Gesture-recognition with Non-referenced Tracking

Paul Keir Jocelyn Elgoyhen Martin Naef
John Payne Martyn Horner Paul Anderson

Digital Design Studio
The Glasgow School of Art

ABSTRACT

This paper presents 3motion™, a novel 3D gesture interaction system consisting of a low-cost, lightweight hardware component and a general-purpose software development kit. The system provides gesture-based 3D interaction for situations where traditional tracking systems are too expensive or impractical due to the calibration and reference source requirements.

The hardware component is built around a 3-axis linear accelerometer chip and transmits a continuous data stream to a host device via a wireless Bluetooth link. The software component receives this data and matches it against a library of 3D gestures to trigger actions.

The system has been validated extensively with various example applications, including a "Battle of the Wizards" game, a character manipulation demonstrator, and a golf game implemented on a mobile phone.

CR Categories and Subject Descriptors: I.3.6 (Computer Graphics) Methodology and Techniques – Interaction Techniques Additional

Keywords: Gesture recognition, 3D input device, linear accelerometer, tracking system

1 INTRODUCTION

Interactive virtual environments must balance the task of immersing the user with the need to instrument their movements. While input data should be gathered unobtrusively, often systems must operate in conditions of reduced lighting, in the proximity of metal objects, or in a mobile context. It is also useful for the system to have an understanding of the intentions behind a user's gestural motions.

Existing motion tracking apparatus may use electromagnetic sensors, computer vision methods or microelectromechanical system (MEMS) components. Such systems typically exist at the higher end of the input device market presenting a cost barrier for some users. Furthermore, implementing gestural interaction on handheld devices with small screens may be impossible using these solutions. This is due to the need to be in the proximity of a calibrated source.

Recent mid-range cost devices such as the Intersense PCTracker utilise sensor fusion methods to combine output from MEMS and ultrasonic sensors. In the home entertainment market In2Games Gametrak™ implements a mechanical system using two retractable cords, while the Sony EyeToy® camera uses techniques from computer vision. Aside from issues of cost, the sensor methods employed are restricted when used outside; whether by light levels, update rates, or the need for an external source.

Commonly available gesture recognition software used in conjunction with such hardware is often limited to matching trajectory shapes based on a single, static orientation. When wireless mobile devices are considered, with undefined and possibly symmetrical ergonomics, a recognition system capable of matching similar shapes regardless of orientation would be valuable.

In this paper we present our handheld inertial tracking device, gesture recognition software and applications. Using low-cost MEMS components our motivation has been to create a product design affordable by the mass-market videogame industry. Similarly the simple, source-less interface that requires no calibration is suitable for a casual end user.

While it is true our device is not a position tracker, our gesture recognition system does not require absolute tracking, operating as it does for only short time periods and matching trajectory data that are seen more like gesture signatures. Therefore we capitalise on the strengths of inertial tracking: fast update rate, high sensitivity, miniature sensor, source-less, low cost and with no line of sight problem. The Bluetooth (BT) wireless communication protocol is used, allowing simple connection with either desktop or mobile host systems.

2 RELATED WORK

Many technical solutions for incorporating gestures within Human Computer Interaction (HCI) tasks have been devised over the years.

2.1 Data Gloves

Instrumented gloves provide an application with information about the hand or hands of a user. Commonly applications using data gloves [3, 21] take advantage of the knowledge of relative finger positions and recognise sign language postures and gestures. Though such gloves report hand states accurately, even the more recent wireless data gloves are still unwieldy and expensive.

2.2 Computer Vision

Computer vision offers the user a less cumbersome interface, requiring of them only that they remain within the viewing frustrum of the camera or cameras. By deducing features and movement in realtime from the images captured from the cameras, gesture and posture recognition [10, 11, 12, 17] can be achieved. However the requirement for pre-positioned cameras means this approach is unsuitable for mobile systems. Computer vision typically also requires good lighting conditions.

Although such systems usually require pre-positioned cameras, some mobile vision gesturing systems do exist [7, 23]. Gesturing

[p.keir | j.elgoyhen | m.naef | j.payne | m.horner]@gsa.ac.uk

IEEE Symposium on 3D User Interfaces 2006
March 25 - 26, Alexandria, Virginia, USA
1-4244-0225-5/06/$20.00 ©2006 IEEE

with such a method involves moving the camera itself, often embedded in a phone or PDA.

2.3 Inertial Motion Sensing

Originally developed for the mass market to deploy car airbags, the falling price, energy efficiency, and portability of MEMS accelerometers has seen their increasing popularity in tilt systems and within motion sensing systems [8, 16, 18]. The miniature size of these inertial units means they can also be used in novel ways [4, 15] within mobile devices such as phones [19, 24] or PDAs [20].

The main issue encountered when using accelerometers within a gesture recognition system is with the sensor signal drift when attempting to derive position. Schemes employing accelerometers for gesture recognition often utilise additional sensors [13, 22] such as the more costly angular rate inertial sensors (gyroscopes) [1, 5] to overcome this lack of stability.

2.4 Gesture Orientation

A circle gestured in the x-y plane creates different data from one drawn in the x-z plane. The problem of obtaining absolute orientation [14] focuses on the calculation of a rotation with which to transform one set of points onto another, minimizing the error between the two sets. Translation and scale values also form a part of the solution, but are simply obtained in comparison to the rotation. Berthold K. P. Horn's paper from 1987 [9] describes a popular closed-form solution using quaternions.

3 System Overview

The 3motion™ technology is the combination of two entities: a handheld controller and a software development kit (SDK).

