Feedback, Latency, Accuracy: Exploring Tradeoffs in Location-Aware Gaming

$^{\diamondsuit}$Kieran Mansley $^{\spadesuit}$David Scott $^{\clubsuit}$Alastair Tse $^{\heartsuit}$Anil Madhavapeddy

$^{\diamondsuit\spadesuit\clubsuit}$Laboratory for Communication Engineering
15 JJ Thomson Avenue
Cambridge CB3 0FD, UK

$^{\heartsuit}$Intel Research Cambridge
15 JJ Thomson Avenue
Cambridge CB3 0FD, UK

{$^{\diamondsuit}$kjm25,$^{\spadesuit}$djs55,$^{\clubsuit}$acnt2}@cam.ac.uk

$^{\heartsuit}$avsm2@cl.cam.ac.uk

ABSTRACT

We are witnessing the development of large-scale location systems and a corresponding rise in the popularity of location-aware applications, especially games. Traditional computer games have pushed the limits of CPU and graphics card performance for many years and experience suggests that location-aware games will place similar demands upon location systems. Unlike traditional gaming platforms however, the mobile devices that interact with location systems are heavily constrained especially in the number of ways that feedback can be provided.

In this paper we describe a location-aware, fast-paced, close quarters action game and use it to experiment with three key components of future location-aware gaming platforms: *(i)* the location system, *(ii)* the network to connect the mobile devices, and *(iii)* the feedback and computational capabilities of the mobile devices themselves.

We investigate the tradeoffs that are possible between these components, the effect of the feedback channel and the suitability of Bluetooth as a network for mobile game devices.

1. INTRODUCTION

1.1 Location-Aware Games

Children[1] have traditionally enjoyed running around in groups playing games such as "hide-and-seek", "musical chairs", or simply shooting at each other with water pistols. In recent years, multi-player computer games have replaced some of these physical activities with networked interaction from a desktop computer or gaming console.

Location-aware technologies have begun to be deployed commercially, and integrated with popular mobile devices. For example, the Federal Communications Commission has mandated that all new mobile phone handsets sold in the United States have Automatic Location Identification capability [9] in order to allow emergency services to quickly locate accident victims. This has led to manufacturers integrating technologies such as Cell Identification, Global Positioning System (GPS) [13], and Assisted GPS [8] into their handsets.

[1] and PhD students

Location-aware gaming aims to take advantage of these developments and combine the social face-to-face aspects of traditional games with the rich complexity that networked computer games provide.

There have recently been a number of projects taking advantage of the presence of GPS integration in mobile phone handsets (see Section 5). Since GPS only works reliably outdoors, these games generally take place in a wide-area urban setting. In this paper, we seek to prototype close quarters, low latency, high accuracy, location-aware games using the current generation of mobile phone hardware and the Active Bat system (see Section 2).

Our novel contributions include: *(i)* creating a fast-paced, close quarters, location-aware game, *(ii)* exploring the tradeoffs between the accuracy of a location system, the I/O capabilities of current mobile hardware, and the latency of user feedback, and *(iii)* investigating the viability of Bluetooth as a component in a low-latency location-aware gaming infrastructure.

We constrain our choice of gaming hardware to standard mobile phones and only use the Active Bat system to simulate what future off-the-shelf location systems may provide. This enables us to explore a new breed of games that will arise when advanced location technologies are integrated into next-generation mobile handsets.

The rest of this paper is structured as follows: Section 2 describes a flexible architecture for creating mobile, location-aware action games while Section 3 describes several game configurations we deployed for testing. Section 4 presents some experimental results and discusses the pros and cons of Bluetooth as a network for mobile games. Section 5 presents some related work and Section 6 concludes.

