Game Action Based Power Management for Multiplayer Online Game

Bhojan Anand, A.L. Ananda, Mun Choon Chan, and Le Thanh Long
School of Computing
National University of Singapore
{banand, ananda, chanmc, lethanhl}@comp.nus.edu.sg

Rajesh Krishna Balan
School of Information Systems
Singapore Management University
rajesh@smu.edu.sg

ABSTRACT
Current mobile devices embrace a wide range of functionalities including high speed network support, hardware accelerated 3D graphics, and multimedia capabilities. These capabilities have boosted the interest for enabling multiplayer online games (MOG) support on such devices. However, the lack of similar growth in battery technology limits the usability of these devices for MOGs. In this paper, we present energy conservation techniques for highly interactive MOGs. These are games, such as first-person shooters, where crisp user interaction is paramount to the overall game experience. Hence, conserving energy while preserving crisp user interaction becomes a critical consideration in this domain. We first present three obvious power management approaches and highlight their limitations. We then discuss two application-assisted approaches for power management that manage to save power while preserving the required user experience. Our results demonstrate that these application-assisted approaches are very promising.

Categories and Subject Descriptors
K.8 [Personal Computing] – K.8.0 General – *games.*

General Terms
Algorithms, Management, Measurement, Performance, Design, Reliability, Experimentation, Human Factors.

Keywords
Mobile games, wireless networks, power management, statistical prediction.

This work is supported in part by the Singapore Ministry of Education Academic Research Fund Tier 2 under the research grant T208B2301. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the granting agency, National University of Singapore or Singapore Management University.

MobiHeld'09, August 17, 2009, Barcelona, Spain.

1. INTRODUCTION
The number of personal portable devices sold each year is increasing rapidly with mobile phone sales outpacing personal computer sales at the rate of 5 to 1 [4]. Current mobile phones are not just devices for voice communication, but are mini computers that have high speed wired and wireless networks, and sufficient processing and memory capabilities to run enterprise, multimedia, and gaming applications. However, the user experience obtained from running multimedia streaming and highly interactive applications on mobile phones is usually much lower when compared to running those same applications on a desktop computer. Beside several user interface factors, there are several technical factors that hinder the user experience for multimedia streaming and interactive applications on mobile phones. One of the key challenges is battery lifetime. For example, a mobile phone's battery may only be able to support 4 to 6 hours of actual voice communications and about 2 to 4 hours of streaming video. This perilous energy situation becomes even worse when the 802.11b or g network interface of the mobile phone is activated. 802.11b/g has higher energy usage compared to GSM based networks as the Carrier Sense function of 802.11b/g's CSMA/CA protocol consumes energy continuously. As various studies have shown [5][9], the wireless interface and CPU are the major power consuming components in mobile devices -- the wireless interface alone can consume up to 50% of the total system power.

In this paper we first present three obvious power management approaches and demonstrate their limitations. We then present two approaches based on predicting a game player's behavior. In these approaches, we first use a simple linear prediction algorithm to decide the player's activity level and then input the activity level into a game action based control mechanism. Based on the outcome, the algorithm decides on the appropriate method of limiting resource usage to save energy while still preserving the user experience. In this paper, we focus solely on limiting the use of the network resource. Our results show that even with a relatively simple algorithmic approach, significant energy savings are possible. Our results show that the first approach is better at saving energy while the second approach achieves a better balance between energy savings and preserving the user experience. Overall, our results show that this is a promising approach for power management in interactive multiplayer online games.

2. RELATED WORK

We broadly classify mobile networked games into turn based games (which are slow and not highly interactive) and real-time games (which are fast and highly interactive). In this project, our main focus is on real-time games.

A wide range of recent research focuses on power aware protocols and techniques for mobile environments. While several of these works are on power management [1][2][3], they focus mainly on standalone mobile applications or latency tolerable applications such as mobile web based applications and mobile audio/video streaming applications. Only a few of these previous approaches focus [8] on power saving schemes for multiplayer networked game applications. These prior work describe solutions for reducing the power usage of the processor, display, and wireless network interface card (WNIC) components of the mobile device. For the display, the display intensity is adapted according to the residual power in the mobile device – trading off quality for longevity. For the processor, various schemes such as CPU scaling, and reducing the CPU load have been proposed [7][10]. Finally, for the WNIC, the basic technique is to switch off the WNIC when it is not needed [3]. Turning off the WNIC introduces losses and latency but this is tolerated by managing the streaming protocols and buffer size, and controlling the traffic between the base station and wireless client. Our work differs from these prior approaches as we exploit a player's game state and focus only on the WNIC.