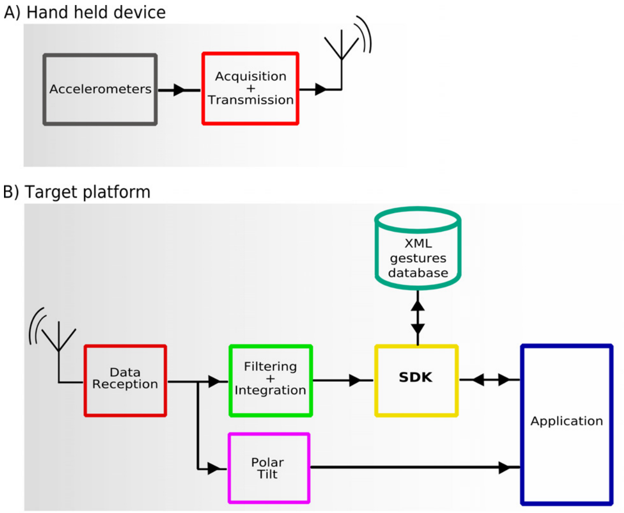

Figure 1: Block diagram of the whole system

The handheld controller is composed of a single chip 3-axis linear accelerometer (see Fig. 1-A) that transmits acceleration measurements in real-time to a host platform. This platform can be a BT-enabled computer, mobile phone, or videogame console.

On the host runs a data packet reconstruction and filtering algorithm followed by the 3motion™ SDK (see Fig. 1-B). On reception, the real-time signals from the device are passed through a pre-filtering stage. As a gesture is performed, the user has to press a switch that triggers a double integration of the signal. When the switch is released at the end of the gesture, the 3motion™ SDK resamples the data set recorded during the gesture. Then the data set is compared with a bank of gesture data sets stored in an XML database. The gestures are compared one by one, first by aligning one with the other, and then calculating statistical variance between the two. The 3motion™ SDK curve-matching algorithm is device independent; working not only with our own hardware but also with any kind of 2D or 3D motion data such as that created by mice or spatial trackers.

The applications that benefit from the 3motion™ SDK can not only make use of the gesture recognition system, they can make use of the raw accelerometer data or tri-accelerometer polar coordinate tilt information from the 3motion™ handheld device (see Fig. 1-B).

4 Hardware

4.1 Sensor Array Types

Due to the commercial targets of the project, the cost of the hardware device was a major contributory factor. We sought to identify a configuration of sensor and interfaces that would provide strong functionality with minimal mass manufacturing cost.

We intend our design to be used by casual users of all ages. With such a wide spectrum of the population, many could find calibration a complex or cumbersome process. Our design had to be built around a cheap, calibration-free sensor array.
As described in the following sections, we considered three types of sensors for our source-less gesture input device.

4.1.1 Angular Rate Sensors

When mounted on three orthogonal axes, MEMS piezoelectric gyroscopes give information about the attitude of the array. Our first concern was regarding the absence of an integrated multi-gyroscope chip. Consequently, all three gyroscopes would have to be precision mounted orthogonal to each other at the factory. A calibration process along all three axes would then be required in order to obtain useful angular rate measurements [2]. These steps have a high impact on cost. Considering mass manufacture, it was decided that at approximately US$10 each, three angular rate sensors were not sufficiently accurate to justify their cost.

4.1.2 Magnetometers

A magnetometric sensor array measures its own absolute rotation around the Earth's magnetic north vector. These data, if used with a sensor fusion system, can give valuable information on the attitude and orientation of the device. Unfortunately common electrical equipment (TV, speakers, mobile phones, etc.) and metallic objects easily perturb the Earth's magnetic field locally, therefore ensuring our decision against adopting magnetometers.

4.1.3 Accelerometers

Any object on Earth is under at least one source of acceleration: gravity. In contrast to the magnetic field, the Earth's gravitational field is stable both in direction (centripetal to the Earth's centre) and intensity (almost 9.8ms^{-2} everywhere). It is a field that does not locally fluctuate or get perturbed and so is a very reliable and measurable vector.

The second source of an object's acceleration is its relative displacement from a reference in space at a fluctuating speed. This acceleration vector gets added to the existent Earth's gravitation vector **g**. As the Earth's gravity vector is constant, all

1-4244-0225-5/06 $20.00 © 2006 IEEE

the variations in acceleration measured by the array originate from its displacement.

The recent availability of inexpensive MEMS 3-axis linear accelerometers allowed us to consider a flat design, single PCB system. A system centred on a single 3-axis accelerometer could be easily manufactured and became an early favourite.

However, recognising gestures with only linear acceleration values is to use only a fragment of the inertial information created when a motion is performed. Our design had to accommodate that restriction.

4.2 Understanding the Sensor Signals

Using only one 3-axis accelerometer array, it is not possible to obtain an indication of the attitude of the device when it is in motion. A 6 Degree of Freedom (DoF) strapdown Inertial Measurement Unit (IMU) configuration overcomes this by utilising gyroscopes to perform attitude correction: a virtual gimbal rotation of the measured acceleration (see Fig. 2). By tracking the gravity vector **g** using the gyroscopes, the acceleration signals can have **g** precisely subtracted. Calculating the trajectory then becomes possible by double integration of the acceleration values resulting only from its displacement.

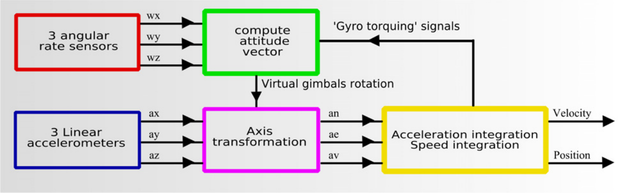

Figure 2: Classical IMU principle

3motion™ takes advantage of the observation that similar motions of our sensor array create acceleration signal data that share similitude over time in both amplitude and frequency. Such data are seen as gesture signature curves, with similar gestures resulting in measurably similar signatures, while dissimilar ones do not (see Fig. 4).

