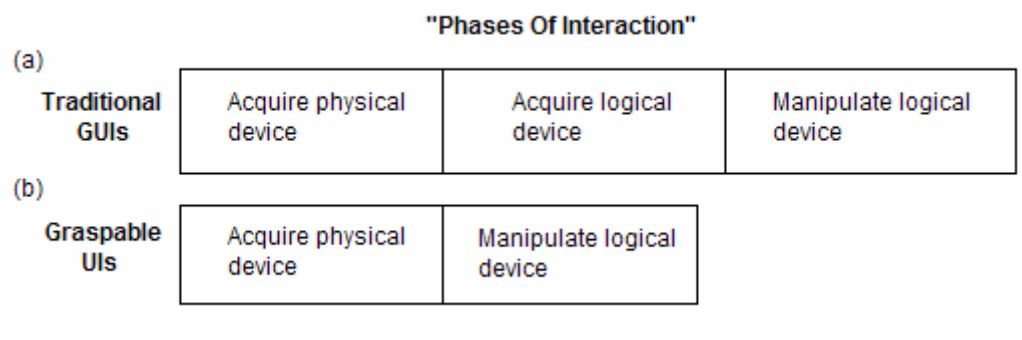
3.5. Graspable User Interfaces

Graspable UIs advocate providing users concurrent access to multiple, specialized input devices which can serve as dedicated physical interface widgets, affording physical manipulation and spatial arrangements. [FB97]

Graspable UIs allow direct control of electronic or virtual objects through physical handles for control. These physical artifacts are essentially new input devices that can be tightly coupled "attached" to virtual objects for manipulation or for expressing action (e.g., to set parameters or for initiating processes). [FB97]

Graspable functions not devices

We, sometimes, may hypothesize "graspable devices" as "graspable functions". If we think about it in a more generic way, all physical input devices can be considered as graspable, because them existing in our 3d-physical world, and thus we can grab and hold them.



{IMAGE 20}: "Phases of Interaction" [FB97]

IMAGE 20 describes the phases of interaction for (a) Traditional GUIs and (b) Graspable UIs.

With traditional GUIs there are often three phases of interaction:

- Acquire physical device.
- Acquire logical device, for example a UI widget such as a button or a scrollbar.
- Manipulate the virtual device.

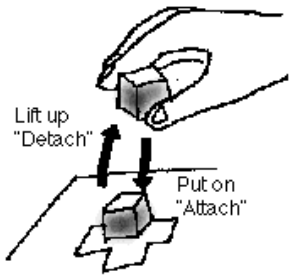
On the other hand, Graspable UIs can often reduce the phases of interaction to:

- Acquire physical device
- Manipulate the virtual device (directly).

This happens because the physical device/devices can be attached to a logical device. [FB97]
As a result, in graspable user interfaces there is a quicker mental connection of the user to the aimed task through his interaction.

Additionally, with a traditional GUI, we have to optically concentrate on a computer monitor, while physically interacting with an object that is not the monitor itself. This is very restrictive, because there is an intermediate translation level involved (to give a result, such as sound, visuals, e.t.c.), that makes interaction feel a bit artificial. With a Graspable UI, our physical movements (when we grab or touch something), do not need a translation, because physical interaction and visual feedback is attached to the same object and haptic feedback as an additional and most instant feedback channel is applied directly to it. Thus, this type of UI can be more user-friendly, because we can concentrate to a specific task only, the movement and the haptic sense.

Graspable UIs can be tracked through different modalities and sensors. One solution is the use of a vision system, such as an expensive video camera or a low cost web camera, which is what has been used in this project.



{IMAGE 21}: “Graspable device” [FB97]

A sub-category of haptic and/or graspable input devices are the so-called “tablets”, to which special reference will be made in a following chapter.

An example of graspable UIs is:

The Hands

Michel Waisvisz conceived of The Hands in the early 1980s. The Hands are aluminum plates containing touch sensitive keys, thumb pressure sensors, and tilt and proximity sensors, held under a performer's hands with velcro fasteners. Waisvisz' idea was to perform with large overt body motions as well as small fingertip control. [Hands]

{IMAGE 22}: “Michel Waisvisz performing, using The Hands” [Hands]

3.6. Combined Input devices

There are also input devices that cannot be included in one specific category, because of their combined technical characteristics and way of use.

One classical example is **"The Lightning II"**, because it combines graspable characteristics (handheld wands, in fact wireless) with a wireless technology for tracking the body movements (infrared transmitters).

{IMAGE 23}: "Lightning II" [Lightning]

LIGHTNING II is a MIDI controller that senses the position and movement of handheld wands and transforms this information to MIDI signals for expressive control of electronic musical instrumentation. In addition to functioning as a MIDI controller, LIGHTNING II contained a 32 voice synthesizer.

Based on principles of optical triangulation, LIGHTNING gathers its information by tracking tiny infrared transmitters that are built into baton-like wands. Unencumbered by wires, these wands provide complete freedom of movement within a performance space that can be as large as 12 feet high by 20 feet wide.

Basically, LIGHTNING II senses the horizontal and vertical position of each hand, for a total of four independent coordinates. From this information, LIGHTNING's digital signal

processor computes velocity and acceleration, and performs detailed analysis of gesture. An easily mastered, musically oriented interface language allows the user to define relationships between various gestures and potential musical responses. [Lightning]

3.7. Bio-sensors

A biosensor is an analytical device which converts a biological response into an electrical signal.

A fundamental example of a bio-sensor based input device is the **"Biomuse"**.

In 1992 BioControl introduced the Biomuse, an eight channel "biocontroller" that acquires and analyzes any type of human bioelectric signal, and then outputs code to control other processor based devices. This unit is intended for use as a platform for a range of applications. BioControl has licensed the Biomuse to over 80 selected research centers including, NASA, United Technologies, Volvo Advanced Research (Sweden), British Telecom, Stanford Medical Center, Loma Linda Hospital, U.S. Air Force and the U.S. Navy – to support the development of specific neural interface and biocontroller applications.

The BioMuse is a bioelectric signal controller in a rack mountable package (17" X 14" X 3"), which allows users to control computer functions directly from muscle, eye movement, or brainwave signals, bypassing entirely the standard input hardware, such as a keyboard or mouse. It receives data from four main sources of electrical activity in the human body: muscles (EMG signals), eye movements (EOG signals), the heart (EKG signals), and brain waves (EEG signals). These signals are acquired using standard non-invasive transdermal electrodes. The BioMuse amplifies and digitizes the biosignals, and then processes the signals using Biocontrol's library of proprietary DSP algorithms. For a given application, the appropriate algorithm then outputs code to control virtually any digitally interfaced device. [Biocontrol]

Stanford researchers R. Benjamin Knapp, and Hugh S. Lusted, used BIOMUSE EMG (electromyograph) sensors to transduce muscular voltage differentials into musical output.

Using BIOMUSE, one is able to create music by coordinating simple to complex muscular movements. The designers of the technology used leg and arm signals for inputting commands to the system. [BioMuse]

{IMAGE 24}: “BioMuse” [BioMuse]

4. Feedback

As humans we are able to use our senses to grab information from the environment. This is how we communicate with each other.

'Ears' ElectroAcoustic Research site gives the following definition for Feedback: "In music, this term refers to an effect produced by having an electronic or analogue circuit in which an output signal is somehow connected to the input signal. For example, this can take place when a microphone feeds back through loudspeakers in its proximity, when a signal played back on an analogue tape is used as (part of) the input for the newly recorded sound. In Artificial Intelligence, feedback can also refer to behaviour modification in which information received can influence future decisions." [Ears, "Feedback"]

A more simple definition can be found in Wikipedia,

Feedback is the signal that is looped back to control a system within itself. This loop is called feedback loop. A control system usually has input and output to the system; when the output of the system is fed back into the system as part of its input, it is called "feedback". [Wikipedia, "Feedback"]

As an example involving musical human action and feedback channels we will mention the case of playing the guitar. When we act on the instrument, we use our fingers to add pressure on the strings and the guitar reacts back, with several ways; we feel the vibration of the strings (vibrotactile) on our fingers and the vibration of the instrument's body on us, we see the vibration of the strings with our eyes and we hear the resulting sound.

Similarly, in electronic musical instruments we may find 3 feedback modes:
Audible, ***Visual and Haptic*** (tactile) Feedback.

Susan J. Lederman (Professor at the department of Psychology and School of computing, at Queen's University (Kingston) said: "If you don't understand the capabilities and limitations of humans, you can't design systems that permit them to operate effectively on remote environments whether they are real or virtual". [LED98]

Also Claude Cadoz mentions in one of his studies, "To allow an instrumental gesture interaction with the computer, we have to take into account the fact that this interaction is bi-directional: from the human being to the computer, and conversely". [CLF03]

Nevertheless, a lack of the haptic feedback channel is often the case in electronic musical instrument design.

4.1. Haptic Feedback

Haptic, from the Greek αφή (Haphe), means pertaining to the sense of touch. Touching is not limited to a feeling, but it allows interactivity in real-time with virtual objects. Thus, haptics are commonly used in virtual arts, such as sound synthesis or graphic design/animation. The haptic device allows the artist to have direct contact with a virtual instrument which is able to produce real-time sound or images. [Wikipedia, "Haptic"]

"Haptics" also refers to the human tactile (cutaneous) and kinesthetic (muscle movement) senses. A haptic interface is a computer-controlled motorized device to be held in the hand by a user, which displays information to that user's haptic senses. It is an extremely powerful modality for interface design because the same device can be used for both displaying output from the computer and accepting input from the user. [MW06]

"Haptic feedback" can be broadly divided into two modalities: **"vibrotactile"** and **"kinesthetic"**.

"Tactile" (or **"vibrotactile"**) feedback results from contact between the body of the performer and the vibrating body of the musical instrument. As these vibrations are reated by the resonating elements of the musical instrument in a traditional instrument and a digital musical instrument may not contain any resonating elements it is necessary to simulate the vibrations in order to provide some form of tactile feedback to the performer [MW06].

Kinesthetic feedback focuses on the gross movement of the human body. Kinesthetic feedback involves contours, shapes and sensations. [BIG04] It's been employed in medical simulation trainers, programmable haptic knobs, video game steering wheels, and virtual reality systems.

"Force feedback" is a term often used to describe vibrotactile and/or kinesthetic feedback. [IMM07]

According to [MC00], "Though musicians rely primarily on their sense of hearing to monitor and adjust the sound being produced by their instrument, there exists a second path through which valuable information about the instrument's behavior can be observed – namely the feedback received via the haptic senses, the senses of touch and kinesthesia..." They were evidence supporting their hypothesis that adding haptic feedback to interfaces for computer based musical instruments improves the player's ability to control these instruments. Moreover, the authors aimed to show that those force conditions where haptic feedback was congruent with auditory feedback, resulted in better performance than conditions where auditory and haptic feedback were not correlated.

Some common examples of haptic feedback are (1) the guitar model, previously mentioned (vibrating strings – finger sense) and (2) the steering wheels used as input devices in computer games such as Sega Rally (force feedback according to road quality, steering straight or taking a turn etc.).

An example of haptic feedback is also, “The Moose: A Haptic User Interface for Blind Persons”.

{IMAGE 25}: “The Moose” [MG97]

The Moose is a haptic interface whose immediate aim is to provide access for blind computer users to graphical user interfaces. Graphical information displayed on a computer’s screen can be made accessible to blind persons who at the moment are denied access to standard Graphical User Interface (GUIs). When white “puck” (connected to double flexures) is moved, the graphical mouse pointer also moves. If the pointer goes over an icon, a haptic representation of it is provided for the user to explore. [MG97]

4.2. Visual Feedback

Visual Feedback is the most known and widely used type of feedback.

{IMAGE 26}: “3D visual feedback” [IMG26]

Most tangible interfaces, mentioned before, use different techniques, whereby through computer screens, or directly from

the object, information is relayed to the user visually. The most common problem with visual feedback is that we must see on a 2D screen, a representation of objects moving in 3D space.
Visibility makes actions observable and improves legibility. It contributes to account-ability. Because of the implicit force to be able to explain publicly visible actions, people tend to reflect more what they do. Seeing actions while they are being done and seeing preparatory movement aids anticipation and improves (peripheral) awareness, supporting coordination. Seeing an action and observing the effect also enables learning by observation. Reciprocity (seeing and being seen) is important for social interaction. Visibility of objects provides focus and shared reference points. It calls for attention. [HOR04]

A sub-category of visual feedback is reflected in GUIs (= Graphical User Interface).

4.2.1. Graphical User Interface (GUI)

{IMAGE 27}: “Apple MacOSX vs Microsoft Windows Vista” [IMG27]

Graphical User Interface (Abbreviated *GUI* (pronounced *GOO-ee*)) is a program interface that takes advantage of the computer’s graphics capabilities to make the program easier to use. Well-designed graphical user interfaces can free the user from learning complex command languages. On the other hand, many users find that they work more effectively with a command-driven interface, especially if they already know the command language. [Webopedia, “GUI”]

Graphical user interfaces, such as Microsoft Windows and the one used by the Apple Macintosh, feature the following basic components:

Pointer: A symbol that appears on the display screen and that you move to select objects and commands. Usually, the pointer appears as a small angled arrow. Text -processing applications, however, use an *I-beam pointer* that is shaped like a capital *I*.

Pointing device: A device, such as a mouse or trackball, which enables you to select objects on the display screen.

Icons: Small pictures that represent commands, files, or windows. By moving the pointer to the icon and pressing a mouse button, you can execute a command or convert the icon into a window. You can also move the icons around the display screen as if they were real objects on your desk.

Desktop: The area on the display screen where icons are grouped is often referred to as the desktop because the icons are intended to represent real objects on a real desktop.

Windows: You can divide the screen into different areas. In each window, you can run a different program or display a different file. You can move windows around the display screen, and change their shape and size at will.

Menus: Most graphical user interfaces let you execute commands by selecting a choice from a menu.

The first graphical user interface was designed by Xerox Corporation's Palo Alto Research Center in the 1970s, but it was not until the 1980s and the emergence of the Apple Macintosh that graphical user interfaces became popular. One reason for their slow acceptance was the fact that they require considerable CPU power and a high-quality monitor, which until recently were prohibitively expensive.

In addition to their visual components, graphical user interfaces also make it easier to move data from one application to another. A true GUI includes standard formats for representing text and graphics. Because the formats are well-defined, different programs that run under a common GUI can share data. This makes it possible, for example, to copy a graph created by a spreadsheet program into a document created by a word processor.

Many DOS programs include some features of GUIs, such as menus, but are not *graphics based*. Such interfaces are sometimes called *graphical character-based user interfaces* to distinguish them from true GUIs. [Webopedia, "GUI"]

4.3. Audible Feedback

The most profound examples of audible feedback come from our everyday life experience in interacting with real objects (through hitting, plucking, rubbing, squeezing etc), which produces accordingly a different sound, to which we are exposed.

At a second level, we may also be exposed to the sound producing mechanism of traditional musical instruments, where our interaction with their body causes musical sound to be produced.

In the case of electronic musical systems with a tangible interface, like the ones we are discussing here, when the user acts on it (e.g. moves objects on its surface), he needs an acoustic feedback, in order to understand the changes of the musical parameters and also the position of the system at each time instance. The most important issue related to this is the latency problem (introduced by the input device's electronics, the communication protocol to the computer, the processing algorithm and the computer's processing power, the sound card

etc), that may limit the user from instantly relating his actions to the produced sound.

A straightforward example for understanding audible feedback is a feature of all mobile phones, which causes a different sound to be generated each time that a key is depressed sufficiently to generate a character on the screen. This is a case where the sound has a denotative meaning for navigating the user through the correct action.

The process of combining (or translating) one type of data (for example gestural) to another type of data, which is controlled by them (for example feedback parameters), has to be implemented through a process, which is called “mapping”. In developing an integrated interactive system for audio control, one may find several layers of data that need to be “connected’, i.e. mapped. An extended reference to this procedure is made here, as recent research has shown that although neglected in previous years, it does play eventually an important role in the efficiency of the system.

5. Music Communication Protocols

A protocol is a standard procedure for regulating data transmission between computers and other multimedia devices like synthesizers. [OSC]
In music science we use most often two types of communication protocols: The “Musical Instrument Digital Interface (MIDI)” and the “Open Sound Control (OSC)”.