3. OUR TESTBED

For the experiments presented in this paper, we used the testbed shown in Figure 1. The test bed contained 4 mobile game clients connected, via 802.11b, to a base station (that was connected, via a 100Mbps Ethernet connection, to the game server). The game clients used Dlink DWL-G630 PCMCIA and Dlink DWA 110 USB wireless interface cards for connectivity. The power characteristics of the two cards used are shown in Table 1.

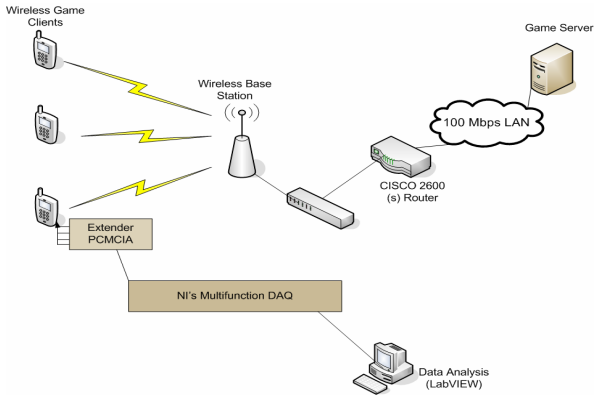

Figure 1 Test Bed for measuring power consumpton

DWA 110 hardware is optimized for low *standby mode* power consumption. The 802.11bg transceivers used in mobile phones today consume even less than 1mW power during *standby mode* due to improvements in the hardware transceiver chipsets. Our schemes focus on reducing *active mode* (transmit/receive mode) power consumption.

Table 1 Power characteristics of the cards used

Mode	DWA-110 802.11bg	DWL-G630 802.11g-b
Standby Mode	4.66 mA	300-300 mA
Power Save Mode	28 mA	Not supported
Receive Mode	248 mA	330-350 mA
Transmit Mode	248 mA	550-580 mA
Power Supply	5 V	3.3 V

The PCMCIA card was connected to the mobile device through a PCMCIA extender card, built by Accurite Technologies (Figure 2a), that extended the PCMCIA bus and provided leads for voltage and current measurements. The USB wireless card was connected through a USB extension cable, which was cut open for connecting the current flow measurement probes. We used a high-speed multifunction Data Acquisition Equipment (DAQ) USB-6251 and a Signal Conditioning Equipment (SC-2345) from National Instruments (NI). The DAQ shown with the USB wireless card in Figure 3, is optimized for superior accuracy and accepts 1.25 million signals per second. It takes signal inputs from the SC-2345 and sends the current and voltage measurements to the computer via the USB module.

One of our mobile clients was a custom made WinCE.net 5.0 based PDA, shown in Figure 2b, from iWAVE (*http://www.iwavesystems.com/)*. The iWAVE PDA provided leads for measuring the power consumption of individual components such as the TFT LCD, WNIC, Processor, Flash, SDRAM; allowing for fine grained power measurement.

Figure 2a Extender Card

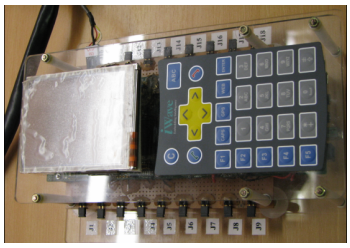

Figure 2b iWAVE PDA

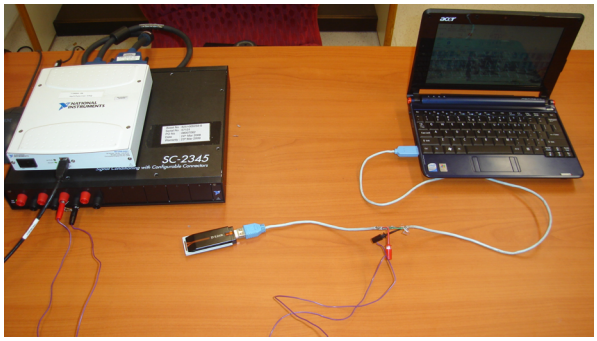

Figure 3 Dlink DWA-110 connected with NI's DAQ

We used the First Person Shooting game Quake II from IDSoftware (*http://www.idsoftware.com/*) for our experiments as a) it is a real-time game, b) the source code was freely available, and c) it could run on our PDA devices.