Our system requires the user to explicitly mark the start and end of a gesture by holding a button while the gesture is performed. Gesture signatures are then obtained by twice integrating the acceleration signal over the gesture's duration (see Fig. 3).

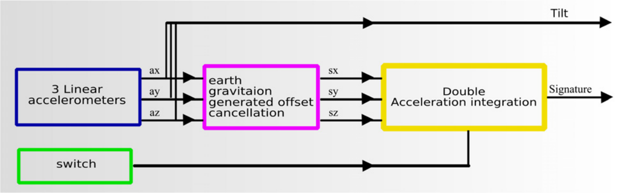
Tilt
3 Linear accelerometers
ax
ay
az
earth gravitation generated offset cancellation
sx
sy
sz
Double Acceleration integration
Signature
switch

Figure 3: 3motion gesture signature principle

Once the signatures are collected they may be compared in order to elicit information on their relative similitude. This step of the input cycle is performed on the host machine, and is described in the following section on software.

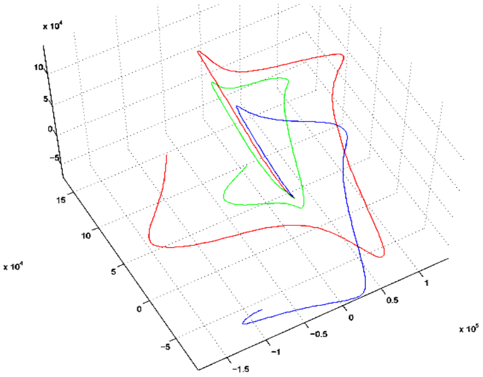

Figure 4: Three signature curves from a "Square" gesture

4.3 Prototype Design Overview

A prototype of the 3motion™ device (see Fig. 5) was created to evaluate the practicality of several concepts. The prototype has become an important tool for testing the requirements of such devices while investigating design concepts for a future commercial design.

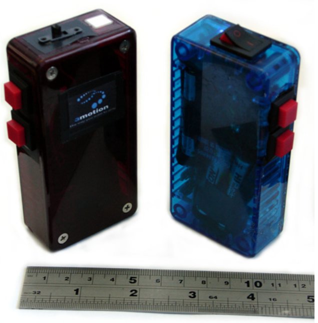

Figure 5: Two 3motion devices

4.3.1 The Accelerometer Sensor

The 3motion™ device is built around a LIS2L02AS4 from STmicroelectronics (see Fig. 6). This single chip 3-axis linear accelerometer provides three analogue output signals and can have its measurement sensitivity configured for either a 2g or a 6g range. The 2g setting provides optimal performance when used as a tilt input device, while the 6g setting handles motions with a greater dynamic range, such as hand and arm movements. The accelerometer's sensitivity may be configured from the host platform by a software command.

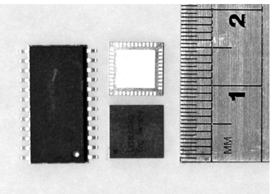

Figure 6: The LIS2L02AS4 from **ST** in SO24 (left) & QFN44 (centre) package

4.3.2 Data Acquisition

Our existing prototype uses the ultra low-power 16-bit Texas Instruments MSP430F1232 micro-controller as shown in Fig. 7. The 10-bit Analogue to Digital Converter (ADC) of the MSP430 is used for data conversion at a sampling frequency of 60Hz. While hand gesture recognition can be achieved with a much lower sampling frequency, the real-time requirements of direct tilt input require a very smooth stream of data. Consequently, the 60Hz sampling frequency was selected as a compromise allowing a highly usable real-time tilt signal, while also permitting an extended battery life.

Each 10-bit sample is wrapped into two 8-bit packets with headers before being sent through the Universal Asynchronous Receiver Transmitter (UART) of the MSP430 to the Bluetooth interface. On reception, the driver on the host reassembles the 10-bit data from the X, Y, and Z sensors.

The firmware makes extensive use of the Low Power Mode (LPM) of the micro-controller. By putting most of the micro-controller core and peripheral in standby between each sampling period, the battery life of the system was maximised.

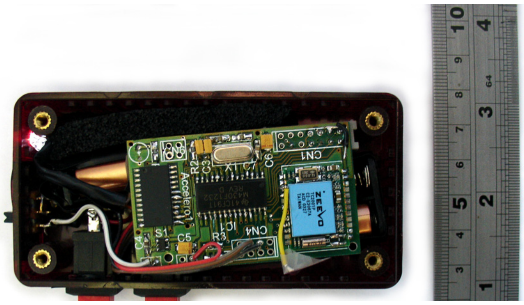

Figure 7: Inside view of the 3 motion prototype

4.3.3 Zeevo Bluetooth Interface

Data communication between the device and the host (PC, mobile phone, game console) is achieved using a Zeevo Z3001Z Bluetooth 1.1 compliant Bluetooth module with a standard Bluetooth serial interface over its UART. The Z3001Z is a standalone device requiring only a power supply and 50Ω antenna (see Fig. 8). Attention commands (AT commands) are used to configure the Bluetooth module by the micro-controller. Once the communication is established, the BT interface becomes transparent and assures the loss-less transmission of all the data while signal strength is sufficient. The serial communications run at 115200 bits per second, although only a small part of that bandwidth is used.

