

Empty-handed Gesture Analysis in Max/FTS

Axel G. E. Mulder, S. Sidney Fels and Kenji Mase
ATR Media Integration and Communications laboratories
Seika-cho, Soraku-gun, Kyoto, 619-02 Japan
{mulder, fels}@mic.atr.co.jp

Abstract

We have developed objects for the computation of human movement primitives within Max/FTS, believing that it will lead to gestural interfaces that will reduce the cognitive load for simultaneous multidimensional control tasks such as in sound design. Focussing on empty-handed gestures for a shape manipulation task called sound sculpting, where shape description parameters are mapped to timbral parameters, we have programmed Max/FTS objects for interfacing serial peripheral devices such as sensors, objects for geometric and kinematic computation and objects for the computation of features of human body parts.

1 Motivation

We report in this paper on work in progress to develop gestural interfaces that allow for simultaneous multidimensional control. An example of simultaneous multidimensional control can be found in music composition and sound design, in which task the control of timbre involves many inter-dependent parameters simultaneously. For any sound designer or sound composer the control of these parameters involves a significant amount of cognitive processing to coordinate his or her motor system when mouse-and-keyboard interfaces are used. Thus, to reduce the cognitive load for the sound designer, it is necessary to design a human computer interface that implements data reduction and/or an interface that exploits the capability of human gestures to effortlessly vary many degrees of freedom simultaneously. In terms of data reduction, sound synthesis models that parametrize timbre independently at a perceptual level only generate a limited number of timbres. While the human hand is well-suited for multidimensional control due to its detailed articulation, most gestural interfaces do not exploit this capability due to a lack of understanding of the way humans produce their gestures and what meaning can be inferred from these gestures. In order to facilitate quick prototyping and experimentation with a multitude of gestural analysis methods, we are approaching the design problem by creating a set of tools that facilitate the computation of various gesture features and parameters.

2 Relevant gestures

When hand shape, hand position and hand orientation are changing simultaneously such as in the physical manipulation of objects shape, the dimensionality

of the control is highest. Although the manipulation of shape (i.e. claying or moulding) in the physical world is most effective with touch and force feedback, our assumption is that these forms of feedback can be replaced by acoustic feedback with some compromises. The generation of appropriate touch and force feedback, while exploiting the maximum range of motion of gestures, is currently technically too challenging in a shape manipulation task. We are also interested in the use of dynamic signs (hand shape is constant, but hand position and/or orientation is changing), although they represent less dimensions of simultaneous control. The other end of the spectrum, in terms of control dimensionality, is represented by static signs, which are not useful in the task we are interested in (i.e. multidimensional control) other than for selection. In previous work on gesture interfaces such as [2] it has been noted that, since humans do not reproduce their gestures very precisely, natural gesture recognition is rarely sufficiently accurate due to classification errors and segmentation ambiguity. Only when gestures are produced according to well-defined formalisms, such as in sign language gesture recognition, will automatic recognition have acceptable precision and accuracy. However, a gesture formalism will require tedious learning by the user. Thus we do not compute or analyse any abstract symbolic representation of the gestures produced by the user, but instead focus on the continuous changes represented by the gestures.

3 Implementation

We are using Max/FTS [5] on an R10000 SGI Onyx with audio/serial option (6 serial ports) to interface two Virtual Technologies Cybergloves (instrumented gloves that measure hand shape - see also [4]) and a Polhemus Fastrak (a sensor for measuring position and orientation of a physical object, such as the human hand, relative to a fixed point). Figure 1 illustrates the hardware device setup. While the Cyberglove is probably one of the most accurate means to register human hand movements, we have found accurate measurement of thumb movement difficult due to the fact that the sensors intended for the thumb do not map to single joints but to many joints at the same time. Nevertheless, with a sufficiently sophisticated hand model it is possible to reach acceptable accuracy for our purposes, but the calibration is tedious as well as individual specific. Max/FTS is a

