“Namuda”

gesture recognition for musical practice

Dr. Godfried-Willem Raes

post-doctoral researcher

Ghent University College, School of Arts & Logos Foundation

2011

Logos Foundation

Kongostraat 35

9000 Gent

Belgium

gwr@logosfoundation.org

Abstract

This paper documents our use of sonar and radar-based systems that use the Doppler effect to record and analyse a performer's gestures in three dimensions. Its 3D gesture recognition hardware is a combination of radar and sonar technology. At the software level, it classifies a performer's gesture as one of 12 gestural prototypes or as non-motion. By means of études for each gesture, the system can be fine-tuned to an individual performer's gestural vocabulary and the performer can become proficient in eliciting a specific response to a given gesture. Hence a performer's movements can be mapped to audio data and used to produce music generated by the entire body. In our artistic implementation, we use the gesture recognition system to steer our robot orchestra, consisting of 48 electromechanical instruments.

Keywords: gesture recognition gesture-controlled music dance robot radar sonar Doppler

Introduction

In this paper, we discuss our use of sonar and radar-based systems that use the Doppler effect to record and analyse a performer's gestures in three dimensions. We have implemented it in the form of our Namuda project at the Logos Foundation in Gent, Belgium; 'Namuda' being a word of our own invention, derived from 'naked music dance.' We felt the need for a new word to describe the very dance-related method of music-making entailed by the development of our invisible instrument.

Maybe one of the most fundamental insights into music gained in the anti-authoritarian late sixties was the realisation that our 'serious' or 'classical' music culture is predominantly historical. More than 94% of all 'serious' music concerts are devoted to the mere reproduction of long-existing and already very well-known music: a repertoire ranging approximately from the late 17th century up to the remainders of late-romanticism at the beginning of the 20th century, if that. [1]

A problematic side effect of this situation is that the introduction of new musical instruments has been excluded from in standard music practice. In fact, our music-culture has failed to create any substantially new instrument since the 19th century.

A musical instrument – generally and philosophically speaking – is an artefact made and designed to extend human expressive possibilities using non-semantic sounds beyond those offered by the body and voice alone. Instruments are tools people use as extensions of their physical possibilities. However tools are not 'general': they are always constructed to make some precise and well-defined class of actions possible. They are characterized by a certain degree of optimization to their task in conjunction with their user. In musicology it can be demonstrated that an instrument such as a violin, is constructed and optimized for mostly tonal music with a strong emphasis on melody-line. Thus the instrument, seen in conjunction with its instruction, 'contains' an image of the music it is made for. For this reason it is quite evident to perform historic music using the tools delivered to us through that very same history. New instruments only have a chance of being accepted in reproductive music performance and practice, if they can produce the same commensurable acoustic result on the one hand and allow more efficient acquisition of the necessary craftsmanship and handling on the other.

The problem is that the general and almost exclusive acceptance of historical instruments in our contemporary music practice limits and compromises our contemporary possibilities for musical expression. The tools we seem to be forced to use do not facilitate new music. When it comes to realizing novel acoustic actualizations and images, players experience their instruments as obstinate and unwilling. As a consequence, new music in general appears as technically 'hard to play' and hence – already for this reason alone – many musicians do not feel very comfortable playing it.

Our personal research and artistic activity over the last twenty-five years has been deeply marked by the issues of musician and instrument. Thus we have experimented widely in the most diverse areas of instrument building and explored the composition of works conceived for newly designed musical tools. In particular, the relationship between the motor gestures of the human body and the instrument it handles, has been one of our prime concerns.

All musical instruments map human motor input onto an acoustic output. In traditional musical instruments, the input-output mapping fully characterizes the instrument and cannot be changed. Since the advent of electronics, it has theoretically become possible to map the input – for instruments with an electrically mediated input, such as most modern keyboard instruments – to any sound-producing device that can be controlled electrically in its turn. In terms of generating input, the use of sonar and radar technologies make it possible to design an absolutely non-contact interface to capture motor, and thus gestural, information. Other than with capacitive and optical systems, these technologies are inherently non-positional and, even more important here, capable of delivering dynamic information straight away.

Such technology allows anyone to participate in music production with full freedom to decide how to balance craftsmanship with the will to interpret or to express. It is theoretically possible to leave all traditional musical craftsmanship to the computer (reading the right notes, playing them at the right moment with the correct pitch and dynamics), allowing the performer to concentrate solely on the overall interpretation of the musical data. Since this type of instrumental user interface is fully programmable, it may open possibilities for the development of a musical culture that is more oriented to the expressive and creative development of individuals and small collectives, which could thus free us from the limiting conditions of commensurability. By adapting the software and mappings, a specific instrument can be tailored to every individual, perfectly in balance with his or her possibilities in terms of gesture control. Gesture is, as I could verify experimentally, just about as different from the one person to another as our fingerprints or handwriting. Furthermore, given the open architecture of this device, it is in principle open to the most varied levels of active participation in music culture. It is not by necessity something you have to be a 'professional' at, since the criteria for measuring this – commensurability – are removed as soon as one begins to program the instrument for a particular player.

This paper is in three sections. The first describes the circuitry developed to create this gesture measurement system; in other words, its hardware aspect. Section Two deals with the software we have developed to process the gestural data, and Section Three delves further into the expressive meaning of gesture and artistic aspects of the Namuda dance project developed at the Logos Foundation.

Main text

Section 1

<Holosound ii2010>: a Doppler sonar and radar-based 3D gesture measurement system

This section provides technical documentation of the circuitry developed in the creation of our 3D gesture recognition system. Our first system capable of recognising gesture was 'Holosound,' created in 1978 and updated in 2000 [2]. Since then, the system has been constantly updated and improved: hence the title, Holosound 2010.

The main modifications to previous designs are as follows:

> 1 the transducers in use here are omnidirectional wide-band sensors operating in the ultrasound range.
>
> 2- the new circuitry supports FM modulation of the base frequency, thus allowing distance measurement as well as a variety of analogue audio applications.
>
> 3- the circuitry can be used as such for audio installation projects and music theatre compositions, but for gesture analysis, the outputs should by sampled and further analysed by a computer. Unlike our previous designs, no special and expensive hardware is required here. Earlier systems invariably made use of National Instruments data acquisition hardware. This version requires up to six audio input channels on the PC, although the system can even be used with a common stereo input. In that case it is restricted to two dimensions.
>
> 4- The signal/noise ratio has been improved by approx. 12 dB over earlier designs.
>
> 5- We have introduced the combined application of radar and sonar technology in order

to achieve full gesture imaging, including positional information.

Receiver circuits for sonar (three are used in a tetrahedral set-up):

Although the highest possible Doppler frequencies for human bodies in movement using a 40 kHz carrier system are limited to less than 2.5 kHz [3], we have designed these receivers with an output bandwidth up to 16 kHz. This was done to allow frequency modulation schemes as well as the capture of ultrasonic components in audio input in general. This became the sensor system of choice, after many circuits using the different transducers available on the market. Figure 1 shows a circuit using a flexible transducer made by Pro Wave Electronics, type number 400FS080 (shown in Figures 2 and 3). We mention it, despite the lower signal noise ratio it offers, because it has interesting directional characteristics: 360 degree omnidirectional in the horizontal plane and +/-40 degrees vertical. It can be imagined as a doughnut shape, or in more scientific terms, a toroid. The bandwidth is limited to 4 kHz, centred around 40 kHz, wide enough to meet the requirements for FM modulation in distance estimation applications. [4]

Figure 1: Ultrasound receiver for electret sensor

Figure 2: Kynar 3 transducer (left) mounted on circuit board

Figure 3: Kynar 3 transducers made by Pro Wave Electronics

This type of transducer is also available with a centre frequency of 77 kHz (type 800FS049). The advantage of this higher frequency is that the Doppler signals are about an octave higher and thus frequency distribution calculation can be performed at approximately twice the speed. However, there are difficulties in building emitters with high enough sound pressure levels. We also have tried the Monacor electret capsule, type MCE2500, since its frequency response extends far into the ultrasonic range. A suitable circuit is given in the notes [5]. The obtainable S/N ratio is not as good as the SPM0204UD5 circuit, but it is considerably cheaper.

The sensor of choice used in our latest design, the SPM0204UD5, is an SMD MEMS component produced by Knowles Acoustics. Essential parameters and values are given in the circuit drawing. Note that the dot-driver is not a VU-meter but a tool to be used for alignment. It reflects the signal strength of the carrier wave. Reception will be best when any of the centre yellow LEDs are turned on. If the red LEDs come on, the preamp is saturated and the signal clipped. When only the green LED lights up, the signal strength of the carrier wave is too low.

Figure 4: SPM0204UD5 sensor in half-opened housing

Figure 4 shows the X and Y receivers, fitted into their half-opened housing. Shielding of the circuit, although not shown in the picture, is very important, since the circuit is very sensitive to electromagnetic disturbances in its vicinity, as commonly encountered with the proliferation of switch mode power supplies and motor controllers.

Figure 5: analogue processor board

The outputs of this analogue computer can be sampled by normal audio cards. For full 3D gesture rendering, 3 channels of audio are required. To allow calculation of distances – requiring FM modulation of the carrier – one should use the fourth analogue channel to sample the modulation signal [6]. The circuit, three channels according to the circuit above, fits nicely on a Eurocard board (100x160 mm) (Figure 6).

Figure 6: 100x160 mm Eurocard board with processor circuit

The limiter circuit with the two back-to-back diodes, is an old classic that will look familiar to anyone who has ever worked with short wave radio circuitry. It is a very non-linear circuit giving good overload protection from large spikes and overloads. We provided for this because we wanted to use the circuit in combination with arbitrary sound sources extending into the ultrasonic range: bats, small bells, key-rattle, breaking glass, gas leaks etc. In a measurement system, this part of the circuit is inappropriate. It distorts amplitude measurement and thus body surface estimation. But if signal levels are kept below the forward voltage of the diodes, the circuit can be used very well for reliable amplitude calculations. To make the adjustment of the input levels easier, we have made a small analogue readout (Figure 7).

Figure 7: Carrier signals analogue readout

Three of these circuits are required and placed at the top of the analogue computing board. [7]

Demodulation takes place in a synchronous multiplier circuit. The advantage is that it makes an intrinsic high pass filter, cancelling out most Doppler frequencies related to extremely slow and involuntary movement. In the circuit used for <ii2000>, we demodulated against an ultra-stable carrier frequency reference and had to cope with very large, slowly moving DC offsets that needed to be removed in the processing software. The synchronous demodulator has a fixed phase angle relation between both of its inputs. Even sharper synchronous carrier separation can be obtained using 40 kHz crystals instead of the resonant LC circuit used here. We did not use crystals in this design however, because of the trouble we had with loss of carrier signal in frequency modulation schemes required for distance estimation.

A deficiency of the circuit presented is that the demodulation frequency is not easily tunable and with the given component values, it centres around 40 kHz, the most commonly used frequency in ultrasonic applications. It would be a benefit to be able to tune to arbitrary carrier frequencies in the range 20 kHz to 200 kHz.