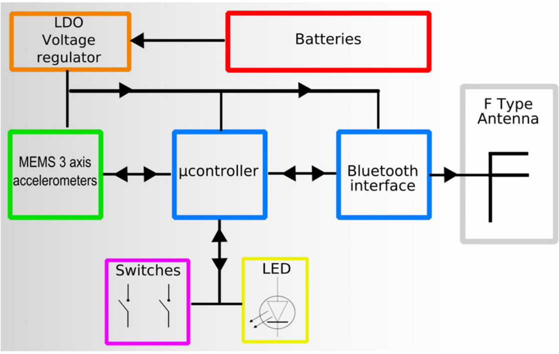

Figure 8: Circuit block diagram

The Z3001Z is connected to a 50Ω micro-strip F-antenna, positioned to minimise hand coverage by the user. By doing so we are reducing the attenuation effect of the human body on the Industrial Scientific and Medical (ISM) frequency band.

4.3.4 Bluetooth Driver

The Bluetooth device on the host platform requires an available Bluetooth serial port interface service. Our serial port driver can then be used by an application to read the packets that arrive. The driver reassembles the packets of data in order to have the 10 bits of X, Y, Z acceleration data and button states available. The communication is full duplex between the two devices, allowing the host application to remotely switch on sampling of the sensors' output and to select the sensitivity of the sensors, between either the 2g or 6g.

4.3.5 Power Supply

For simplicity a standard Low Drop-Out (LDO) based power supply provides a steady 3.3V to the entire circuit from three AAA rechargeable Ni-MH batteries (see Fig. 8). A more advanced DC to DC converter could be used in order to employ either a single Li-ion rechargeable battery or two Ni-MH batteries. In standby mode (no Bluetooth connection) the circuit takes 20mA. Most of that current is used by the BT interface in wait mode. During operation (active BT data communication) the current drawn by the circuit is 37.1mA and can peak at 48mA. This peak is reached when the RF signal is strongly attenuated either by the close proximity of walls or body parts to the host or device antenna, and by the magnitude of the distance between them.

4.3.6 Performance

Clear acceleration signals are reliably transmitted from the device to any Bluetooth interface connected to it, in a radius of approximately eight metres. The lag time during the acquisition, transmission, and processing of the data is well below 10ms making the direct tilt input a true real-time experience.

The system has shown during tests that it could perform a single session continuously for over 24 hours on one set of batteries.

4.4 Filtering

As explained before, only the DC-free signal should be double integrated. The DC (Direct Current) part of the signal from the accelerometer is filtered out with DC blockers (see Fig. 9 and 11), which remove the two sources of DC offset in accelerometer measurements:

- The accelerometers' 0g output is non-zero, unique and constant for each sensor.
- As soon as an accelerometer is not perfectly horizontal, it will measure the effect of **g**, the Earth's gravitation.

When the attitude of the device varies, the sensors are measuring the effect of the relative movement of **g**. This turns **g** into a non-constant component of the signal. During those transitional states the filters need time to properly cancel the effect of gravitation. It can be seen during transition T1 and T2 on Fig. 10. In our system we are not trying to calculate the trajectory of the device, but only to extract a signature. In that case it is not a problem to imperfectly remove the component of the signal created by the relative movement of **g**.

Proceedings of the 2006 IEEE Symposium on 3D User Interfaces (3DUI'06)
1-4244-0225-5/06 $20.00 © 2006 IEEE

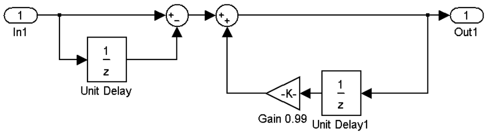

Figure 9: DC blocker block diagram

Our system accommodates hand and arm gestures of between 0.5 and 5 seconds duration. Attitude changes of the device while performing a gesture tend to be repeated based on a user's interpretation. A gesture is described not only by the trajectory of the device but also by its changing attitude, as both contribute to the output of the accelerometers.

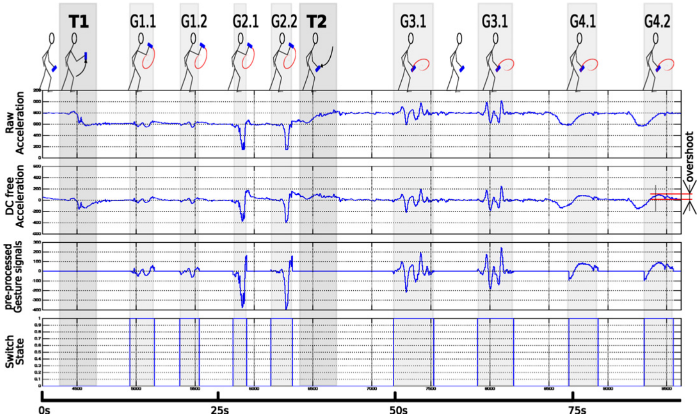

Figure 10: Acceleration signal and filtering

By using a DC blocker with a time constant τ of ~1 second, we make the whole DC cancellation process efficient for most of the gestures we are targeting, while reducing to an acceptable level the time required by the filter to adapt to the new attitude of the device prior to the gesture (see T1 and T2 on Fig. 10).

During long gestures where the acceleration is particularly focused in one direction it is normal for the DC blocker to start correcting the signal by bringing it back to DC. This behaviour can be seen on gesture G4.1 and gesture G4.2 on Fig. 10. In this graph the DC-free acceleration curve displays an overshoot above 0. Those distortions in the signals used for computing the signature are not an issue because they appear every time the same gesture is performed as gesture G4.1 and gesture G4.2 shows.

