Ubiquitous Game Control

Robert Headon[1] and Rupert Curwen[2]

[1] Laboratory for Communications Engineering
Cambridge University Engineering Department
rph25@eng.cam.ac.uk
[2] AT&T Laboratories Cambridge
rcurwen@uk.research.att.com

Abstract. This paper considers the control of computer games through players interacting with the physical environment around them in a natural and appropriate manner. This is achieved using a sentient computing system. Such a system senses the location, motions, actions and even physiological responses of users. This sensory data can be used to interface and control the game. A good example is 3D first-person games, and we demonstrate a system in which actions in the game are mapped to similar actions in the real world. The aim is to give a more involving experience through the unencumbered ubiquitous control of the game.

1 Introduction

A sentient environment uses appropriate sensing technologies, data processing and supporting middleware to maintain a representation of physical space in a world model, allowing shared perception between computers and people [1]. An example is the sentient floor which ubiquitously senses a person's position, orientation and recognises their physical movements. A sentient floor is the combination of a weight-sensitive floor to sense ground reaction forces, data processing and statistical pattern recognition to give "awareness", and the use of middleware to make this awareness ubiquitous. We consider human movement to be actions such as a step, crouch, jump, rising to stand, whatever, and not conditions based solely on position changes such as "moving forward". Furthermore, these movements can be characterised to more accurately *describe* the movement, e.g. the height of a jump, the pace of a walk, etc.

The sentient computing paradigm depends on sensors of diverse modalities, including biosensors, which can give awareness of the physiological states, both mental and physical, of a person. Useful signals in determining these states are heart and breathing rates, skin conductance (GSR), brain waves (EEG) and muscular activity (EMG). These sensors are worn by the user and connected to a portable acquisition device (e.g. [2,3]). The physiological response of the player during game play can be used to adapt the game, according to their mood, excitement, stress level, etc, tailoring it to maintain captivation. EMG sensors are used to observe muscular activity and can be used to detect movement for explicit control, e.g. detect finger clench.

In a sentient environment, the sensors and their information are ubiquitous. Sensors are dispersed in the environment (e.g. floor) or worn by people (e.g. biosensors). This information is acquired, processed and published using a distributed computing model. This makes the shared awareness accessible to all networked devices, from desktops to handheld computers, facilitating the control of games on these devices using information that is otherwise unavailable.

Section 2 presents related work on movement recognition and other alternative techniques for interfacing with games. Section 3 describes the sentient floor, a sentient system with perception of the player's position and movements. Section 4 outlines our technique for interfacing this new control with existing games. Section 5 describes three applications for which the awareness of position and movement is used to control games. Section 6 concludes with observations on our experience of ubiquitous gaming.

2 Related Work

Player position, motion and physiological state are already control inputs to some games, although not in a ubiquitous manner. Human movement tracking has been an active area of vision research for several years, and should be distinguished from movement *recognition.* An interesting untagged vision system for movement recognition uses motion history images to preform recognition using temporal templates [4]. This method has been used to give perception to environments, e.g. [5, 6]. However, vision systems suffer when occlusion or background movement is present and operate in a limited workspace. Using a weight-sensitive floor it is possible to recognise whole body movements of a person by tracking the position of the force [7] and/or the magnitude of the force [8] over time, the latter being more powerful. Other surfaces can also be used to determine the player's position. Floor mats with nine pressure sensors exist as commercial input devices to some games. When a pressure switch is depressed it gives crude positioning. Another is a trampoline with a magnetic tracker attached to the bottom [9]. These two devices are not capable of movement recognition, neither do they blend into the environment like a weight-sensitive floor can.

The orientation of a control device (e.g. game-pad) can be measured using a micromachined accelerometer. The cursor keys are emulated by tilting the device, which controls direction. This mechanism is used in several commercial game controllers (e.g. [10]), though these are tethered to the games console and are not ubiquitous. More interesting is the Itsy portable computing device [11] that uses tilting of the unit to represent the cursor keys. Although not explicitly designed as a gaming interface, they demonstrate its use for controlling the games of Cat and Mouse, and Doom.