2. GAME SYSTEM ARCHITECTURE

A mobile, location-aware, gaming platform will have a number of key components: *(i)* the location system, *(ii)* the network to connect to the mobile devices, and *(iii)* the feedback and computational capabilities of the mobile devices. As there is no current consensus on which technologies will eventually be deployed in

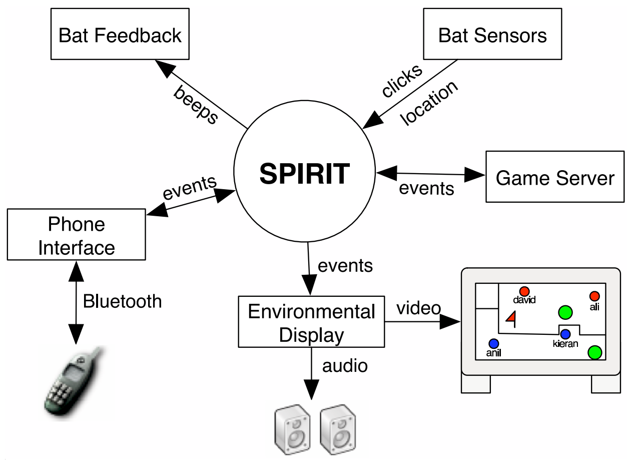

Figure 1: Architecture of the location-aware gaming platform

each of these roles, we have created an architecture that allows us to experiment with the properties of each component, determine what the requirements of each component are and what tradeoffs are involved.

2.1 The Location System

The Active Bat [11] system is a sophisticated indoor location system in which small battery powered tags ("Bats") are signalled periodically by radio to emit narrowband ultrasound pulses. These pulses are received by a network of sensors embedded in the ceiling and the time-of-flight information is used to multi-laterate the position of the Bats – a process which is accurate to about 3cm 95% of the time. Each Bat also contains a pair of push-buttons, a pair of status LEDs and a buzzer for basic I/O.

The Active Bat system can trigger 50 location updates per second per radio zone; there is one such zone in the current installation[2]. The system adaptively schedules Bat updates and offers highly mobile Bats a higher Quality of Service by triggering them more frequently. Applications are also able to signal the scheduler to request more frequent updates for particular Bats for limited periods of time.

The Active Bat system was deliberately over-engineered to provide higher update rates and levels of accuracy than are likely to be found in any deployed commercial location system in the near future. Although the Bat System itself is unrealistic, it is nevertheless an ideal experimental platform for determining the properties *future* location systems must have in order to support fun location-aware games.

2.2 Game Event Distribution

We use the SPIRIT [1] spatial indexing middleware layer to take raw location events from the Active Bat system and convert them to more useful events such as *"Player X has entered Room Y"*. This is augmented by a game server which listens for SPIRIT events, and determines their impact on the game. It in turn issues (through SPIRIT) game events such as *"Player X has picked up a flag"*. This is illustrated in Figure 1.

A number of feedback components can connect to the SPIRIT middleware to receive the game events, and so augment the game.

[2]A previous deployment in AT&T Laboratories Cambridge had three separate radio zones – one for each floor of the building.

As each of these components is fully independent, we can add and remove them to experiment with different methods of feedback. These components include: *(i)* triggers to feedback through the Active Bat, *(ii)* displays of the game map, *(iii)* sound servers, and *(iv)* a Bluetooth proxy to allow mobile phones in the vicinity to receive game events.

In the following section we use combinations of these components to simulate a series of mobile gaming platforms with different feedback, location-accuracy and latency properties.

3. LOCATION-AWARE GAMING

The games we have chosen to evaluate are based on the popular *Capture the Flag* and *Counter-Strike* PC games. Each game has two teams, and each team has a "base" positioned somewhere within the building. The teams compete to either capture a flag from the opponents base and return it to their base, or plant a bomb at the opponents base. To thwart the other team each player is equipped with weapons, including land-mines and a shotgun.