4. BASELINE APPROACHES

First, we investigated the potential gains of three baseline approaches. These approaches were classified as either *white box* or and *black box*. A "white box" approach is used where the source code of the game is fully available and the solution is implemented in the source code itself while a "black box" approach is used when the source code is not available and the syntax and semantics of the network protocol are not known. There is an intermediate "grey box" approach (where the game semantics are known) that we do not use in this paper.

The three baseline approaches were (1) a "black box" approach where clients drop packet with no support from the application; (2) a "white box" approach where a game's Dead Reckoning threshold is varied, and finally (3) a "black box" approach where the game server varies its sending rate.

4.1 Baseline 1: Packet Dropping by Client

We first tested a black box approach where packets are dropped by the client without any knowledge about the game state. Our expectation was that every packet dropped would save energy at the expense of game quality. The drop rate was dynamically adjusted based on the remaining battery power in the mobile client. The drop rate was also not allowed to rise to levels where the game quality dropped below an acceptable threshold – defined as the point where a human player started noticing the packet drops and resulting latency. We found that this simple method resulted in very small amounts of energy savings. . This was because Quake II uses very small packet (~ 24 bytes) as shown in Figure 4. Hence, dropping an individual packet results in just a very small opportunity for saving energy by not transmitting that packet – the transmission time of a 82 byte frame (24 byte date with 58 bytes UDP/IP/802.11b overheads) is only 0.0074 ms when using an 11Mbps 802.11b link.

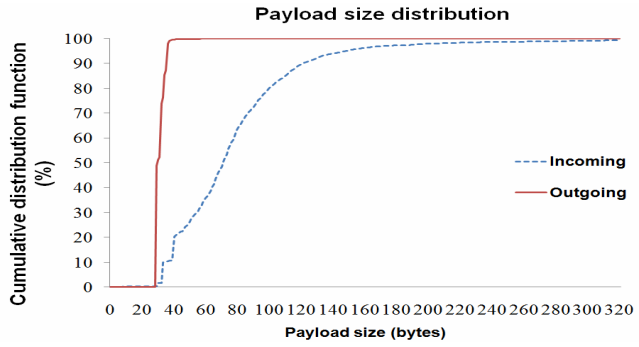

Figure 4 Quake II Client/Server Packet Size

In addition, when not transmitting, most wireless cards go into 'receive mode', where they consume about half of the power used in 'transmission mode'. For some cards, such as the DLink DWA 110 USB wireless interface card and ORiNOCO PCMCIA wireless interface card (SILVER), the receive mode power consumption is equal to or close to the transmission mode. Finally, some PDAs, such as the "i-wave PDA", are not optimized for power usage and do not show any difference in power consumption when in different operating modes.

We tested our PDA using two extreme scenarios. In the first scenario, the i-wave PDA, running Quake II, is connected directly to the game server with no game traffic being generated. In the second scenario, a real game was played, generating real game traffic between the PDA and the server. The difference in power consumption between the two scenarios, using 802.11b CAM mode, was only 0.002 to 0.004 watts as shown in Figure 5 and Figure 6. As such, there is no real benefit in reducing traffic volume.