Awareness of the physiological state of the player enables entirely new games, such as Brain Ball [12] where you beat your opponent by being more relaxed, as determined from EEG signals. In traditional games there is an opportunity to use physiological states of the player to adapt game play. This type of interface may not necessarily be used to explicitly control a game, as physiological responses can be difficult to control. For example, heart rate and excitement may be difficult to control when playing a traditional game, but can allow the subtleties of the game to change in response.

3 Sentient Floor

The sentient floor is a combination of sensing, data processing and information sharing in a distributed computing environment. The sensor is a weight-sensitive or *active* floor, the basic sensor being the strain gauge, contained within a load-cell. The data processing consists of information extraction to determine the location of a force on the floor, and pattern recognition to classify movements. These components are "glued" together using middleware which caters for the distributed acquisition and processing of the sensor data, and provides an easy interface to the perception of the environment.

3.1 The Active Floor

The Active Floor[13] was developed by Olivetti and Oracle Research (ORL)[1] in 1995. It provides a mechanism for determining the vertical ground reaction force (GRF) experienced by a floor. The current implementation of the Active Floor

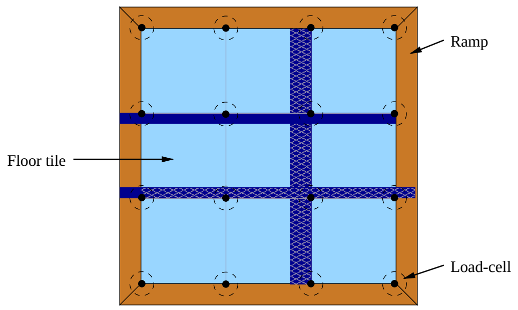

Fig. 1. Plan view of the Active Floor

consists of a four by four array of load-cells with a three by three array of false tiles resting on top (figure 1). Each tile is composed of a steel plate, with a three quarter inch plywood board bonded to it, and dimensionally matches a standard carpet floor tile (i.e. 500mm × 500mm), which are placed on top. The corner of each tile sits on a load-cell. At the corners of the floor the load-cell supports only one tile, and each vertex on the outer edge supports two. The inner load-cells support four tiles each. The floor is robust, with no flexion so it feels like a regular floor.

The Active Floor provides the data necessary to extract important parameters of the environment: user position, tracking, movement recognition and characterisation. Weight-sensitive floors have also been used to identify people by their gait [13, 14]. The active floor currently replaces part of the floor in one room only, and is in everyday use. It is straightforward to extend the coverage.

3.2 Data Processing

The data from the load-cells needs to be processed to determine the position of the player and to generate the feature vectors used to classify movements.

Position: The matrix of load-cells under the floor observe the reaction force at each corner of the tile. Assuming a single force acting on the tile (e.g. one person standing on the tile), the asymmetry of a non-centric load in the reaction forces can be used to determine the position of the centre of mass of this force. This gives a 2D position of the force with a resolution less than 1cm. Tracking is achieved by recording these positions at a desired sample rate.

Movement Recognition and Characterisation: The GRF generated by human movements can be classified using pattern recognition, in particular hidden Markov models (HMMs) [8]. For each movement a pattern class exists that defines the movement. These patterns can be joined in sequence to create more complex movements, a technique known as syntactic pattern recognition. Movements can also be characterised, extracting key parameters of the movement, for example the

[1] now AT&T Laboratories Cambridge

height of a jump. This characterisation allows games to respond more faithfully to movements.