5.1. Musical Instrument Digital Interface

MIDI is an industry-standard protocol that enables electronic musical instruments, computers, and other equipment to communicate, control, and synchronize with each other. MIDI allows computers, synthesizers, controllers, sound cards, samplers and drum machines to control one another, and to exchange system data. [Wikipedia, “MIDI”]

MIDI does not transmit an audio signal or media – it transmits digital data “event messages” such as the pitch and intensity of musical notes to play, control signals for parameters such as volume, vibrato and panning, cues, and clock signals to set the tempo. As an electronic protocol, it is notable for its widespread adoption throughout the industry, and for continuing in use since its introduction in 1983. [Wikipedia, “MIDI”]

5.2. Open Sound Control

OSC is a new protocol for communication among computers, sound synthesizers and other multimedia devices that is optimized for modern networking technology. [OSC]

A network is a group or system of electric components and connecting circuitry designed to function in a specific manner. [OSC]

The demand for lower costs, increased reliability, greater user convenience and more reactive musical control were all factors important in OSC's development. [OSC]

OSC is a transport-independent protocol which means that it is a format for data that can be carried across a variety of networking technologies (ETHERNET, USB, IEEE-1394). [OSC]

The unit of transmission of OSC is a packet (or datagram) rather then as a constant stream of data traveling along an established connection. Rather then assuming that the receiver holds some state from the previous communication, OSC sends information in larger, self contained chunks that include all pertinent information in one place. This leads to a protocol that is as stateless as possible. Any application that sends OSC Packets is an OSC Client; any application that receives OSC Packets is an OSC Server. This packet-based delivery model provides a mechanism for synchronicity. Synchronicity means that messages in the same packet (for example, messages to start each of the notes in a chord) will all begin their effects at the same time as each other. [OSC]

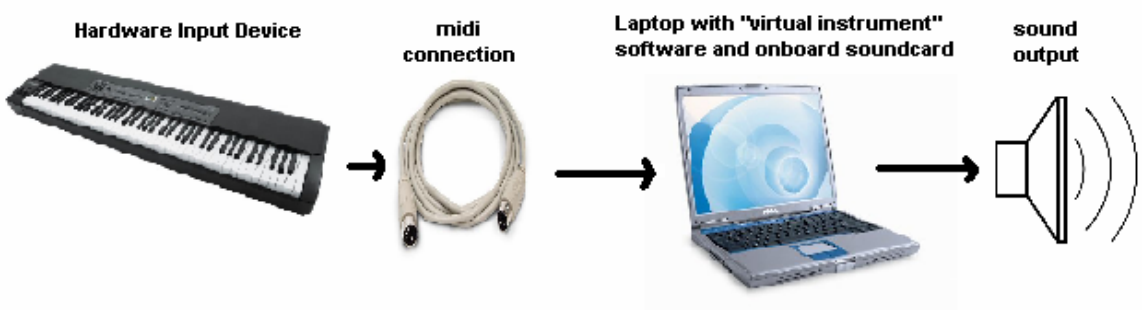
OSC uses UDP ports to make connections between applications. A program chooses any number of ports to send on, and any number of ports to receive messages on, and each message has an associated path or a URL. [Wikipedia, "OSC"]

Comparison

The advantages of OSC over MIDI are primarily speed and throughput, internet connectivity and database resolution. OSC messages arrive as fast as the underlying network stack can transfer them, and can be delayed to take effect at a specific time, whereas MIDI insures synchronicity of messages by transferring them at a specific clock rate. [Wikipedia, "OSC"] Finally, there is the problem of bandwidth. Standard, serial MIDI can send only about 3,000 bytes, or 1,000 messages, per second. [MIDI]

6. Mapping

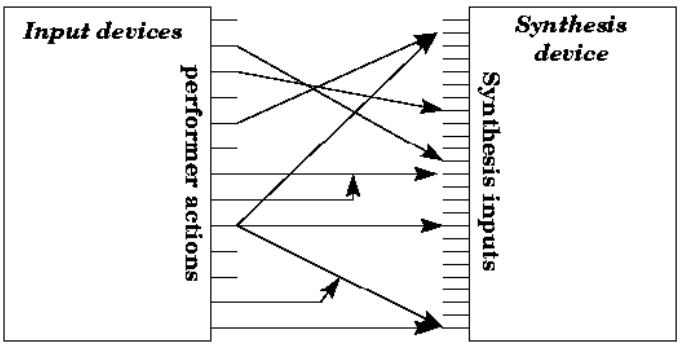
"If we need to give a specific definition of the word 'mapping', we can say that it is any algorithmic or formal process that involves the translation of data" [EARS].



{IMAGE 28}: "Connection Setup / Mapping"

In order to better describe this translation process we came up with the following simple example (IMAGE 28): Imagine we have one mother keyboard that we connect through midi with a laptop computer that is running a kind of virtual instrument software. The mother keyboard is the "hardware input device", which sends out user-action information data (such as note number, key velocity etc). This information is transduced to the computer via a hardware cable over a (MIDI) protocol. This data needs to be translated into sound through a specific process; the typical 'MIDI translation' assumes that note number corresponds to pitch, key velocity corresponds to volume etc). This procedure of matching user action parameters to audio parameters is one example of mapping (here 1-1 mapping, later explained).

We can also say that the word "mapping" refers to the liaison or correspondence between *control parameters* (derived from performer actions) and *sound synthesis parameters*. This concept is illustrated in Figure 3 that represents a general computer-based musical instrument; what might be called a "composite electronic" musical instrument. [HWK00]



{IMAGE 29}: “Mapping of performer actions to synthesis parameters” [HWK00]

In the musical literature, many tried to categorize the strategies of mapping in different ways.

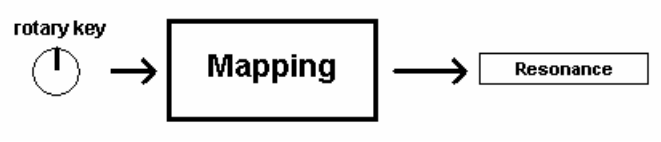
Ryan [RYA91] has categorized different strategies by proposing Euclidean analogies (points, lines and curves). Therefore, mapping one event to a set of musical parameters would be between a point and a curve. Conversely, various performer actions relating to one musical parameter would be considered as a curve to a point. Other possible relationships could then be between lines, curves, and so on.

Rovan et al. [RWD97] have identified the three basic categories, using the words *convergent* (many-to-one) and *divergent* (one-to-many).

Garnett and Goudeseune [GG99] have also considered the general case with three strategies: *direct mapping* from individual controls to individual synthesis parameters, *one control driving several parameter*s and *one parameter being driven by several controls*.

A short overview of some proposed mapping strategies follows here:

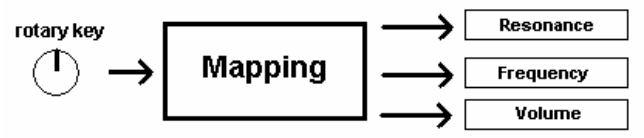
6.1. One-to-One (Single)



{IMAGE 30}: “Single mapping”

Each independent gestural output is assigned to one musical parameter, usually via a MIDI control message. This is the simplest mapping scheme, but usually the least expressive. It takes direct advantage of the MIDI controller architecture. This mapping scheme is preferred by beginners and not as much by experienced performers, who spent less time to master it [HK00].

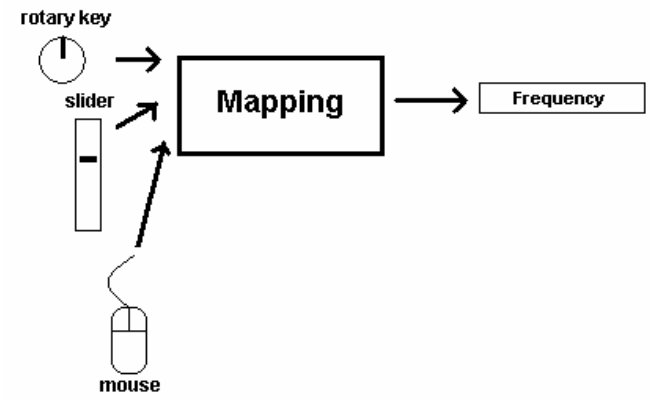
6.2. One-to-Many (Divergent)



{IMAGE 31}: “Divergent mapping”

One gestural output is used to control more than one simultaneous musical parameter. Although it may initially provide a macro-level expressivity control, this approach may nevertheless prove limited when applied alone, as it does not allow access to internal (micro) features of the sound object [HK00].

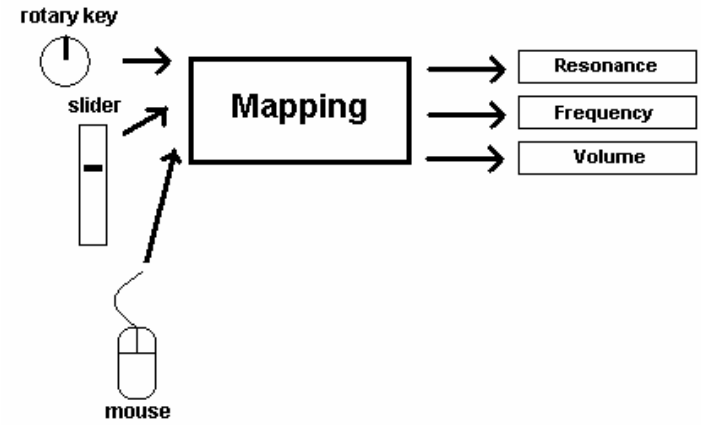
6.3. Many-to-One (Convergent)



{IMAGE 32}: “Convergent mapping”

In this case many gestures are coupled to produce one musical parameter. This scheme requires previous experience with the system in order to achieve effective control. Although harder to master, it proves far more expressive than the simpler unity mapping [HK00].

6.4. Many-to-Many (Complex)



{IMAGE 33}: “Complex mapping”

This is a mixture of the above strategies, where many gestures are used to control a number of synthesis parameters. Although the performers need more time to learn using interfaces with this complexity, they found them more engaging [HK00].

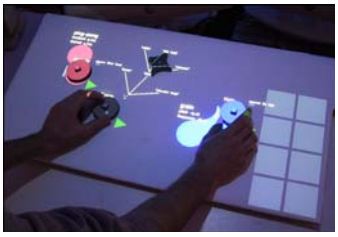
As the practical part of our project will focus on a tablet input device, an extensive (but not exhaustive) list of previous attempts and good practices will be presented here.

7. Tablets

Some of the following tablets gave us the required inspiration for this project. Some technical (and other) piece of information will be given, about the way they work.

7.1. Audiopad

“Audiopad” is a composition and performance instrument for electronic music which tracks the positions of objects on a tabletop surface and converts their motion into music. One can pull sounds from a set of samples, juxtapose recordings against synthetic melodies, cut between drum loops to create new beats, and apply digital processing at the same time.



{IMAGE 34} : “Audiopad” [PRI02]

“Audiopad” also creates a visual and tactile dialogue between itself, the performer, and the audience.

“Audiopad” has a matrix of antenna elements which track the positions of electronically tagged objects on a tabletop surface. Software translates the position information into music and graphical feedback on the tabletop. Each object represents either a musical track or a microphone. [PRI02]

7.2. Audio D-Touch

"Audio d-touch" is a collection of 3 tangible interface applications for music composition and performance: the *Augmented Stave*, the *Tangible Drum Machine* and the *Physical Sequencer.*

They can be thought of as toy-like computer music instruments. Each instrument includes a number of wooden blocks, and the interaction takes place on a flat surface, like a table top. The blocks represent different sounds, by spatially arranging the blocks on the surface users can create sequences of piano notes, drum sounds or generic audio clips.

{IMAGE 35}: "Audio D-Touch" [dtouch]

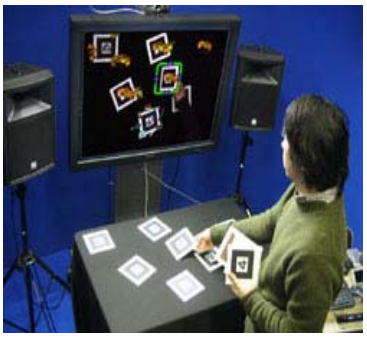
A standard personal computer observes the blocks through a low cost web-cam, and thanks to the d-touch computer vision system it can localize them precisely. The information about the position and orientation of each block is used to control a digital audio synthesis process. The system was designed to be extremely low cost (it only requires a consumer-grade PC with a sound card and a web-cam, inexpensive wooden bricks tagged with the markers and a printed piece of paper covering the interactive surface) and easy to set up.

From the technical point of view audio d-touch is written entirely in C++ in Linux, using a multi-threaded architecture to achieve low latency and robustness. [dtouch]

7.3. The Music Table

"The Music Table" provides a tactile and visual representation of music that can be easily manipulated to make new musical patterns. It lets people experience their own music as patterns in musical space.

Patterns of cards are arranged on a tabletop to become musical phrases. Completed phrases are stored on other cards and combined as multi-layered patterns. Animated characters provide visual feedback as we edit and arrange phrases, in order to make abstract musical structures visible and tangible.



{IMAGE 36} : "The Music Table" [BMH03]

This physical/tactile representation reinforces the visual contour of the melody through large muscle actions of the arms and body as the user places and moves the cards on the table. The hypothesis made by the group for the development is that this physical connection will help guide the early development of musical abstraction in younger users.

A camera placed above the table recognizes special patterns on the cards. By determining the position and rotation of each card on the table, the computer can interpret these positions as musical events. A MIDI sequencer sends this information to a synthesizer to produce the music. Through the use of *The Augmented Reality Toolkit* and set of C programming libraries, the software can also superimpose computer-generated images into the real scene according to the positions of the markers. [BMH03]

7.4. Interactive Surround Sound Cube (ISS Cube)

The “ISS cube” is aimed to work as a surround sound mixer that tracks the position of several physical objects on a tabletop surface to provide the user with an intuitive way of creating different soundscapes. Sounds of different categories shall be added, combined and positioned in space by simply moving these objects. The sound space is provided by four surrounding speakers. Each object functions individually and simultaneously to modify the volume and position of different sound sources. The philosophy behind the development of this tablet has been to let the user mix the sound sources all together or just a number of them, but at the same time; with common audio mixing consoles and systems this wouldn’t have been possible, as each sound source is usually mixed one after the other. The limitation is due to the nature of the input devices; joysticks on surround consoles and single mouse input on software systems. The ISS interface consists of a wooden box with a glass surface. Due to the multiple input devices, the square tabletop display, which enables equal access from all sides, supports collaborative interaction. [ISScube]

{IMAGE 37}: “ISS Cube” [ISScube]

ISS Cube has 4 coloured pucks, called carriers, which allow the users to select a predefined sound sample by moving it to the edge of the surface, where a selection menu will appear. Once a sample is selected, moving the carrier across the surface will spatially position the sound using a 4 speaker set-up. Each corner of the surface representing one of the speakers, so the sounds pan between each speaker based on the position of the

carrier relative to the surface corner. A second type of tactile object, a smaller white puck, controls the volume of each sample. The closer the sample carrier to the volume, the louder it becomes within the space. [LO05]

7.5. Audiocube

"AudioCube" is an interactive installation that allows a group of users to create their own three-dimensional soundscape. Four cubes, containing different symbols on each side, can be placed on a glass plate. Each symbol on the cubes represents a different sound; each cube refers to a category of sounds, such as drums, base, leads and strings.

{IMAGE 38}: "Audiocube" [AudioCube]

These cubes can then be turned to change the associated sound. The position of the cube on the glass plate refers to the position of the sound in the room. Rotating symbols, projected on the glass plate, indicate the active state of a cube. Due to this combination the users can create an individual soundscape within 3D-space. Direct acoustic feedback, intuitive interaction, and a clear connection between the action of the user and the reaction of the system are implemented with the aim of building an environment, where spatial audio can be explored creatively and collaboratively. [AudioCube]

7.6. Audiocubes

{IMAGE 39}: “Audiocubes” [AudioCubes]

“Audiocubes” is a tangible user interface (TUI) consisting of a number of cubes made out of a plastic material. Each cube contains a digital signal processor (DSP) with optical sensors and emitters (infrared, red, green and blue LEDs). The sensors and emitters receive and send audio signals which are generated or processed by the signal processor in the cube. Each cube is powered by a rechargeable battery pack. By positioning the cubes relative to each other and moving them around, a signal processing network can be created. A musician can use this interface to learn a new way of interacting with sound and music. [AudioCubes]

7.7. Loopqoob

“Loopqoob” is a physical performance system consisting of one or more sensor-equipped cubes connected to a computer based music generation/synthesis system.
The orientation of the cubes determines an aspect of the music to be played. In the implementation presented, there are three cubes.

{IMAGE 40}: “Loopqoob” [loopqoob]

Each face of each cube is mapped to a musical motif or loop. The 'cubist' controls which motifs are played by orienting the cubes so that the face corresponding to the desired motif faces up. [loopqoob]

7.8. Reactable

The “reactable”, is a multi-user electro-acoustic music instrument with a tabletop tangible user interface. Several performers share complete control over the instrument by moving physical artefacts on the table surface and constructing different audio topologies in a kind of tangible modular synthesizer or graspable flow-controlled programming language.

{IMAGE 41}: “Reactable” [KMC04]

The “reactable” hardware is based on a translucent round table. A video camera situated beneath, continuously analyzes the table surface, tracking the nature, position and orientation of the objects that are distributed on its surface, representing the components of a classic modular synthesizer. These objects are passive without any sensors or actuators, users interact by moving them, changing their position, their orientation or their faces (in the case of volumetric objects). These actions control the topological structure and parameters of the sound synthesizer. A projector, also from underneath the table, draws dynamic animations on its surface, providing a visual feedback of the state, the activity and the main characteristics of the sounds produced by the audio synthesizer. [KMC04]

The tracking of the objects, used for Reactable, is being achieved by an open source, cross-platform computer vision framework called “reacTIVision”.