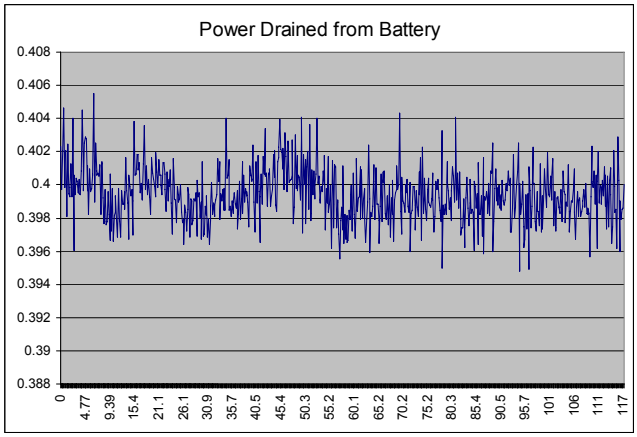

Figure 5 Power consumption with no traffic: 0.3957 W to 0.3973 W

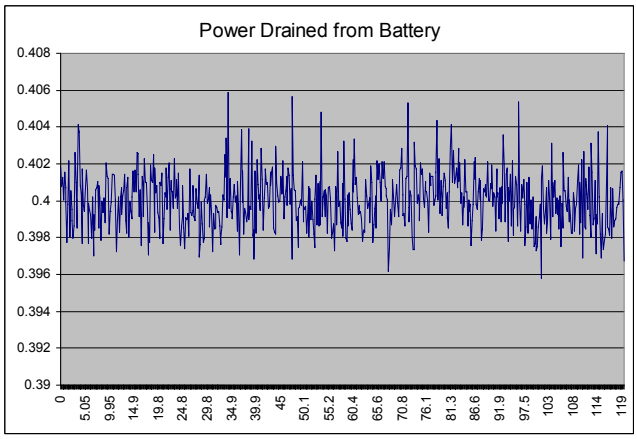

Figure 6 Power consumption with traffic: 0.3994 W

4.2 Baseline 2: Adjusting the Dead Reckoning Threshold

Our second baseline approach is a *white box* approach that directly manipulates and optimizes a parameter in the game source code to save energy. The parameter chosen is the dead reckoning (DR) threshold. Dead reckoning is a technique used in games to smooth player movement in the absence of real update information. A smaller threshold is more sensitive to packet losses but accurately reflects player movements while a larger threshold can tolerate higher packet losses but could result in inconsistent movement (for example, the player turns but the on-screen avatar continues moving straight).

For this approach, we tuned the DR algorithm, implemented in the Quake II source, so that the DR threshold value became higher (or lower) when the battery level went down (or up). By default, a Quake II client sends 25 packets per second (Figure 7a) and the Quake II server sends 10 packets per second (Figure 7b). By increasing the DR threshold, we can reduce the number of packets

sent by the client with little drop in the quality of game play. However, similar to the previous section, due to the small packet sizes, we did not notice any significant power savings with this approach.

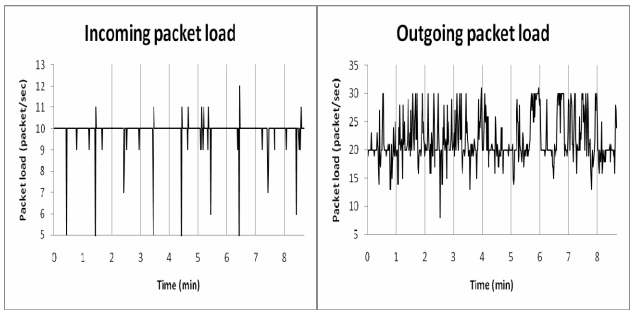

Figure 7a. Client to Server **Figure 7b. Server to Client**

4.3 Baseline 3: Reduction of Server Rate

Our third approach is a server assisted or proxy assisted balancing where the proxy/server reduces the number of packets to the client when the client's battery level is low. Unfortunately, this *black box* approach does not help much as the client's WNIC has to constantly remain in the energy consuming receive mode. This is because the inter arrival time of the Quake II packets (similar to most real-time games) is very short. As reported in prior work [11], a delay of more than 200 ms will noticeably impact game quality.

Hence, even if the proxy/server and client are time synchronized and the client knows the time of next incoming packet, it cannot turn its WNIC to sleep mode as a) the inter arrival time between packets is very short, and b) unlike media streaming applications, we cannot buffer packets as they are time-critical. Indeed, using large queuing buffers is frequently counterproductive in real-time games.

4.4 Discussion

The previous approaches were all game agnostic and they are likely to result in quality degradations if they were stretched to save as much energy as possible during any phase of the game. For example, dropping one or more packets that carry details about a shooting action (which probably killed the enemy), could result in the enemy appearing alive on other player's screens even though the enemy is dead in the current player's screen.