An easy way to make the demodulation frequency independent is to feed to multipliers with the signals of a pair of transducers, i.e. xy, yx and zx. The output signals will now be the product of both inputs, and although the demodulated signals reflect the gesture very well, the maths involved to bring the data back to real world units (body surface and movement speed) becomes very complicated. (Raes, 1993) A proposal – using pure analogue circuitry – to solve this problem is given in this block diagram:

Figure 8: Carrier-independent Doppler demodulation circuit for invisible instrument

The multipliers can be either AD633 type (low cost analogue devices), or MPY632, AD534 etc. (expensive but a lot more precise). The dividers can be created using the same chips in the appropriate divider configuration.

This idea can also be applied to individual receiver circuits, as we found out. In order to achieve this, each receiver is equipped with two sensors placed closely together. After pre-amplification, both signals are multiplied in an analogue multiplier, functioning here as a phase comparator. A fifth order low pass follows and some further amplification in order to get a sufficient signal level for sampling. This is the circuit as we tested it out:

Figure 9: Phase demodulator receiver circuit

Performance of the circuit as a gesture-capturing device is good, but with the given components, the noise level (S/N ratio worst case, using 40TM transducers, 42 dB) is inferior to previous designs. This is mainly due to the 1 mV noise found on the output of the (cheap) multiplier. Another issue with this circuit is that it is very sensitive to large DC offset variations on the multiplier output in reaction to amplitude changes of the input signal. This could only be remedied by applying a phase shifter between both inputs as suggested in the application notes for the multiplier chip by Analog Devices. Unfortunately phase shifters only work well if the carrier frequency is constant and thus that solution would rule out the neat feature of this circuit, its carrier frequency independence. However, a phase shifter can also be realised mechanically, as follows.

When we consider only the sinusoidal ultrasonic carrier signal component, leaving the Doppler shifted components out of consideration, the multiplier operates as a squarer and thus, in the frequency domain, as a frequency doubler since we then have:

Equation 1

This transfer equation shows a very high DC offset depending on input amplitude. To get rid of this, we can create a phase shift between the X and Y inputs of 90 degrees, and applying the textbook equation,

Equation 2

we can obtain:

Equation 3

Since both inputs in our application use individual sensors, this makes it possible to realize the phase shift without resorting to electronic networks. It is sufficient to place one of the transducers a quarter wavelength further back from the other. For a 40 kHz carrier, this means 2.125 mm, since a full wavelength corresponds to 8.5 mm.

By way of an aside: the circuit works very well for bat detection without any need for tuning, if used with the SPM0204 transducers.. If the output is to be sampled and the sampling rate in use is lower than 32kS/s, an extra low pass filter on the output becomes mandatory.

Although not discussed in full depth in this paper [8], we have been using this board in combination with our 24 GHz microwave radar Doppler receivers. The radar board is piggy-back mounted on the ultrasound receiver board with the sensor aligned with the ultrasound sensors. This combined sensor offers the possibility to deduce the absolute distance to the moving body part, by measuring the time difference between both demodulated Doppler signals. Since ultrasound travels at the velocity of sound (340 m/s) and radar at the velocity of light, the time between the reception of the two signals allows us to deduce distance: s = 340.t Of course the resolution will be a function of the sampling rate. If the sampling rate is the common 44.1 kS/s, resolution will be 7.6 mm. Here is the full circuit for the combined sensor as we made it:

Figure 10: Combined radar/sonar receiver for gesture imaging

The principle behind the distance measurement technique made possible by this circuit is shown in the graph below, showing the signals for a single small gesture:

Figure 11: radar and sonar signals for a small gesture

It is generally not possible to find the start of the gesture signal reliably, and hence it is better to look for a local maximum in the data. Of course latency is introduced this way. Correlation of the two waveforms is a better alternative, in the sense that it gives more reliable distance data, but it is mathematically more intensive and may lead to even more latency. For non real-time applications this is obviously not an issue.

This circuit was built in 3 copies to serve as X, Y and Z receivers. The breadboarded circuits for the upper part (sonar) came out looking like this:

Figure 12: breadboarded circuits for sonar receivers

The signal noise ratio measured with this circuit – properly enclosed – came to 60dB. Further improvements are still conceivable and possible: by using B&K ultrasonic measurement microphones in combination with the best available multipliers on the market and by ruling out all offset errors, an improvement of approx. 12dB can be anticipated. The price, however, would also rise by some 40dB... The S/N ratio obtained with the radar section is at least 20dB poorer than that of the sonar section. Hence data analysis should be based on the sonar signal leaving the radar signal for redundancy checks and distance determination only. Spectrum measurements with this system, operated with a 40kHz carrier, revealed that for normal human gestures, no gesture-related frequency components higher than 4kHz are traceable in the sonar signal. A low pass filter with a cut off at 4kHz would have been enough, but can be implemented in software quite easily if the sampling rate is high enough. [9]

In order to test this hardware gesture recognition platform with both sonar and radar sensors, we have made a board to make interfacing to a computer based audio input device an easy matter (6 channels of audio line level input are required for a complete 3-D gesture rendering). The circuit looks like this:

Figure 13: combined radar/sonar circuit for interfacing to computer-based audio input device

The 7456 divider chip used here is an obsolete TTL part produced by Texas Instruments, no longer in production. Since we had still a good quantity in stock, we used it nevertheless. Software for gesture control of the robot orchestra using this hardware platform has been developed by Kristof Lauwers at the Logos Foundation using PD; sample code is available on request.

The test board looks like this:

Figure 14: test board for radar/sonar interface

Emitter board:

This small board receives the carrier signal – either fixed frequency or an FM modulated signal – from the analogue computer board described above. The design goal was to achieve a sound pressure level of approx. 120 dB (measured at 1 m distance) with wide area coverage. [10] Basically the circuit is a bridge amplifier delivering an output voltage swing almost twice as large as the power supply voltage. The LF356 was selected for its excellent behaviour in driving capacitive loads.

Figure 15: emitter board circuit

The practical realization looks like this:

Figure 16: emitter board

The input potentiometer can be used to adjust the output level such that an optimum S/N ratio is achieved. It also allows adjustment to variable distances in the set-up. In fact, the placement of the emitter does not have to be strictly on the vertex of the ground plane of the imaginary tetrahedron, but can be placed much farther away. The distance to the X,Y and Z receivers however has to be the same. Although not drawn in the circuit drawing, our practical realization of the circuit uses an array of four 400ET080 transmitters, connected in parallel and aligned vertically. The beam angle for these transducers is specified as 125 degrees. The vertical stack alignment was done in an attempt to meet the sound pressure levels that we obtained in using the Murata piezoelectric transducers in earlier designs. These 16mm Murata piezoelectric transducers, however, are quite unidirectional and do not allow frequency modulation due to their inherent sharp resonance at 40 kHz. Sound pressure measurement with this circuit and the 400ET080 transducers revealed that we could not obtain the required 120 dB. Hence we also made a similar circuit, this time using two 10 mm diameter 400ST100 ceramic transducers mounted in a horizontal plane. The modulation bandwidth with these transducers is limited to 2.5 kHz, but 120 dB sound level pressure could be reached. However the voltage over these transducers should be kept below 15 V_{rms}. The capacitance of the transducers is 1900 pF and the beam angle limited to 72 degrees.

The required sound pressure level could be reached with the circuit below. Here we used an array of four Murata MA40S4S emitters, specified for 120dB SPL at 30cm. The beam angle for each transducer is specified as 80 degrees, thus in order to have a somewhat wider coverage, we arranged the array of four transducers in a square on an imaginary sphere segment. Each transducer has a capacitance of 2.55nF and hence the total capacitance (10.2nF), if we connect them all in parallel, would marginally exceed the capacitive drive specification for the LF356 opamps. We tried this, and in fact observed a very odd phenomenon in that the lower opamp undergoes a complete phase reversal as soon as the input signal exceeds a certain limit, yielding almost no output from the circuit. Hence we used the series/parallel connection as shown in the circuit below. Since we had plenty of headroom – the transducers should not be driven with voltages higher than 20Vpp – we suffered no penalty from this.

Figure 17: alternative emitter board circuit

Although we now meet the sound pressure level requirements, this circuit does not lend itself very well to FM-modulation due to the small bandwidth of the transducers.

Other experiments carried out have been:

1.- Extended range tweeter loudspeakers: highly inefficient, very bulky and... way too directional. Imagine: we needed a 150 W amplifier to feed a 200 W titanium dome tweeter in order to get 120 dB sound pressure at 40 kHz at 1 meter distance.

2.- Capacitive transducers requiring a DC bias voltage in the order of 300 V. Their principle of operation is identical to that of electrostatic loudspeakers. These work pretty well and they are indeed suitable for FM applications, but due to their large diaphragms, they are also way too directional (12 degrees) for our application here. The type we tested is 500ES430, although specified for 50 kHz, they also operate well at 40 kHz. A particular problem is raised by the required +300 V bias: if this voltage is applied through the 5 conductor cable, the latter becomes highly microphonic itself. Generating this voltage with a small switcher on board, on the other hand, causes a lot of interference with the ultrasound carrier. A cleverer circuit, in which the DC voltage is obtained through diode multiplication starting from the carrier signal itself, should be designed.

3.- Plasma ion sources are a perfect match on the design table: They are by nature omnidirectional and can easily cope with FM. However, they are highly sensitive to air flow – they sputter – and they cause extreme EMC in the environment. The fact that they are intrinsically dangerous as they operate with 20 kV voltage levels, also limits their usability. [11]

The ideal transmitter has not been found so far. The importance of the search for very powerful transmitters lies in the simple fact that the signal to noise ratio at the receivers end, tracks perfectly with the sound pressure level of the transmitters. Research goes on.

A final note on interference.

In general, sonar technology is far less sensitive to environmental interference, although care must be taken to avoid turbulences and wind flow as well as temperature gradients. However, another source of interference is formed by the extended spectral contents of quite a few sounds of real musical instruments. This interference does not occur with loudspeaker-generated environmental music, since audio systems filter out all frequency components above 20 kHz by their very nature. Since we use this technology in the implementations of our invisible instrument to control our robot orchestra, composed of purely acoustical sound sources, we have often noticed disturbances with certain loud sounds such as those that originate from our robot saxophone <Autosax>, struck metal shells in <Llor> and edge whistle tones generated by organ pipes in our <Qt>, <Bourdonola> and <Piperola> robot. The problem can be avoided only by placing the sensor system far enough from the acoustical sound sources. Unfortunately, on our premises at the Logos Foundation this is impossible because of lack of space...

Section 2

Namuda Gesture Recognition on the Holosound ii2010 platform

This chapter describes the software layer for gesture recognition using Doppler-based hardware systems.

The holosound hardware is connected to the PC for sampling and further analysis, treated in this chapter. It is very important to stress that all the data acquisition and classification procedures covered here were designed to work concurrently and in real time. The latency was to be better than 10ms. This restriction accounts for the fact that we did not optimize the procedures for maximisation of precision nor for maximisation of feature extraction. It will be obvious that taking this restriction away, and performing the analysis off-line, can greatly improve the functionality of the gesture sensing technology for analytical and scientific purposes.