As a minimum each player must carry an Active Bat in order to participate. This device provides both player location information and allows the player to perform actions in the game. In the system described here game actions are invoked by pressing one of the two push buttons on each Bat. The first button is used to fire weapons; the choice of weapon is indicated using *gestures*. Landmines are laid by clicking near the floor, while the shotgun can be fired by clicking twice (to determine direction) at chest level. The second Bat button is used to request a status update (giving information about the player's health and ammunition levels).

Flags are picked up and dropped automatically simply by being in the right place at the right time. It does not require an explicit request from the player.

All objects in the game (flags, bombs, land-mines, shotguns etc.) are entirely virtual and have no physical representation. This allows for a very flexible deployment of the game (you can play it anywhere you have access to the location system) but removes a key feedback component from traditional games: the ability to see where things are.

Three configurations of the system were created in order to investigate what levels of feedback, accuracy and communication latency were required in order to ensure an enjoyable game. Each configuration was extensively play-tested while parameters — such as the Bluetooth communications latency — were varied. After playing the games an informal survey assessed how enjoyable the games were, and what the major problems or limiting factors might be. Each configuration is described in detail in the following sections.

3.1 Feedback through the Bat buzzer

The most basic mechanism for providing feedback to the player simply utilises the I/O capabilities of the Active Bat itself. Bats are extremely minimalistic devices, possessing only the most rudimentary forms of I/O: a buzzer and a pair of push-buttons. Active Bats are therefore good prototypes of ultra-cheap consumer location devices of the future.

The buzzer on each Bat can be remotely triggered to play one of a set of pre-programmed simple tunes. By associating different tunes with key game events we are able to convey approximations of the game state to the player. There is an inherent tradeoff between signalling every event (possibly overloading both the user and the Bat with lots of trivial events) and signalling only selected events (which risks leaving out a critical event and confusing the user). We found the following set of events to represent a usable compromise: *(i)* object picked up, *(ii)* object dropped, *(iii)* gun fired: hit *(iv)* gun fired: missed, and *(v)* you have died. The user can also request their current status, and this is again conveyed using tones on the Bat: either "you are dead", "you are alive and have the flag", or "you are alive and have *N* spare mines".

This configuration, while primitive, could be seen to represent one possible commercial system. The mobile device is extremely small, low power, and has no expensive components such as a CPU or screen. It operates very much as a *thin client*. However, the extreme limitations of the mobile device required greater sophistication in other components to compensate. For example, the location system must be able to support gestures in order to distinguish different weapons. The very low bandwidth of the feedback channel emphasises the need for low-latency and jitter in order to guarantee timely (and hence meaningful) feedback to the user.

In practice, this game configuration enjoyed only partial success. The use of gestures (made possible by the high-accuracy of the location system) was indeed successful in mitigating problems caused by the limited number of buttons on the Bat. Despite this, the game was found overall to be confusing by the game participants. This was attributed to two factors: firstly, fast action games like *Capture the Flag* often generate events at a rate faster than we could remotely trigger the buzzers on the Bats, even when being very selective about which events are transmitted (consider a gun battle where each shot is associated with a beeping sequence). Secondly, when a beep is late there is no way for the player to tell if the event was missed completely (i.e. they should repeat the command/gesture) or whether the game is experiencing temporary lag (e.g. if the Bat radio channel is currently congested and the event notification signal has been delayed).

3.2 Sound Servers and Ambient Displays

In contrast to the first configuration which used only the buzzer on the Active Bats for feedback, this configuration adds a small number of "sound servers" and "ambient displays" to enhance the game.

The game events can be divided into two categories: *(i)* those that apply to an individual player (e.g. "You are dead"), and *(ii)* those that apply to all players (e.g. "Team A has captured Team B's flag").

The simple mobile devices are personal, and so naturally lend themselves to the "individual" events. They can be triggered so that players do not hear the events that only apply to others[3].