3.3 Middleware

The middleware layer enables all networked applications to access the awareness of the sentient floor via distributed computing. The sensor bus of the Active Floor is extended using a CORBA interface to the data acquisition device. The data processing services connect to the sensor source and also present their information using a "pull" interface for the position and use the notification service [15] to send movement events to interested clients. The middleware layer allows distributed access to the information from the floor, which can be used as a control input to suitable games.

4 Interface with Games

The physical position and movements of the player are novel control inputs, and few existing games are designed to use this information. However, many games are suited to such an interface; consider the control of a character in a 3D first-person game, where the player's movements can be mapped almost directly to those of the character. Most games are closed-source, perhaps becoming open-source several years after release, so it is generally not easy to incorporate new input modalities. A generic technique is required for a simple interface to all existing games.

Most games have a small command set with actions being triggered through the use of core input devices such as a keyboard or mouse, or by extended controls such as a joystick. To include our new awareness as a control device to existing games we have created *virtual* input devices in software, such as a virtual keyboard and mouse. These programs enable the simulation of key strikes, mouse motion and button press events to the window manager. This layer of abstraction allows one-way communication between the controller that uses awareness and the game, in an extremely easy and reusable manner.

The implementation of a game controller that uses position and movement information is extremely simple, and often generic. The command set of the game must be defined, which maps the control and the corresponding key, e.g. walk left corresponds to a left cursor key press, etc. This mapping identifies which event should be generated (via a virtual keyboard) for a desired action. Then a mapping between the player's context, obtained from the sentient model, and this command set is specified. The controller receives position and movement updates and uses the mappings to generate the necessary events, thus controlling the game (figure 2). For many games rate of progression is tri-state (stopped, walk, run). When a key is depressed, the keyboard generates continuous events to the window manager. Using a virtual keyboard it is possible to vary the key-event interval and thus control the rate of key strikes. This gives control of character motion velocity within the game.

Simulating input device events is the easiest way to interface with existing games, and add the awareness of player position and movement. A drawback of this interface is the one-way communication with the game which precludes feedback to the controller. For example, the control has no knowledge of the position of the character in the game, which limits the interface. Ideally games should have an open interface from which controllers can query the state of the game and player. We have created virtual input devices for X-based clients, and the same technique will work equally well with non-X windowing systems, even on hand-held devices.

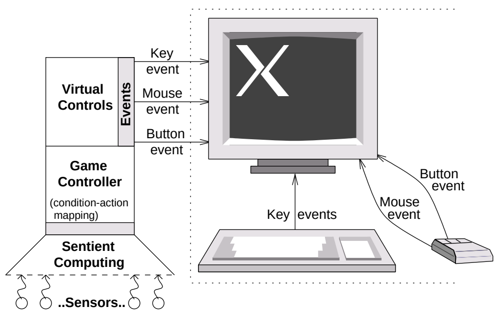

Fig. 2. Virtual Controls

5 Applications

To demonstrate the use of physical position and movement recognition as entertaining ubiquitous controls for games we consider three applications. The first maps the player's position to the cursor keys, which are common controls in many existing games. The second describes a new game in which the position of the user is again relevant, but on a larger scale. This game is more ubiquitous in that it can be played without a display using auditory feedback alone. The third application describes the development of a control interface for a 3D game, in this instance Quake™. The control uses position and movement information to control the character on screen.

5.1 Cursor key emulation

Many games use the cursor keys to control game play. In this case the virtual input device can map the physical position of the player into these cursor key-events. The four cursor keys are represented by regions defined on the floor. When the player's centre of mass is contained within a region the corresponding key strike is generated. These regions can be defined anywhere, and overlap, thus making the control ubiquitous. The regions could be defined as areas on a single tile, on four adjacent tiles, or any other suitable configuration. The regions could even be defined relative to the player's current position thus enabling a "follow me" control, always being with the player as they move. This allows a very simple and effective interface for the basic directional control of many existing games. As the control is abstracted from the game, no modification is required for other games with the same control set. We have used this interface to control games such as Snake and similar.