8. reacTIVision

This software is very useful for someone, who wants to experiment on his own reactable construction, but he/she does not have the appropriate programming knowledge background.

ReacTIVision tracks objects that are attached with a specific type of images/ graphs, called “Fiducials”. Fiducials are targets/symbols designed to be identified and tracked using a set of algorithms. Pattern recognition seems to be robust and is much less prone to problems with lighting and background due to the nature of the tracking process. [KB07]

Fiducials began their life as variations of the d-touch fiducials, where identification is performed using topological pattern matching on a region adjacency graph of a binarised input image. Once fiducials have been identified, d-touch uses geometric techniques including line detection and relative position of regions to determine the location, orientation and identity of each marker (marker geometry decodes to a unique id number). Their design evolved toward using only the region adjacency graph and the bounding rectangles of each region to determine all necessary information about each marker. This allowed them to remove a number of steps from the computer vision algorithm; however it also introduced increased complexity in laying out the fiducials. To solve the layout constraints while minimizing the size of the fiducials they successfully employed a genetic algorithm. [KB07]

ReacTIVision framework has been developed by Martin Kaltenbrunner and Ross Bencina at the Music Technology Group at the Universitat Pompeu Fabra in Barcelona, Spain as part of the reactable project, a novel electro-acoustic music instrument with a tangible user interface.

ReacTIVision is a standalone application, which sends Open Sound Control (OSC) messages via UDP port 3333 to any connected client application. It implements the TUIO protocol, which was specially designed for transmitting the state of tangible objects and multi-touch control on a table surface. This framework includes a set of free example client projects for various programming languages, which can serve as the basis for the easy development of tangible user interface applications. Alternatively reacTIVision is also able to send MIDI messages, although TUIO is the recommended messaging format. [KB07]

The reacTIVision application is available for Windows, MacOS X and Linux. Under Windows it supports any camera with a proper WDM driver, such as USB, USB2, FireWire and DV cameras. Equally under MacOS X any such camera supported by QuickTime will work in reacTIVision. Under Linux, FireWire cameras are best supported, as well as a few Video4Linux USB cameras.

ReacTIVision was designed to enable gestural control of musical sound, and can track many markers at a high frame rate. The development of reacTIVision involved not only computer vision algorithms, but also the design of a new marker system. The development team co-designed the computer vision system and markers, applying computation to minimize marker size while meeting geometric constraints required to efficiently compute the location and 2D orientation of the markers. [KB07]

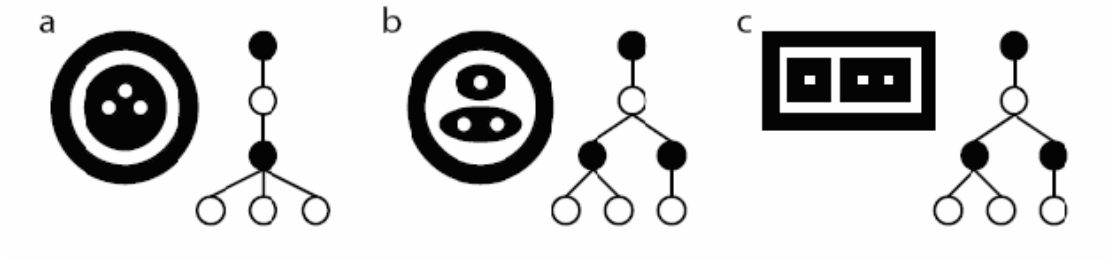
ReacTIVision was developed after initial prototyping with Costanza and Robinson's d-touch system for tracking the location and orientation of fiducials (markers) in a real-time video stream.

9. Fiducials

Fiducials are binary image markers, specially designed for being identified by the reacTIVision application on the basis of topological structure identification algorithms.

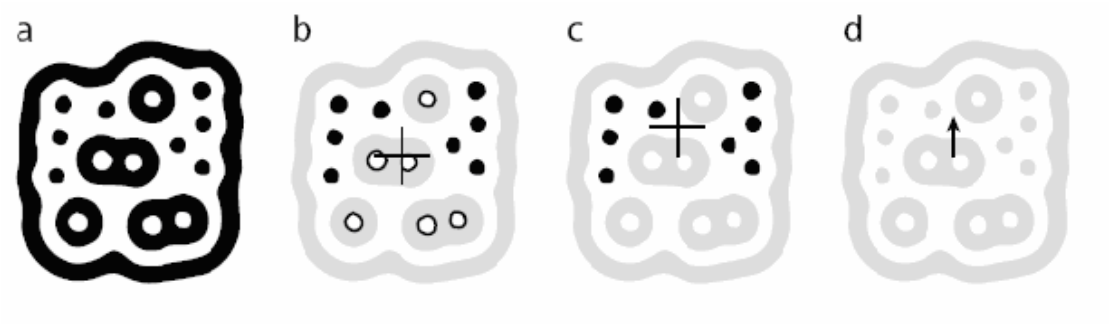
9.1. ReacTIVision Fiducials & Reactable

Reactable uses ReacTIVision Fiducials introduced by Costanza and Robinson in the d-touch system and developed by Bencina et al. 2005. They look for adjacent colour changes and the number of these colour changes within each boundary. We can understand this method of detection, more clearly, on a Region Adjacency Graph (RAG).



{IMAGE 42}: “Some simple topologies and their corresponding region adjacency graphs” [BK05]

On “IMAGE 42 (a)” we see one black outer ring followed by one white inner ring and one black inner circle, inside of which lies three small white circles. This fiducial pattern is known as the “Topology”. Beside each simple topology, we can see a basic form by the RAGs, which follow in sequence from top to bottom. As we notice, the RAGs of “IMAGE 42 (b)” and “IMAGE 42 (c)”, are the same although the topologies are different. In relation to Audio D-touch, that uses geometry to identify (ID) fiducials, ReacTIVision fiducials are identified by their RAG or Topological structure. [BK05]



{IMAGE 43}: “(a) a reacTIVision fiducial, (b) black and white leafs and their average centroid, (c) black leafs and their average centroid, and (d) the vector used to compute the orientation of the fiducial” [BK05]

On “IMAGE 43 (a)” we see a reacTIVision Fiducial. Its centre and orientation is specified using its smallest entities (the last parts of the RAG tree). These leaf centres, both black and white, are computed using a weighted average centroid of either all the white leafs or black leafs. ReacTIVision selected this method of computation, because it is easy to find the centre of an object, irrespective of their shape (if they are square, circular and/or relatively small). [BK05]

10. Spatialization

“Spatialization is perhaps the most all-embracing and general term used to describe the means by which loudspeakers are used to articulate or create a spatial musical experience for listeners in playback or performance. The term is wide-ranging from a technical and arguably aesthetic point of view. It includes formats (e.g. Stereophonic, Ambisonic, Dolby), the placement and movement of sounds in space in any number of listening situations (e.g. concert hall, installation, virtual environment, cinema), and performance practices (e.g. diffusion, Octophony, and more recent developments in automated performance systems).” [Ears, “Spatialisation”]

The movement of sound through space creates dramatic effects and can serve as an important structural element in composition. Composers can articulate the voices in a contrapuntal texture by giving each one a unique spatial location. The virtual and physical “sound stage” around the audience can be treated as a landscape, with its background and foreground, and fixed and moving sources. This sound stage can be fixed in playback, or controlled by gestures in concert [ROA96].

Digital simulations of moving sound pose special problems. In many concerts the audience is surrounded by a number of loudspeakers. How does one create the illusion of a sound traveling about the hall, moving away from or toward the listener as it goes? In listening situations with only two loudspeakers or with headphones, the illusion of sounds moving freely in space is even more difficult.

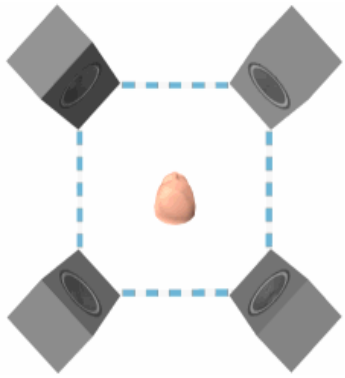
The most popular spatial illusions are “horizontal panning” – lateral sound movement from speaker to speaker – and “reverberating” – adding a dense and diffuse pattern of echoes to a sound to situate it in a larger space. “Vertical panning” (up and

down and overhead) can also create striking effects in electronic music. [ROA96]

There are lot configurations for different types of spatialization. We are not going to analyze all of them, but we will be focused in one specific type of spatialization, the “Quadraphonic sound”.

10.1. Quadraphonic Sound

Quadraphonic sound is remembered with mixed feelings by many in the industry; as it represents a failed attempt to introduce surround sound to the consumer. A variety of competing encoding methods, having different degrees of compatibility with each other and with two channel stereo, were used to convey four channels of surround sound on two channel analogue media such as vinyl LPs (so-called 4-2-4 matrix systems). Unlike Dolby Stereo, quadraphonic sound used no center channel, but was normally configured for a square arrangement of loudspeakers, two at the front and two behind the listener. The 90° angle of the front loudspeakers proved problematic because of lack of compatibility with ideal two channel reproduction, and gave poor front images, often with a hole in the middle.



{IMAGE 44}: “4 speaker setup”

While a number of LP records were issued in various quad formats, the approach failed to capture a sufficiently large part of the consumer imagination to succeed. It seemed that people

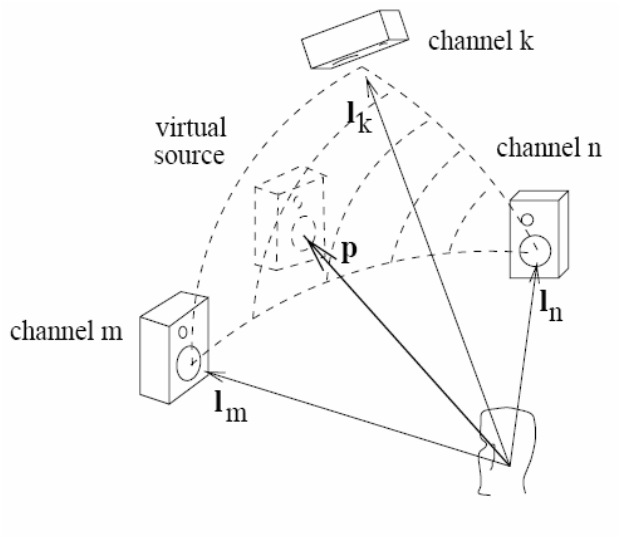
were unwilling to install the additional loudspeakers required, and there were too many alternative forms of quad encoding for a clear ‘standard’ to emerge. Also, many people felt that quad encoding compromised the integrity of two channel stereo listening 9the matrix encoding of the rear channels was supposed to be two-channel compatible but unwanted side effects could often be heard). [RUM04]

10.2. VBAP – Vector base amplitude panning

In computer music it is a common problem that the loudspeakers are in different positions in different concert halls and computer music studios. The ideal is a system that would be able to create identical soundscapes using different loudspeaker configurations. Vector base amplitude panning (VBAP) is a generic method for virtual source positioning. It generalizes the pair-wise paradigm to triplet-wise panning paradigm, which can be used in 3-dimensional loudspeaker setups. Virtual sources are positioned by defining the virtual source direction as (azimuth, elevation) coordinates. [PUL00]

In conventional amplitude panning a sound signal is applied to 2 loudspeakers in front of the listener (Blumlein 1931). The listener perceives a virtual source between the loudspeakers. When loudspeakers are placed on horizontal plane around the listener, virtual sources in all azimuth directions can be produced using the “pair-wise” paradigm (Chowning 1971). One sound signal is applied to 2 loudspeakers at one time. [PUL00]

With loudspeaker systems that also include elevated loudspeakers, the pair-wise paradigm is not appropriate. “Triplet-wise” panning can be formulated for such loudspeaker configurations. The loudspeakers in a triplet form a triangle from listener’s view. The listener will perceive a virtual source inside the triangle depending on ratios of the loudspeaker amplitudes. [PUL00]



{IMAGE 45}: “Three-dimensional panning” [PUL00]

In three-dimensional VBAP a loudspeaker triplet is formulated with vectors as on IMAGE 45. The Cartesian unit-length vectors I_m, I_n and I_k point from listening position to the loudspeakers. The direction of the virtual source is presented with unit-length vector p. Vector p is expressed as a linear weighted sum of the loudspeaker vectors

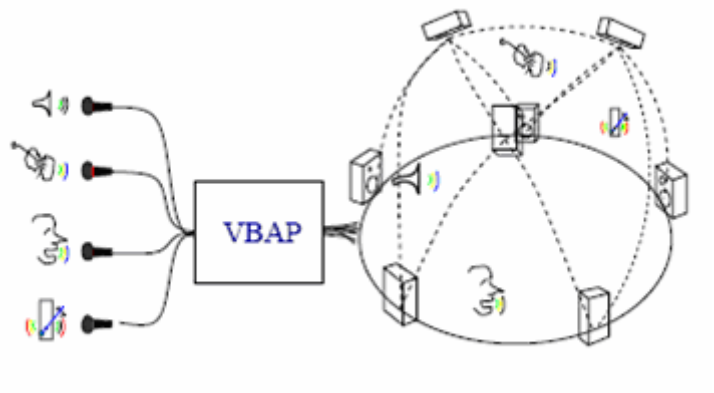
$$p = g_m I_m + g_n I_n + g_k I_k$$

Here g_m, g_n and g_k are the gain factors of respective loudspeakers. The gain factors can be solved as

$$g = p^T L_{mnk}^{-1}$$

Where $g = \left[g_m g_n g_k \right]^T$ and $L_{mnk} = \left[I_m I_n I_k \right]$. The calculated factors are used in amplitude panning as gain factors of the signals applied to respective loudspeakers after suitable normalization, e.g. $\|g\| = 1$. [PUL00]

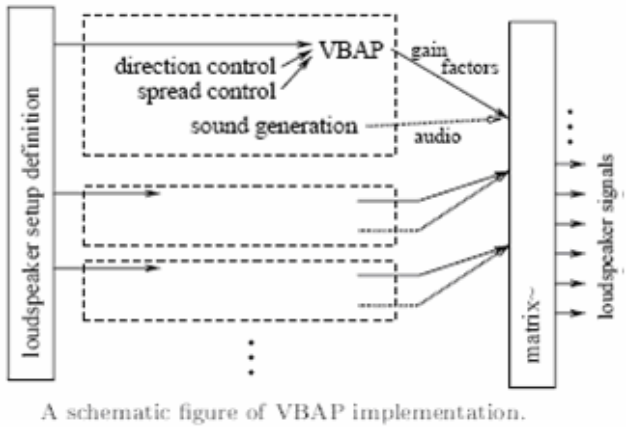
If more than three loudspeakers are available, a set of non-overlapping triangles are formed of the loudspeaker system before run time. One of the defined triplets is used in panning at one time (check IMAGE 46). There can be, of course, several virtual sources applied to one triplet. [PUL00]



{IMAGE 46}: “Using VBAP with arbitrary loudspeakers setups” [PUL00]

10.2.1. VBAP and panning in MAX/MSP

MAX/MSP is a graphical programming environment produced by Cycling ’74. MAX is designed for processing of events, and MSP is an extension of it designed for real-time audio applications.