This leaves the game events that apply to many or all players to be broadcast by a different channel. We have augmented the simple (single feedback channel) above by adding "sound servers" to convey the events that relate to all players. These connect to the SPIRIT middleware (as shown in Figure 1) from any PC in the building, and will then broadcast suitable sound effects through the PC speakers when they receive game events. These sound effects are much more meaningful than the simple beeps. This greatly reduced confusion, and made it much easier for people who had not played the game before to participate.

An additional "ambient display" component was provided which consisted of a PC near each team's base displaying a continuously updated map of the virtual world. Due to the inherent immobility

[3]This is greatly assisted by the mobile devices using earphones — without them it can be difficult to distinguish tones from your device from tones from the devices of other nearby players

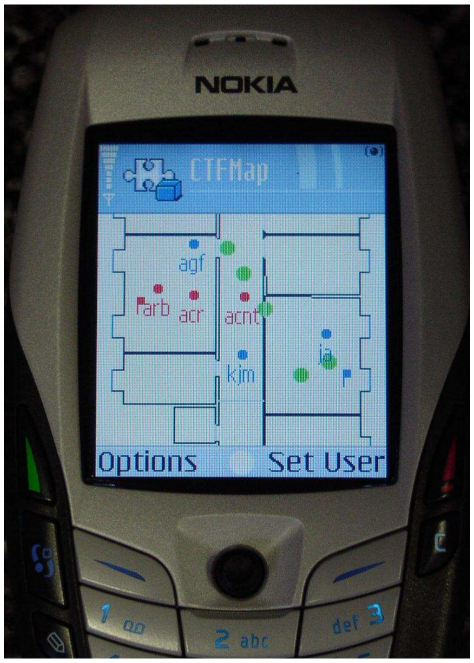

Figure 2: The game map displayed on a mobile device
The six small initialled circles are the game players. The larger (green) blobs are mines that have been laid. The flags are both currently in their bases at either side of the map.

of these components they do not broadcast events to players "in the field" in the way the sound servers do, but instead allow them to retreat to their base to gain a view of the entire current game state. As the game objects do not have any physical representation this is the only way that players can view where land-mines (for example) are.

This game configuration proved to be much less confusing than the first configuration, mainly because the players knew they could always retreat to their bases to examine the state of the game on the ambient displays. Additionally, the use of the sound servers to broadcast game-wide events kept everyone very much "in the loop" and reduced the burden placed on the limited Bat buzzer feedback channel.

3.3 Adding Mobile Phones

Mobile phones are becoming ever more powerful and now incorporate colour screens, cameras, sound capabilities, and networking in the form of GPRS and Bluetooth.[4] It is likely that future phones will also incorporate some form of location-sensing technology. Such devices would make an attractive platform for running mobile location-aware games.

With this in mind, the third game configuration gave players a mobile phone in addition to their Active Bat. The sound server and map display components from the above setup and are then run on the phones. A Bluetooth proxy bridges between SPIRIT and the phones, transmitting the relevant game events. This gives those players with phones a detailed map of their surrounding area (Figure 2).

The inclusion of the mobile phones into the architecture sig-

[4]We used a number of Nokia 6600 phones for our experiments.

nificantly changed the game dynamics. Those users with phones tended to act as the eyes and ears of their team, directing their team-mates, and coordinating play. The capabilities of the phone rendered the fixed map display and sound server redundant, allowing for a much more ad-hoc game (assuming a omnipresent network connection and location system). Interestingly we found that the game became *less* social if everyone had their own phone because then the players did not have to collaborate explicitly with their team-mates. It seems that using less hardware made the game more social.[5]

The increased feedback to the players with phones also reduced the requirements on the other components in the system. It could, for example, tolerate more latency before becoming confusing. Weapon selection could also easily be done on the phone, removing the need for gesture support and tolerating lower location-accuracy. Increased feedback did however make increased demands of (and expose the critical role of) the network connection to the mobile device. This is discussed in detail in the following section.