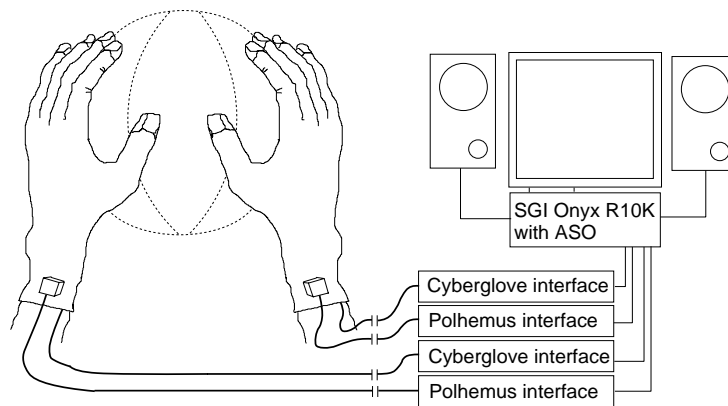


Figure 1: Hardware devices used. The dotted lines represent a virtual surface which the sound designer or sound composer manipulates.

visual, interactive programming environment for real-time sound synthesis and algorithmic music composition developed at IRCAM in Paris. Figure 2 and 4 show typical Max projects. We chose Max/FTS as our platform due to its real-time computation and visual programming capabilities, the access to various synthesis models implemented as editable patches and the fact that it runs on an SGI, thus needing no special sound cards. Max/FTS functionality is extended by linking in dynamic shared objects (DSO) at run-time. In order to facilitate quick and easy prototyping of various gestural analysis computations and allowing for application of the computations to different bodyparts we have developed new Max/FTS objects. We exploit the strong datatype checking of Max/FTS to introduce new datatypes such as *position* and *orientation* as well as lists of these datatypes and a *hand* datatype. This way the user cannot apply objects at an inappropriate abstraction level, which would be possible if computed values were only represented as a number data type. The motivation for distinguishing between abstraction levels is that they help the user when designing and using a mapping of human movement to sound. Both sound and human movement can be represented at various abstraction levels. A mapping will be faster to learn when movement features are mapped to sound features of the same abstraction level. The new objects are listed below according to their level of abstraction.

4 New Max/FTS objects

At the lowest level we have defined two Unix character device drivers:

- *serial* interfaces peripheral devices that communicate using one of the SGI's serial ports
- *client* allows for communication between Max and other processes using TCP sockets.

We are using the client object to communicate data to an OpenInventor 3-D graphics server to display the acquired hand shape, hand position and hand orientation data as a graphic hand. Next, we have defined



Figure 2: Typical example of geometric computing using the new Max/FTS *position* and *orientation* objects. This Max patcher takes a list of *positions*, projects these on the plane that best fits them, then rotates the plane (with projected *positions*) and eliminates z. This patcher enables subsequent fitting of 2-D parametric curves to the x-y coordinates (e.g. the x-y coordinates computed from the *positions* marking the thumb and index fingers).

two sensor interfaces, i.e. peripheral (character) device interfaces that implement a specific protocol:

- *cyberglove* interfaces the Virtual Technologies Cyberglove
- *polhemus* interfaces the Polhemus Fastrak

The following level of abstraction defines new datatypes *position* (x, y and z as floats in one data structure) and *orientation* (a rotation matrix) to facilitate geometric and kinematic computation such as distances between points, orientations of planes etc. and their derivatives. We aim to expand this level with datatypes for points, lines, planes, volumes etc. Currently it consists of the following objects:

- $+P$ and $-P$ add and subtract a *position* (i.e. translate)
- $*P$ multiplies each of x, y and z of a *position* by a constant
- $dotP$ computes the dot-product of two *positions*
- $\|P$ and $normP$ compute the magnitude and the norm of a vector represented by a *position*
- $Pxyz$ and $xyzP$ (un)packs a *position* as three floats (x, y and z)
- $O * P$ multiplies an *orientation* with a *position* (i.e. re-orient a vector)
- TO transposes an *orientation* (i.e. changes the frame of reference of a rotation matrix)
- $*O$ multiplies an *orientation* with another *orientation*
- $EtoO$ computes an *orientation* from Euler angles
- $OtoAA$ and $AtoO$ compute an angle axis representation from an *orientation* and vice versa