1. Data Acquisition

Since the low pass filtering is performed in the hardware, we can set up a three-channel sampler with a sampling rate of 1024 S/s. Using a National Instruments USB data acquisition device (type: NI-USB 6210 or NI-USB 6212) and calling in the drivers and libraries offered by NiDAQmx, this is pretty easy. The resolution is 16 bits and the voltage sensitivity for the inputs is programmed for -10 V to +10 V. There is plenty of headroom, so we should not fear clipping, since the signals from our hardware stay well within the range of 4 V_{pp} under normal circumstances. The data is read in a callback function operating at a rate of 256 times a second. So on each call, 3 x 4 = 12 new samples are read. Reading the data at the sampling rate itself is impossible due to limitations in the USB driver implementation provided by National Instruments, despite the fact that the devices themselves can handle sampling rates well above 250 kS/s.

These samples are written to the fast circular data buffers, 256 entries in size, one for each of the three vectors. Thus the time window available here is 250 ms and the sampling rate remains at 1024S/s. A medium size time window is set up in the same callback for which we down-sample to 256 S/s and write the result to the medium data buffers. Thus these circular buffers cover a time window of 1 second, or, 256 samples. The down-sampling applies a simple low pass filter to the data. A long time window, covering a 4-second timespan, is then implemented by down-sampling again to 64 S/s. These circular buffers again contain 256 samples. Pointers to these nine circular buffers are made available to the user through a specific structure. All buffer data is in double precision format and normalized, bipolar -1 to +1.

Further preprocessing performed from within the same callback consists of:

A. setting up a running integrator with rectification in order to calculate the signal strength proportional to the size of the moving and reflecting body. These values are returned in the normalized .xa(), .ya(), .za() fields of the structure. The integration depth can be controller with the parameter in the .dta field.

The algorithm here also implements a high pass filter to eliminate slow DC offsets and artefacts originating from very slow and mostly involuntary movements of the body.

To summarise, body surface derivation uses:

```
high pass filter
rectifier
low pass filter
integrator
```

B. Calculation of the frequencies in the Doppler signals in order to grasp information on the speed of movement. Here we encounter quite a few fundamental problems: first of all, the signals are in no way periodic but in fact bandwidth limited noise bands. Second, the frequency limits of these Doppler signals depend on the angle of the movement: the cosine in the Doppler formula $f_d = 2\ v\ f_o \cos(a) / c$. Only when movements are performed towards or away from a particular vector in line with the transducer, will the cosine be unity and the absolute value of the bandwidth trustworthy. If the performer knows this, he/she can obviously take it into account. For general gesture analysis though, ideally the coordinates of the movement in space have to be known. As we have proved in our papers and notes on hardware for gesture recognition, this can be solved either by combining sonar and radar systems, or by FM modulating the carrier wave. This adds a lot of complexity to the software and we will not delve deeper into the related problems at this point.

However, even without cosine correction and lack of positional information, we can reduce induced errors greatly by making use of the redundancy offered by the fact that we have three vectors of data available. Placing the sensors on the vertices of an imaginary tetrahedron means that the maximum value for detected speed in the three vectors together can never be more than 50% off, the cosine of 60 degrees being equal to 0.5.

The algorithm performed in the callback function first converts the data in the fast buffer to a square wave with a Schmitt trigger and hysteresis to avoid noise interference. Then the zero crosses of the signal are counted, disregarding any periodicity. Thus the results, returned in the .xf(), .yf(), .zf() fields of our structure, reflect noise densities of the signal proportional to movement speed. The numeric values returned are not normalized but reflect real world values expressed in Hz, but have to be interpreted carefully within the constraints as explained. If normalization is required, the values can be simply divided by 200, a realistic maximum value as determined empirically.

C. Calculation of vectorial acceleration. This is done by differentiating the vectorial frequency information. The delta-t value can be user controlled via the .dtacc parameter. Larger values lead to better resolution, to the detriment of responsiveness. The results are returned in .xac(), .yac(), .zac(), with the scaling being independent from the setting for .dtacc.

This callback function updates the parameter fields in the Doppler structure 256 times a second. This is the first level of processing of the movement data. It yields information on the moving body surface as well as a first approximation of speed.

Performing FFTs on the data buffers from within the same callback is impossible, even on a quad core PC. Therefore the spectral analysis of the data must be performed as a different thread. The shape of the spectrum (the distribution of the frequency bands) reflects the characteristics of the gestures very well. This can be demonstrated convincingly by mapping the output of the transform on the keys of a piano, whereby the power of each frequency band is mapped on the velocity of the attack for each key.

The complete structure as declared in the software is:

```
TYPE DopplerType DWORD        ' for ii_20xx with NiDAQmx
      xa     AS SINGLE        ' reflection amplitudes for the x-vector
      ya     AS SINGLE        ' idem for y-vector
      za     AS SINGLE        ' idem for z-vector
      xf     AS SINGLE        ' doppler frequency shift density for the x-vector
      yf     AS SINGLE
      zf     AS SINGLE
      xac    AS SINGLE        ' acceleration for the x-vector
      yac    AS SINGLE
      zac    AS SINGLE
      pxfast AS DOUBLE PTR    ' pointer to the x-vector data(0). Most recent
                              data are in data(255) 250 ms buffer  1024 S/s
      pyfast AS DOUBLE PTR
      pzfast AS DOUBLE PTR
```

```
    pxm    AS DOUBLE PTR    ' 1s buffer, sampling rate 256 S/s
    pym    AS DOUBLE PTR
    pzm    AS DOUBLE PTR
    pxslow AS DOUBLE PTR    ' 4s buffer, sampling rate 64 S/s
    pyslow AS DOUBLE PTR
    pzslow AS DOUBLE PTR
    pxfbuf AS SINGLE PTR    ' 64 values for the frequency measurement, used
                            for calculation of acceleration
    pyfbuf AS SINGLE PTR    ' sampling rate: 256 S/s
    pzfbuf AS SINGLE PTR    ' most recent data in data(63)
    dtacc  AS WORD          ' dt for acceleration derivation. Valid values: 1-
                            63. Do not exceed range!
    dta    AS WORD          ' amplitude integration depth  (0-255)
    noise  AS SINGLE        ' noise level threshold   (1E-3, default, for 60
                            dB signal noise ratio)
    hpf    AS WORD          ' high pass filter differentiation depth
END TYPE
```

2. Gesture recognition

Although the callback function described above already performs quite a lot of data processing and allows us to get elementary information on the movement of the body, it is not capable of performing gesture recognition. Hence the next steps. In our approach, rather than trying to recognize predefined gesture models, we attempt to maximize the number of gestural characteristics retrievable from the use of this particular hardware and its signals as used throughout thousands of experiments with more than 20 different human bodies. The subjects were either trained musicians or dancers operating with direct auditory feedback. We baptised these retrievable gestural characteristics as Namuda gesture prototypes. Each prototype is defined within a certain time frame (100 to 1000 ms) and has a calculated normalized property strength as well as a persistence value, expressed in time units (generally 1/256 second units, corresponding to the best possible resolution within the constraints of our sampling function).

Namuda gesture types:

So far we are able to distinguish twelve types of gesture (not counting the no-movement property, 'Freeze'). We certainly do not exclude the possibility of deriving a few more, but on the other hand we have found that these twelve micro-gestures might very well be already at the upper limit of what we can clearly control given the motor skills of our bodies. Many gestural characteristics can be seen as dipoles: they exclude each other. Dipoles we distinguish are:

1. Speeding up versus slowing down

speedup
definition: the movement speed goes up within a time frame of maximum 500 ms

slowdown
definition: the movement speed goes down within a time frame of maximum 500 ms

Algorithms for feature extraction:

Speedup/slowdown uses a gesture recognition algorithm based on a FIR filter (as does implode/explode, discussed below). The coefficients used are the elementary gesture type that we are trying to evaluate. In the first approach these coefficients are linear functions. Further refinement consists of applying different functions to calculate the coefficients: quart sine or cosine, exponentials or even beta functions. However, this should only be done on the condition that the pattern recognition triggers the property with linear coefficients. The reason being that we simply do not have enough computing power to handle each recognition algorithm with many different coefficient models in real time.

The procedure is implemented as a function to which the pointer to the frequency data array (covering 500 ms) is passed as well as the vector. Vectors are numbered 0,1,2 for the x, y, and z vertexes where the transducers are placed physically. Vector 3 always refers to a sum or a maximum value. [12]

```
SUB GestureProp_Speedup (BYREF ar() AS SINGLE, BYVAL vektor AS LONG)
    'time weighted moving average approach using a Nth order FIR filter
    'vector: 0,1,2  for x,y,z
    'speed data   - the coefficients are the shape of a speedup gesture (rising)

    LOCAL i AS DWORD
    LOCAL av AS SINGLE
    STATIC order AS LONG
    STATIC Fscale  AS SINGLE
    DIM oldF(0 TO 2) AS STATIC SINGLE

    ' speedup coefficients, these have only to be recalculated if the window
       size is changed with .Forder:
    IF order <> gesture.Forder THEN
        order = gesture.Forder
        REDIM bs(0 TO order) AS STATIC SINGLE
        RESET Fscale
        FOR i = 0 TO order
            bs(i) = i / order                 'rising series of coefficients:
                                                 this is the gesture model
        NEXT i
        Fscale = (order + 1) / 2              'for normalization
    END IF

    'FIR with time dependent linear coefficients:
    FOR i = (UBOUND(ar) - order) TO UBOUND(ar)
```

```
        av = av + (ar(i) * bs(i-UBOUND(ar) + order))
    NEXT i

    'now we calculate the matching value (fuzzy):
    IF av > oldF(vektor) THEN
          gesture.speedup(vektor) = (av - oldF(vektor)) * 2
          INCR gesture.speedup_dur(vektor)
    ELSE
          RESET gesture.speedup(vektor), gesture.speedup_dur(vektor)
    END IF
    gesture.speedup_val(vektor) = av/ Fscale 'this is the renormalised output
                                              value of the filter itself
    oldF(vektor) = av

    'reconsider the global property (vektor 3):
      Gesture.speedup(3) =  MAX(Gesture.speedup(0),
      Gesture.speedup(1),Gesture.speedup(2))
      Gesture.speedup_val(3) =  MAX(Gesture.speedup_val(0),
      Gesture.speedup_val(1),Gesture.speedup_val(2))

END SUB
```

Here is a plot of the result of this algorithm:

Figure 18: plotted result of speedup algorithm

The upper three graphs give the frequency signals for the x,y and z vectors as found in pDoppler.xf, yf, zf. The vertical scale is in Hz. The window shows a time frame of 4 seconds. The graphs below show the output of the recognition. The vertical scale shows property strength, again for the three vectors. The last graph shows the property for all three vectors together.

The procedure for slowdown recognition is identical, but the coefficients are a linearly descending series.