4. BLUETOOTH

Bluetooth [5] is a low-power short-range wireless communication protocol designed for connecting peripherals. Devices conforming to the Bluetooth specification operate on the 2.4GHz ISM radio frequency band using frequency hopping spread spectrum. Physical channels are defined by a frequency hopping code and devices that are sharing this code cooperate in a time-division duplex (TDD) scheme. These devices using the same physical channel form a Bluetooth *piconet* which contains a single master and one or more slaves.

Bluetooth is the natural choice for providing location-aware gaming communications channel as it *(i)* exhibits low power consumption, *(ii)* low latency operation, *(iii)* has reasonable transmission range[6] and *(iv)* is widely implemented on portable commodity devices such as mobile phones, PDAs, USB adapters and notebook computers.

Other networking solutions were also considered. GPRS proved inadequate because of the high latency [6] and high monetary cost of data transfer. 802.11 wireless LAN provides great coverage, bit-rate and response times. However, 802.11 devices are generally require more power and are less portable than Bluetooth devices, making them difficult to incorporate in fast-paced location-aware gaming.

Our experiments with fast-paced action games exposed some limitations of Bluetooth which should be considered if one were to utilise it for future location-based gaming.

4.1 Coverage

The gaming area is dictated by two constraints: the coverage of both the location system and the communication network. Despite the rated 100m outdoor range for class 1 Bluetooth devices, we found (as expected) the indoor coverage to be significantly less. Our experiments show devices more than 20 metres away from the class 1 Bluetooth transmitter in a typical indoor office environment could not reliably maintain a usable connection.

By using the location system, we were able to make detailed signal strength coverage maps to survey the areas of acceptable reception. Figure 3 shows more than 30000 samples obtained by

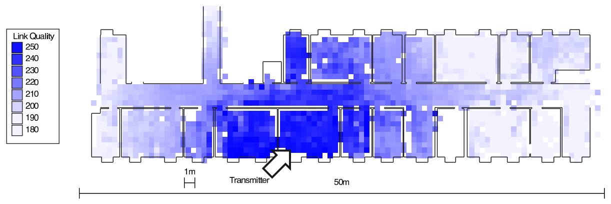

Figure 3: Bluetooth signal strength in an office environment
In this map the darker colours represent stronger signal strength. The occasional white spots in an otherwise dark area are due to inability to collect data because of obstructions (e.g. furniture) in the physical environment. (Grid size: 0.5mx0.5m)

querying the master of a piconet for the perceived link quality of a connection to a single slave, plotted against spatial location as reported by the Active Bat system.

The results shows that a single class 1 Bluetooth device is not sufficient to cover the same area as the Active Bat system. At locations where coverage is weak, the game client will experience degraded performance that manifests as increased latency (*lag*) due to retransmissions over the Bluetooth ACL[7] connection. We could overcome this by using high powered antennas or managing handovers between multiple Bluetooth devices.

Handovers are not well supported under the Bluetooth specification as it was originally only designed as a cable replacement technology [2]. When considering making a handover, a client or server must decide itself when it is appropriate to switch zones. Under 802.11, such a decision is made on the basis of signal strength. Under Bluetooth, obtaining signal strength requires an established connection. To form connections, the client would have to regularly perform an inquiry scan to discover reachable servers. This inquiry scan would introduce detrimental latency for the existing connections. However Bluetooth connections do remain open when a device is out of range for a period of time.

4.2 Latency

Real-time feedback is important for many games. Special attention must be given to the latency of the Bluetooth network when supporting multiple clients. Typically, Round Trip Times (RTT) between the server and the game client range between 20-40ms with an average of 34ms. Experiments were conducted to determine the effects on latency when supporting more than one client on the network.

Figure 4 shows ping times between the game server (using the Linux 2.4.25 BlueZ Bluetooth stack) and client (on a Nokia 6600). At two points in the experiment we introduce a second client (another Nokia 6600), which performs a file transfer from the server. The first file transfer is done using the same Bluetooth interface as the ping measurements, while the second transfer is done using a different Bluetooth interface. There is a marked degradation in the response time of the ping connection when it is sharing an interface. In both transfers, there is an initial jump in ping times that corresponds to the establishment of a new connection (known as *paging*). For the shared interface scenario, response times increase up to 1120ms compared to just an increase of up to 200ms in the separate interface scenario.