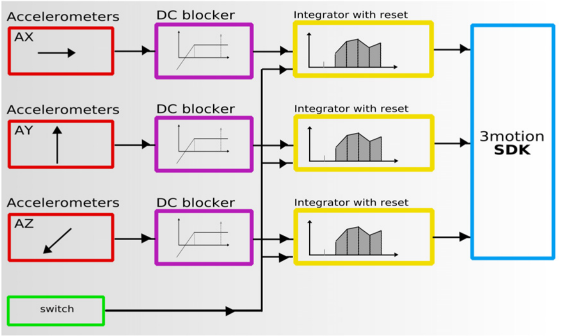

Figure 11: Signal processing block diagram

In the end, the gesture signature is a set of 3D data (see Fig. 4) that can be easily processed by the 3motion™ SDK.

5 SOFTWARE

In this section we describe our algorithm that finds a best match when comparing a gesture trajectory against a list of pre-defined gesture trajectories using a non-parametric curve-matching system. We apply a method originally developed for photogrammetry by Berthold K. P. Horn [9] to the problem of absolute orientation, and for clarity this is described in section 5.2. For each pair of curves, the algorithm calculates the absolute orientation of the first curve relative to the second using section 5.2's non-iterative, closed-form solution. Their translation offset is determined using the difference between the two centroids, while the relative scale between each pair is the ratio of each curve's root-mean-square deviation from its centroid. Using the error between a candidate and transformed library curve as a fit metric allows us then to recognize three-dimensional gestures independently of their orientation, position, and size.

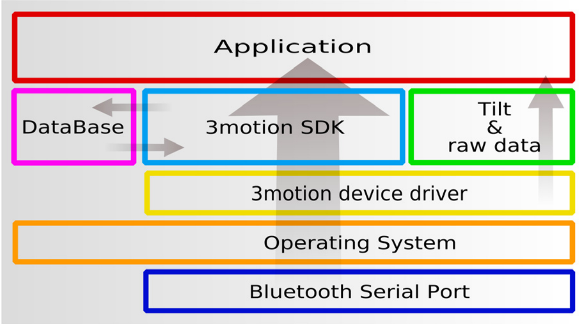

Figure 12: Software layers containing the 3motion SDK

A collection of gestures gathered in advance of the realtime phase, referred to as the gesture library, is prepared to allow useful comparisons. The software expects the stream of gesture data to be time sequential and delimited. In the case of our hardware a delimiter symbol is sent when the primary button is pressed or released. Applications running on the host are configured using our SDK (see Fig. 12) and custom XML gesture schema, and need not use our hardware.

5.1 Resampling

We refer to a series of sampled gesture data as a trajectory curve. Curve A (C_A) represents a trajectory from the gesture library, while curve B (C_B) is a newly performed candidate looking for a match (see Fig. 13). First of all either C_A or C_B is resampled to ensure both curves contain the same number of sampled points. The choice of curve is a configurable option.

5.2 Calculate Transform

Curve A (C_A) and curve B (C_B) now contain the same number of sampled points. Before we can compare the two curves, C_A is first transformed into the same coordinate system as C_B. This transformation is comprised of a translation, scaling and rotation.

5.2.1 Translation using Curve Centroids

The centroids (or barycentres) of each curve, C_A and C_B, are determined by averaging the sum of their sampled coordinate vectors. Subtracting C_A from C_B provides a vector representing the translation between the curves. Both curves then have their positions translated such that their centroids are both at the origin, to C_A^I and C_B^I.

This is a convenient point to obtain expressions of scale. By evaluating the ratio of C_A^I's root-mean-square deviation from the origin, to C_B^I's, the relative size (S_{AB}) of one with respect to the other is found.

An absolute scale value for C_B^I is also calculated (S_{Babs}) by expressing its root-mean-square deviation relative to that of a unit radius sphere. S_{Babs} will be used later to normalize the final match result.

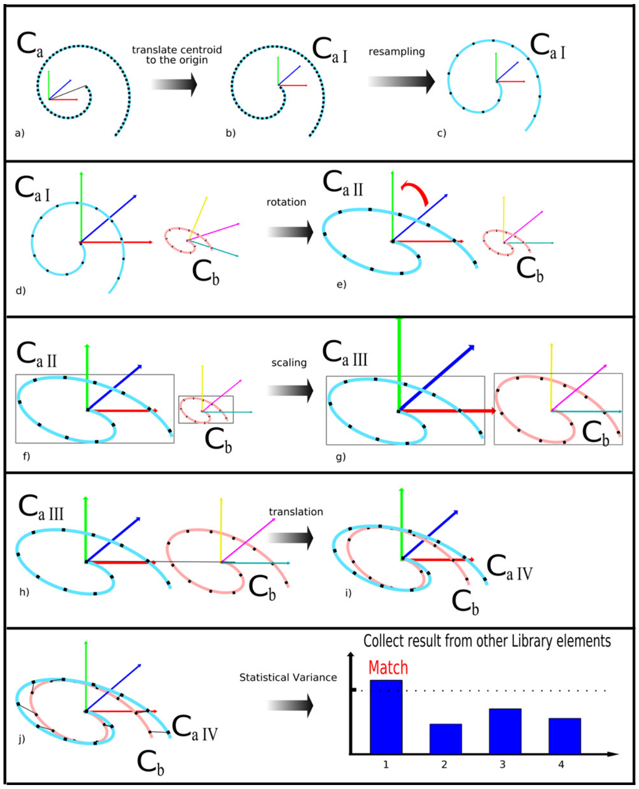

Figure 13: Diagram of the four stages of $C_A^{I\text{-}IV}$

5.2.2 Construct Covariance Matrix

A 3x3 covariance matrix is then built. The process of constructing this matrix M_C begins with a 3x3 zero matrix to which is added the matrix resulting from the tensor multiplication (a.k.a. Kronecker multiplication) of the first vector sample from both C_A^I and C_B^I. This process is repeated, adding to M_C in turn the tensor product of every sample pair from the two curves, resulting in our complete covariance matrix.