Here is a plot of the result of the slowdown recognition algorithm:

Figure 19: plotted result of slowdown algorithm

2.- Expanding versus shrinking (or explosion – implosion)

exploding

definition: the size of the moving body surface increases within a time frame of maximum 1000 ms

imploding

definition: the size of the moving body surface decreases within a time frame of maximum 1000 ms

For a shrinking/expanding body surface we use the same approach as for speeding up and slowing down, but found that the time window has to be made larger for good reliability. We have it set at 1 second. The FIR filter order can be changed by the user using the parameter in the .Sorder field.

Here is the procedure for shrinking gesture recognition with linear coefficients:

```
SUB Gestureprop_Shrink ((BYREF ar() AS SINGLE, BYVAL vektor AS LONG)
    ' pattern recognition code using FIR approach with the gesture model based
      in the coefficients
    ' since this works with our omnidirectional ii2010 receivers , no angle
      correction is required
    ' the amplitudes of the reflected signals are independent from the frequency
      of the doppler shift.

    LOCAL i,j AS DWORD
    LOCAL av, s AS SINGLE
```

```
    STATIC order AS LONG
    STATIC Sscale  AS SINGLE
    DIM oldS(2) AS STATIC SINGLE
    DIM bs(gesture.Sorder) AS STATIC SINGLE

    'calculation of a descending series of coefficients:
    IF order <> gesture.Sorder THEN
        order = gesture.Sorder
        REDIM bs(order) AS STATIC SINGLE
        RESET Sscale
        FOR i = 0 TO order
            bs(i) = 1 - (i/(order+1))       'this could also be the result of a
                                     beta-function describing the gesture type
            Sscale += bs(i)                 'sum of factors, required for
                                             normalisation.
        NEXT i
    END IF

    'FIR filter with time dependent linearly descending coefficients:
    RESET j
    FOR i = (UBOUND(ar) - order + 1) TO UBOUND(ar)
            av += (ar(i) * bs(j))
            INCR j
    NEXT i

    'now classify: shrinking surface
     IF (av < oldS(vektor)- (@pDoppler.noise/ SQR(order)))  AND (av/Sscale >
@pDoppler.noise) THEN
          IF gesture.implo_dur(vektor) THEN
                 gesture.implo(vektor) = MAX(MIN((oldS(vektor) - Av)* 2.5, 1),0)
                   'scaling factor for commensurability with the explo property
```

```
                    'observed range, before rescaling with * 2.5:
                    '19.04: x<= 0.156, y<=0.224, z<=0.08  [gwr with clothes]
                    '19.04: X<= 0.335, y<=0.373, z<=0.268 [gwr naked]
          ELSE
                  RESET gesture.implo(vektor)
                    'the property will only be set if it lasts at least 7.8ms
          END IF
          INCR gesture.implo_dur(vektor)             'in 1/256s units (3.9ms).
     ELSE
                  RESET gesture.implo(vektor), gesture.implo_dur(vektor)
     END IF
     gesture.implo_val(vektor) = av/ Sscale
                  'we always return this normalized value for research purposes.
     oldS(vektor) = av

    gesture.implo(3) =  (gesture.implo(0) * gesture.implo(1) * gesture.implo(2))
^0.33   'to be further investigated
    gesture.implo_val(3) =  (gesture.implo_val(0) * gesture.implo_val(1) *
gesture.implo_val(2)) ^0.33

END SUB
```

Here is a plot of the result of the implosion algorithm:

Figure 20: plotted result of implosion algorithm

In this case we plot the normalized amplitudes of the Doppler signals against time (4 seconds) in the upper three graphs. Below, again, is property strength.

For explosion we get similarly:

Figure 21: plotted result of explosion algorithm

3. Fluency (steadiness) and speed properties

Not all gesture properties are calculated with this or a similar FIR algorithm. For the properties fluency (constancy of moving body surface area within the time frame) and fixspeed (constancy of movement speed within the time frame), we found traditional statistics useful. Here we look in the data buffer and calculate the running average as well as the significance. The smaller the deviation from the mean, the stronger the property will be evaluated. Here is, as an example, our coding for fluency detection:

```
SUB Gestureprop_Fluent ((BYREF ar() AS SINGLE, BYVAL vektor AS LONG)
    LOCAL avg, d, s AS SINGLE
    LOCAL i AS DWORD
    STATIC maxval, maxprop AS SINGLE

         ' calculation of the average in the body surface data array:
        FOR i = 0 TO UBOUND(ar)
            Avg +=  ar(i)
        NEXT i
        Avg /= (UBOUND(ar) +1)        'normalised 0-1

        'calculate the standard deviation:
        FOR i = 0 TO UBOUND(ar)
            d += ((ar(i) - Avg)^ 2)      ' method of the smallest squares
        NEXT i                           ' d =  average fault for each sample
            s = SQR(d/(UBOUND(ar)-1))    ' s = standard deviation for a random
                                          population
```

```
        ' statistics math memo:
        ' 68% of all values in dta are between Avg - s and Avg + s
        ' 95%                      are between Avg -2s and Avg +2s
        ' 97%                      are between Avg -3s and Avg +3s

        IF Avg > @pDoppler.noise * 5  THEN
            IF gesture.flue_dur(vektor) THEN
                gesture.flue(vektor) = MAX(0, 1 - (5*s / Avg))
                                           'gives quite a good range
            ELSE
                RESET gesture.flue(vektor)
            END IF
            INCR gesture.flue_dur(vektor)
        ELSE
            RESET gesture.flue(vektor), gesture.flue_dur(vektor)
        END IF
        gesture.flue_val(vektor) = Avg
                                           ' always returned.

    Gesture.flue_val(3) = (Gesture.flue_val(0) * Gesture.flue_val(1) *
Gesture.flue_val(2)) ^ 0.33
                                           ' questionable
    gesture.flue(3) = (gesture.flue(0) * gesture.flue(1) * gesture.flue(2))
^ 0.33

END SUB
```

Here is a plot of the result of the fluency property algorithm:

Figure 22: plotted result of fluency algorithm

And here, again for the speed constancy property:

Figure 23: plotted result of speed constancy algorithm

4. Collision - Theatrical Collision

For the dipole collision/theatrical collision, we use yet another approach, based on an analysis of the acceleration data acquired in the sampling callback. The time window used in this analysis is 100 to 390 ms. The smaller values make the algorithm more responsive.

Definition of collision: acceleration rises up to the point of collision, where we have a sudden sign reversal.

Definition of theatrical collision: acceleration decreases until the point of collision - a standstill - , where we have a small sign reversal followed with a rise of acceleration.

Algorithm:
a first leaky integrator is applied to the first three quarters of the acceleration data (upval) and a second integrator on the last quarter of the data (downval). The sensitivity is set with the variable 'sens.'
If upval > sens and downval < - sens, we set the collision property.

The practical coding is as follows:

```
SUB Gestureprop_Collision (BYREF ar() AS SINGLE,  BYVAL vektor AS LONG)
       ' the array passed should be the amplitude array, we need it to look up the impact
         value on the moment of the collision
       ' we could also pass the value ar(230) alone.
       ' the algo uses a leaky integrator, not FIR!, on the acceleration data
```

```
  ' the procedure returns the gesture properties collision and theatrical
    collision. These are mutually exclusive.
STATIC tog AS LONG
STATIC xslope, yslope, zslope, xdown, ydown, zdown, sens AS SINGLE
IF ISFALSE tog THEN
  sens = 0.4                      ' this is a critical value to experiment with
  DIM xc(25) AS STATIC SINGLE     ' 101 ms buffers for acceleration and collision
       DIM yc(25) AS STATIC SINGLE ' empirically determined
       DIM zc(25) AS STATIC SINGLE
       tog = %True
END IF

SELECT CASE vektor
       CASE 0
         ARRAY DELETE xc(), @pDoppler.xac ' circular buffer with acceleration
                                          data. Most recent data now in xc(last)
         'calculate the shape:
         xslope = (xslope + xc(0)) / 2    ' low pass filter - should cut off around
                                            25Hz
         xdown = (xdown + xc(25)) /2
         IF (xslope > sens) AND (xdown < -sens) THEN
                                          ' this is the pattern recognition condition
               Gesture.collision(0) = xslope - xdown
                                        ' always positive, since xdown is negative !
               INCR Gesture.collision_dur(0)
               Gesture.impact(0) =  ar(230)
                                          ' further research required for optimum
                                            value in the array
               RESET Gesture.theacol(0), Gesture.theacol_dur(0)
         ELSEIF (xslope < -sens) AND (xdown > sens) THEN
                                          ' condition for theatrical collision
               Gesture.theacol(0) = xdown - xslope
                                          ' always positive, since xslope is negative
               INCR Gesture.theacol_dur(0)
```

```
            RESET Gesture.collision(0),  Gesture.collision_dur(0)
            Gesture.impact(0) =  ar(230)
      ELSE
            RESET Gesture.collision(0), Gesture.theacol(0), Gesture.impact(0)
            RESET Gesture.collision_dur(0), Gesture.theacol_dur(0)
      END IF
 CASE 1
      ARRAY DELETE yc(), @pDoppler.yac
                  ' the acceleration values are updated 256 times a second
      yslope = (yslope + yc(0)) / 2
      ydown = (ydown + yc(25)) / 2
      IF (yslope > sens) AND (ydown < -sens) THEN
            Gesture.collision(1) = yslope - ydown
            INCR Gesture.collision_dur(1)
            Gesture.impact(1) = ar(230)
            RESET Gesture.theacol(1), Gesture.theacol_dur(1)
      ELSEIF (yslope < -sens) AND (ydown > sens) THEN
            Gesture.theacol(1) = ydown - yslope
            Gesture.impact(1) =  ar(230)
            INCR Gesture.theacol_dur(1)
            RESET Gesture.collision(1),  Gesture.collision_dur(1)
      ELSE
            RESET Gesture.collision(1), Gesture.theacol(1), Gesture.impact(1)
            RESET Gesture.collision_dur(1), Gesture.theacol_dur(1)
      END IF
CASE 2
      ARRAY DELETE zc(), @pDoppler.zac
      zslope = (zslope + zc(0)) / 2
      zdown = (zdown + zc(25)) / 2
      IF (zslope > sens) AND (zdown < -sens) THEN
            Gesture.collision(2) = zslope - zdown
            Gesture.impact(2) = ar(230)
            INCR Gesture.collision_dur(2)
            RESET Gesture.theacol(2), Gesture.theacol_dur(2)
```

```
            ELSEIF (zslope < -sens) AND (zdown > sens) THEN
                  Gesture.theacol(2) = zdown - zslope
                  Gesture.impact(2) =  ar(230)
                  INCR Gesture.theacol_dur(1)
                  RESET Gesture.collision(2),  Gesture.collision_dur(2)
            ELSE
                  RESET Gesture.collision(2), Gesture.theacol(2), Gesture.impact(2)
                  RESET Gesture.collision_dur(2), Gesture.theacol_dur(2)
            END IF
    END SELECT

      Gesture.collision(3) = MAX(Gesture.collision(0), Gesture.collision(1),
      Gesture.collision(2))
      Gesture.theacol(3) = MAX(Gesture.theacol(0), Gesture.theacol(1),
      Gesture.theacol(2))
      IF Gesture.collision(3) THEN
       Gesture.impact(3) = MAX(Gesture.impact(0), Gesture.impact(1), Gesture.impact(2))
      ELSEIF Gesture.theacol(3) THEN
       Gesture.impact(3) = MAX(Gesture.impact(0), Gesture.impact(1), Gesture.impact(2))
      ELSE
       RESET Gesture.impact(3)
      END IF

END SUB
```

Here is a plot of the result of the collision detection algorithm:

Figure 24: plotted result of collision detected algorithm

In this case the upper graphs show the vectorial acceleration channels. Theatrical collision detection resulted in the graphs below:

Figure 25: plotted result of theatrical collision algorithm

5. Roundness – Edginess

The spectral analysis of the Doppler signals in the short data buffer allow us to discriminate between gesture properties situated on a dipole that can be described as 'roundness' versus edginess. Other ways to describe the gesture characteristic referred to could be smooth as opposed to shaky or jagged.