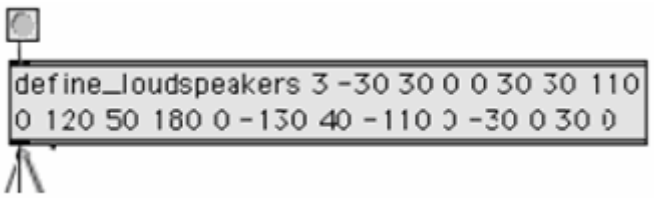
{IMAGE 47}: “A schematic figure of VBAP implementation” [PUL00]

The VBAP implementation consists of three objects, as we see on IMAGE 47:

- “define_loudspeakers”
- “vbap”
- “matrix~”

A “vbap” object is attached to each generated sound signal. The user may design controls for direction and spreading for it. The loudspeaker setup is defined using “define_loudspeakers”. The “matrix~” object performs distribution of sound signals to loudspeakers. When the patch is applied for different loudspeaker setups, only the settings of “define_loudspeakers” and “matrix~” object has to be updated. [PUL00]

Object “define_loudspeakers” perform all needed calculations to form a data set that describes the loudspeaker setup (see IMAGE 48). [PUL00]



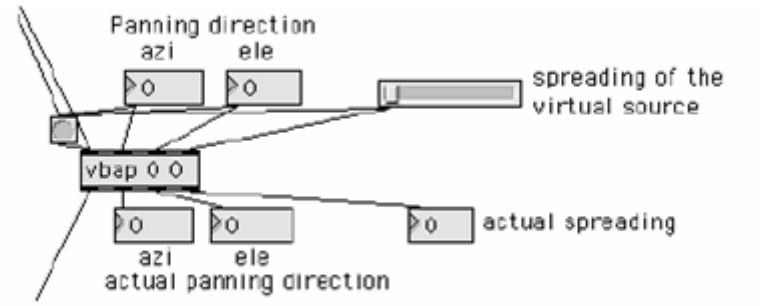
{IMAGE 48}: “The loudspeaker configuration is specified in define_loudspeakers object” [PUL00]

It is initialized by specifying the directions of the loudspeakers. It chooses loudspeaker pairs or triplets and calculates an inverse matrix L_{mnk}^{-1} for all of selected triplets. It outputs the matrices and the loudspeaker numbers of each triplet to all “vbap” objects. It performs these calculations when it receives a “bang” message. It can read the loudspeaker configuration as a list and read the loudspeaker triplets as a list. The latter list is used if the user wants to select the triangles by hand. Normally this is not needed. [PUL00]

Describing IMAGE 48, the first parameter is the dimensionality of the loudspeaker setup, which can be “2” or “3”. If the dimensionality is “2”, the following entries are the azimuth angles of the loudspeakers. If the dimensionality is “3”, following

number pairs are (azimuth, elevation) coordinates of the loudspeakers. The loudspeaker directions are presented in order of the loudspeaker channel numbers. [PUL00]

The “vbap” object calculates gain factors depending on specified panning direction and on received loudspeaker setup information. See IMAGE 49. [PUL00]

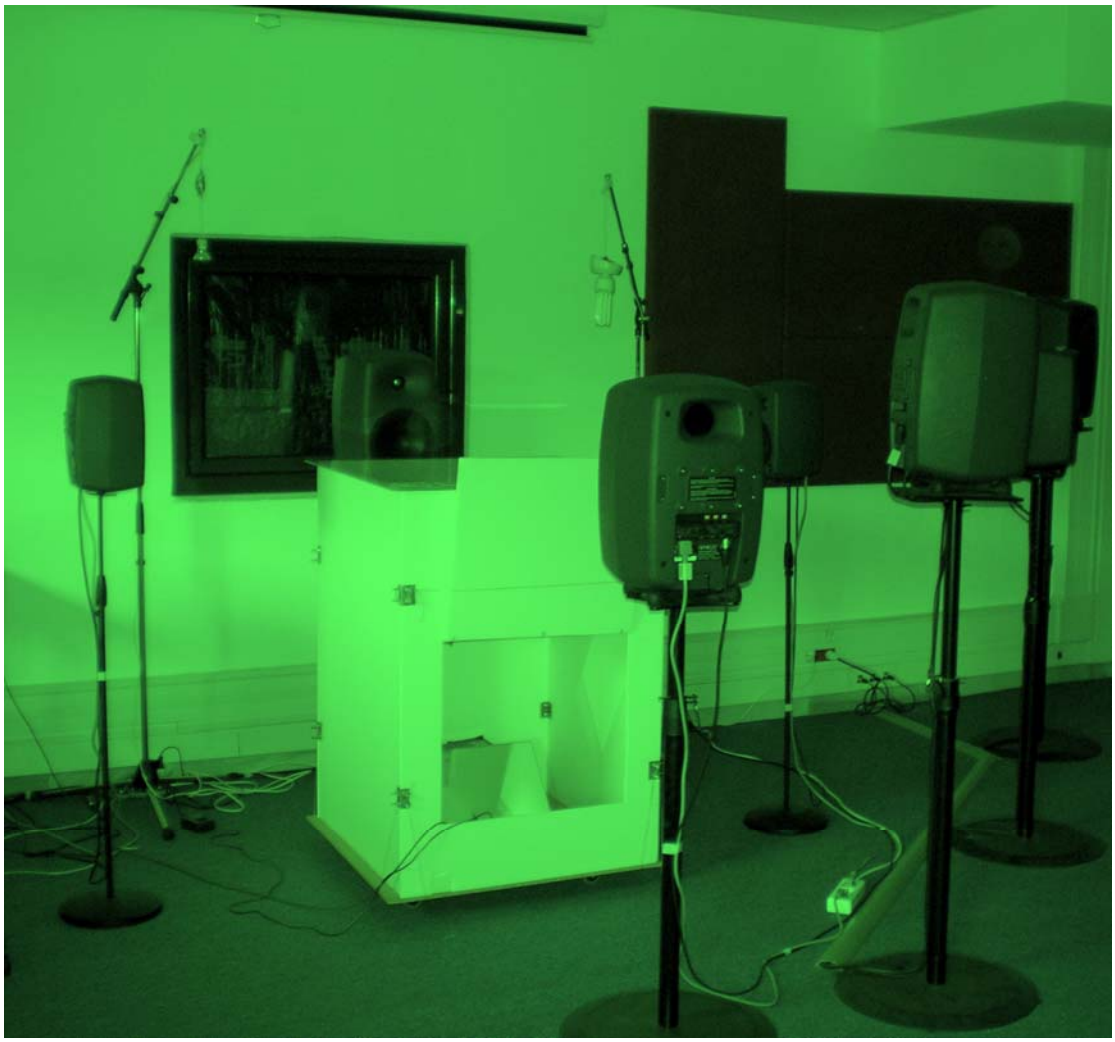


{IMAGE 49}: “Object vbap calculates the gain factors of the loudspeakers and sends them to matrix~ object” [PUL00]

It takes as input the loudspeaker setup data from object “define_loudspeakers”, panning angle as azimuth and elevation parameters, and a parameter that controls spread of the virtual source. When the “vbap” object receives a “bang”, it performs calculations of VBAP and MDAP (multiple-direction amplitude panning – Pulkki 1999) and outputs the gain factors for all loudspeaker channels. [PUL00]

The VBAP technique we just presented is based on the VBAP implementation by Ville Pulkki. Our project is based on another implementation of the VBAP technique (by Yves Gigon, found in the Trajaudioapp.mxf patch described later in this dissertation).

11. Technical Chapter

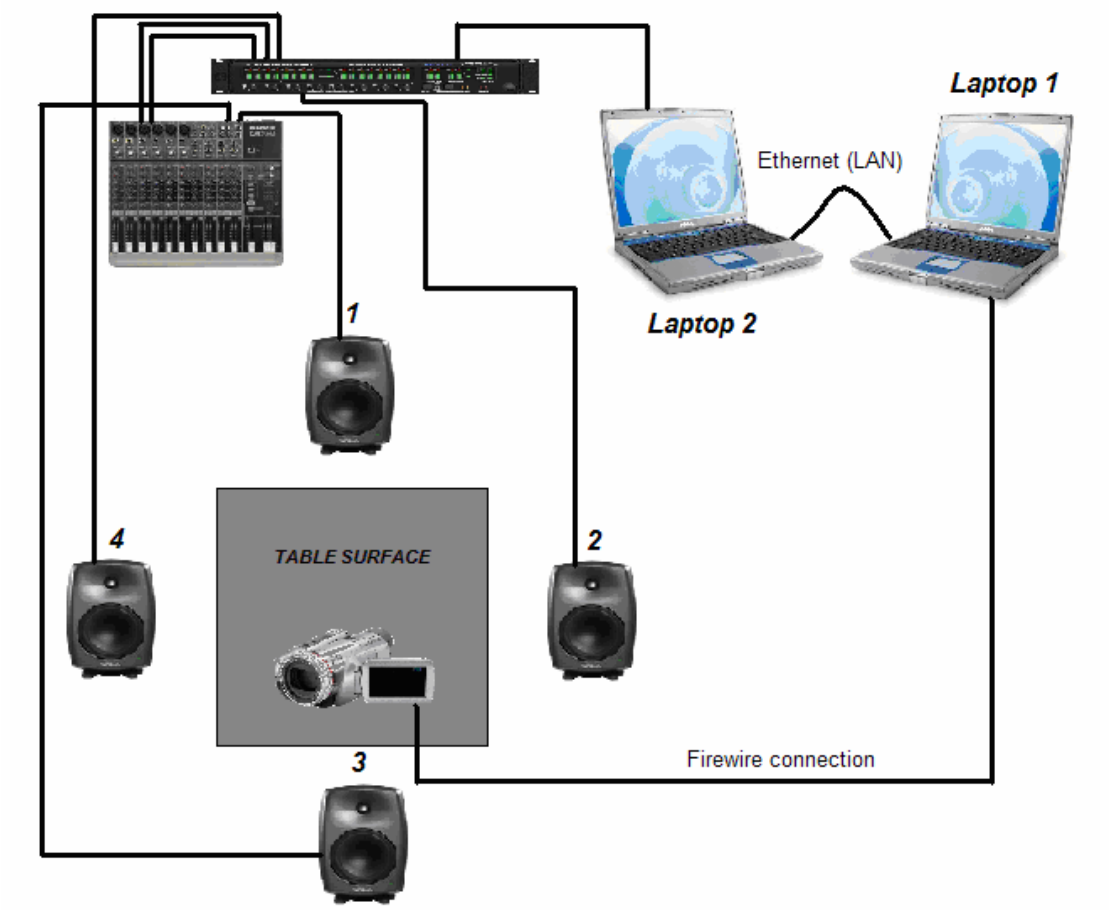


11.1. THE HARDWARE SETUP

Our project needs a number of hardware connections and software installations to operate. These are:

- Two personal computers (connected over LAN network),
- A simple webcamera or an analog camera (connected through a video capture-card) or a digital firewire camera (DV) or any other (we used a firewire video camera). In our project a firewire camera was used.
- An audio interface (with at least 4 output channels)
- At least four loudspeakers
- A specially constructed table, with Plexiglas top surface (detailed description follows)

Connections should be made according to the following image:



{IMAGE 50}: "The Hardware Setup for our project"

It is preferable to use two laptops, in order to distribute the processing power required and for these to communicate we used the OSC protocol through a specific IP address and port, via an Ethernet cable.

11.2. Construction and Testing

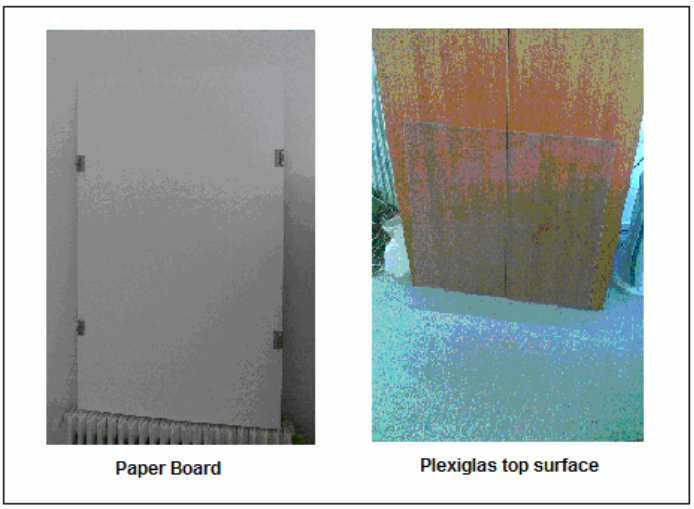
The project started off with the aim of developing a low-cost and easy-to-use device, which dictated the use of specific hardware, such as mainstream commercial webcameras, for image tracking. Nevertheless, testing of these cameras (to be discussed later in this chapter) showed high fluctuation of data at normal room lighting conditions and led to a change of the required hardware.

In line with our decision for the table to be relatively portable, light, and large enough for easy experimentation with the fiducials, we constructed the table out of "paper board" (a material that architects use for building maquettes) in white color (for better diffusion of light, if its surface needs to be illuminated from underneath). A thorough search of similar attempts for constructing such table-based devices showed that more materials can be tried, but availability is limited for some of them.

On the bottom of the table we placed a mirror with the proper angle for tracking larger surface-space. Then we placed a video camera for the optical tracking of the fiducial images, which –as already discussed—can be a simple web-cam, an analog camera (connected through a video capture-card), a digital firewire camera (DV) or any other type (we used a firewire video camera), in parallel with the bottom of the table and connect it with "Laptop 1" via firewire.

{IMAGE 51}: "Camera and mirror placing"

The four surfaces of the box were maid out of paper board at a size of 1.2m x 0.7m (IMAGE 52a).



{IMAGE 52}: “Construction materials – (a) Paper Board, (b) Plexiglas top surface”

For the top surface we used a 4mm thick Plexiglas of 75cm x 75cm size (IMAGE 52b). For the bottom we used a 75cm x 75cm wooden board based on 4 wheels to make it portable.

For sound spatialization we used a MOTU 896HD firewire soundcard, two Genelec 8050A active studio monitors, two Genelec 8030A active studio monitors and a Mackie 1402-VLZ3 sound mixer.

{IMAGE 53}: “(a) Motu Soundcard, (b) Genelec studio monitors”

We connected speakers 1, 2, 3 and 4 to Motu 896HD outputs. Because we used 2 different pairs of Genelec active studio monitors we had to connect the 8050A pair through a Mackie 1402-VLZ3 mixer for controlling their volume. The other pair (8030A) had a separable volume control each, so we connected

them directly to the soundcard's outputs (this was due to the lack of space in the lab and is mentioned here only for reporting purposes, but ideally four speakers of the same type should be used).
We also connected the Motu soundcard to "Laptop 2" via firewire.

For the visual tracking of the fiducials, we used a Panasonic NV-GS500 video camera.

{IMAGE 54}: "Video Camera with firewire connection"

This camera (IMAGE 54) carries Leica Dicomar Lens and 12x Optical Zoom.

11.3. THE SOFTWARE SETUP

- "Laptop 1" handles the image processing part of our project by running the following applications (on Windows XP):

  (a) reacTIVision application - "reacTIVision client example", which is a standalone program for recognizing fiducials and sending their parameters to a MAX/MSP client, to be downloaded for free at:
  http://mtg.upf.edu/reactable/?software,

  (b) Cycling '74 - MAX/MSP, in specific the 'TUIO_IMG_final.pat' patch (for receiving the reacTIVision parameter values and sending them over the Ethernet connection (OSC - Open Sound Control) to "Laptop 2").

- "Laptop 2" handles the audio processing part based on the parameters received from "laptop 1" by running the following application:
  MAX/MSP, in specific the 'main_patch_morebanks.pat' patch, which contains the samplers and spatialization (it needs the firewire connection for the motu soundcard and the Ethernet connection for the OSC max external).

We connected "Laptop 1" and "Laptop 2" via Ethernet and we need to establish a network connection between the 2 laptops (we need to assign them a static IP address). The OSC external for MAX has to be downloaded and installed.

‘TUIO_IMG_final.pat’:

According to the readme.txt file for the TUIO client patch in MAX/MSP, this is what the different parameters are related to:

This package contains a simple patch that receives and decodes TUIO messages received from reacTIVision or the simulator. This package provides a MaxMSP extra for Windows and MacOS X.

The TuioClient object takes an alternative port number as an optional argument. The default TUIO port is 3333.

There are two outputs, the first one is sending all relevant TUIO events such as addObject, updateObject and removeObject as well as addCursor, updateCursor and removeCursor.
The second output sends simple bangs for each fully received frame, and every second while no new messages are received, in order to indicate that the connection is still alive.

The list format of the messages received at the first output is in analogy to the TUIO message format:

addObject session_id fiducial_id
updateObject session_id fiducial_id xpos ypos angle xspeed yspeed rspeed maccel raccel
removeObject session_id fiducial_id

addCursor session_id
updateCursor session_id xpos ypos xspeed yspeed maccel
removeCursor session_id

As a more detailed explanation of the previous text, we may mention the following things:

“Object” is used to describe the detection of a fiducial image and “Cursor” is used to describe the detection of a fingerprint image (another simple fiducial that may be attached to one’s finger in

order to track a finger print (this wasn't used in this project, as it has shown no robustness at all)).

"addObject": Sends messages, each time the camera detects a fiducial image.

"session_id": Describes how many times the camera detects the concrete fiducial_id. For example, how many times the camera "saw" fiducial 68!

"fiducial_id": Describes which fiducial image we use each time. We can choose between 89 different fiducial images. Each fiducial image is unique and has its own id number.
"updateObject" : Sends messages during we are using the fiducial in front of the camera.

"xpos" : Describes the position of the fiducial on the x axis and its value is a decimal number between 0 and 1.

"ypos" : Describes the position of the fiducial on the y axis and its value is a decimal number between 0 and 1.

"angle" : Describes the position of the fiducial for its rotation. The value of angle is a decimal number between 0 and 6.28. This value describes radians.

"xspeed" : Describes the speed of the fiducial movement on the x axis and its value is a decimal number between 0 and 1.

"yspeed" : Describes the speed of the fiducial movement on the y axis and its value is a decimal number between 0 and 1.

"rspeed" : Describes the speed of the fiducial rotation and its value is a decimal number between 0 and 1.

"maccel" : Describes the acceleration of the fiducial movement on x and y axes.

“raccel” : Describes the acceleration of the fiducials rotation.

“removeObject” : Sends messages when we remove a fiducial from the visual range of the camera.

The optical tracking of the fiducial images has shown great fluctuations according to light conditions, as already mentions and a brief history of our experimentations to enhance tracking follows.