5.2.3 Quaternion from Most Positive Eigenvector

The nine elements of M_C are then used to construct a new 4x4 symmetrical matrix M_S from their sums and products,

$$M_C = \begin{bmatrix} A\ B\ C \\ D\ E\ F \\ G\ H\ I \end{bmatrix}$$

$$M_S = \begin{bmatrix} A+E+I & F-H & G-C & B-D \\ F-H & A-E-I & B+D & G+C \\ G-C & B+D & -A+E-I & F+H \\ B-D & G+C & F+H & -A-E+I \end{bmatrix}$$

The four eigenvalues and eigenvectors of M_S are then calculated (using Wild Magic SDK [6]). The eigenvector with the largest positive eigenvalue, taken as a unit quaternion Q, represents a rotation. Rotating curve C_A^I by Q gives curve C_A^{II}. Q is the rotation that minimizes the root-mean-square error between the coordinates of curve C_A^{II} and of curve C_B^I. For a complete derivation please refer to [9].

5.3 Apply Transform to First Curve

Using the obtained values (from 5.2), the transform of curve C_A onto C_B is applied as follows. C_A is first translated to the origin (C_A^I) as an intermediate step to allow scaling and rotation to be applied relative to its centroid. After converting the quaternion Q to a 3x3 rotation matrix M_R, C_A^I is rotated to C_A^{II} by multiplying its sampled vectors with M_R. The lengths of C_A^{II}'s sampled coordinate vectors are then scaled uniformly by S_{AB} to C_A^{III} until their average length is equal to curve C_B. Finally the scaled and rotated curve C_A^{III} is translated to C_A^{IV} a position where its centroid is co-incident with that of the unmodified C_B's.

5.3.1 Least Squares Minimization

The statistical variance between the shapes of our two curves can be expressed as the average of the sum of the squared distances between the sample points in both curves: C_B, and the transformed C_A^{IV}. The standard deviation σ is found by taking the square root of this variance. By dividing σ by the absolute scale of the second curve (S_{Babs} from 5.2.1) a normalized value D_{AB} is found which can compare the geometrical similarity of gesture curve pairs independently of rotation, translation, and scale.

If two idealized curves were tested, one being simply an affine transformation of the other, the resulting D_{AB} value would be zero.

5.4 XML Recognition Configuration

While the C++ API itself can be used to configure application specific recognition settings, it is often more convenient to separate such requirements from the need to compile. Our system uses a proprietary XML schema to serialise an application's gesture configuration and resources. XML was itself chosen for its extensibility, human readability and document validation facility. There follows a brief description of the design methodology.

5.4.1 Gesture Data

Within the XML document, a 'data' child element of the categorical element 'gesture_data' contains the raw data defining an individual gesture: an unbroken sequence of number values that describe the gesture's geometry. At present the C++ SDK can accept two or three-dimensional vector data, and so the count of number values for each set is a product of its vector count with the vector element dimension. Many gesture data sets can be held in such a document, each located by its unique 'id' attribute.

5.4.2 Gesture Configuration

XML child elements of the categorical element 'gesture_config', tagged as 'gesture', define how data described in the 'gesture_data' section mentioned previously should be matched and interpreted. These 'gesture' elements have attributes handling name, unique id, data id, active status, data size and type.

As our recognition algorithm is capable of detecting gestures by their trajectories, regardless of affine transformation, it is often useful to restrict recognition based on orientation, scale and position. Gestures based on one trajectory, but with varying affine transformations, can then each be used to trigger separate events. To this end, child elements of 'gesture' governing deviation, orientation, scale, and position each have an offset, tolerance, and active status attribute.

Of these, the deviation element's tolerance attribute determines if a possible match is close enough to trigger a response. With a constant gesture set, reducing this tolerance value results in fewer, more accurate matches, i.e. badly performed gestures would not be matched.

5.5 Special Cases

If two gestures are simple, the rotation reported may be non-intuitive. For example, a straight-line gesture may be well matched with another in the library; yet considering the rotation as an axis-angle pair, both axis and angle may be substantially different from that expected intuitively. To account for this, the system may be configured to seek a better rotation after the match has been established, or to avoid orientation considerations altogether.

6 Applications

Both to gain feedback from users and illustrate the system's faculty as an entertainment medium, a number of interactive entertainment demonstrators have been created. Of these, three are now described.