Definition: roundness occurs if the gestures are smooth and continuous; edginess, if the gestures contain many abrupt components and discontinuities. The property reflects aspects of the shape of a gesture very well.

The analysis is based on the spectral power distribution in the fast data buffer (250ms). The code as we wrote it at first for the Fourier transforms is:

```
SUB DFT_Dbl (Samp#(), Sp#()) EXPORT

    'discrete fourier transform on double precision data
    LOCAL x, i, maat, halfsize AS LONG
    LOCAL kfaktor#, kt#, Rl#, Im#, g#, pw#
    DIM a(UBOUND(Samp#)) AS LOCAL DOUBLE
    maat = (UBOUND(Samp#)) + 1 ' must be a power of 2
    halfsize = maat / 2
    'note: size of Sp# must be Sp#(halfsize-1)
    kfaktor# = Pi2 / maat ' this is eqv. to ATN(1)* 8# / maat
    MAT a() = (1.0/maat) * Samp#()
    i = 0
    DO

        kt# = kfaktor# * i
```

```
            RESET Rl# , Im#
            FOR x = 0 TO maat - 1

                 g# = (kt# * x)
                 Rl# += (a(x) * COS(g#))
                 Im# -= (a(x) * SIN(g#))

            NEXT x
            pw# = (Rl# * Rl#) + (Im# * Im#) ' sum of squares
            IF pw# THEN Sp#(i) = 2 * SQR(pw#)
            INCR i

       LOOP UNTIL i = halfsize ' the second half would be empty

  END SUB
```

In a later 10-fold improvement for speed, we recoded this using lookup tables for the sine and cosine factors and added the application of a Hanning window inline. The resulting code obviously becomes a lot less readable. All DFTs in our gesture recognition code are performed with a 25% overlap. To classify these properties, we split the power spectrum frequency bins in Sp#() in two parts at a variable point i. Note that our spectrum is 8 octaves wide (4 Hz to 512 Hz) and thus, taking into account the linear nature of the frequency distribution in the transform, we have to set i=15 for a normal midpoint at 64Hz. Then we sum up the power in the spectral bands up to i, as well as the power in the spectral bands i+1 up to the highest frequency. The power found in the lowest bin at Sp#((0) is dropped. If lw is the first sum, and hw the second, the smoothness property for each vector is calculated as 1 - (lh / (lw+lh)) and for the edgy property as 1 - (lw/(lw+lh)). The structure also returns the durations of these mutually exclusive properties in 1/256 s units. The algorithm for this property runs at a much lower speed of 16Hz, thus guaranteeing a data overlap of 25% on each refresh. Performing the required transforms much faster causes too much jitter in the behaviour of our real time multitasker. A consequence of this is that the durations are incremented in 16 unit steps (62.5 ms).

As we found that the analysis of the power spectrum offers a good basis for the analysis of some gestural features, we created a thread in the software to perform the spectral transforms on all three data buffers continuously. The results are returned to the user via pointers in the gesture structure: gesture.pspf() for the fast transform (4 Hz-512 Hz in 128 bands, with a refresh frequency of 25 Hz); gesture.pspm() for the medium buffer transform (1 Hz- 128 Hz in 128 bands, with a refresh frequency of 5 Hz), gesture.psps() for the slow buffer transform (0.25 Hz- 32 Hz in 128 bands, with a refresh frequency of 1.25 Hz).

6. 'Airborneness'

We came upon this gesture property when doing experiments with balls thrown through the sensor system. When looking into the spectral analysis data obtained, we noticed that for flying objects, the spectrum is discontinuous and reveals a gap at its low end. By looking at the size of the gap, we can distinguish flying objects – or for that matter, similarly, acrobatic jumps – from bodies in contact with the ground. The reason for this is quite easy to understand, if we realize that a moving body normally always has some parts at rest, even when extremities such as arms, legs, head are moving vigorously. The Doppler noise bands returned are therefore continuous and always start at zero. Airborneness might not be the most useful feature to extract from musicians' gestures, but forms an attractive bonus if it comes to dance movement analysis. It will be obvious that the time frame for this property detection has to be made very short and that the persistence values of the property cannot attain high values.

When coding for recognition of this property, one has to be very careful to take into account the spread of the background noise in the spectrum. To clarify this, we provide here a plot of the maxima encountered in the spectral transform over the 250ms time window (x-vector) without any body in the set-up:

Figure 26: spectral noise over two minutes

The values on the horizontal axis are the frequency bins and have to be multiplied by a factor of four to obtain real-world frequency values. The measurement time for this plot was 2 minutes. For proper evaluation of spectrum-based gesture properties, this basic noise spectrum has to be subtracted from the acquired data arrays. From a scientific point of view, a superior way of handling this would be by letting the software measure the basic noise spectrum on initialisation. Taking into account the stringent conditions for a meaningful measurement (no moving bodies or objects in the neighbourhood as well as a measurement time of a few minutes), we have instead used a rough approximation of the curve in our coding. Here are two typical spectrum plots thus obtained during a jump:

Figure 27: typical spectrum plot during jump

Figure 28: typical spectrum plot during jump

The vertical scaling was adapted to a factor of 10, to cope with the higher dynamic range obtained here. Unfortunately, due to the refresh speed limitations of the transforms in our coding, this property can only be calculated at a rate of 8 times a second. Hence, the duration field returns only a rough estimate of the time the body is airborne and cannot be used for precise measurement of jumps. However, the shape of the gesture during the jump is reflected pretty well in the spectrum obtained. In order to optimize the recognition code for speed, we converted the spectra to octave bands first. Then we evaluated the proportion of power distribution in the lowest two bands to the sum of all the higher bands. If the total power is higher than a measured noise level and this proportion exceeds an empirically determined value (a value of 17 was found), the property is set.

The reliability of jump detection using this algorithm is higher than 90%.

7. Tempo and periodicity (repetitive movements)

The most difficult gesture property to retrieve in real time is periodicity. It represents a well known problem that has been treated in depth by a lot of different authors in very different contexts. The problem is the same whether applied to music or to gesture. We need information about periodicity in the gesture because it allows us the derive a tempo from gesture input.

There are a number of different algorithms possible, each with pros and cons. Since we are dealing with slow periodic phenomena, the fastest algorithm consists of measuring the time between periods (1/t equals frequency then) and checking the deviation from the mean on each new periodic phenomenon. The fastest algorithm starts from the collision detector, and yields a delay time in the order of 100ms. Synchronization is well possible, but latency and sudden jumps cannot be avoided. For clearly colliding periodic gestures, this method is very reliable and precise as long as the periodicity stays below 4 Hz (tempo MM240). However it imposes many restrictions on the type of gesture input. The jitter on the periodicity can be used as a base to calculate a fuzzy value for the tempo property. For the kind of gestures made by a conductor, we found that this path can be followed although the results are not always very convincing.

The FFT approach can be followed as well, but since it has to work on the 4 second buffers if it is to resolve tempi as low as MM30, the latency will be very large. The advantage is that it also works on less clear-cut gesture input. A serious disadvantage is that synchronizing with the input is difficult. The code as we wrote it, tries to cancel out the frequent occurrences of harmonics in the spectral transforms. Yet, it is far from perfect although we have been using it to synchronize music coded in midi files or calculated in real time with dancers on stage.

In the FFT approach, we coded a task that tries to derive a musical tempo expressed in MM numbers by interpolating between the frequency bins from the spectrum transforms. Here is a code snippet:

```
' spectra obtained after a DFT with a hanning window in the FFT thread
' vectors are 0,1,2 for the x,y,z buffers and 3 for the sum.
DIM sps(128) AS LOCAL DOUBLE AT @gesture.psps(vector)
i = ArrayMax_Dbl (sps()) 'find the strongest frequency bin
IF i THEN

      gesture.periodic(3) = ((i+1) * 15) - (7.5 * sps(i-1)/sps(i)) +
      (7.5 * sps(i+1)/sps(i))
      gesture.jitter(3) = (SQR(sps(i) + sps(i-1) + sps(i+1))) * 1E9
ELSE
      gesture.periodic(3) = 15 + (7.5 * sps(1)/sps(0))
      gesture.jitter(3) = (SQR(sps(0) + sps(1))) * 1E9

END IF
' now we try to evaluate the spectrum
RESET ct
FOR i = 0 TO 2
      IF ABS(gesture.periodic(i) - gesture.periodic(3)) <
      gesture.periodic(3)/ 20 THEN
      ' 5% deviation allowed
                  INCR ct
      ELSEIF ABS((gesture.periodic(i)/2) - gesture.periodic(3)) <
      gesture.periodic(3)/ 20 THEN
                  INCR ct 'in this case the i vector must be
                  an octave above the sum of vectors
      ELSEIF ABS((gesture.periodic(i)*2) - gesture.periodic(3)) <
      gesture.periodic(3)/ 20 THEN
                  INCR ct 'in this case the i vector must be
```

```
                an octave below the sum of vectors
        ELSEIF ABS((gesture.periodic(i)/3) - gesture.periodic(3)) <
        gesture.periodic(3)/ 20 THEN
                INCR ct 'in this case the i vector must be
                an fifth above the sum of vectors
        END IF
NEXT i
SELECT CASE ct
        CASE 0
                gesture.jitter(3) = gesture.jitter(3) ^4 ' no periodicity
        CASE 3 ' here we are certain, although we may have the wrong octave.
                gesture.jitter(3) = gesture.jitter(3) ^ 0.25
        CASE 2 ' high probability
                gesture.jitter(3) = gesture.jitter(3) ^ 0.5 ' we should see how the
                one deviating period relates to the equal ones.
                ' if it is a harmonic, we could increase confidence
        CASE 1
                gesture.jitter(3) = gesture.jitter(2) ^ 1.5 ' decrease certainty
END SELECT
```

A final approach, leading to quite reliable results under restricted conditions, although still with a fair amount of latency, makes use of cepstrum analysis. The usual method was followed here: starting from the power spectrum obtained on the slow data buffer, the log of the spectrum is calculated. On this dataset an inverse DFT is performed, yielding a series of periodicities defined in the time domain. In the literature these are referred to as 'quefrencies'. If this transform shows up a clear peak in the section after the initial impulse response burst, this peak is likely to correspond with a clear periodicity in the gesture. It will take time to investigate all the variants in coding and implementation, but all our results have been quite disappointing. Presumably the relatively high amount of simultaneously present non-periodic signal components in the input spectrum are at the

origin of the failures. More sophisticated filtering techniques on the spectrum data prior to performing the cepstrum transform might be a remedy. However, even when we get it to work, synchronization with the input in real time remains very problematic and presumably can only be done if the calculated tempo is applied to an event taken from the fast data buffer.
Suggestions for improved code and maths here are most welcome.