11.4. Tracking Problems

According to our experimentation with the fiducial tracking we realized that there were several problems involved with it and that a larger tracking surface needed to be used, as well as bigger fiducial images, so we had to:

- Experiment with the size of the mirror we had to use for catching a larger space on the surface.

- Experiment with the distance of the mirror, from the camera and its angle with the bottom of our table (we had to find a way to adapt this mirror inside the box, to prevent the imaging of unwanted parts of the table).

- Experiment with the placement of the camera (distance, height), but always trying to keep the camera stable and parallel with the bottom of the table.

- Fix the zoom parameter of the camera and try to adjust manually other parameters, like iris and white balance.

- Try to use larger fiducial images.

- Experiment with the illumination in many different ways.

“Infrared tracking”: IR is just below the visible spectrum of light in frequency and is radiated strongly by hot bodies. Many objects such as people, vehicle engines and aircraft generate and retain heat, and as such, are especially visible in the infra-red wavelengths of light compared to objects in the background. In infrared photography, infrared filters are used to capture the near-infrared spectrum. Digital cameras often use infrared blockers. Cheaper digital cameras and camera phones have less effective filters and can "see" intense near-infrared, appearing as a bright purple-white color. This is especially pronounced when taking pictures of subjects near IR-bright areas (such as

near a lamp), where the resulting infrared interference can wash out the image. We tried to use this kind of illumination, but the lamps we had (we tested a “philips TUV 8W G8 T5” UV-C radiation fluorescent lamp, a “DURA lamp SpA 30W” fluorescent lamp, an “OSRAM Halogen 50W” and an 18 leds headlight) were emitted very week in the IR area. According to our tests, only the OSRAM Halogen lamp behaved better, because of its construction (something like a mirror-based material on its back). We also tried placing the lamps under the surface, over the surface, far and close to the table (needs attention, to avoid mirroring effects from the mirror or the Plexiglas surface).

{IMAGE 55}: “Testing hardware and techniques”

- Use a high definition, with high frame rate, video camera. That is because we tested one usb Microsoft webcam and one usb Logitech webcam and the tracking was too slow (due to frame rate and interfacing).

- Check latency issues with the camera.

11.5. User Instructions Manual

1. Execute on “Computer 1” the MAX patch “TUIO_IMG_final.pat”.

2. Execute on the same computer the program “reacTIVison.exe” and press “v” for the appearance of which fiducial_id is on. We also need to press “g” for enabling the Frame thresholder for better view.

3. Execute the MAX patch “main_patch_morebanks.pat” on “Computer 2” (computers 1 and 2 must be connected through a Local Area Connection).

4. Each Fiducial marker has a unique role:

   - FIDUCIALS 3, 4, 5: We can imagine these 3 fiducials like 3 different cd’s. When we add them on the surface of the table they start playing the samples that have been loaded on each one. When we take them off the surface, the samples stop sounding and their volume fades out gradually (not instantly). When we rotate each one, we increase (clockwise) or decrease (anticlockwise) the volume of each sample independently (straight alignment corresponds to muting). When we move the fiducials within the surface’s active area, we are able to diffuse each sample to a polyphonic sound system (from one up to four speakers). Attention: The 4 channel polyphonic sound system is set clockwise, evenly spaced in a circle: Speaker 1 at 12:00 (top of the circle), speaker 2 at 15:00, speaker 3 at 18:00 and speaker 4 at 21:00 o’clock.

   - FIDUCIAL 6: This is the start/stop button for turning the whole system on/off. When we add it on the surface, it triggers the start button and sets a 4-channel speaker setup as default.

   - FIDUCIAL 7: We use this fiducial for playing back all the samples (that are on the surface) reverse-wise. With this

type of effect, we can try to represent the scratching effect of a DJ.

- FIDUCIAL 8: When we rotate this fiducial on the table surface, we can increase or reduce the playback speed. Attention: If we increase the playback speed and take away the fiducial from the table surface, the samples' playback speed continues at the same last value.
- FIDUCIAL 9: When we add or remove this fiducial from the surface, we change with a serial way the sound banks (5 different sound banks).
- SOS: The first thing we have to do before starting the playback process is:
    i. Drag n' Drop the 15 samples that we are going to use.
    ii. Put on the table surface the fiducial 9, for loading the first sound bank.
    iii. Start the playback process, by putting on the surface the fiducial 6.

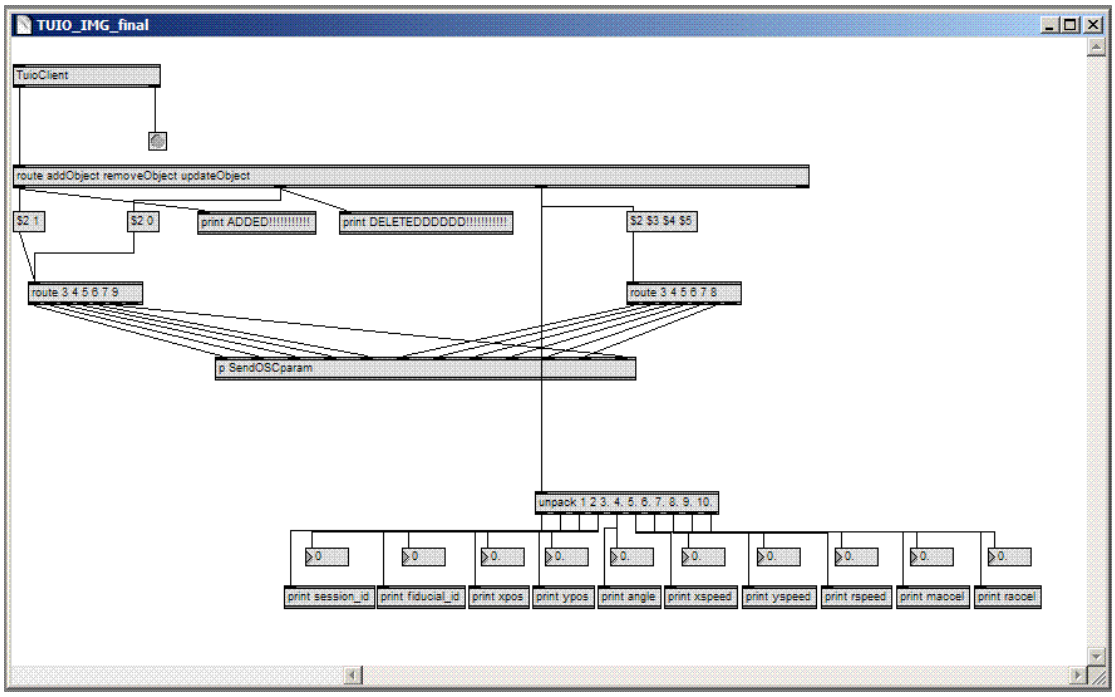
11.6. Max/Msp Patches Explanation

Our patch is based on the TUIOclient for the fiducial recognition (laptop 1) and the 'trajaudioapp.mxf' MAX patch for the sound distribution (laptop 2).

ReacTIvision.exe sends all fiducials parameters to the TuioClient max patch (listening to default port 3333) and we receive through it all the required information. TuioClient can provide us with information about its fiducial_id, position on x and y angle, rotation, whether is on or off the table and many more.

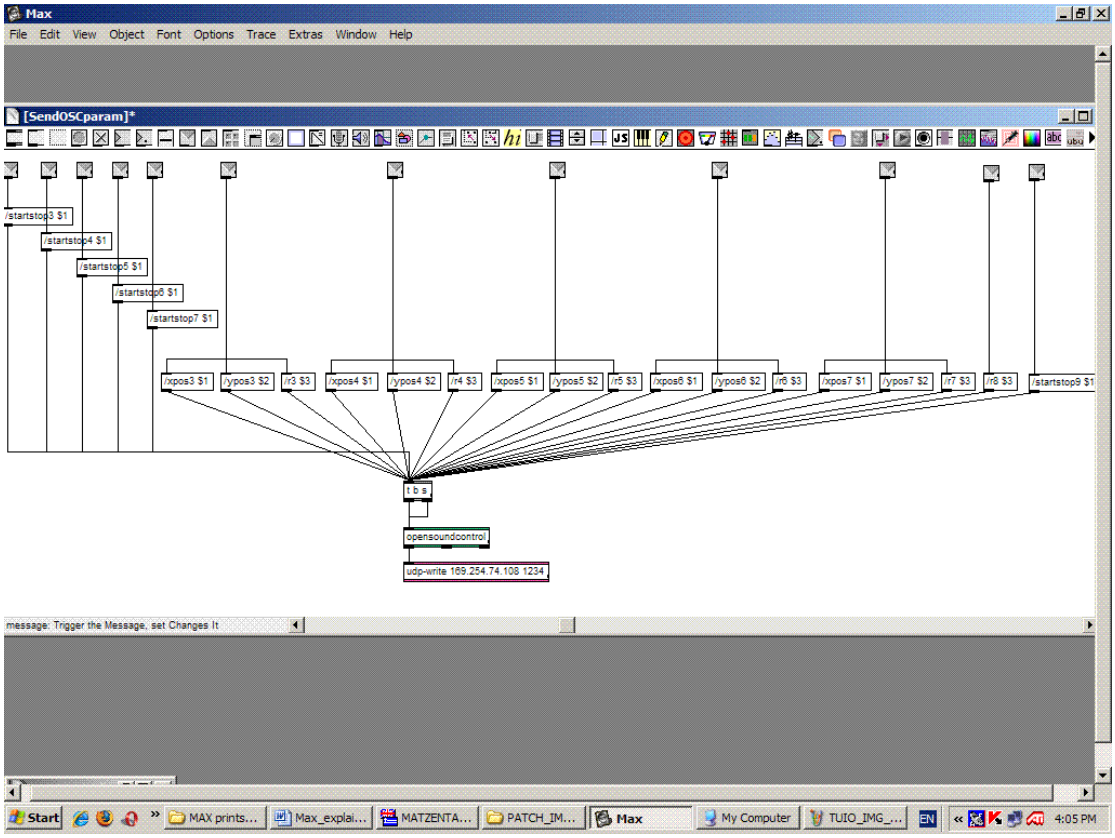
Because we are using two computers, we have decided to use one computer for the image recognition and its parameters and the other one for the sound part. The first computer loads the "TUIO_IMG_final.pat" that contains the TuioClient and sends through the Open Sound Control (OSC) all the parameters required for the sound part of our project.

MAX PATCH ON COMPUTER 1 (VISUAL TRACKING)



{IMAGE 56}: “Main MAX patch for fiducial image tracking”

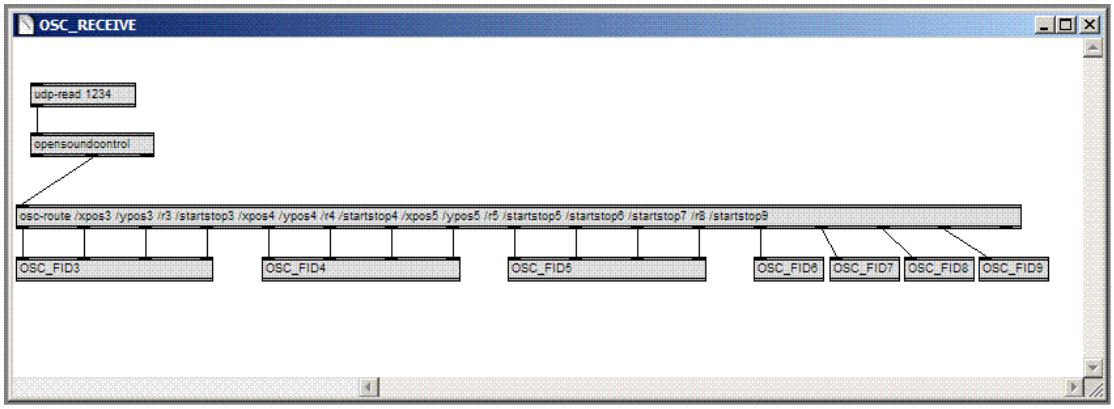
We are using a “route” object box, which helps us route the addObject, removeObject and updateObject. From the addObject we select the second parameter. The same thing happens for the removeObject. This means that when we add a fiducial on the surface, we receive a “1” and when we remove a fiducial from the surface, we receive a “0”. These values refer to the start/stop of a fiducial recognition process. From the updateObject we select the parameters refer to “x position”, “y position” and “rotation”. We are using a second route, that sends the parameters of specific fiducials id’s such as fiducials 3, 4, 5, 6, 7, 8 and 9.



{IMAGE 57}: “Sending parameters via Open Sound Control”

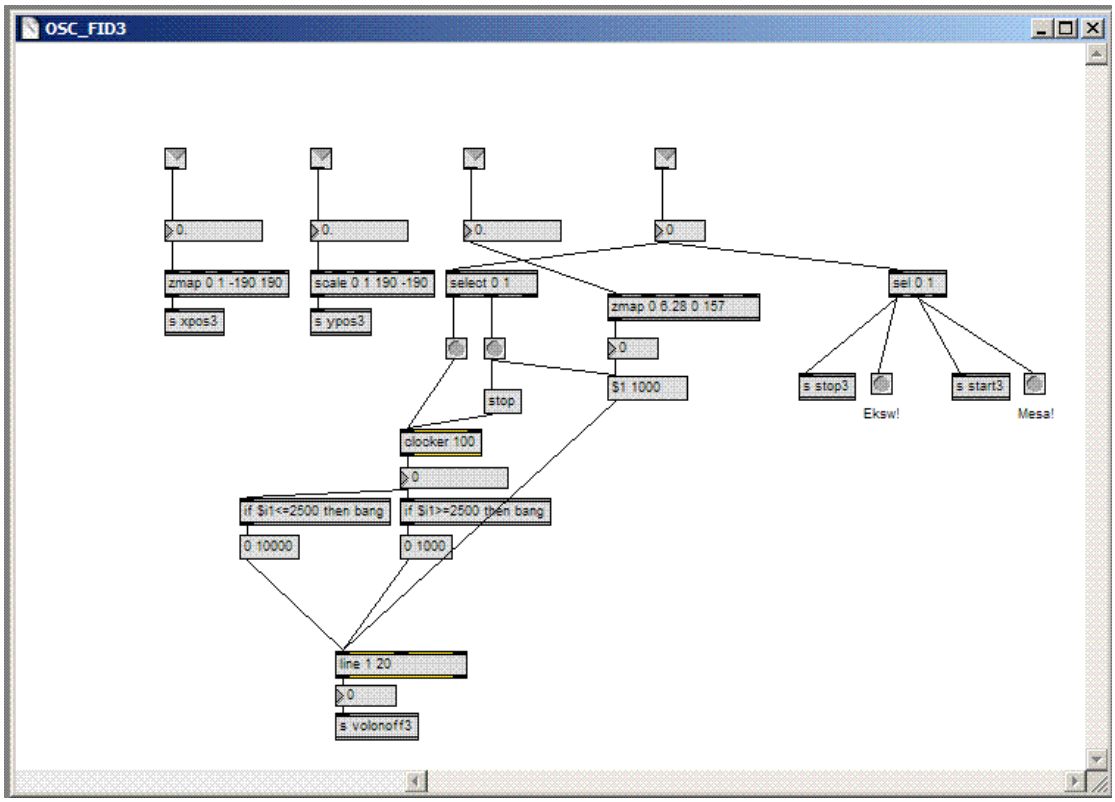
We made this subpatch that sends through OSC the values of start/stop (value: 1/0), x position (value: decimal 0 to 1), y position (value: decimal 0 to 1) and rotation (value: decimal 0 to 6.28) for each fiducial separable. The only thing we have to pay attention in this subpatch is to give the right ip-address (169.254.74.108) and port number (UDP 1234) to achieve the desired communication between the 2 computers.

MAX PATCH ON COMPUTER 2 (RECEIVING OF FIDUCIALS' PARAMETERS AND SOUND SPATIALIZATION)



{IMAGE 58}: "Receiving parameters via OSC to a second computer"

On the second computer, we use this patch to receive (from the port we have set as default from the previous patch) all those parameters for each fiducial. We have a separable patch for each fiducial, which contains all we need to use for our project.

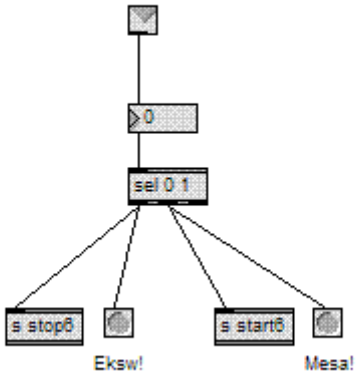


{IMAGE 59}: "Fiducial 3 – receiving and adjustments of the fiducials parameters"

We can see on this patch 4 inlets. The fist one is about x position and gives us values from 0 to 1. We want to convert these values from -190 to 190 that is why we use a "zmap" object

box. The second inlet is about y position; there also we use a “zmap” for values from -190 to 190. The third inlet refers to rotation parameter. Its value is from 0 to 6.28. We use a zmap to convert them from 0 to 157. The fourth inlet is about start and stop (add or remove). We take a “1” when we add a fiducial and a “0” when we remove it. Because there was a problem with the fiducial tracking method, we used a “clocker” that makes the following computation: If we loose the tracking of a fiducial for less than 2500 ms, the clocker starts fading out the volume to zero level in 10000 ms from its last value and when the camera tracks again the fiducial the fading out process stops and the volume takes a new value. If we loose the tracking for more than 2500 ms, then the volume fades to zero in 1000 ms.