5.2 Smash Game

It was desirable to develop a new game, more ubiquitous in nature, with inherent use of the player's position. The game is akin to arcade games where a hidden object pops up in one of several locations and the player must hit it with a soft hammer to score points. Each floor tile corresponds to a possible pop-up location, and the player scores by landing on the relevant tile. Feedback is both visual and auditory. A display shows the nine floor tiles, and icons over-lay these to indicate on which tile the pop-up has occurred. This is accompanied by the tile number being announced. The position of the next popup can be given relative to the player's current position and orientation, e.g. two tiles behind. Upon a successful hit the icon is overlaid with a footprint, which is accompanied by a squishing noise to give feedback to a player not facing the display. The game [16] is implemented as a Java applet and can be played with either a high resolution location system (e.g. sentient floor, Active Bat[17]) or a mouse.

5.3 Controlling Quake™with real movements

Awareness of human movements and position can be used to directly control the movement of a character in a virtual environment. In this paper we use the game Quake™to demonstrate this control. Although earlier versions of this game are now open-source and it is possible hook movement recognition directly into the game, we use virtual controls (section 4) to illustrate the ease at which 3D first-person games can be controlled. Firstly the control set is defined. It is necessary to control direction ($\times 8$), rate of progression (0-100%), jumping and shooting, each of which have a corresponding key-event.

The direction can be forward, backward, left, right and combinations. This is determined by the location of the player on the floor by defining regions similar to that of cursor control(§4), with example regions indicated in figure 3. These regions can be set during a calibration phase anywhere on the floor. In the configuration shown, three tiles are used, with the ones on the left and right representing those directions correspondingly. A more natural approach to turning is desired, whereby a turn to face another direction results in the on-screen character turning too. However the player is constrained to view the display at all times and this inhibits playability (see §6 for discussion). Progression is achieved by on-the-spot stepping

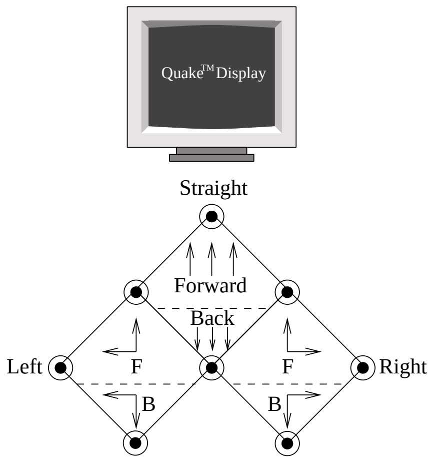

Fig. 3. Plan view of selected floor tiles with markings indicating the regions used in determining direction

on any of the defined progression regions. The frequency of steps is used to control the speed at which the character moves, if at all. Using the simulated key strikes it is possible to control progression over a range of speeds rather than the two state (walk or run) model exhibited when playing Quake™with a keyboard. When the player jumps, the online movement recogniser observes this and notifies interested clients, one of which is the virtual Quake™control. This then sends a jump event to the game and the character jumps on-screen.

The interface is incomplete without being able to control shooting through movement. It is possible to define another region or movement to activate shooting, though a more natural interface is to react to the retraction of a finger to simulate a trigger pull. Finger movements are monitored by carefully placing an EMG sensor on the forearm. When the player retracts their shooting finger the activity of the muscle is observed, and results in a shoot action if it exceeds a threshold.

6 Conclusion

This paper describes a ubiquitous sensing technology that determines a person's position and movement, which can be used to control games. This is illustrated by three applications, two considering the control of existing games and also a new game which has inherent use for the position of the player in physical space. A generic mechanism for interfacing with existing games is also described.