This degradation would appear to be due to problems schedul-

[5]In the same way that "hide-and-seek" requires less hardware in that it does not require the purchase of a games console.
[6]100m outdoors with class 1 devices

[7]Asynchronous Connection-orientated Link

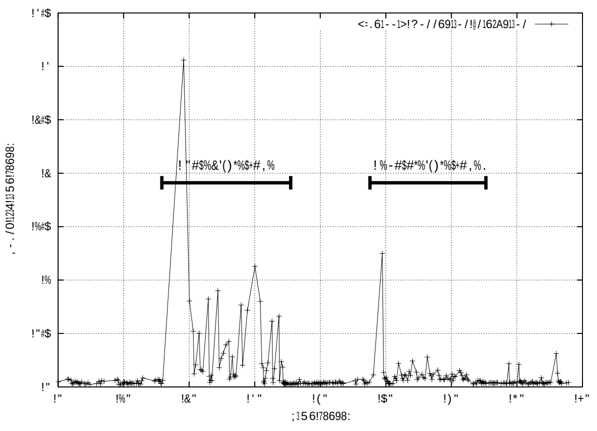

Figure 4: Round trip times under single and multi user usage

ing two connections within the Linux 2.4.25 BlueZ stack rather than a general issue. Performing similar experiments using FreeBSD 5.2.1 with the same Bluetooth adapter, the server did not show the same degradation in ping time. This is demonstrated in Figure 5.

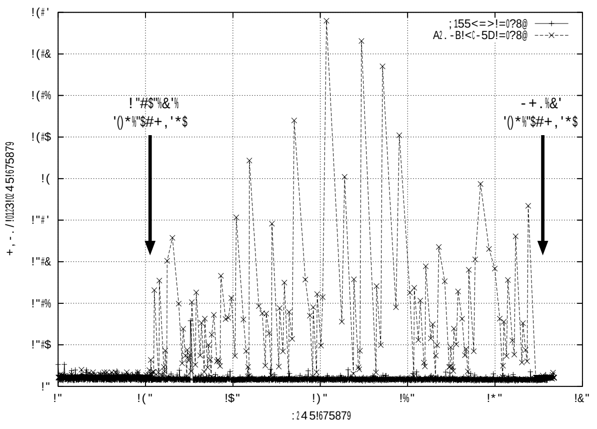

Figure 5: Round trip times on different Bluetooth stacks

These results suggest that in order to achieve acceptable latency, some currently available Bluetooth implementations[8] require separate interfaces for each game client. Furthermore, some Bluetooth implementations cannot support a large number of game clients without significant degradation of response times. This can be attributed to Bluetooth's intended use for networking peripherals where in many cases contention is unlikely. These observations may be relevant to developers and users of latency sensitive equipment such as Bluetooth headsets.[9]

4.3 Other Limitations

Our game does not require communications between clients since the mobile phone is simply acting as an extended display mechanism. If inter-client communication is required, the clients need to be capable of establishing two or more Bluetooth connections. In our experiments, we found that Nokia 6600 phones were incapable of reliably establishing more than two Bluetooth connections.

It is prudent to take into account Bluetooth's relatively low data rates compared to other networking technologies. The maximum bit rate is 1Mb/s with up to 725kb/s data rate available. The highest data rate demanded by each client in our experiments was 24kb/s, suggesting Bluetooth is capable of supporting many clients. Game designers should, however, take this limit into consideration when utilising such a communication channel.

Despite the described shortcomings with the Bluetooth specification and implementation, it is still a practical networking protocol given its cost, simplicity and relatively low latency. Games with more players demand a larger coverage area; however for small numbers of players (up to 4), the ideal game area size does not exceed the Bluetooth coverage area.