The same patch describes fiducial 4 and 5.

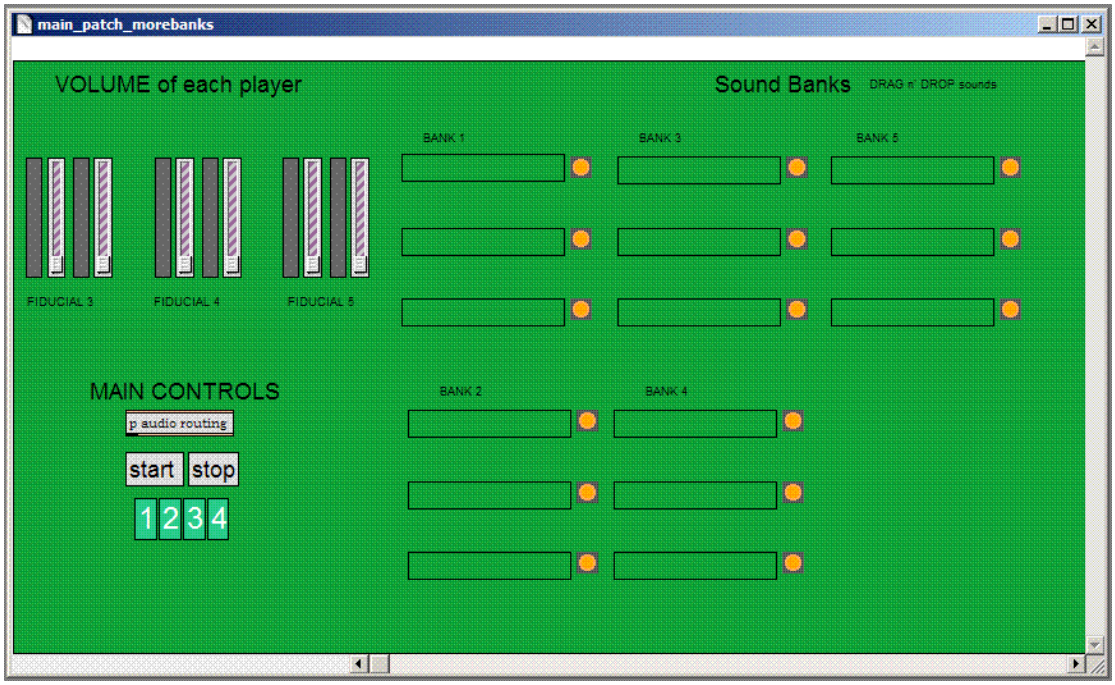


{IMAGE 60}: “Fiducial 6 – Start / Stop”

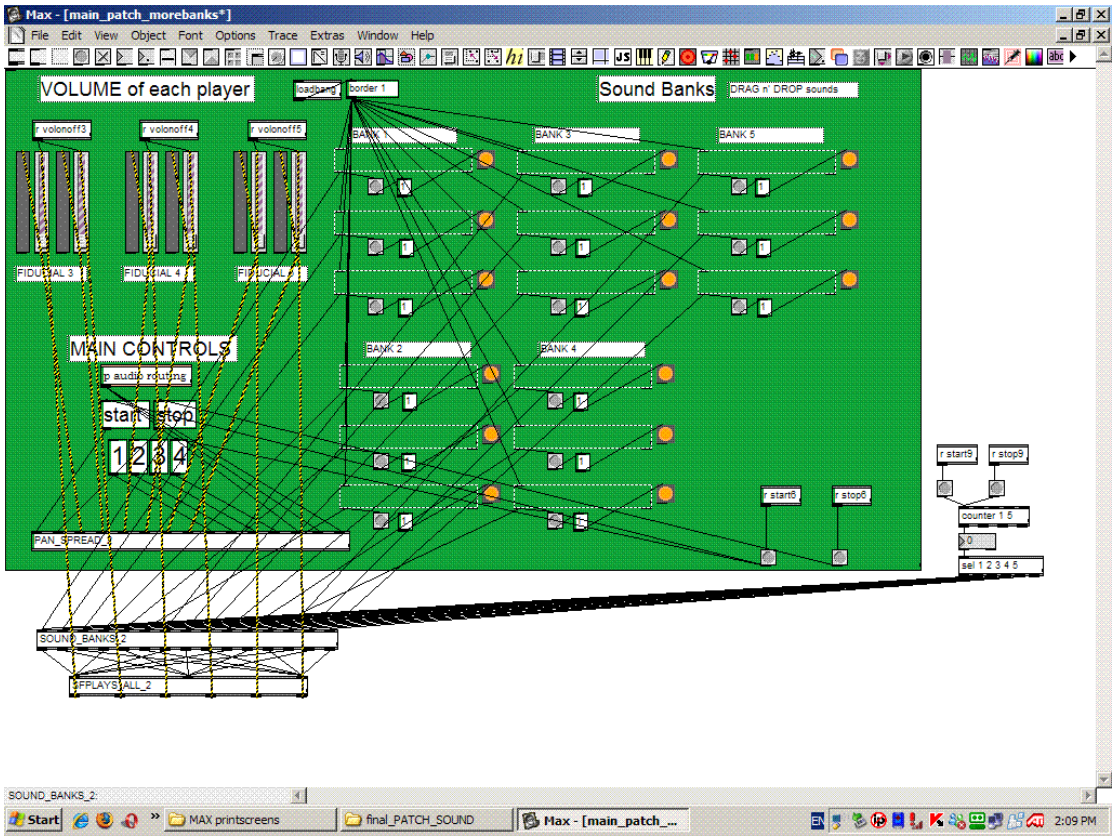
For fiducial with id=6, we only need the start/stop parameter. The same patch describes fiducial 7 and fiducial 9.

{IMAGE 60} : “Fiducial 8 – rotation”

For fiducial 8, we receive only its rotation values (from 0 to 6.28).



{IMAGE 62}: “Main Sound patch - Locked”

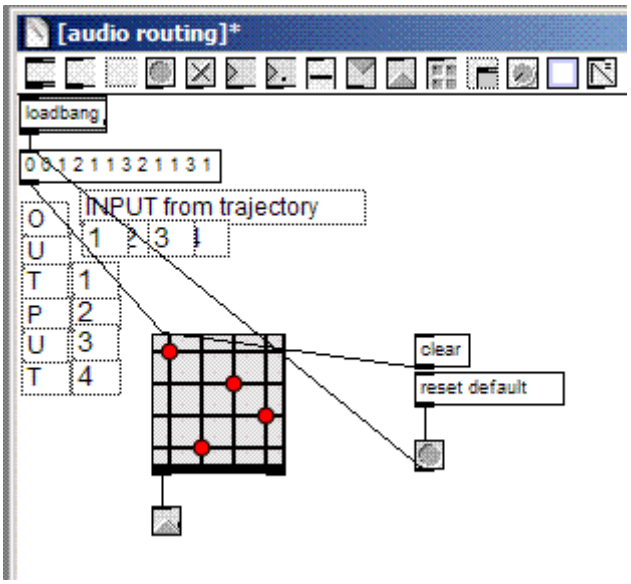


{IMAGE 63}: “Main Sound patch - unlocked”

This is the main patch that includes all the parameters concerning the sound (loading samples, sample players, sound spatialization, audio routing and many more).

We can see here the “Main Controls” for the patch such as:

- **“audio routing”**: This sets default values for the sound output parameters, for a 4 channel sound system, but the user is able to change manually these values. We can see that we have a “loadbang” that triggers specific values on this switching matrix control, based on our speaker configuration.



{IMAGE 62}: “audio routing”

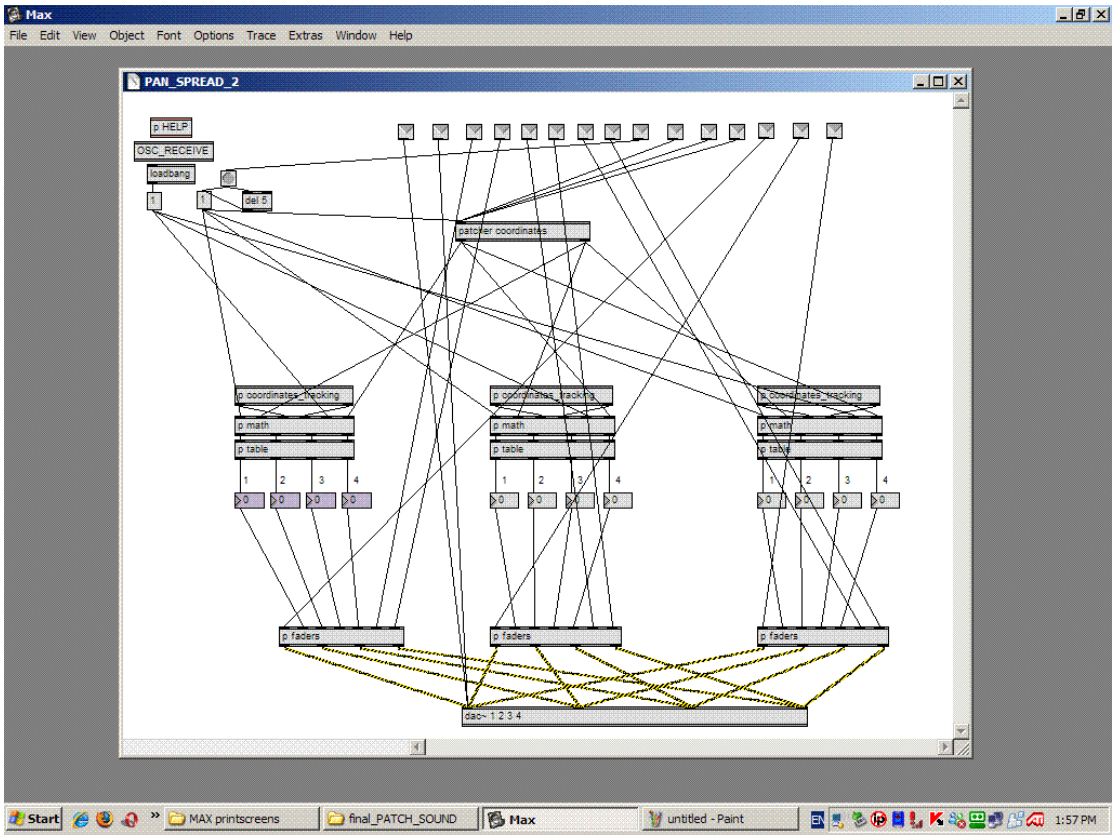
- **“Start/ Stop”**: These two buttons turn on or off the whole system sound engine. When we put on the surface the fiducial 6, we trigger the start button and the 4 speaker configuration.
- **“1, 2, 3, 4”**: These buttons trigger the speaker configuration.

We can also see the “volume control of each player”. Each player is a stereo playback engine (sfplay~), which is why we have 2 faders for each player; with a separable vu meter each. We have connected each fader pair with “receive volonoff”. This means

that when we rotate fiducial 3, 4 and 5, we can increase or reduce the main volume level of each player.

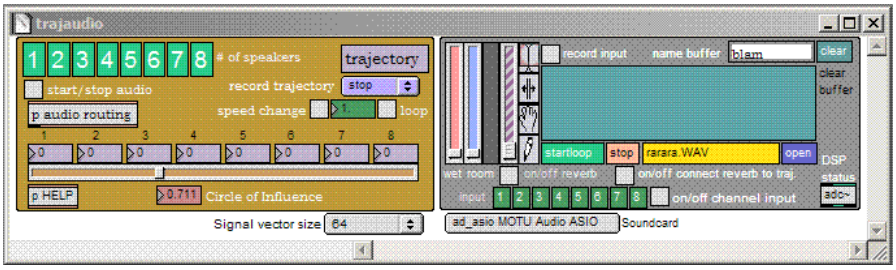
We are using “dropfile”, for defining a region for dragging and dropping audio files, for loading each sample bank. Because we want this region to be viewable, we use a “loadbang” charging a “boarder 1”. We also added leds near each file dropping region, for helping the user to understand when a sample is loaded.

For triggering the five different sound banks, we use fiducial 9. Whether we add or remove this fiducial, a counter gives us the opportunity to select (with a serial way) one sound bank a time.

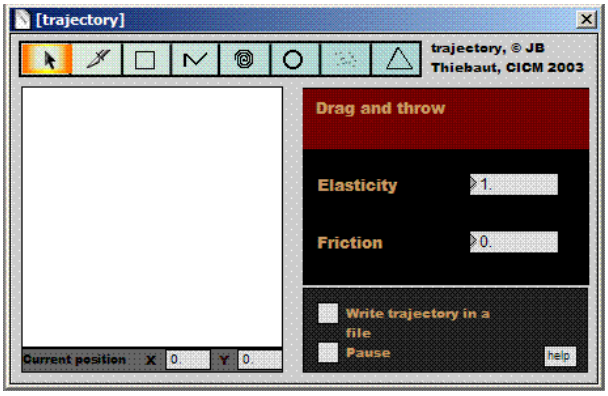


{IMAGE 65}: “Spatial subpatch”

This is a patch that describes the spatialization parameters. It is based on a patch that we found on the internet called “trajaudioapp.mxf”. This patch allows us to pan a soundfile and/or input in up to eight channels using the trajectory object (by JB Thiebaut), math by Yves Gigon and the rest by Carey Dodge, 2005.



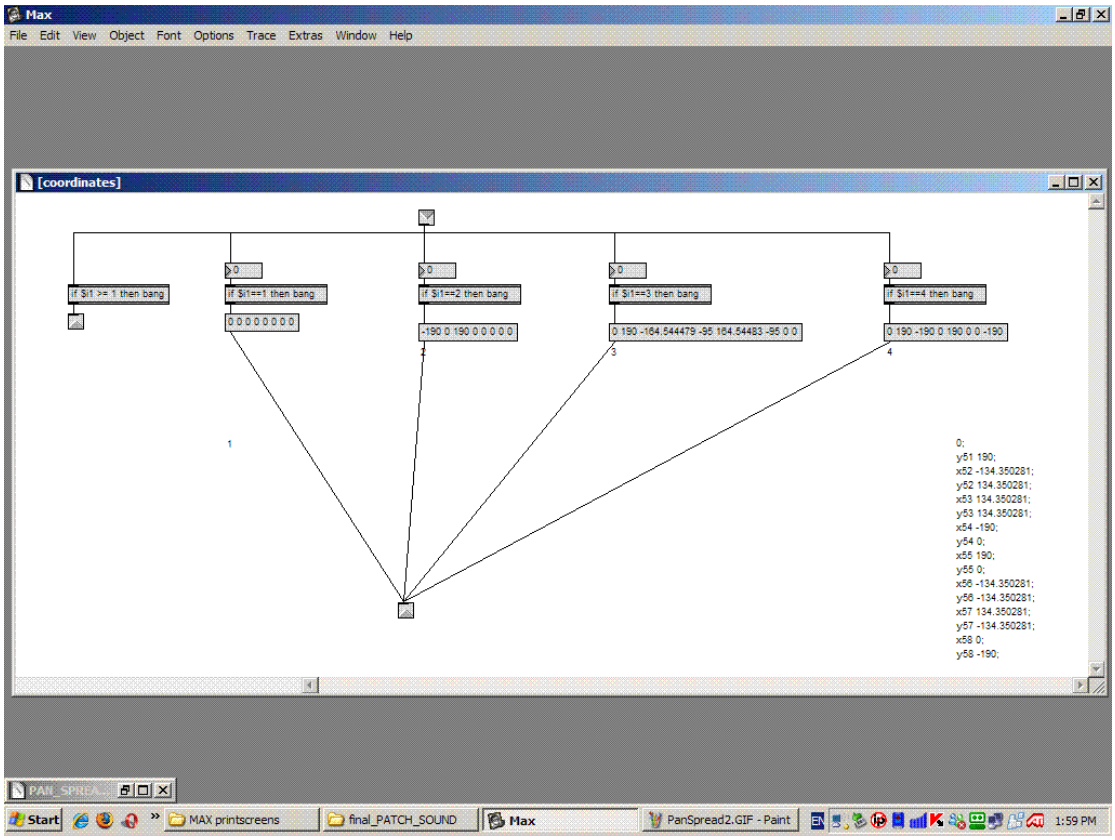
{IMAGE 66}: “trajaudioapp.mxf”



{IMAGE 67}: “trajectory”

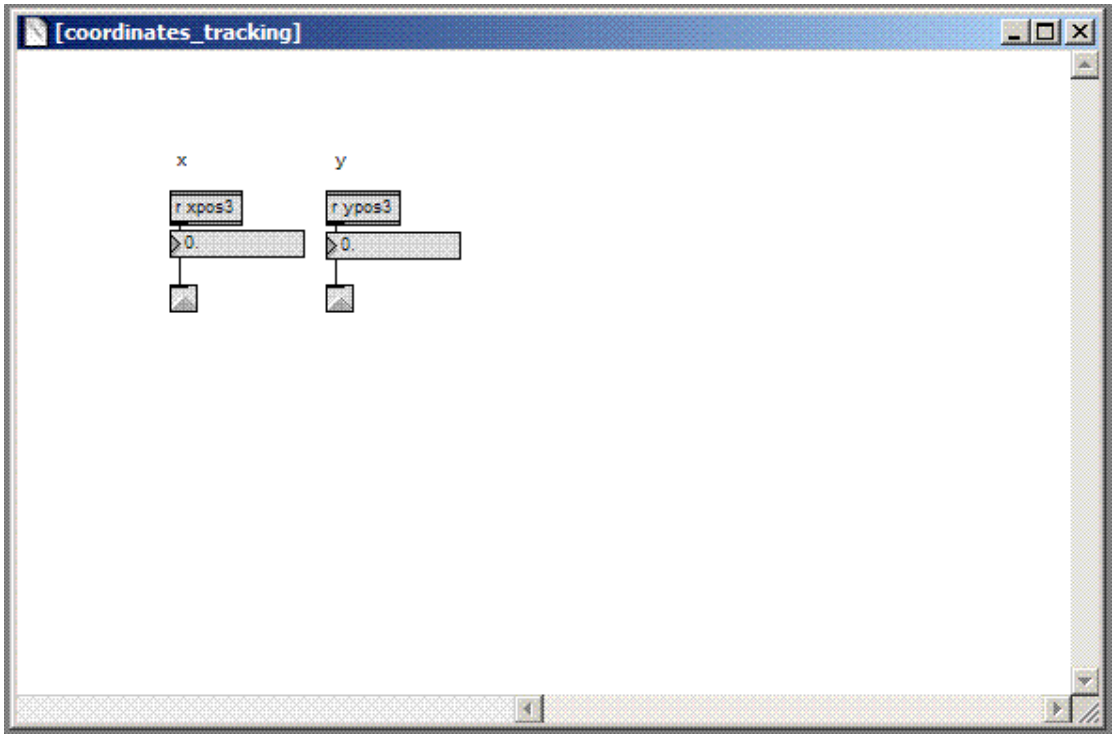
The virtual speaker placement is in a clockwise formation evenly spaced in a circle (except for 1 and 2 channel panning). Channel 1 is always at the top of the circle; however output channels can be remapped. This is a completely modular panning patch.