Therefore, in order to achieve a significant impact on power savings while preserving game quality, we have to:

- Change the mode of the WNIC card to off/deep sleep whenever possible with minimum possible mode switch latency.
- The packets that are lost during the sleep period should be as insignificant as possible. That is, packets that convey important game events/actions should not be dropped as far as possible. For example, update packets containing absolute values may be more valuable than update packets containing incremental values.

To achieve the two stated goals, the power saving algorithm must understand the current game status. In the next section, we present two different application-assisted approaches for power savings.

5. APPLICATION-ASSISTED APPROACH

5.1 Statistical Prediction based Scheme

We defined the *player activity level* (PAL) as the frequency of his interaction with the game during the game play -- the activity level is high when the frequency of user interactions is high. From our measurements, we observed that the player's current activity level correlates well with the previously observed player activity levels.

We observed that if the PAL is higher than the *PAL threshold*, then all the resources must operate to their full potential to minimize quality losses. Otherwise, the system has opportunities to enter various power saving modes. We set our *PAL threshold* value, to a value of 6, based on our initial measurements of various trial game runs with real game players. This threshold should be adjusted, in practice, depending on the level of the player (expert, novice…). Energy can be saved if the user activity level is predicted in relation to the game state, and the power consumption is reduced when the PAL is low.

We predict PAL using a simple standard linear prediction model -- to avoid additional power drain due to complication prediction models. The prediction algorithm was implemented in the Quake II game source, together with routines to turn off the wireless interface, and for reducing the CPU clock frequency based on user activity levels. Our algorithm can be mathematically stated as follows:

$$x'(n) = \sum_{i=1}^{p} w x(n-i)$$

where x'(n) is the estimation of current activity level of the player and x(n-i) are actual measurement of previous activity levels of the player. w_i is the weightage given for historical PALs used for predicting the current PAL. We used a history window of size 5. The weightages (w_i) are initialized to [.3,.25,.2,.15,.1], with higher weightage to the latest computed PAL values. The error generated in this method is,

$$e(n) = x'(n) - x(n)$$

The error is used as feedback to dynamically adjust the weightages (w_i); thus reducing the overall error. The algorithm predicts the appropriate times for turning off the WNIC interface without degrading the quality of the game play significantly. During the periods in the game play where the user action is not highly fluctuating, the algorithm results in correct prediction close to 97% on whether to turn-off (below threshold) or not (above threshold). This accuracy with is a simple algorithm is good enough to preserve the quality of the game. As shown in next section, swift action oriented event periods (eg. shooting) are significantly less than the simple smooth periods (eg walking) in a game play. Hence, the game play in general, is dominated by stable or gradually changing user actions.

There are two issues in changing the power mode of the WNIC card.

- *Mode switching latency* – this affects the quality of the game play as the card is not usable while it is switching modes. The Dlink card we used takes less than 250ms to go between on and off states. Most of this latency is du to the re-association of the

WNIC with the base station. By saving the wireless association before turning off the WNIC card, this latency can be further reduced. Moreover, in our approach, since the mode switching happens only when the user activity level is low, the loss in quality is unnoticeable. Certain cards, such as the Dlink DWA-110, have special power save and very lower power standby modes. Switching to these modes, instead of the off mode, will reduce the switching latency even further.

- *Mode Switching Penalty (Additional power consumption during mode switching)* – this could lead to increased power use if mode switching is used naively. In our experiments, we optimally set the off/sleep duration such that the total amount of power saved is more than the overhead power lost during mode switching. Since we chose the optimal settings, the player perceived quality is maintained at acceptable levels. Another positive effect we encounter is that frequently switching to a low power consumption state allows the batteries to recover, exploiting the battery recovery effect [6].

The optimal value for off/sleep duration d_{sleep} should be in-between minimum *l* and maximum *h* such that,

(eps > 0) AND (h < (tda - qualityThreshold)), where,

eps (effective power saved) = power saved during the period d_{sleep} – mode switch penalty
tda (total duration available) = time between successive turn off – mode switching latency
qualityThreshold = the minimum duration that the card should be on before next turn-off/sleep state to ensure the playability of the game.

The duration d_{sleep} can be tuned according to the requirement. When d_{sleep} is close to *l*, it results in low power savings and better game quality. When it is very close to *l*, the amount of power saved will become insignificant. When d_{sleep} is close to *h*, it results in high power savings and lower game quality.