5. RELATED WORK

The Global Positioning System (GPS) [13] is the most widely deployed location system in use today. It requires a lock from at least four satellite signals to determine three-dimensional location (accurate to around 30m horizontally). However, a receiver can take a few seconds to report an accurate result, and the system fails to operate effectively indoors.

Despite these short-comings, GPS has proved a popular platform for wide-area location-aware gaming. *Geocaching* [20] allows players to share the physical location of caches on the Internet, and use GPS receivers to find and retrieve the contents of other players' caches. *Can You See Me Now?* [10] was a mobile mixed-reality game played on a city-wide scale. Up to twenty players on the Internet were chased across a map of a city by three performers running through the streets. *Human Pacman* [7] upgrades the classic Pacman arcade game with elements from ubiquitous computing, tangible user interfaces and augmented reality. Players wear headsets displaying their state in the game, and are tracked via GPS as they move around the Pacman grid. Physical objects act as "virtual cookies" that players pick up and interact with. In countries such as Japan, Singapore and Hong Kong where mobile handsets already support wide-area location information, games such as "Mogi, item hunt" [16], "Gunslingers" [15] and "Undercover" [21] are proving popular in dense urban areas.

Accurate indoor location systems are still an area of active research. Cricket [17] is a system which operates over ultrasound and radio. It differs from the Active Bat system by using active beacons and passive listeners. However, this makes Cricket less suited to fast-paced location-aware games since the passive listeners can only listen to a single beacon at a time, resulting in reduced accuracy for rapidly moving targets. They describe their experiences with porting Doom, a popular first-person shooting game here [3]. The Bristol Indoor Positioning System [18] also provides low-cost location tracking using a combination of ultrasonics and RF, but to a lower accuracy than the Bats system.

Researchers have used indoor location systems to prototype some novel games. *Pirates!* [4] merges some of the traditional social aspects of game-play by using the physical world as a component of a computer game. Local wireless network and PDAs with RFID proximity-sensors attached are used allowing players to walk around a physical "game arena" and map locations, ob-

[8]We conducted these tests using interfaces from three different manufacturers, and found similar results in all cases.

[9]Although note that headsets should use Synchronous Connection-Oriented (SCO) connections rather than ACL connections as SCO connections can reserve bandwidth.

jects and nearby players into the game. *PingPongPlus* [14] improves the classic game of ping-pong by providing an "athletic-tangible interface" which tracks the progress of the game using microphones, and projects additional effects onto the game table (e.g. water ripples where the ball bounces). Headon et al. proposed the *Active Floor* [12], which measures the Ground Reaction Force of a user and allows movement-based control over a game, including simple gesture recognition.

Other researchers have investigated using Bluetooth as a communication channel for multiplayer games. Ritter et al [19] overcame shortcomings with Bluetooth handovers by using a second wireless infrastructure to support ad-hoc multiplayer games.

6. CONCLUSIONS

This paper presented the design of a location-aware mobile game platform designed for fast-paced mobile games in the style of *Counter-Strike* and *Capture the Flag*. The platform was sufficiently flexible to allow experiments with different feedback mechanisms, location-accuracies and communications latencies.

These experiments revealed insight as to how weaknesses in one of these three components can be offset against strengths in the other two. In particular, fundamental tradeoffs between event latency and feedback bandwidth, as well as location-accuracy and device I/O capabilities were highlighted.

We investigated the suitability of Bluetooth as a communications network for low-latency, location-aware mobile gaming. Overall it showed promise, with sufficient bandwidth and an average application-to-application round trip time of 34ms. However a number of critical bugs in some implementations were evident, particularly when handling multiple concurrent connections. The fundamental issue of handover between Bluetooth interfaces remains to be addressed if games are use a larger playing arena; we found 20m to be the practical limit in a typical office environment.