Our own patch consists from other smaller patches. These are:



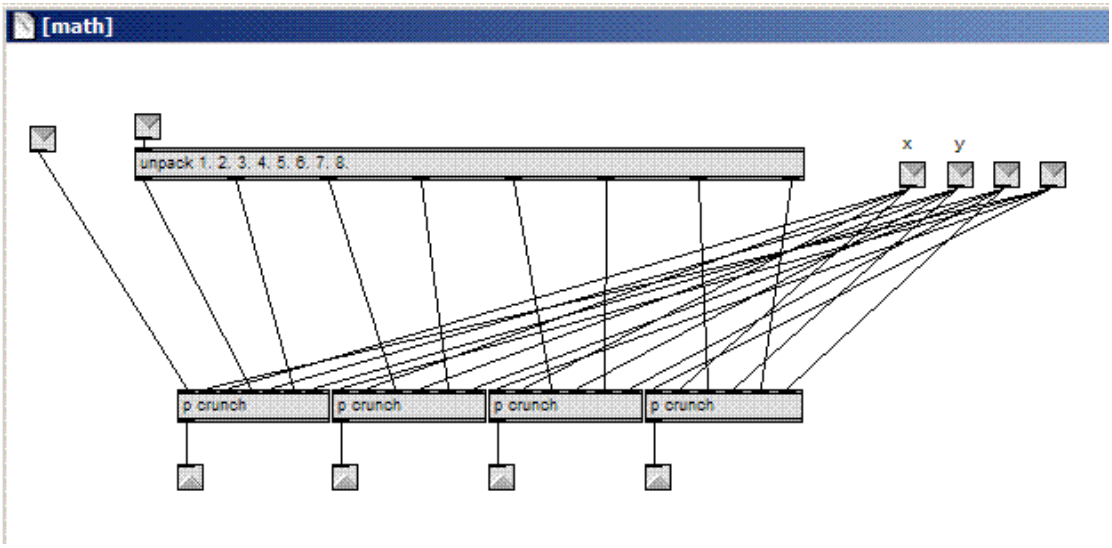
{IMAGE 68}: “Number of speakers and coordinates”

In this patch, we can see the four different cases about each different speaker configuration. To understand each case, let’s say we have a 4 speaker configuration. For each specific case, we give specific coordinates for each speaker separable. These coordinates describe the placement for each speaker in the room. We have x and y axes. Their value is from -190 to 190. We see the numbers “0 190 -190 0 190 0 0 190”. So the coordinates for each speaker are: (0, 190) for speaker 1, (-190, 0) for speaker 2, (190, 0) for speaker 3 and (0, 190) for speaker 4.



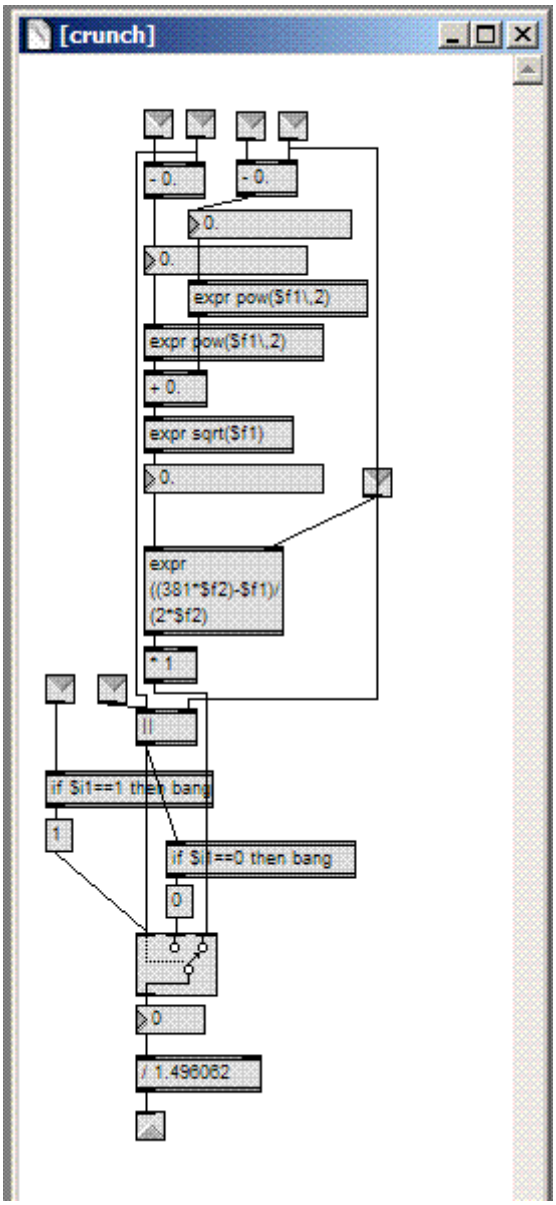
{IMAGE 69}: “Fiducial 3 – x and y position values”

In this patch we receive the values for x and y position.



{IMAGE 70}: “subpatch containing all the math processing”

Here we unpack the values of the coordinates, to send them to subpatch crunch, together with other parameters, such as x and y position.



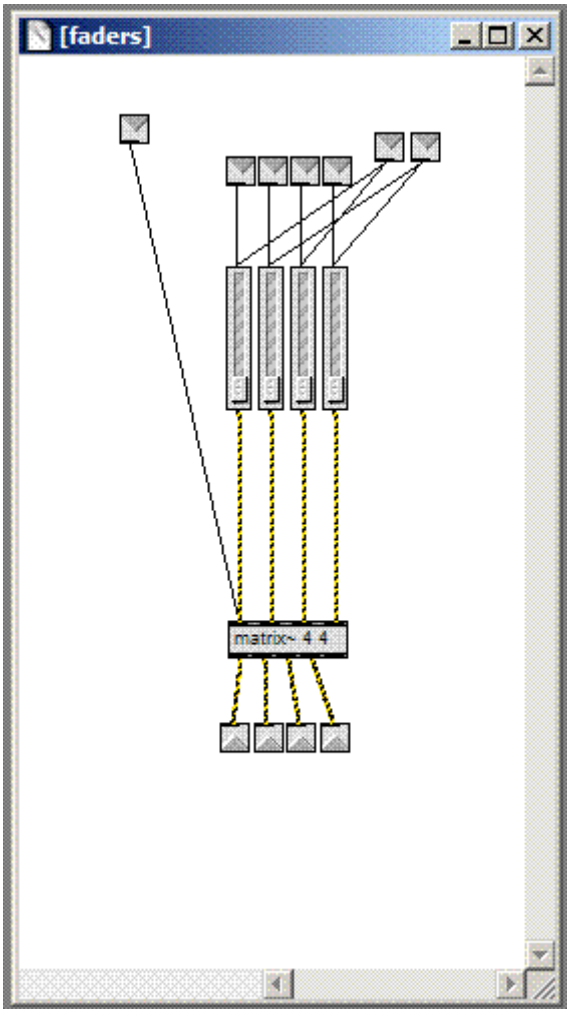
{IMAGE 71}: “subpatch crunch – calculate VBAP”

This patch makes all the calculations for the VBAP. There are 2 cases, a switcher will decide for the result (If the number in either inlet is not 0, then the output is 1 (cases of 2, 3, 4 speakers). If the number in both of the inlets is 0 (case of 1 speaker setup), then the output is 0. A number in the left inlet triggers the output). We need to know the values of inlets 3, 4, 5, and 6 for the calculation.

Inlet 3 refers to “x position” value, inlet 4 refers to the first coordinate value from the “patcher coordinates” subpatch, inlet 5 refers to “y position” value and inlet 6 refers to the second coordinate value from the “patcher coordinates” subpatch. For

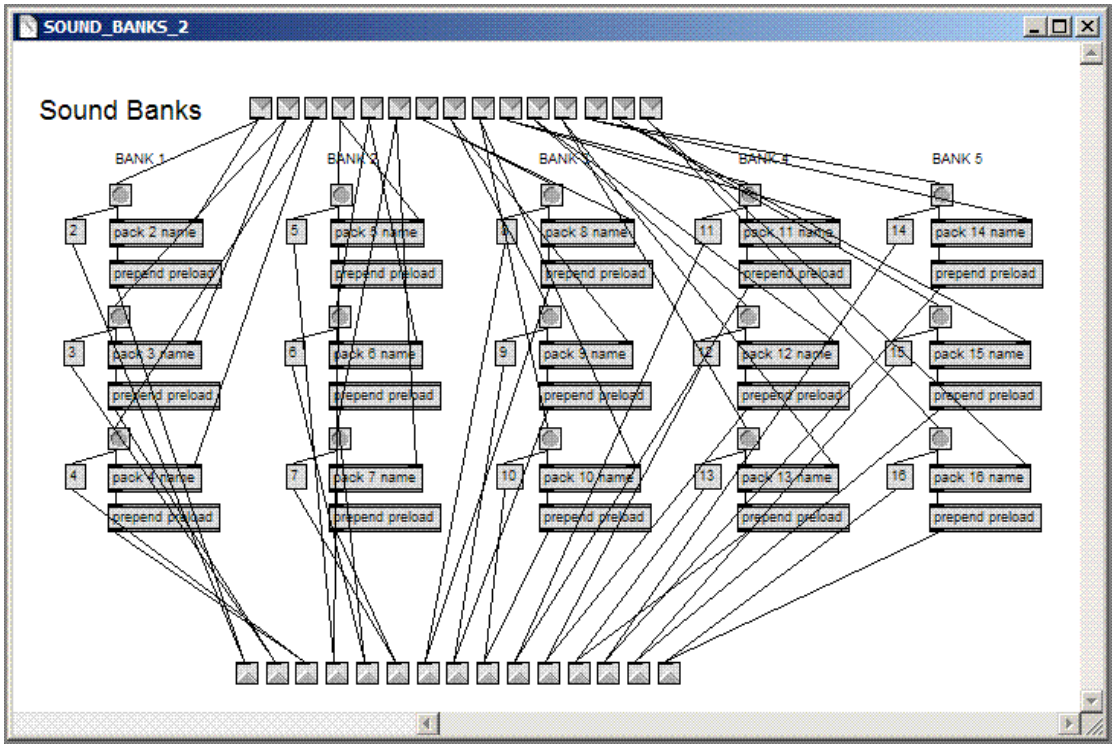
the case of 2 or 3 or 4 speakers' setup, we have to make the following calculation: $\frac{(381\times1)-\sqrt{(inlet_4-inlet_3)^2+(inlet_6-inlet_5)^2}}{2}=A$

Where: $result=\frac{A}{1,496062}$



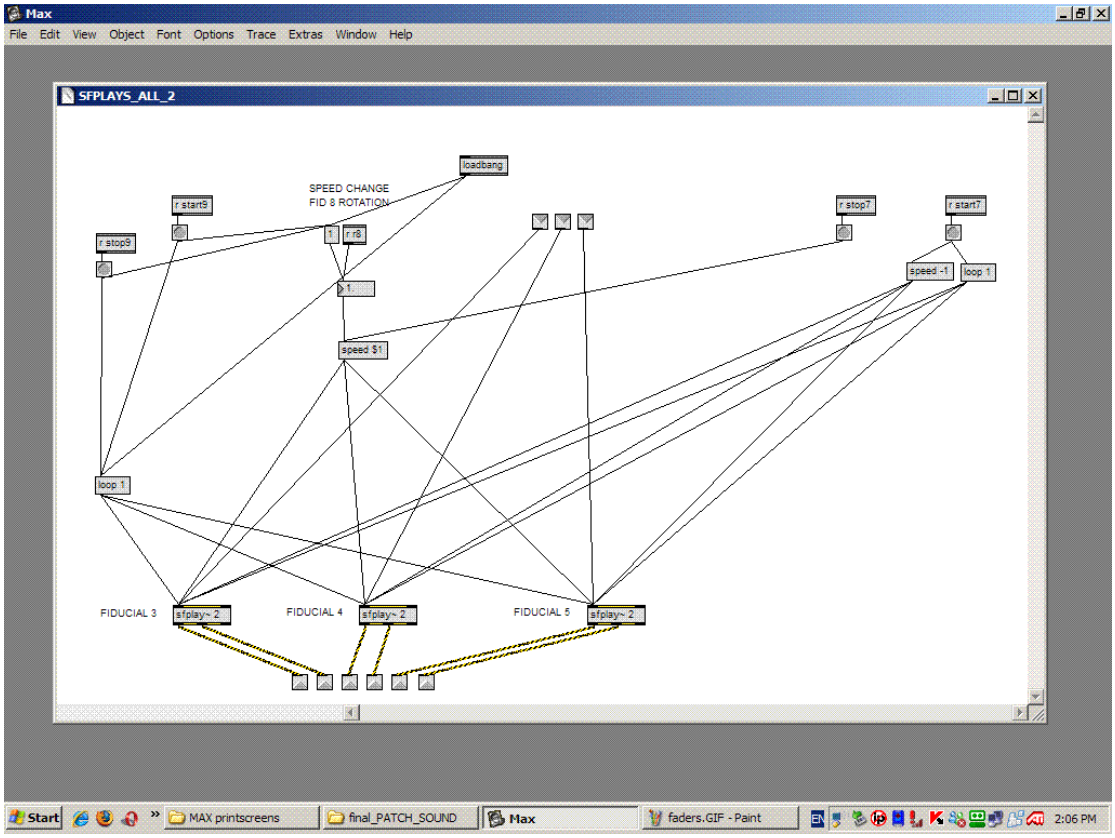
{IMAGE 72}: "A fader for each speaker – volume control – matrix"

In this patch we have 4 faders connected to 4 x 4 matrix. Through them we take the value for the volume for each channel (each speaker). We can control the main volume of them from the faders shown on the main patch.



{IMAGE 73}: “Sound Banks patch – preloads – cues for 15 different samples”

Another subpatch that we can find in the main patch is this that refers to the sound banks. Each sfplay~ can load a large number of samples, but for this occasion we have to preload the sample and start its playback with a specific way. We have 15 different samples and we need 15 different preloads and 15 different playback start buttons. All those are triggered with the fiducial 9 (bang start/stop).



{IMAGE 74}: “sfplay~ x 3 – playback engines for fiducials 3, 4, 5”

This is the last part of our main patch and contains three “sfplay~” playback engines. We can see that we receive the rotation value of fiducial 8, for changing the playback speed of all samples together. Values from 0 to <1 make the playback speed slower and values from 1 to 6.28 make the playback speed faster. We also use the on / off of fiducial 7 to play the sample reversed. With a loadbang, we make all the samples to loop, for a continuously playback.