We observe that the static location of a game display limits the range of position and movements a player can make. When playing Quake™, for example, it is natural for the player to turn and face another direction and expect the game character to do so too. With a static display this is not feasible. We are posed with several alternatives. Options include head-mounted, hand-held, or projector displays. The "everywhere" projector [18] can change the position of its display using a rotating mirror. This is an interesting option as the display is available to everyone, the player is unencumbered with wearable devices, and the display is large. However, most real-world surfaces don't make particularly good projection screens. The size and manner of the interface has grown enormously, and the display technology must be sympathetic to this.

The use of movement awareness can be extend by characterisation. By doing so, we can extract parameters such as the height and distance of a jump, the interval between footfalls, whatever. Certain games, e.g. track and field style, are suited to this method of interaction. Such games are particularly suited to multi-player instances, adding the important dimension of competition. Multiple players are supported by each defining their own operating regions on the floor.

Adding physical-world perception to the game allows exciting new interactivity, making the play more natural and compelling by integrating perceptive capabilities to the interface.

References

1. Andy Hopper. The Clifford Paterson Lecture, 1999 Sentient Computing. *Phil. Trans. R. Soc. Lond.*, A (2000)(358):2349–2358, 2000.
2. Thought Technology's ProComp. http://www.thoughttechnology.com/procomp.htm.
3. Digital angel - technical data. http://www.digitalangel.net/da/tech.htm.
4. G Bradski and J Davis. Motion Segmentation and Pose Recognition with Motion History Gradients. In *WACV*, pages 238–244, 2000.
5. Aaron Bobick et al. The KidsRoom: A Perceptually-Based Interactive and Immersive Story Environment. *PRESENCE: Teleoperators and Virtual Environments*, 8(4):367–391, August 1999.
6. James W. Davis and Aaron F. Bobick. Virtual PAT: A Virtual Personal Aerobics Trainer. In *Workshop on Perceptive User Interfaces*, San Francisco, California, November 4-6 1998.
7. Jaana Leikas, Antti Väätänen, and Veli-Pekka Räty. Virtual Space Computer Games with a Floor Sensor Control: Human centred approach in the design process. In *Workshop on Haptic Human-Computer Interaction*, pages 119–122, University of Glasgow, 31 August - 1 September 2000.
8. Robert Headon and Rupert Curwen. Recognizing Movements from the Ground Reaction Force. In *Workshop on Perceptive User Interfaces*, 15-16 November 2001. To appear.
9. Pär Hansson, Anders Wallberg, and Kristian Simsarian. Techniques for "natural" interaction in multi-user CAVE-like environments, 1997. ECSCW'97 Poster Description.
10. Microsoft freestylepro. http://www.microsoft.com/hardware/sidewinder/devices/FSpro/.
11. Joel F. Bartlett. Rock 'n' Scroll is Here to Stay. Research Report 2000/3, Western Research Laboratory, Compaq, Palo Alto, CA, USA, May 2000.
12. Sara Ilstedt Hjelm and Carolina Browall. Brainball - using brain activity for cool competition. Demo at NordiCHI 2000, Stockholm, Sweeden.

13. M.D. Addlesee et al. The ORL Active Floor. *IEEE Personal Communications*, 4(5):35–41, October 1997.
14. Robert J. Orr and Gregory D. Abowd. The Smart Floor: A Mechanism for Natural User Identification and Tracking. In *Proceedings of the 2000 Conference on Human Factors in Computing Systems*, The Hague, Netherlands, 1-6 April 2000.
15. Object Management Group. Notification Service Specification, June 2000. http://cgi.omg.org/cgi-bin/doc?formal/00-06-20.ps.
16. Smash Game. http://www-lce.eng.cam.ac.uk/̃rph25/floor/SmashGame.html.
17. Andy Ward, Alan Jones, and Andy Hopper. New Location Technique for the Active Office. *IEEE Personal Communications*, pages 42–47, October 1997.
18. Claudio Pinhanez. Augmenting Reality with Projected Interactive Displays. In *Proc. of International Symposium on Virtual and Augmented Architecture (VAA '01)*, Dublin, Ireland, 2001.