The next level of abstraction addresses the fact that the computations now involve ordered and linked lists of *positions* representing for instance human anatomical structures. A new datatype *hand* is used to represent a human hand:

- $packP$ and $unpackP$ (un)pack a number of *positions* into (from) a list of *positions*.
- $packH$ and $unpackH$ (un)pack a number of *positions* into (from) one *hand*, the data structure representing a hand.
- $avgP$ computes the center of mass of a list of *positions*
- $plane$ computes the plane normal vector from a list of *positions* (figure 2).
- $geoHand$ computes an ordered list of *positions*, packed as a *hand* that are relative to the Polhemus transmitter from Cyberglove joint data and Polhemus receiver position and orientation data.

Features such as the distance between thumb and index fingertips, distance between left and right hand palm, palm orientation are easily calculated using the above position and orientation objects in a patcher. We intend to make other objects for the computation of features such as finger and hand curvature using a curve fitting algorithm, estimated grip force using a grasping taxonomy etc..

5 Application

The set of objects is expanding while we are exploring the application of the objects to implement sound sculpting [3]. Figure 3 illustrates the approach we are taking in mapping human movement parameters to sound parameters. It is our strategy to use shape as a means to relate hand movements to sound variations. Hand movements, shape and sound are all multidimensionally parameterized. We exploit the intuitive relations between shape (of physical objects) and timbre as well as shape and manipulation for the design of a sound editing environment where the user can change the sound by applying shape operations to a virtual object using a shape processor. Shape features are subsequently computed and mapped to sound parameters. Hand features that are computed from the acquired hand shape, hand position and hand orientation data using the above set of objects are mapped to shape parameters of a similar abstraction level. If a shape parameter is specified at a different abstraction level than the parameters of the shape processor, it will need to be translated to the parameter space of the shape processor. A shape processor is an algorithm that computes surface data points from shape parameter values. The simplest shape processor is the identity, i.e. it takes surface data points as shape parameters and outputs the same surface data points. If the hand feature computation and shape parameter translation are also taken as the identity, the *positions* that represent the hand will also specify the virtual object surface or in other words, the hands are always "touching" the virtual object. A shape feature is computed from surface data points and represents an abstraction of the shape. For shape feature computation we have developed a number of objects. The object *plane*, discussed also above, computes the plane normal vector from a list of *positions*. Another type of shape feature computation involves superquadrics, a mathematical method to describe a wide variety of shapes with two parameters specifically related to shape (vertical and horizontal "squareness") and 9 others for size, orientation and position of the virtual object. A superquadric surface is defined by the 3-D vector

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_1 \cos^{\epsilon_1}(\eta) \cos^{\epsilon_2}(\omega) \\ a_2 \cos^{\epsilon_1}(\eta) \sin^{\epsilon_2}(\omega) \\ a_3 \sin^{\epsilon_1}(\eta) \end{bmatrix} \quad \text{where} \quad \begin{array}{l} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega < \pi \end{array}$$

Based on [6] we have programmed an *superquadric* object that fits a superquadric shape to a set of *positions*. As the fitting process is iterative, it is computationally expensive and as yet too slow (order of 100 ms) for real-time control of timbre. A simpler approach is

