eWatch: A Wearable Sensor and Notification Platform

Uwe Maurer[1], Anthony Rowe[2], Asim Smailagic[3], Daniel P. Siewiorek[3]
[3]School of Computer Science, Carnegie Mellon University, Pittsburgh
[2]Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh
[1]Computer Science Department, Technische Universität München, Germany

Abstract

The eWatch is a wearable sensing, notification, and computing platform built into a wrist watch form factor making it highly available, instantly viewable, ideally located for sensors, and unobtrusive to users. Bluetooth communication provides a wireless link to a cellular phone or stationary computer. eWatch senses light, motion, audio, and temperature and provides visual, audio, and tactile notification. The system provides ample processing capabilities with multiple day battery life enabling realistic user studies. This paper provides the motivation for developing a wearable computing platform, a description of the power aware hardware and software architectures, and results showing how online nearest neighbor classification can identify and recognize a set of frequently visited locations.

1. Introduction

The eWatch is a wearable sensor and notification platform developed for context aware computing research. It fits into a wrist watch form factor making it highly available, instantly viewable, and socially acceptable. eWatch provides tactile, audio and visual notification while sensing and recording light, motion, sound and temperature. The eWatch power management was designed to operate similar to a cellular phone, requiring the user to recharge overnight. The eWatch needs to be small and energy efficient enough to allow for multiple day user studies by non-technical participants. Given these energy and size constraints eWatch should provide the most computation and flexibility to allow an assortment of applications. The goal was to move beyond simple sensor logging and allow for online analysis that could query the user for feedback while collecting data or provide services to showcase context aware applications.

eWatch can be used for applications such as context aware notification, elderly monitoring and fall detection, wrist PDA, or a universal interface to smart environments. The ability to sense and notify allows for a new variety of enhancements. For instance, much work has been done on fall detection for the elderly [7]. Existing systems do not function appropriately when a patient loses consciousness and cannot press a button. Current automatic systems have a high rate of false positives. An eWatch system could sense if the user was in distress and then query to confirm that it is an emergency. If the user does not respond, then the eWatch could use its networked abilities to call for help. The use of online learning could profile a patient's daily activity and notify a caretaker if a patient no longer performs their daily routines. The eWatch can also notify a patient when they should take certain medication.

2. Related Work

Several groups have developed wearable computing platforms and wearable sensor recording and processing devices [2, 1, 8]. However, most of these devices do not interact directly with the user, have insufficient battery life, or are too cumbersome for a long-term study with non-technical subjects. The idea of a smart wrist watch dates back as early as the 1930s [3] and first took a functional form with the IBM Linux Watch [6]. In its original form, the Linux Watch was a PDA on the wrist, and did not possess sensors. Later revisions of IBM's Linux Watch added acceleration and audio sensors; however they lacked light and temperature sensors and have not targeted user context or location tracking applications. The power consumption of the Linux Watch is too great for day long operation.

Current location tracking systems offer high accuracy [10, 5] using triangulation methods, however, they require infrastructure support. In this paper we demonstrate a simple, coarse-grained location tracking method to show how eWatch can use sensor information to reason about the environment. Our method relies only on sensor samples from the environment in order to categorize the user's location.

3. Hardware Architecture

Figure 1 shows the eWatch architecture which consists of: the main CPU, sensors, power control, notification mechanisms, and wireless communication. The main CPU is a Philips LPC2106 ARM7TDMI microcontroller with 128Kb of internal FLASH and 64Kb of RAM. The LPC2106 is a 32bit processor capable of software controlled CPU scaling up to 60Mhz. eWatch communicates wirelessly using a SMARTM Bluetooth module and an infrared data port for