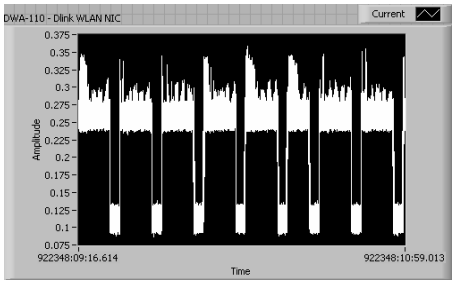

Figure 8 Dlink DWA-110 OFF-to-ON Penalty

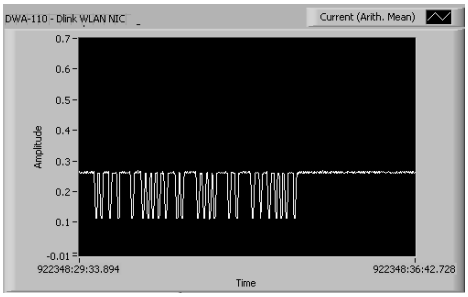

Figure 9 Dlink DWA-110 ON-OFF Pattern with the Linear Prediction Algorithm

The power consumption profile for the DWA-110 card, shown in Figure 8, shows the transitions between the *Receive state* (260mA current consumption) and the *Radio-OFF state* (100mA current consumption). During the mode switch from OFF to ON there is a short period of higher power consumption (350mA current consumption) – resulting in a fairly high 90mA switching overhead (350mA – 260mA). Hence, as long as the *Radio-OFF* time is higher than this overhead (in our case it is set to optimal), significant amounts of power can be saved.

The following calculations are based on the data collected from several 5 minute game sessions with different players and with an activity threshold of 6. Above the *PAL threshold*, the player is highly active and no data should be lost. Figure 9 shows the pattern of ON-OFF states for this data. From our trace, we observed 131 possible places where we could turn-off the Radio after eliminating the periods in which *tda-l <= 0,* that is, the periods which are too small for saving energy. On average, with a 100ms mode switch latency (from off to on) the time between successive turn-offs is 2300 ms. Hence, the total duration available (*tda*) is 2200 ms. In these experiments, whenever an opportunity to turn off the radio occurred, it was turned off for one second (1s falls within *l* and *h*). In a 2200 ms period, if we turn-off the wireless radio for just 1 second, the radio will be ON for 1200 ms. We can thus compute a back-of-the-envelope savings, using the card DWA-110, in the following way. First, the total current consumption (without any idle/off period) for 2300ms is 598 mA.

Total current consumption for 2300 ms with on-off cycle:
35mA switch overhead + 100mA radio off period + 312mA radio on period = 447mA.

Total current saved in one cycle = 598mA – 447mA
= 151mA
*Total current saved for 5 minutes = 131 cycles * 151mA*
= 19.78 A
Hence, percentage of current saved is ~ 25.3%

If we assume a mobile device with 4 hours of battery supply for continuous game play in normal mode, our power saving scheme above extends the lifetime by approximately 0.6 hours.

5.2 Game Action based Prediction Scheme

In this section, we present a Game action/state based prediction decision method that is complementary to the simple user activity level based prediction method presented in the previous section. It is based on the experimentally observed correlation between the game state and the player activity level and it improves the amount of energy saved while maintaining the quality of the game play at an acceptable level.

The proposed method uses a 2 stage prediction scheme. In essence, the method first classifies various types of games activities in Quake II (such as, shooting, chasing, running, walking, hiding, etc.) and then observes the user activity level during each of these activities to make power saving decisions. The advantage of using a 2 stage approach is that it helps to better identify the most appropriate period and duration to put the WNIC into a sleep/off state. For example, we can increase the frequency and duration of the sleep/off state while the game action is walking or hiding and reduce the frequency and duration

of the sleep/off period when the game action is very dynamic such as shooting or chasing.
Based on our study of the Quake II game, we identified the following major game activities:

1) Walking, 2) Running, 3) Hiding
4) Moving up and down (across the floors), 5) Jumping,
6) Shooting and other weapon use
7) Changing weapons / collecting new weapons
8) Collecting utility items (such as bullets and health packs)

In order to understand the frequency of these activities, we collected data from more than 50 game play sessions of a human player. We believe that this data collection effort will provide a good gauge on how effective our 2 stage approach could be. The results of our data collection are shown in Figure 10. The high frequency activities such as walking (27% of the time) and hiding (21% of the time) provide the highest chance for turning off the wireless radio.