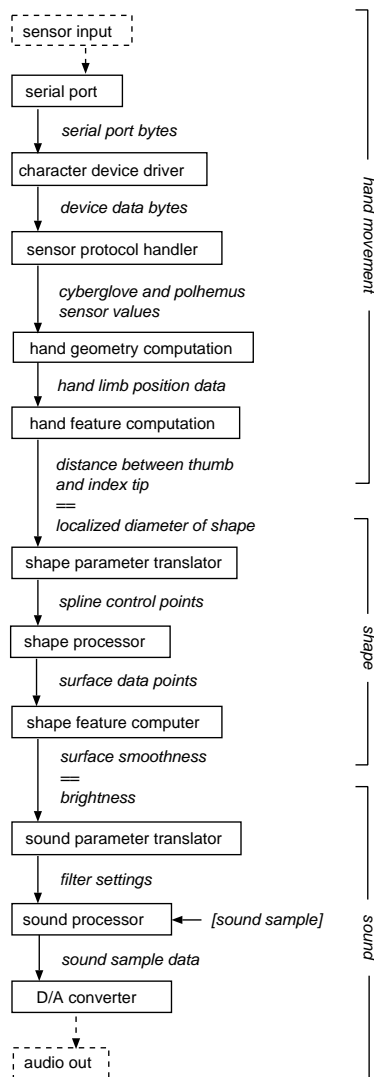


Figure 3: Functional diagram of hand movement to sound mapping. Shape is used as a means to relate hand movement features to sound features. The diagram intends to illustrate mapping at a similar abstraction level of hand movement parameters to shape operators and of shape features to sound parameters. The example shows mapping at the feature abstraction level, but mapping at other levels is equally possible. While the hand feature and shape feature concepts appear in this diagram at the highest level of abstraction we are planning to develop other computations that use the features to derive higher abstractions of hand movement and shape.

implemented in a feature computation object *ellipsoid* which fits only an ellipsoid to the list of *positions* and computes within real-time. Currently our fitting algorithms do not take into account that the virtual object should be bounded by the hands, so that the virtual object intersects with the hands. Thus, we need to add more constraints. We are further studying 2-D and 3-D Fourier transforms for use as shape features [1]. Figure 4 shows a typical example Max project that allows a user to control FM synthesis parameters using hand movement primitives.

6 Summary and Conclusions

We have initiated the development of objects for the computation of human movement primitives within Max/FTS, believing that it will lead to gestural interfaces that will reduce the cognitive load for simultaneous multidimensional control tasks such as in sound design. Focussing on empty-handed gestures for a shape manipulation task called sound sculpting, where shape features are mapped to timbral parameters, we have programmed Max/FTS objects for interfacing serial peripheral devices such as sensors, objects for geometric and kinematic computation and objects for the computation of features of human body parts. Further work is necessary to define shape operators that modify the virtual object shape and shape features that can be mapped to timbral parameters. We hope to show through experimentation that shape features can be meaningfully mapped to sound parameters.

References

- [1] Ch. Brechbuhler, G. Gerig, and O. Kuhler, "Parametrization of closed surfaces for 3-D shape description," *Computer Vision and Image Understanding*, Vol. 61, No. 2, March, pp. 154-170, 1995.
- [2] S. S. Fels, G. E. Hinton, "Glove-Talk: a neural network interface between a data-glove and a speech synthesizer," *IEEE Transactions on Neural Networks*, Vol. 4, No. 1, pp. 2-8, 1993.
- [3] A. G. E. Mulder, "Getting a GRIP on alternate controllers: Addressing the variability of gestural expression in musical instrument design," *Leonardo Music Journal*, Vol. 6, pp. 33-40, 1996.
- [4] A. G. E. Mulder, "Human Movement Tracking Technology," *Technical Report, NSERC Hand Centered Studies of Human Movement project*, Burnaby, BC, Canada: Simon Fraser University, 1994. Available through the WWW at <http://fas.sfu.ca/cs/people/ResearchStaff/-amulder/personal/vmi/HMTT.pub.html>
- [5] Miller Puckette, "FTS: A real time monitor for multiprocessor music synthesis," *Computer music journal*, Vol. 15, No. 3, pp. 58-67, 1991.
- [6] Franc Solina and Ruzena Bajcsy, "Recovery of parametric models from range images: the case for superquadrics with global deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 2, February, pp. 131-147, 1990.