Acknowledgements

This work was funded by AT&T Labs — Research, Intel Research Cambridge, The Royal Commission for the Exhibition of 1851, and the Schiff Foundation.

7. REFERENCES

[1] M. Addlesee, R. Curwen, S. Hodges, J. Newman, P. Steggles, A. Ward, and A. Hopper. Implementing a sentient computing system. *IEEE Computer*, 34(8):50–56, August 2001.

[2] S. Baatz, M. Frank, R. Göpffarth, D. Kassatkine, P. Martini, M. Schetelig, and A. Vilavaara. Handoff support for mobility with IP over Bluetooth. In *Proceedings of the 25th Annual Conference on Local Computer Networks*, pages 143–154, November 2000.

[3] H. Balakrishnan, R. Baliga, D. Curtis, M. Goraczko, A. Miu, N. B. Priyantha, A. Smith, K. Steele, S. Teller, and K. Wang. Lessons from developing and deploying the Cricket indoor location system, November 2003.

[4] S. Björk, J. Falk, R. Hansson, and P. Ljungstrand. Pirates! Using the physical world as a game board. In *Interact, IFIP TC.13 Conference on Human-Computer Interaction*, July 2001.

[5] Bluetooth SIG. *Bluetooth Core Specification v1.1*, April 2003. `http://www.bluetooth.org/`.

[6] R. Chakravorty and I. Pratt. WWW performance over GPRS. In *Proceedings of the IEEE International Conference on Mobile and Wireless Communication Networks*, August 2002.

[7] A. D. Cheok, S. W. Fong, K. H. Goh, X. Yang, W. Liu, and F. Farzbiz. Human Pacman: a sensing-based mobile entertainment system with Ubiquitous Computing and tangible interaction. In *Proceedings of the 2nd Workshop on Network and System Support for Games*, pages 106–117. ACM Press, 2003.

[8] G. M. Djuknic and R. E. Richton. Geolocation and assisted GPS. *IEEE Computer*, 34(2):123–125, 2001.

[9] Federal Communications Commission. Enhanced 911. `http://www.fcc.gov/911/enhanced/`.

[10] M. Flintham, S. Benford, R. Anastasi, T. Hemmings, A. Crabtree, C. Greenhalgh, N. Tandavanitj, M. Adams, and J. Row-Farr. Where on-line meets on the streets: experiences with mobile mixed reality games. In *Proceedings of the conference on Human factors in computing systems*, pages 569–576. ACM Press, 2003.

[11] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom '99)*, 1999.

[12] R. Headon and R. Curwen. Movement awareness for ubiquitous game control. *Personal and Ubiquitous Computing*, 6:407–415, December 2002.

[13] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System: Theory and Practice*. Springer-Verlag, 5th edition, March 2001.

[14] H. Ishii, C. Wisneski, J. Orbanes, B. Chun, and J. Paradiso. Pingpongplus: design of an athletic-tangible interface for computer-supported cooperative play. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 394–401. ACM Press, 1999.

[15] Mikoishi Studios. Gunslingers, Urban Combat Evolved. `http://guns.mikoishi.com/gunsSingTel/`.

[16] Newt Games. Mogi, item hunt. `http://www.mogimogi.com/`.

[17] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43. ACM Press, 2000.

[18] C. Randell and H. L. Muller. Low cost indoor positioning system. In *Proceedings of the 3rd international conference on Ubiquitous Computing*, pages 42–48. Springer-Verlag, 2001.

[19] H. Ritter, T. Voigt, M. Tian, and J. Schiller. Experiences using a dual wireless technology infrastructure to support ad-hoc multiplayer games. In *Proceedings of the 2nd workshop on Network and system support for games*, pages 101–105. ACM Press, 2003.

[20] R. Webb. Recreational Geocaching: The South-East Queensland Experience. In *A Spatial Odyssey, Australian Surveyors Congress*, September 2001.

[21] YDreams Team. Undercover. `http://hk.playundercover.com/`.