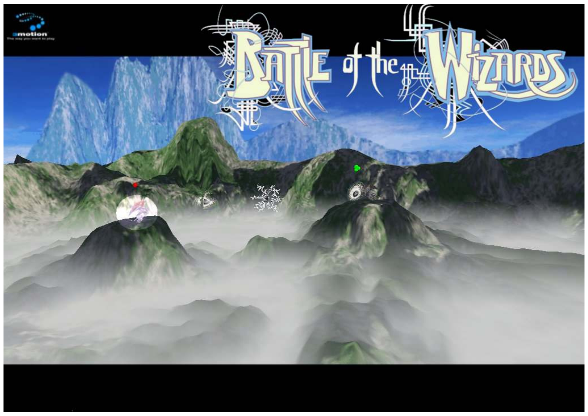

Figure 14-A: Battle of the Wizards

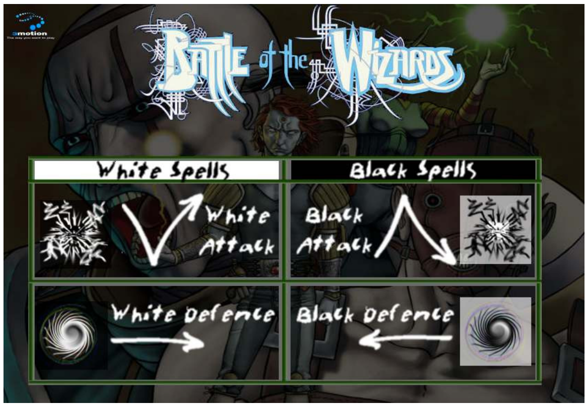

Figure 14-B: Gesture Tutorial

6.1 Battle of the Wizards

"Battle of the Wizards" takes place in a fantasy 3D gaming scenario where two mountaintop-dwelling enemy sorcerers launch magical projectiles at each other (see Fig. 14-A). Each player takes a sorcerer as their avatar and must cast offensive and defensive spells by gesturing an appropriate rune-shape in the air with the wireless controller. Offensive spells impacting with a wizard reduce his health by one unit.

In use, four gestures are used; two for attack and two for defence (see Fig. 14-B). The gestures are pre-defined, geometrical, simple and easily demonstrable. Users seemed to enjoy the gesticulation and drama of the application setting. Once familiar with the system they would become immersed in the escalating mental and physical challenge.

6.2 Character Manipulator

Another demonstration application encourages the use of more natural, personal gestures. Here a bipedal alien character stands ready to respond to a configurable selection of gesture signals (see Fig. 15). One particular movement from the user results in a key being drawn from the alien's pocket, then unlocking a chest. Another gesture causes a lightning bolt to strike.

In this application, user-defined gestures are utilised effectively. One animation shows the character throwing a hammer away. In this instance the user will often perform a similar movement. Gestures that cause external events, like rain, to which the character merely responds, remain more geometrical and abstract through choice. Forehand and backhand gestures make the character play the respective table-tennis shot. Typically around fourteen gestures are used simultaneously, demonstrating construction, sport and adventure contexts.

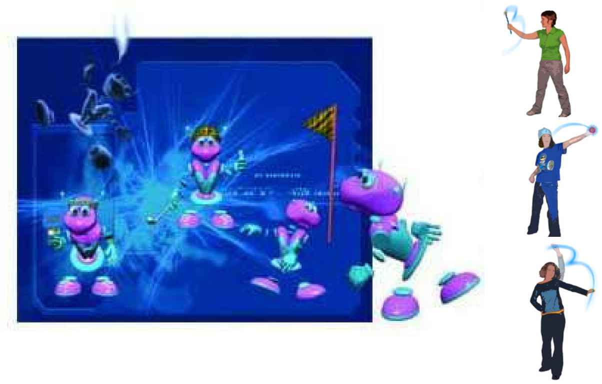

Figure 15: Pink Pod - Natural Gesture character manipulation

6.3 Mobile Phone Golf

After close coupling our device electronics on the back of a Symbian mobile phone, a golf game was created where players swing the phone like a golf club. In this case the Series60 Nokia 6630 phone battery was used as a power source for the 3motion™ remote sensor unit to save space. The 6630's Bluetooth serial port service was used to facilitate realtime communications between the application and our device. The program was written in C++ and uses the GapiDraw graphics library.

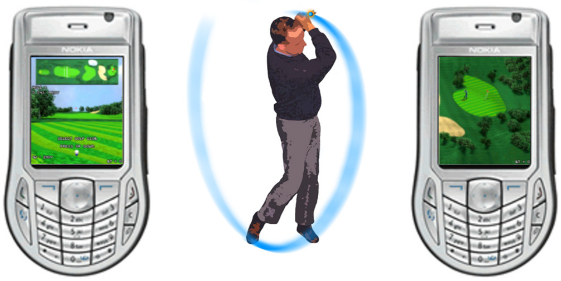

Figure 16: Series-60 mobile phone Golf game

The application's concept and visual appearance follow golf gaming conventions, with the exception that at the point where a well-timed button press normally occurs, the player's swing is substituted (see Fig. 16). Upon swinging, it is the peak in acceleration and deceleration and choice of club, not gesture recognition, which determines how far the ball travels.

7 CONCLUSIONS

This paper presented a novel 3D interaction system consisting of a low-cost, lightweight hardware component and a general-purpose gesture interaction software development kit. The software component is robust enough to reliably recognize a usable set of 3D gestures, yet sufficiently processing efficient and portable to run on low-power mobile devices such as cell phones.

The interaction device was built around a 3-axis linear accelerometer single chip and provides wireless communication facility to a large range of host devices through a standard Bluetooth serial interface. The hardware was optimised for low power consumption and the potential to mass-manufacture for just a few dollars per device.

Over the last year, the system was presented and tested at various occasions, including trade-shows, to potential investors in the gaming industry, and during in-house demonstrations to a wide audience. Both hardware and software have proven very reliable. Users could immediately interact with our demonstrator system with very little introduction required. Especially children were enthusiastic about the new interaction experience, which confirms the suitability for our envisioned main application area, the gaming market. We have also received significant interest to use the 3D gesture paradigm in areas where traditional mouse and keyboard interaction is not suitable, such as in immersive environments or for mobile and wearable computing.

Current development effort now focuses on adding additional data input sources, such as vision-based motion detection, and exploiting accelerometers built into an upcoming generation of mobile phones [19, 24]. The software development kit will be extended to provide continuous gesture recognition through a windowed comparison, doing away with the requirement to press a button to signal a gesture. We are also currently testing a new matching algorithm based directly on the inertial signature, therefore doing away with the double-integration step.