11.7. User Tests

{IMAGE 75}: “exploring our interface”

A short testing phase followed the development of the project. We invited some people to try and explore our music mixing approach. The objects were first given an operation manual with instructions on how it works. Then they were left alone to try the interface and fill out a question-form according to with their experience.

11.7.1. Questions & Answers

- **Did you find our interface functional?** (scale 1 to 5)

  - *Nick (age: 23, music technology student)* = 3
  - *Marina (age: 22, music technology student)* = 3
  - *Makis (age: 21, music technology student)* = 4
  - *Theodore (age: 23, music technology student)* = 4
  - *George (age: 20, music technology student)* = 5
  - *Spiros (age: 18, music technology student)* = 4

- **Was it easy to learn how to use it?** (scale 1 to 5)

  - *Nick (age: 23, music technology student)* = 5
  - *Marina (age: 22, music technology student)* = 5
  - *Makis (age: 21, music technology student)* = 4
  - *Theodore (age: 23, music technology student)* = 5
  - *George (age: 20, music technology student)* = 4
  - *Spiros (age: 18, music technology student)* = 5

- **Are you interest to experiment more with this new interface?** (scale 1 to 5)

  - *Nick (age: 23, music technology student)* = 5
  - *Marina (age: 22, music technology student)* = 5
  - *Makis (age: 21, music technology student)* = 5
  - *Theodore (age: 23, music technology student)* = 4
  - *George (age: 20, music technology student)* = 5
  - *Spiros (age: 18, music technology student)* = 5

- **Is there something more that you wish this interface have?**

  - *Nick (age: 23, music technology student)*: I wish the whole interface reacted faster. I also wanted to had smaller size so it can be portable and I would like if was having some effects, like reverb for example.
  - *Marina (age: 22, music technology student)*: I wish it had more sound banks.
  - *Makis (age: 21, music technology student)*: I would like the present of an effect processor and I would also love to have more fiducials dedicated as playback engines for more sounds on the table at the same time.
  - *Theodore (age: 23, music technology student)*: I would like to see some more parameters on it.

- *George (age: 20, music technology student)*: More Sample Banks please!
- *Spiros (age: 18, music technology student)*: More Sample banks and more sounds together on the surface.

After these user tests, we decided to add more sound banks to the interface (when the tests took place the interface had only two sound banks).

11.8. Conclusions and Future Improvements

On this project we came across a large number of parameters to be used for great experimentation. Unfortunately, the absence of the required hardware forced us to use some of those parameters and find ways to overcome any limitation that showed up.
I think this project can be further developed in many ways.

- We can use some of the fiducials parameters for controlling an effect processor.

- We can have more fiducial based playback engines and bigger table surface. I think that 24 playback engines will be something near to an analog mixing console. We can also add a kind of equalizing.

- We can use an ideal type of camera (armed with a filter that lets the camera track in IR luminance conditions) for faster and more reliable fiducial tracking and also use more expensive and proper materials for the construction part, as a lot of other familiar projects do.

- The use of a projector for visual feedback on the surface of the table would enhance the user experience.

- We can give the user the possibility to record his/her performance.

- We can add the finger recognition for controlling other parameters.

- We can use a room properly installed, and people inside it wearing T-Shirts with fiducial images stuck onto them, moving all the time. This would be like a big table surface...

12. REFERENCES

[AudioCube] – AUDITE, “AudioCube”, available at: http://www.audite.at/en/projects_audiocube.html [accessed 21/04/08]

[AudioCubes] – Stephane Bronckers, Peter Vuchelen and Jimmy Verhulst, “AudioCubes”, available at: http://www-ccrma.stanford.edu/~bschiett/audiocubes/ [accessed 21/04/08]

[BIG04] – Stephen J. Bigelow, “Haptics Make It Happen”, General Computing article, 2004

[Biocontrol] – BioMuse, available at: http://www.biocontrol.com/biomuse.html [accessed 15/05/08]

[BioMuse] – Stanford researchers R. Benjamin Knapp and Hugh S. Lusted, available at: http://ccrma.stanford.edu/CCRMA/Courses/252/sensors/node26.html [accessed 21/04/08]

[BK05] – Ross Bencina and Martin Kaltenbrunner, “The Design and Evolution of Fiducials for the reacTIVision System”, Music Technology Group, Audiovisual Institute Universitat Pompeu Fabra, Barcelona, Spain, 2005

[Blob] – “Active Blobs”, 1997, 1998 Image and Video Computing Group – Boston University, available at: http://www.cs.bu.edu/groups/ivc/ActiveBlobs/Home.html [accessed 22/04/08]

[BMH03] – Rodney Berry, Mao Makino, Naoto Hikawa and Masumi Suzuki, “The Augmented Composer Project: The Music Table”, Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR), 2003

[CLF03] – Claude Cadoz, Annie Luciani, Jean-Loup Florens and Nicolas Castagne, “Artistic Creation and Computer Interactive Multisensory Simulation Force Feedback Gesture Transducers”, Proceedings of the 2003 Conference on New Interfaces for Musical Expression (Nime03), 2003

[dtouch] – Enrico Costanza, “Audio d-touch”, available at: http://web.media.mit.edu/~enrico/research/research.php?projectTitle=Audio%20d-touch [accessed 21/04/08]

[Ears, “Feedback”] – Ears: ElectroAcoustic Resource Site, available at: http://www.ears.dmu.ac.uk/spip.php?rubrique87 [accessed 21/04/2008]

[Ears, “Interactivity”] – Ears: ElectroAcoustic Resource Site, available at: http://www.ears.dmu.ac.uk/spip.php?rubrique40 [accessed 20/04/2008]

[Ears, “Interface”] – Ears: ElectroAcoustic Resource Site, available at: http://www.ears.dmu.ac.uk/spip.php?rubrique274 [accessed 20/04/2008]

[Ears, “Mapping”] – Ears: ElectroAcoustic Resource Site, available at: http://www.ears.dmu.ac.uk/spip.php?rubrique384 [accessed 21/04/2008]

[Ears, “Spatialisation”] – Ears: ElectroAcoustic Resource Site, available at: http://www.ears.dmu.ac.uk/spip.php?rubrique240 [accessed 21/04/2008]

[EyeCon] – Available at: http://eyecon.palindrome.de/ [accessed 21/04/08]

[FB97] – George W. Fitzmaurice, William Buxton, “An Empirical Evaluation of Graspable User Interfaces: towards specialized, space-multiplexed input”, Human Factors in Computing Systems, CHI ’97 Conference Proccedings, ACM Press, 1997, available at: http://www.sigchi.org/chi97/proceedings/paper/gf.htm [accessed 20/04/2008]

[FELGT] – S. Sidney Fels, Geoffrey E. Hinton. "Glove-TalkII: A neural network interface which maps gestures to parallel formant speech synthesizer controls." IEEE Transactions on Neural Networks, 9 (1998), No. 1, 205-212

[GG99] – Guy E. Garnett and Camille Goudeseune, “Performance Factors in Control of High-Dimensional Spaces”, Proc. 1999 International Computer Music Conference, Beijing, 1999, available at: http://zx81.isl.uiuc.edu/camilleg/icmc99.html [accessed 21/04/08]

[GOT06] – Suguru Goto, “The Case Study of An Application of The System, BodySuit and RoboticMusic-Its Introduction and Aesthetics”, Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME060), Paris, France

[Hands] – Michel Waisvisz, available at: http://www.crackle.org/TheHands.htm [accessed 21/04/08]

[HBC96] – Hewett-Baecker-Card-Carey-Gasen-Mantei-Perlman-Strong-Verplank, “ACM SIGCHI Curricula for Human-Computer Interaction”, 1996, available at: http://sigchi.org/cdg/cdg2.html [accessed 20/04/2008]

[HK00] – Hunt A. and R. Kirk. "Mapping Strategies for Musical Performance." In M. Wanderley and M. Battier, eds. Trends in Gestural Control of Music. Ircam - Centre Pompidou. 2000

[HOR04] – Eva Hornecker, "A Design Framework for Designing Tangible Interaction for Collaborative Use" Paper at the Danish HCI Symposion, University of Aalborg, Nov. 16. HCI Lab Technical Report no 2004/1. pp.57-61

[HUN03] – Andy Hunt, Lecture Notes in "Electronic Musical Instruments", University of York, UK, 2003

[HWK00] – Andy Hunt, Marcelo M. Wanderley and Ross Kirk, "Towards a Model for Instrumental Mapping in Expert Musical Interaction", in proceedings of the 2000 International Computer Music Conference, 2000

[HWP02] – Hunt, A., Wanderley, M.M., and Paradis, M., 'The Importance of Parameter Mapping in Electronic Instrument Design', in Proc of the 2002 Conference on New Interfaces for Musical Expression (NIME-02), 2002

[IMG3] – taken from: http://www.cse.unsw.edu.au/~waleed/thesis/node33.html and http://www.base2john.com/itp/pcomp/ [accessed 08/06/08]

[IMG6] – taken from: http://hct.ece.ubc.ca/research/glovetalk2/ [accessed 08/06/08]

[IMG16] – taken from: http://www.inition.com/inition/product.php?URL_=product_mocaptrack_vicon_motioncapture&SubCatID_=19 [accessed 08/06/08]

[IMG17] – taken from: http://www.physikinstrumente.com/en/products/prdetail.php?sortnr=901200 [accessed 08/06/08]

[IMG26] – taken from: www.gersic.com/plugins/index.php?daCat=-2 [accessed 08/06/08]

[IMG27] – taken from: www.thoosje.com/vista-forum/viewtopic.php?t=115

and predatorramboxxx.blogspot.com/ [accessed 08/06/08]

[IMM07] – Immersion, "The Value of Haptics", 2007

[InfoMus] – InFomus Lab, "The EyesWeb Project", available at: http://www.infomus.dist.unige.it/ [accessed 21/04/08]

[Inition] – Inition, available at: http://www.inition.co.uk/inition/product.php?URL_=product_mocaptrack_polhemus_patriot&SubCatID_=18 [accessed 21/04/08]

[ISScube] – somethingonline.org, "Interactive Surround Sound (ISS) Cube", available at: http://reactable.iua.upf.edu/?related [accessed 21/04/08]

[jazzmutant] – jazzmutant, “Lemur”, available at: http://www.jazzmutant.com/lemur_overview.php [accessed 14/05/08]

[JSM94] – R.J.K. Jacob, L.E. Sibert, D.C. McFarlane, and M.P. Mullen, Jr., "Integrality and Separability of Input Devices," ACM Transactions on Computer-Human Interaction, vol. 1, no. 1, pp. 3-26, March 1994

[KB07] – Martin Kaltenbrunner, Ross Bencina, ‘reacTIVision: A Computer-Vision Framework for Table Based Tangible Interaction”, Proceedings of the first international conference on “Tangible and Embedded Interaction” (TEI07). Baton Rouge, Louisiana, 2007

[KMC04] – Kaltenbrunner M., O'Modhrain S., Costanza E., “Object Design Considerations for Tangible Musical Interfaces”, Proceedings of the COST287-ConGAS Symposium on Gesture Interfaces for Multimedia Systems, Leeds (UK), 2004

[LBird2] – Ascension Technology Corporation, available at: http://www.ascension-tech.com/products/laserbird.php [accessed 21/04/08]

[LED98] – Susan J. Lederman , from: http://www.apa.org/monitor/jun98/object.html [accessed 08/06/08]

[Lightning] – Buchla, available at: http://www.buchla.com/lightning/descript.html [accessed 21/04/08]

[LO05] – Livingstone, D., O'Shea, C. 'Tactile Composition Systems for Collaborative Free Sound', *Proceedings of the International Computer Music Conference*, Barcelona, 2005

[loopqoob] – murat n konar interaction design, “loopqoob”, available at: http://www.muratnkonar.com/id/loopqoob/ [accessed 21/04/08]

[MC00] – O'Modhrain, S., Chafe, C., "Incorporating Haptic Feedback into Interfaces for Music Applications" in proceedings of ISORA, World Automation Conference, 2000

[MG97] – Maura Sile O’Modhrain and Brent Gillespie, “The Moose: A Haptic User Interface for Blind Persons”, Proceedings of the Third WWW6 Conference, Santa Clara, CA, April 1997

[MIDI] – available at: http://www.bikexprt.com/cakewalk/midiprob.htm [accessed 11/06/08]

[MOO00] – James A. Moorer, “Audio in the new millennium”, Sonic Solutions, Novato, CA 94945, USA, 2000 – available at: http://www.aes.org/technical/heyser/Heyser.cfm [accessed 20/04/2008]

[MOR81] – Tom Moran, 1981, available at: http://lipas.uwasa.fi/~mj/hci/hci2.html [accessed 20/04/2008]

[MUL07] – Meinard Muller, “Information Retrieval for Music and Motion”, Springer Press, 2007

[MW06] - Mark T. Marshall and Marcelo M. Wanderley, “Vibrotactile Feedback in Digital Musical Instruments”, In Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME06), Paris, France, pp 226-229, 2006.

[NBT03] – Camilla Bannebjerre Nielsen, Malene Benkjær, Maria Tønnesen, Mikkel Byrsø Dan, Morten Wang and Simon Larsen, “The Human Remote Control”, Automatic Perception, Medialogy, 4th semester, Aalborg University, Esbjerg – Copenhagen, June 4, 2003

[OSC] – Shannon Simpson, ‘Open Sound Control, An Overview’, available at: http://www.ixi-software.net/content/info/osc.html [accessed 09/06/08]

[OXTS] – OXTS Inertial+GPS, available at: http://www.oxts.co.uk/default.asp?pageRef=63 [accessed 21/04/08]

[OZS06] – Timur M. Ozsan, “An alternative tangible interface for manipulating 3D audio and multiple media”, MSc in Music Technology, University of York, 2006

[PRI02] – Patten, J., Recht, B., Ishii, H., "Audiopad: A Tag-based Interface for Musical Performance", in Proceedings of Conference on New Interface for Musical Expression (NIME '02), Dublin, Ireland, May 24 - 26, 2002

[PUL00] – Pulkki Ville, ‘Generic panning tools for MAX/MSP’, *Proceedings of the International Computer Music Conference 2000*, Berlin, Germany: International Computer Music Association.

[ROA96] – Curtis Roads, “The Computer Music Tutorial”, The MIT Press, 1996

[RUM04] – Francis Rumsey, “Spatial Audio”, Focal Press, 2004

[RWD97] – Joseph Butch Rovan, Marcelo M. Wanderley, Shlomo Dubnov and Phillipe Depalle, “Instrumental Gestural Mapping Strategies as Expressivity Determinants in Computer Music Perfomance”, Analysis-Synthesis Team/Real-Time Systems Group – IRCAM – France, 1997

[RYA91] – Joel Ryan, “Some Remarks on Musical Instrument Design at STEIM”, 1991, available at: http://www.steim.org/steim/texts.php?id=3 [accessed 21/04/08]

[SAP00] – S. Sapir, “Interactive Digital Audio Environments: Gesture as a musical parameter”, In Proc. COST-G6 Conference on Digital Audio Effects (DAFX-00), 2000

[SONLG] – Laetitia Sonami, Lady’s Glove, available at: http://www.sonami.net/lady_glove2.htm [accessed 21/04/08]

[Theremin] – Leon Theremin, available at: http://www.thereminworld.com/article.asp?id=17 [accessed 21/04/08]

[Thunder] – Buchla, available at: http://www.buchla.com/historical/thunder/ [accessed 21/04/08]

[TroikaTronix] – Troika Tronix, “Isadora”, available at: http://www.troikatronix.com/isadora.html [accessed 21/04/08]

[TUF89] – E. R. Tufte, “Visual Design of the User Interface”, IBM Corporation, Armonk, N.Y., 1989

[VER94] – Roel Vertegaal. “An evaluation of input devices for timbre space navigation”, MPhil, dissertation, Bradford, UK: Department of computing, University of Bradford, 1994

[vvvv] – Available at: http://vvvv.org/tiki-index.php [accessed 21/04/08]

[Webopedia, “GUI”] – Webopedia: Online encyclopedia dedicated to computer technology, available at: http://www.webopedia.com/TERM/G/Graphical_User_Interface_GUI.html [accessed 21/04/08]

[Wikipedia, “Feedback”] - Wikipedia: The Free Encyclopedia, available at: http://en.wikipedia.org/wiki/Feedback [accessed 21/04/08]

[Wikipedia, “Haptic”] - Wikipedia: The Free Encyclopedia, available at: http://en.wikipedia.org/wiki/Haptic#Arts [accessed 21/04/08]

[Wikipedia, “Human-computer interaction”] – Wikipedia: The Free Encyclopedia, available at: http://en.wikipedia.org/wiki/Human-Computer_Interaction [accessed 20/04/2008]

[Wikipedia, “MIDI”] - Wikipedia: The Free Encyclopedia, available at: http://en.wikipedia.org/wiki/MIDI [accessed 09/06/08]

[Wikipedia, “OSC”] - Wikipedia: The Free Encyclopedia, available at: http://en.wikipedia.org/wiki/OpenSound_Control [accessed 09/06/08]