8. Freeze

A gesture property almost so trivial that we might almost forget to mention it: the absence of movement. We could also have named it rest, referring to the musical meaning. Absence of movement can be detected as soon as the received signals start sinking away into the noise floor. This happens when no body is in the set-up. A gesture freeze with a body in the field of the sensors will always have signal levels above absolute noise, since a living body always moves. Therefore, this property is not binary, but again a fuzzy value with a duration of its own.

A plot of input data versus analysis results is shown below. In this example, only the Y-vector has triggered the freeze property:

Figure 29: plotted result of freeze algorithm

As of the time of writing, the complete structure used to return our gesture properties to the user looks like:

```
TYPE GestureType DWORD
     collision (3) AS SINGLE        '0=X, 1=Y, 2=Z, 3= total
                                    acceleration based, the values are the correlation magnitude
     collision_dur (3) AS LONG      'in 1/256 s units
     theacol   (3) AS SINGLE        'theatrical collision       acceleration based
     theacol_dur (3) AS LONG        'in 1/256 s units
     implo     (3) AS SINGLE        'surface based  - normalized property strength
```

```
implo_dur (3) AS LONG          'in 1/256 s units 0 at the start of detection, climbing up as
                               long as the property is valid.
implo_val (3) AS SINGLE
explo     (3) AS SINGLE        'surface based  - normalized property strength
explo_dur (3) AS LONG          'duration of the property in 1/256 s units
explo_val (3) AS SINGLE        'prediction value of the property (FIR output)
speedup   (3) AS SINGLE        'speed based
speedup_dur (3) AS LONG
speedup_val (3) AS SINGLE
slowdown  (3) AS SINGLE        'speed based
slowdown_dur (3) AS LONG
slowdown_val (3) AS SINGLE
periodic  (3) AS SINGLE        'should give the tempo in MM units
jitter    (3) AS SINGLE        'should give the degree of certainty of the above tempo.
flue      (3) AS SINGLE        'gives the standard deviation for the surface buffer  (
                               in fact 1-s)
                               ' constant body surface property
flue_val  (3) AS SINGLE        'gives the average value for the surface buffer
                               (normalized 0-1)    size of body surface
flue_dur  (3) AS LONG          'duration in 1/256 s units
fixspeed  (3) AS SINGLE        'speed based, gives standard deviation for the frequency buffer
fixspeed_val (3) AS SINGLE     'gives the average value of the constant speed
fixspeed_dur (3) AS LONG       'duration in 1/256 s units
impact    (3) AS SINGLE        'surface of the body on the moment of detected collision
                               (not a gesture property!)
smooth    (3) As SINGLE
smooth_dur(3) AS LONG
edgy      (3) AS SINGLE
edgy_dur  (3) AS LONG
airborne (3) AS SINGLE
airborne_dur (3) AS LONG
freeze    (3) AS SINGLE      'property set when no movement is detected above the noise level
freeze_dur (3) AS LONG       'in 1/256 s units
freeze_val (3) AS SINGLE     'normally = 1- freeze, value always set
distance  (3) AS SINGLE      'requires the combination of radar and sonar.
pspf (3) AS DOUBLE  PTR          'pointer to the spectral transform data for the fast buffer
pspm(3) AS DOUBLE PTR            'pointer to the spectral transform data for the medium buffer
psps(3) AS DOUBLE PTR            'pointer to the spectral transform data for the slow buffer
algo              AS DWORD       'parameter for algorithm to use (for research)
```

```
    Sorder            AS DWORD      'FIR filter order for the surface related properties
    Forder            AS DWORD      'FIR filter order for the speed related properties
END TYPE
```

Since the entire structure is recalculated 256 times a second, responsiveness is pretty good, certainly in comparison with competing non-radar or sonar based technologies. We took measurements to estimate the processor load and found out that the complete gesture analysing procedure takes between 12 MIPS and 25 MIPS, using a quad core Pentium processor. That is about half of the available capacity. More than half of this processor load is due to the spectral transforms.

The full source code as well as compiled libraries (DLLs) are available from the author for evaluation. The software, with slight modifications, can also be used without the National Instruments data acquisition device [13], if a decent quality four channel sound card is used. We have sensors available that can connect directly to audio inputs of a PC. These do not require a analogue computing and processing board. We welcome contributions from other researchers and invite them to use, test and improve our approach.

Section 3

Namuda in practice: gesture recognition as artistic expression

This concluding chapter describes the application layer for gesture recognition using Doppler-based hardware systems.

Being capable of recognizing a defined set of gestures in a piece of software is of little significance if an application layer fails. Namuda dance technique requires a mutual adaptation of the performer and the software parameters. In order to make the study of Namuda dance possible, we have designed a series of études in which a single gesture prototype can be practised. Since visual feedback to the performer is very problematic in the context of performance, for it greatly hinders freedom of movement and is by nature too slow, we have opted for auditory display. The robot orchestra [14] as we have designed and built it, makes a good platform for such auditory display, particularly since the sounds are not virtual (loudspeakers) but real acoustic sounds emanating from real physical objects. In fact just about any musical instrument can be seen as an example for auditory display as it by its very nature truthfully converts a certain subset of fine motor skills and gestures into sound. The gestures underlying music practice may very well constitute a basis for the embodiment underlying the intelligibility of music. [15] The motor skills and gestures entailed by playing traditional musical instruments are obviously instrumental in nature. They are dictated by the mechanical construction of the instrument. Therefore, as an extension of the body, an instrument can, at most, be a good prosthesis. By removing the necessity of a physical object, the body becomes the instrument. But this in no way removes the need for motor skill and gestural control.

1. Namuda études

The scheme followed for each étude is always the same: starting with the default parameter settings in the software, the performer has to practice each gesture prototype until the recognition is optimum, as can be judged from the response of the robots. The default parameters are not arbitrary, but have been determined as a result of hundreds of measurement sessions with about twenty different subjects, male and female. Each gesture prototype is mapped to a different subset of responding robots. In this respect, the study of Namuda gestures is quite similar to the study of any musical instrument. A certain level of fine motor control has to be developed in the player. Only once that level has been reached can the recognition software be modified by changing the parameters slightly. One would never buy a new and better violin for a child every time it makes a handling and playing mistake. Only once it knows the basics reasonably well should buying a better instrument become an option. Fortunately, in the case of the invisible instrument, we do not have to buy a new instrument but we can improve the software and adapt it to the player. This last possibility opens a whole new perspective for future developments in instrument building.

1.1: Speedup

To study steady accelerating movements, we made a mapping to the x, y and z vectors for the oboe (<Ob> robot), the cornet (<Korn> robot) and the saxophone (<Autosax> robot) respectively. The speedup property strength is mapped to the pitch the instruments will sound whereas the sound level is controlled by the value of the speed parameter. It is a good exercise to try to have the instruments play as long as possible by stretching the time over which the property can remain in a set state. As soon as the property is set in all three vectors, the large stainless steel shells that make up <Llor> will come into play. The sensitivity is set at a much lower level for this to happen than is the case for the vectors taken separately.

1.2: Slowdown

The slowdown property can obviously only be set when the starting gesture is in movement already. Thus the property can never be triggered from rest. The étude maps all three vectors to pitches in the <Piperola> and <Bourdonola> robots. These are flue pipe organs. The pitch depends on the value of the slowdown, not on the property strength. When the property is set in all three vectors, the lights on the piperola robot will flash.

1.3: Expanding

To trigger this property a movement is required whereby the amount of moving body surface gradually is increased. The property is associated with growth, explosion,

enlargement and can be triggered from a standstill. The x-vector is mapped to our <Klung> robot, an automated Indonesian anklung with brass chimes. Pitch selection is mapped to property strength whereas volume is correlated to the value of the property. The y-vector similarly is mapped to <Simba>, a cymbal playing robot. The z-vector is mapped on <Springer>, a somewhat hybrid robot combining shakers, very large springs mounted on resonators as well as a big motor-driven siren. The selection of the elements playing in this robot is mapped to the duration of the property in the z-vector.
When all three vectors are set, the lights on the <Simba> robot will come up.

1.4: Shrinking

Being the other side of the dipole to the previous property, the gestural associations can also be formulated as imploding, diminishing, getting smaller. Clearly the property presupposes movement to start from. The mapping is as follows: x-vector to <Xy>, a quarter tone xylophone robot; y-vector to <Tubi>, a quarter-tone robot made with thick aluminium tubes; z-vector to <Vibi>, an automated vibraphone. For all three vectors, property strength is mapped to the pitch and property value on the sound level. When the property is triggered in all three vectors the lights on <Xy> will flash.

1.5: Steady

In order to trigger this property it is required that the amount of body surface in motion remain constant within the time frame of measurement. The mapping for this property is on our quarter-tone organ robot <Qt>. When the property is set in all three vectors, the blue lights on the robot will come on. In this mapping the amount of body surface remaining constant determines the pitches of the notes. The attack velocity of the notes is controlled – in a rather subtle way – by the property strength.

1.6: Fixspeed

This property is set as soon as the detected speed of a movement stays reasonably constant within a time frame of 500 ms. It is pretty difficult at first to trigger this property more than just accidentally. The reason is not only due to our control, but also in part due to the cosine factors on the Doppler frequency shifts. The latter can be much improved and even cancelled out by keeping the angle of the movement axis and the sight of the vectorial transducer constant. The mapping of all three vectors here is on one of our smallest robots: <Toypi>, an automated toy piano. When all three vectors have the property set, the lights on the little robot come up.

1.7: Collision

Since determination of this gesture property is based on acceleration followed by a sudden stop, it implies a well-defined sudden stop in the gesture. Rebounding movements should be avoided as they can result in false or double triggering. To make the étude convincing, we mapped the output to nothing but percussion, the collision-based instrument family par excellence. The étude should be performed using all possible parts of the body: not only arms and legs, but also the head, the entire torso, the feet, the elbows and even the belly if musculature allows it... The x-vector is mapped to the drums in <Troms> (a set of drums) and the <Snar> robot, an automated snare drum. The y-vector is mapped to the the cowbells that make up the <Vacca> robot. The z-vector is mapped to the set of woodblocks in the <Thunderwood> robot as well as to the thin metal sheets in the <Psch> robot. When collision is detected in all three vectors, the cymbal in the <Troms> robot will play. If collisions are detected but they are below the sensitivity level set in the software, the white lights on the robots will flash.

1.8: Theatrical Collision

As this prototype is defined as a decelerating movement ending in a stop and accelerating again – avoidance of real collision – it tends to be set more easily over relatively larger time spans. In our étude we mapped the gesture to our <Puff> robot, a percussive organ-like instrument tuned in quarter tones.