Figure 10 Game Activity based Analysis for Power Management

An interesting observation is that less significant activities such as walking and hiding happens most of the times; especially compared to important game changing activities (that cannot experience any quality loss) such as shooting. This suggests that substantial power saving might be possible.

We used the same set of 5 minute game traces used in Section 5.1. For each of the 131 cases where the WNIC could have been turned off, we fully eliminate the periods where the game actions during that period were critical. From the activity based analysis shown in Figure 10, we identify the "Shooting and other weapon use" event as being critical – accounting for 17% of the total player actions.

Using back of the envelope pessimistic calculations, by eliminating 17% from the total current saved, we can still effectively save 25.3% x (100%-17%) = 21% without affecting the game quality during the critical periods. We can achieve even higher savings if we increase the radio off period; at the expense of slower response to the player.

It is also possible to save power, even during the critical period, if the activities during the critical period have enough gaps between them, or if the player is in safe region. Hence, in reality, it is possible to save even more power compared to the conservative calculations shown above.

6. FUTURE DIRECTIONS

Learning the game activity automatically in real-time is a highly complex task. An easy and efficient way is to let the game itself provide such information for power optimization purposes. For this purpose, we are in the process of defining a *power-aware game API* that will let game engines inform of the system of the current game status/activity.

In this work, we only considered the WNIC component of the PDA. We plan to combine this work with additional work that saves power by changing the *display brightness* and/or adjusting the *CPU frequency*. Together, these techniques, coupled with energy-aware network traffic optimization, will create a highly efficient power optimization solution for mobile games that balances energy efficiency with quality.

7. REFERENCES

[1] Krashinsky, R., and Balakrishnan, H., "Minimizing Energy for Wireless Web Access with Bounded Slowdown", in *proceedings of the eighth annual ACM/IEEE international conference on mobile computing and networking (mobicom 2002),* September 2002.

[2] Chandra, S., and Vahdat, A. "Application-specific Network Management for Energy-aware Streaming of Popular Multimedia format", in *proceedings of the 2002 usenix annual technical conference,* June 2002.

[3] Anand, M., Nightingale, E. B., and Flinn, J., "Self-tuning Wireless Network Power management", in *proceedings of MOBICOM'03,* September 2003.

[4] Findlay Shearer, "Power Management in Mobile Devices", *Communications Engineering Series,* Newnes-Elsevier,2008.

[5] Chen B, Jamieson K, Morris R, Balakrishnan H. "Span: An Energy-Efficient Coordination Algorithm For Topology Maintenance In Ad Hoc Wireless Networks", *Proceedings Of MOBICOMM*, 2001.

[6] Carla F. C. and Ramesh R. Rao, "Pulsed Battery Discharge in Communication Devices", *in Proceedings of MOBICOM '99*, Seattle, WA, August 1999, pp 88-95.

[7] Wanghong Yuan, Klara Nahrstedt, "Practical voltage scaling for mobile multimedia devices" in *Proceedings of the 12th annual ACM international conference on Multimedia*, 2004.

[8] Yan Gu, Samarjit C., "Control Theory-based DVS for Interactive 3D Games", *Proc. of 45th Design Automation Conference (DAC), Anaheim, California, USA*, June 2008.

[9] Anastasi G, Passarella A, Conti M, Gregori E, Pelusi L, "*A power-aware multimedia streaming protocol for mobile users*" , Proceedings of International Conference on Pervasive Services, 2005.

[10] Shivajit M., Radu C., Nikil D., Alex N. & Nalini V., "Integrated Power Management for Video Streaming to Mobile Handheld Devices*", Proc. of ACM-MM Conf.*, 2003.

[11] Beigbeder T, Coughlan R, Lusher C, Plunkett J, Agu E, Claypool M, "The Effects of Loss and Latency on User Performance in Unreal Tournament 2003" , *Proc. of ACM SIGCOMM 2004 workshop on Netgames*, Portland, USA, August 2004.