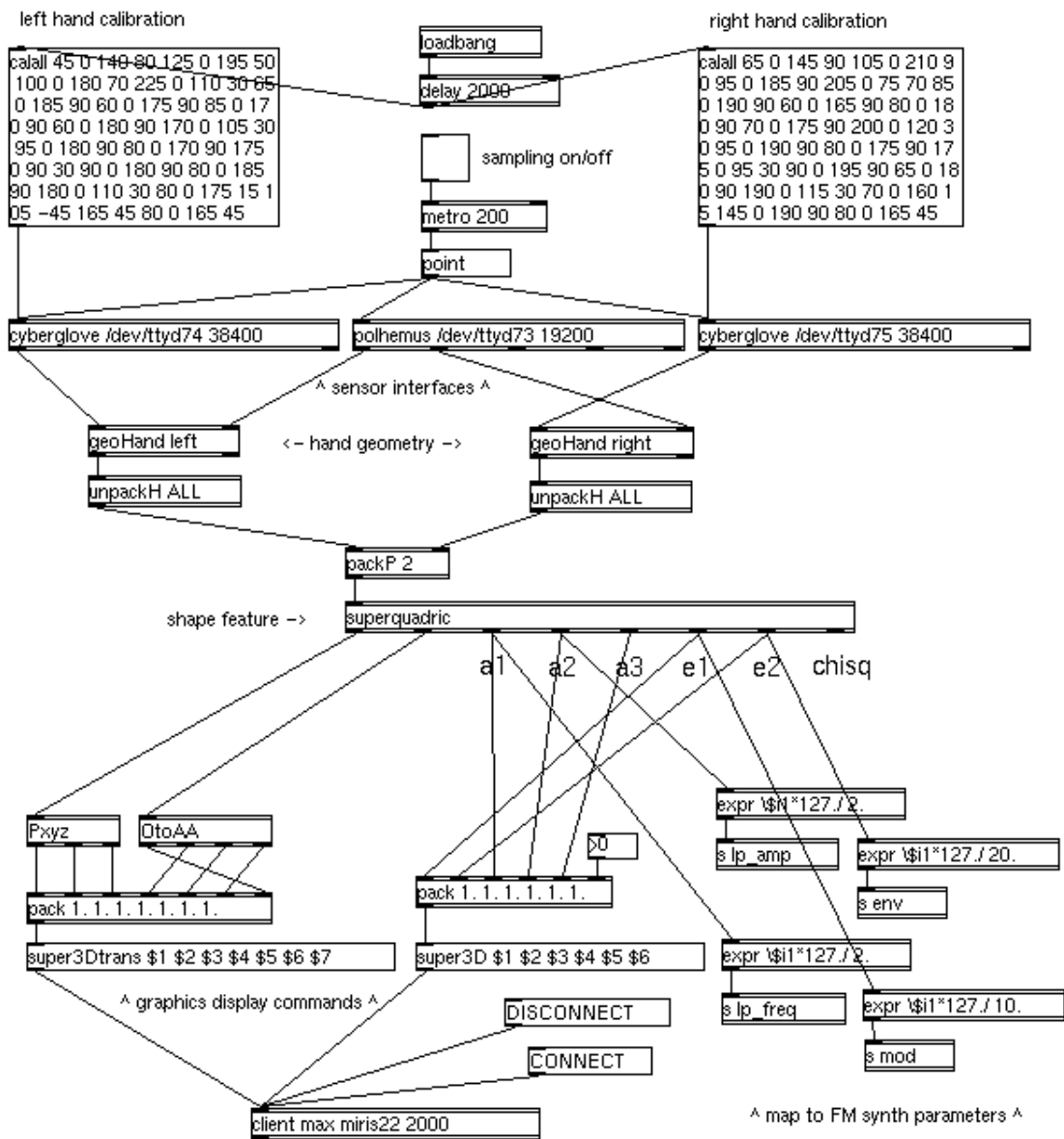


Figure 4: Typical example of a Max project using the new Max/FTS objects to compute a fit of the superquadric shape parameters to the list of *positions* as computed by the *geoHand* objects. The superquadric shape parameters or features are subsequently sent as messages to the client object which sends them to the OpenInventor 3-D graphics server. The parameters are also mapped to FM synthesis parameters like modulation and envelope and sent to another Max project for good old FM sound synthesis. With reference to figure 3, the character device driver is not visible because it is a subroutine called by the protocol handlers *cyberglove* and *polhemus*. Also, in this example we have set the hand feature computation, shape parameter translator and shape processor to the identity operator so that the list of *positions* generated the *geoHand* objects are taken as the new surface data points as input for the superquadric shape feature computation. The sound parameter translator and processor are represented as another Max/FTS project, not included in the figure.