1.9: Smooth (roundness)

1.10: Edgy

These gesture prototypes can be practised together. The edgy property strength is mapped to the piano notes, played by our player piano robot. The smooth property is mapped to the pitches of our quarter-tone organ robot, <Qt>. The attack velocity is controlled by the momentary value of the vectorial moving body surface. The property is set based on an analysis of the spectrum of the Doppler signals. Edgy movement with many sudden changes causes an overweight of higher partials in the spectrum. The sound volume from <Qt> is made to be a (slow) function of the value of the low side of the power spectrum. It had to react slowly because it steers the compressor on the organ. Those motors, due to their inertia, cannot change speed suddenly. The recognition quality was rated as excellent by all performers we have subjected to this étude so far.

1.11: Jump (airborneness)

The airborne gesture property is set when the performer jumps and no part of the body is at rest. False triggering can occur on large stepping movements though. The larger or higher the jump the stronger the property will be defined. For this étude the x, y and z-vectors are respectively mapped to <Heli> (a helicon robot) <Bono> (a trombone robot) and <So> (the sousaphone robot). When the property is defined for all three vectors, both castanet-playing robots will join in. The sound volume is mapped on the amount of body mass involved whereas the pitch will be proportional to the property strength.

When all three vectors have the airborne property set, certainty is nearly 100%. If only two vectors trigger the property, certainty is still higher than 90%. Most often it is the Z-vector that fails to trigger, which is easily explainable by the fact that the transducer for that vector is suspended above the performer. In order to be certain that the Z-vector also triggers the property, the jump must also include a positional displacement. Needless to add, this étude is quite exhausting to perform.

1.12: Periodic

This is the least well-functioning property in our set and it definitely needs further improvement. Response time is too slow and false triggers do occur regularly. The mapping to the drums in our <Troms> robot allows practice and evaluation. We have also developed software allowing performers to synchronize midi or audio file playback with gestural input. Within strict limitations such as no sudden tempo changes, avoidance of rhythmical complexities (doublings and triplets) it can even be got to work. Further research is being done based on cepstrum analysis.

1.13: Freeze

To practice this 'non-gesture', we applied an inverse mapping, whereby sound will be heard as long as the freeze property is not triggered. The robot used in this mapping is <Bourdonola>, a low register open organ pipes with a string-like sound. The étude is very fundamental as it lets the performers experience the very high sensitivity level of the system.

Parallel to these recognition-based gesture properties, the implementation also offers a full set of very direct mappings of movement parameters on sound output:

- moving body surface: The most intuitive mapping for this parameter seems to be to sound volume or density.
- speed of movement: The most intuitive mapping for this parameter seems to be to pitch.
- spectral shape of the movement: The most intuitive mapping for this complex parameter seems to be to harmony.
- acceleration of the movement: The most intuitive mapping for this parameter seems to be to percussive triggers.

Of course there is nothing mandatory in the way the mappings of gestural prototypes have been laid out in these études. It is pretty easy to devise mappings more suitable for use out of the context of our own robot orchestra. The simplest alternative implementations consist of mappings on any kind if MIDI synth or sampler. However mapping the data from our gesture recognition system to real time audio streams (as we did in our 'Songbook' in 1995, based on human voice sound modulation via gesture) is an even better alternative.

2. Extending the time frame

The gesture prototypes practised in the études reflect a gestural microscale in the time domain. Their validity may be as short as 7ms and can for most properties seldom exceed 2 seconds. The only gesture properties that can persist over longer time spans are freeze, periodic, edgy, smooth, fluent and fixspeed. Some can, by their nature, only be defined over very short time intervals: airborne and collision. These gesture prototypes are to be compared to what phonemes are in spoken language, although they already carry more meaning than their linguistic counterparts.

Meaning here being understood as embodied meaning.

By following assignment and persistence of the gesture prototypes over longer time spans, say between 500 ms and 5 seconds, it becomes possible to assign expressive meanings to gestural utterances. Here we enter the level of words and short sentences, to continue using linguistic terminology as a metaphor. When we ask a performer to make gentle movements in order to express sympathy, then the statistical frequency of a limited subset of gesture properties will go up significantly. When we ask for aggression, the distribution will be completely different.

It is on this level of time scale that quite a few of our gestural prototypes may be correlated to the classifications of 'effort' set up by Rudolf Laban in his text 'Effort ; Rhythmic Control of Man's activities in Work and Play'. This text was written well before 1950 and only got published as an appendix to the book mentioned in the bibliography of our paper. (Laban, 1980). As far as we can judge, his classification and notation proposals are difficult to hold in a more generalized context of a theory of expressive gesture. Clearly none of Laban's analysis are based on any other objective measurement than visual observation and the dancers' own experience of effort.

It would be an interesting research project to find out how comparable such distributions for a limited set of sentic forms [16] are amongst a large set of different dancers and musicians. Unfortunately this is beyond the scope of our own mission. In part, such research is being done now by our colleague Dr. Jin Hyun Kim.

3. Relation to other dance practices.

Although as soon as we gained some insight in the potential for dance offered by our technology – in the mid nineteen-seventies – we carried out artistic experiments with dancers trained in classical ballet as well as modern dance, we quickly found out that such an approach to dance was unsuitable to work well with this technology. Classical dance forms concentrate on elegance and – in general –

avoid collision and a sense of mass. Position in space and visual aspects are very dominant. Immediately alternative dance practices came into consideration. In the first place butoh dance, an avant-garde dance practice with its roots in Japan, where we also came in contact with it (through Tari Ito). Thus we got in contact with dancers such as Min Tanaka, Tadashi Endo, Emilie De Vlam and later on Lazara Rosell Albear ... This has led to quite a considerable list of collaborative performances. However, butoh is only vaguely defined from a technical dance point of view. Its non-avoidance of roughness, its nakedness [17] and its concentration on bodily expression, leaving out any distracting props and requisites, formed the strongest points of attraction. Only in some forms of contact improvisation did we find other links, but in this dance form we ran into problems with our technology, which is not capable of distinguishing the movements of more than a single moving body at the same time. As far as couple dances go, we have also investigated tango (and the related milonga) quite a bit, in part also because we happen to be a tanguero ourselves. In this type of dance the problem of the two inseparable bodies poses less of a problem since movements are always very well coordinated. Acrobatic tango is of particular interest for the gestures used are very well defined. However, good dance couples in this genre are pretty rare. Moreover, acrobatic tango is generally highly choreographed, leaving little space for improvisation, thus not lending itself so well to genuinely interactive applications.

Other dance forms we have experimented with include pole dance, flamenco and break dance. For the first dance style, we even developed a special set of microwave radar sensors that can be mounted at the top of the pole. Unfortunately we found out that the professionals in this dance form have little if any affinity with the art forms we are interested in ourselves, which are after all still quite experimental. As for break dance, we are forced to admit that despite the fascination we have for the virtuosity in movement that can be found in the genre, it seems to be strongly bound to a certain age group that we have long since left behind us...

Although not normally thought of as dance forms, the movement practices that can be found in some sports and martial arts, seem to offer a good technical introduction to the Namuda gestures considered here: karate and to a lesser extent judo. Karate is of a particular interest here because it exploits in full the development of very fast accelerating gestures.

4. Interactive composition and choreography

It will be clear that the mastering of Namuda opens wide perspectives for the development of real time interactive composition with a strong theatrical component. Over the 35 years that we have been developing the system, many hundreds of performances have been staged. Here is a short overview of our own main artistic productions in which the technologies described here were developed and/or applied:

- Holosound (1980)
    - This was the first large scale music theatre performance using the purely analogue version of my invisible instrument. The performer was invariably Moniek Darge and the piece was performed at least 200 times all over the world. An alternative version functioning as an audio art installation was realized as well. A copy of the installation was purchased by the Musical Instrument Museum in Brussels.
- A Book of Moves (1992)
    - This was the first large-scale composition using our invisible instrument. Each section of the piece demonstrates another way of directly mapping gesture derived parameters to sound generated by an assembly of samplers and synthesizers. The piece was performed over 350 times all over the world by the Logos Duo (the author and Moniek Darge). Together with the Holosound installation, the hardware and software for this production is also available at the Musical Instrument Museum in Brussels. Another copy was purchased by 'het Muziekgebouw' in Amsterdam, where it is part of their sound garden.

- Songbook (1995)
    - In this full evening production we used our invisible instrument to control the sounds of the performers' voices in real time through pure wireless gesture commands. Thus gesture-controlled pitch shifting, a variety of effects, harmonizers and filters as well as the build-up of a large choir from a single voice, were implemented and demonstrated. The piece makes use of a bank of DSP sound processors. It has been performed over a hundred times all over the world. Performers have been Moniek Darge, Karin Defleyt, Joachim Brackx and the author.
- Gestrobo Studies (2000)
    - This is the first collection of pieces using the sonar-based invisible instrument as well as the robot orchestra. Different sections of the suite have been performed many times by performers Emilie De Vlam, Tadashi Endo, Marian Deschryver, Nicoletta Branchini, Dominica Eyckmans and others.
- Quadrada Studies (2001)
    - The first collection of pieces using the microwave radar-based version of the invisible instrument with mappings to the robot orchestra. Performers of different pieces in this suite have been: Angela Rawlings, Helen White, Moniek Darge and Nicoletta Branchini.
- TechnoFaustus (1998-...)
    - Although try-outs of many different acts in this large scale opera project have been presented throughout our regular concerts, the full production as yet still awaits its premiere. The try-out acts have been performed by Moniek Darge, Emilie De Vlam, Kristof Lauwers and the author.
- Hanaretemo (2009)
    - A full evening music theatre production using almost all of the technologies we have

hitherto developed for gesture sensing. It was performed by Emilie De Vlam and Nan Ping as butoh dancers.

- Namuda Studies (2010-)
    - As the most recent development in our artistic production, it remains as yet unfinished. Nevertheless we present new contributions to the collection on a very regular basis at our concerts with the robot orchestra. Performers are Emilie De Vlam, Zam Ebale, Lazara Rosell Albear and Dominica Eyckmans and we hope to extend the number of performers willing to undertake some training in Namuda mastership.

The entire Namuda system including the invisible instrument is open for use by other composers and performers. Scientists interested in research into human gesture are also invited to explore the possibilities of the system. Other composers that have made use of it so far are Kristof Lauwers and Yvan Vander Sanden. The system has also been investigated by Hans Roels, Troy Rogers, Jin Hyun Kim, Dirk Moelants and others. Many applications have been developed on other platforms than our own GMT-programming environment. To facilitate this, we have designed a limited gesture sensor with built-in signal processing using a PIC micro-controller. This 'Picradar' sensor, as we have baptized it, makes use of microwave radar (9.35 GHz) and outputs its data following the midi protocol [18]. It can readily be used in PD,and is available for the PC, for Linux and can even run on a Mac.

There is still a lot of work left to be done on improvements to the hardware and recognition software as well as in terms of its artistic implementations. An open invitation.

Conclusions

Combined radar and sonar technology enables full gesture imaging, including both movement and positional data. Omnidirectional wide-band sensors operating in the ultrasound range are used for data capture; support for FM modulation allows distance measurement.