REFERERNCES

[1] A. Benbasat, and J. Paradiso: An Inertial Measurement Framework for Gesture Recognition and Applications, In Ipke Wachsmuth, Timo Sowa (Eds.), "Gesture and Sign Language in Human-Computer Interaction," International Gesture Workshop, GW 2001, London, UK, 2001 Proceedings, Springer-Verlag Berlin, 2002, pp. 9-20

[2] B. Danette Allen, Gary Bishop and Greg Welch, Tracking: Beyond 15 Minutes of Thought, Course 11, SIGGRAPH 2001

[3] J. Eisenstein, S. Ghandeharizadeh, L. Golubchik, C. Shahabi, D. Yan, R. Zimmermann. Device Independence and Extensibility in Gesture Recognition. IEEE Virtual Reality 2003, p. 207

[4] L. Emering, R. Boulic, D. Thalmann: Body Expression in Virtual Environments, Proc. IEEE Int. Workshop on Robot and Human Interaction, RO-MAN'99, ISBN 0-7803-5841-1, Sept. 27-29, Pisa, Italy (1999)

[5] Thomas Fuhrmann, Markus Klein, and Manuel Odendahl: The BlueWand as Interface for Ubiquitous and Wearable Computing Environments, Proceedings of the 5th European Personal Mobile Communications Conference (EPMCC'03), Glasgow, Scotland, 22-25 April, 2003

[6] Geometric Tools, Inc., Wild Magic Version 3, http://www.geometrictools.com/SourceCode.html (2005)

[7] GestureTek Inc., Introduction to Gestpoint, http://www.gesturetek.com/gestpoint/introduction.php (2005)

[8] Hofmann, F.; Hommel, G.: Analyzing Human Gestural Motions using Acceleration Sensors. Progress in gestural interaction: Proceedings Gesture Workshop '96 pp. 39-59, Springer Verlag, 1996

[9] Horn, B.K.P., Closed-Form Solution of Absolute Orientation using Unit Quaternions, Journal of the Optical Society A, Vol. 4, No. 4, pp. 629–642, April 1987

[10] Ming Hsuan Yang, Narendra Ahuja, Mark Tabb: Extraction of 2D Motion Trajectories and Its Application to Hand Gesture Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 24, no. 8, pp. 1061-1074 (2002)

[11] A. Jaimes and J. Liu: Hotspot Components for Gesture-Based Interaction, Interact 2005, Rome, Italy, Sept. 12-14, 2004

[12] Hyun Kang, Chang Woo Lee, Keechul Jung: Recognition-based gesture spotting in video games, Pattern Recognition Letters, v.25 n.15, p.1701-1714 (2004)

[13] K. Van Laerhoven, N. Villar, A. Schmidt, G. Kortuem and H.-W. Gellersen. "Using an Autonomous Cube for Basic Navigation and Input". In Proceedings of ICMI/PUI 2003. ISBN: 1-58113-621-8; ACM Press. Vancouver, Canada. 2003, pp. 203-211

[14] S. Z. Li: Similarity Invariants for 3D Space Curve Matching, Proceedings of the First Asian Conference on Computer Vision, pp. 454-457, Osaka, Japan, November, 1993

[15] Lumsden, J. and Brewster, S.A. A Paradigm Shift: Alternative Interaction Techniques for Use with Mobile & Wearable Devices. In Proceedings of 13th Annual IBM Centers for Advanced Studies Conference CASCON'2003 (Toronto, Canada), Stewart, D.A Ed., pp. 97 - 110 (2003)

[16] Jani Mäntyjärvi, Juha Kela, Panu Korpipää, Sanna Kallio: Enabling fast and effortless customisation in accelerometer based gesture interaction, Proceedings of the 3rd international conference on Mobile and Ubiquitous Multimedia October 2004

[17] Vladimir I. Pavlovic, Rajeev Sharma, Thomas S. Huang: Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review, IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. 19, NO. 7, JULY 1997

[18] Hernandez-Rebollar, J., R. Lindeman and N. Kyriakopoulos: A Multi-Class Pattern Recognition System for Practical Finger Spelling Translation, Proc. Fourth IEEE International Conference on Multimodal Interfaces, IEEE Computer Science Press, 2002, pp. 185-190

[19] Samsung Electronics Co. Ltd., SCH-S310 Press Release, http://tinyurl.com/b4jkv (2005)

[20] S. Strachan, R. Murray-Smith, I. Oakley, J. Ängeslevä, Dynamic Primitives for Gestural Interaction, Mobile Human-Computer Interaction – MobileHCI 2004: 6th International Symposium, Glasgow, UK, September 13-16, 2004. Proceedings. Stephen Brewster, Mark Dunlop (Eds), LNCS 3160, Springer-Verlag, p325-330, 2004

[21] Su, S. A., and Furuta, R., A Logical Hand Device in Virtual Environments, in VIRTUAL REALITY SOFTWARE & TECHNOLOGY: proceedings of the VRST'94 conference, Singapore, August, 1994, pp. 33-42

[22] Sukada, K. and Yasumura, M.: Ubi-Finger: Gesture Input Device for Mobile Use, Proceedings of APCHI 2002, Vol. 1, pp.388-400 (2002)

[23] V-Enable, Inc., V-Enable Turns Phones Into Mice, www.v-enable.com/pr/V-Enable_phones_to_mice.doc (2005)

[24] Vodaphone KK, V603 Press Release, http://www.vodafone.jp/english/release/2005/050131e.pdf