Gesture recognition is a matter of capturing the coordinates of a movement in space, and then creating a 'library' of gestures from different performers from which patterns of gesture characteristics can be extracted to serve as gestural prototypes. We have classified gesture into twelve fundamental categories (plus the motionless state as a thirteenth category), many of which are mutually exclusive dipoles.

In order to enable a relevant artistic application for the technology, a series of études have been devised to practice a single gesture prototype. Mapping the gestural data to the robot orchestra, with each gesture prototype mapped to a different subset of robots, gives the performer audio feedback (and in some cases coloured light displays) on how the system is interpreting the performer's gestural input.

The result is not a complete elimination of craftsmanship: although the Namuda technology does not use the skills required to play a traditional musical instrument, it does demand motor skill and gestural control to produce predictable sound results. It is perfectly possible to operate the sensors without skill, training or specific expressive intent, as numerous workshops with school children have proven: clearly the sensors pick up the children's movements just as they record the gestures of trained dancers, albeit slightly less efficiently (it is impractical to tailor the software to each individual child within a short time frame, and of course the children try out the system wearing their clothes). However the dancers and multidisciplinary artists who use Namuda to create artistic productions generally spend considerable time exploring and practising the sounds that their gestures can produce. The innovative potential of the Namuda system lies primarily in the use of the entire human body to produce music, and also in the capacity of the software to adapt to individual performers by means of minor changes in the parameters. Available for use and further research by composers, performers and scientists, the Namuda gesture recognition system harbours considerable capacity for further development.

Competing interests

None

Acknowledgements

Thanks to my dance and music collaborators who helped me perform the many necessary experiments and measurements using their bodies: Moniek Darge, Angela Rawlings, Helen White, Dominica Eyckmans, Lazara Rosell Albear, Zam Ebale, Marian De Schryver, Nicoletta Brachini, Emilie De Vlam, Tadashi Endo, Nan Ping, Jin Hyun Kim, Marjolijn Zwakman and many others. Thanks also to my collaborator Kristof Lauwers for helping out with code development, data analysis and debugging.

Endnotes

[1] This summary of the background to our experimental instrument-building projects is largely taken from “An Invisible Instrument”, Godfried-Willem Raes, 1997: http://logosfoundation.org/g_texts/invisins.html, which deals in greater depth with the philosopy behind the design.

[2] This project is part of the ongoing research of the author into gesture-controlled devices over the last 35 years. Earlier systems, based on sonar, radar, infra-red pyro-detection and other technologies are fully described in "Gesture controlled instruments" (1999; http://logosfoundation.org/ii/gesture-instrument.html) as well as in our doctoral

dissertation 'An Invisible Instrument' (1993). Artistic productions and compositions using these interfaces and devices have included:

<Standing Waves> http://www.logosfoundation.org/mp3/lpd003/stanwaves.rm

<Holosound> http://www.logosfoundation.org/scores_gwr/holosound/holosound.html

<A Book of Moves> http://www.logosfoundation.org/scores_gwr/bom-songbook/bom.html

<Virtual Jews Harp> http://www.logosfoundation.org/instrum_gwr/virtualjewsharp.html

<Songbook> http://www.logosfoundation.org/scores_gwr/bom-songbook/songbook.html

<Slow Sham Rising> http://www.logosfoundation.org/scores_gwr/Slow_sham_rising.html

<Gestrobo> http://www.logosfoundation.org/scores_gwr/gestrobo.html

including: <Ices> (Gestrobo Study #19) and <Bodies of revolution> (Gestrobo Study #20)

<Quadrada> Studies http://logosfoundation.org/ii/quadrada.html

<Technofaustus> http://www.logosfoundation.org/scores_gwr/TechnoFaustus/technofaustus.html (in Dutch)

<Butoh> http://www.logosfoundation.org/images/robody/robody9.html

<Differentials> http://www.logosfoundation.org/scores_gwr/Differentials/differentials.html

etc.

[3] As of August 16th 2009 the world record for running in the 100m sprint was 9.58 s. This corresponds to a movement speed of 10.52 m/s or 37.8 km/h. Needless to say, such speeds are not encountered amongst 'normal' people, even when they engage in the wildest forms of dancing.

$$f_d = 2\ v\ f_o \cos(a) / c$$

- f_o = frequency of the carrier (ca. 40 kHz)
- c = propagation speed of the wave (340 m/s)
- f_d = Doppler frequency
- v = movement speed of the body
- a = movement angle with the axis of the transducer

[4] Here is the circuit as we developed and tested it with the Prowave 400FS060 sensor:

Figure 30: Ultrasound receivers for Kynar transducers

[5] Here is the circuit we developed for the Monacor electret MCE2500 microphone:

Figure 31: receivers for Monacor electret

Figure 32: Monacor transducer mounted on circuit board

This circuit has very good wide band characteristics but suffers from a poorer signal noise ratio.

[6] for a more in-depth treatment of FM modulation for distance determination see: RAES, Godfried-Willem "Microwave Gesture Sensing" (Ghent, 2009): http://logosfoundation.org/ii/dopplerFMradar.html.

[7] A word of warning with regard to the measurement of the signal voltage levels may be appropriate here: if using a multimeter with a true RMS scale, check the characteristics of the meter beforehand. The large majority of such instruments cannot handle AC signals with frequencies up to and above 40 kHz. Even good and expensive ones, such as the Agilent U1252A, is only reliable up to 30 kHz. The Fluke 87 performs well in this respect. The instrument of choice for voltage measurement therefore remains the oscilloscope, though is is a bit clumsy to take on the road...

[8] The radar system for gesture measurement is described in more detail in Raes, Godfried-Willem, "Microwave Gesture Sensing"(Ghent, 2009): http://logosfoundation.org/ii/dopplerFMradar.html

[9] The system described here was used for an exhaustive measurement session lead by Dr. Jin Hyun Kim during the second week of May 2010. The sonar receiver signals were recorded as audio tracks simultaneously with the audio of the M&M robot orchestra playing under the control of our own Namuda gesture recognition software referred to in the previous note. At the same time, a high speed video recording (300fps) of all gesture input was made, thus providing a very wide data set for further analysis and research. A paper presenting the results of this investigation will be published in due time.

[10] Most data sheets for ultrasound transmitters specify the SPL measured at a distance of only 30 cm, whereas the standard for acoustic measurements specifies it at a distance of 1 m. This means that we have to subtract approx. 12 dB to bring the data back to common standards. Also note that in our set-up, the distance between emitter and receiver is normally 3 m. This means that the SPL at the point of the receiver will be 20 dB down compared to the SPL as given in the data sheets.

[11] See our design for a real digital loudspeaker, “Talking Flames” (http://www.logosfoundation.org/scores_gwr/talkflam.html). Since the sound source here is a virtual point, the radiation pattern is inherently spherical.

[12] All our code is written in PowerBasic and compiled with their Windows compiler, version 10.0. The code is part of a fairly large programming environment that we have developed for real time composition: GMT. We have put it in the public domain and all source code is available on the Logos website.

[13] A great advantage to using the National Instruments devices is that they are per definition very well supported in Labview, undeniably the leader in professional instrumentation software and tools for analysis.

[14] Detailed information on the robot orchestra and the robots constituting it can be found at http://www.logosfoundation.org/instrum_god/manual.html

[15] These matters were discussed in depth in my texts on the invisible instrument: Raes,1993 ,1994 and 1999.

[16] The notion of sentic forms was introduced by Dr. Manfred Clynes, with whom we had many conversations and discussions at the time we visited him in Sydney when we were demonstrating our invisible instrument at the Sydney Conservatory. References to his publications on the subject can be found in the bibliography section of this paper.

[17] We wrote an essay on nakedness some time ago, after realizing that even nowadays there are still people around that seem to have difficulty in coping with this utmost human property... The text can be read at: http://logosfoundation.org/g_texts/naked.html.

[18] A paper describing the Picradar project as well as some of its applications can be found at: http://www.logosfoundation.org/ii/picradar.html

---

References

- Analog Devices, AD633 data sheet and application notes, 2002.
- BECKMANN, Petr & SPIZZICHINO, Andre "The Scattering of Electromagnetic Waves from Rough Surfaces", Pergamon Press, Oxford, 1963
- CLYNES, Manfred "Sentics, the touch of the emotions" , Anchor Books (NY, 1978)
- DROITCOUR, Amy Diane "Non contact measurement of hearth and respiration rates with a single-chip microwave Doppler radar", Stanford PhD thesis, June 2006.

- KRAMER, Gregory (ed.) "Auditory Display", Addison-Wesley Publishing Company, (Reading MA, 1994)
- LABAN, Rudolf "The mastery of movement", Macdonald & Evans Ltd (Plymouth, 1980)
- MENDE, Ralph, BEHRENS, Marc "A 24GHz ACC Radar Sensor", 02/2005, Smart microwave sensors Gmbh
- RAES, Godfried-Willem "Een onzichtbaar muziekinstrument" (Gent, 1993, Ph.D. thesis)
- RAES, Godfried-Willem "An Invisible Instrument (1994): http://logosfoundation.org/g_texts/invisins.html
- RAES, Godfried-Willem "Gesture controlled virtual musical instruments" (Ghent, 1999): http://logosfoundation.org/ii/gesture-instrument.html
- RAES, Godfried-Willem "Quadrada" (Ghent, 2003): http://logosfoundation.org/ii/quadrada.html
- RAES, Godfried-Willem "PicRadar" (Ghent, 2004-2005): http://logosfoundation.org/ii/picradar.html
- RAES, Godfried-Willem "Distance sensing" (Ghent, 2007): http://logosfoundation.org/ii/distance_sensing.html
- RAES, Godfried-Willem "Naked" (Ghent, 2008): http://logosfoundation.org/g_texts/naked.html
- RAES, Godfried-Willem "Microwave Gesture Sensing" (Ghent, 2009): http://logosfoundation.org/ii/dopplerFMradar.html
- RAES, Godfried-Willem "Holosound 2010, a doppler sonar based 3-D gesture measurement system" (Ghent, 2010): http://logosfoundation.org/ii/holosound2010.html
- RAES, Godfried-Willem "Holosound 2010, a doppler sonar based 3-D gesture measurement system " (Ghent, 2010): http://www.logosfoundation.org/scores_gwr/Namuda_Links/namuda_studies.html
- ROADS, Curtis “The computer music tutorial”, (MIT press, 1996)
- ROELS, Jetty(ed.) "Images of corporeality, traces of Butoh in Flanders", VCIT, (Ghent, 2002)
- SAMRASH Company "Users Manual for the Bumblebee", (05.2008) http://www.samraksh.com/support.html
- Microwave Solutions Ltd. 'MDU11xx series tunable transceivers', 2008
- SEIFERT, Uwe, HYUN KIM, Jin (eds) "Paradoxes of Interactivity", (Bielefeld, 2008)
- SINCLAIR, Ian Robertson., "Sensors and Transducers" (London, 1992) , ISBN 0 7506 0415 8
- TECK, Katerine "Movement to Music", Greenwood Press (Westport CT, 1990)

Figure legends

Figure 1: Ultrasound receiver for electret sensor