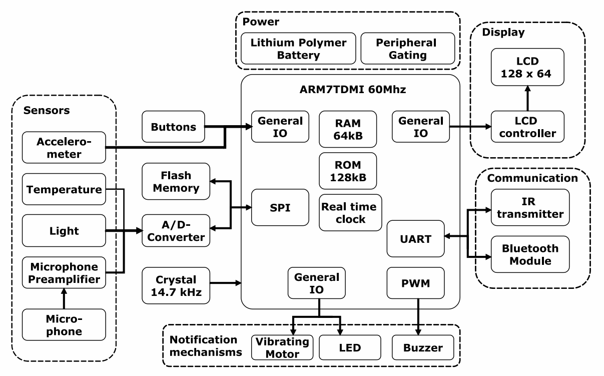


Figure 1. eWatch hardware architecture

Part	Avg. Power(mW)	Peak Power(mW)
ARM7 processor	29.7	132
ADC	4.95	4.95
Microphone Amp	2.5	2.5
Accelerometer	2.0	2.0
LCD Controller	0.23	0.33
Serial Flash Memory	0.003	13.2
Bluetooth Module	0.09	90
Vibration Motor	0	63
Backlight LED	0.03	33
Light Sensor	0.825	0.825
Temperature Sensor	0.825	0.825
Average Life Time: 56 hours		

Table 1. Average and peak power

control of devices such as a television. A previous version of eWatch based on different hardware is described in [9].

Sensor data is acquired using an external TLV1544 10bit ADC and can be stored in a 1Mb external FLASH device. eWatch is capable of sensing temperature, light, two axes of acceleration and audio at user controllable sampling intervals up to 100Khz. A MAX4061 amplifier is used for audio conditioning. We use an ADXL202 MEMs accelerometer to measure the planar acceleration of the user's hand. The user can be notified using a 128x64 pixel display, an LED, vibrating motor and tone generating buzzer. Three push buttons are distributed around the outside of the housing in the standard configuration of a digital watch.

eWatch is powered by a 3.6 volt 700mAh rechargeable lithium polymer battery with a linear regulator active during peak voltages and a DC to DC voltage pump as the battery drains. Table 1 shows the power consumption of each component in the system. The chart shows the maximum possible power usage of each device followed by an average power consumption computed from a trace of daily use.

The final housing is made from epoxy resin that was cast in a silicon mold and measures 50mm x 48mm x 17.5mm. The limiting factor with resepect to eWatch's size is the battery which can later be reduced as the device is customized for a specific application.

4. Software Design

The eWatch system was designed as a platform for developing context aware applications. The main goals that influenced the design decisions were *ease of use* and *flexibility*. eWatch provides the developer with an API that enables rapid prototyping. The eWatch software system consists of three layers: Application, System Functionality, and Hardware Abstraction.

Applications access functionality of lower layers to render screen images, interact with the user and retrieve information from the storage, sensors or wireless network. The System Functionality Layer provides an API for shell, task and power management. The Hardware Abstraction Layer contains the drivers for all the hardware components providing access to all eWatch functionality.

The layered architecture helps to achieve our goal of flexibity by reducing the effort necessary to port to another hardware or software environment. For example, we developed a Linux port of the software system that replaces the hardware abstraction layer with simulated hardware. This enables rapid development cycles since the code can be tested on the developers machine without actually updating the eWatch firmware.

4.1. Interface

eWatch offers two interfaces for a user or developer to control its functionality: the eWatch shell and the Graphical User Interface (GUI) on the built-in display.

The eWatch shell allows users to execute functions and configure variables via Bluetooth. A text-based protocol is used to transmit commands similar to a Unix shell. The applications on eWatch can register functions and variables, making them accessible through the shell. The commands can be typed by a user or developer through a keyboard or sent from a program running on the PC. This enables automated scripting of the system and allows remote applications to access eWatch functionality.

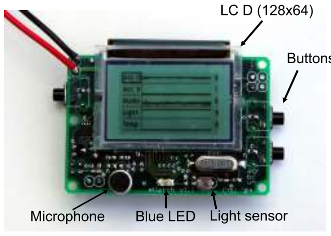
The primary GUI of eWatch is the menu system. As shown in Figure 3(a), the menus allow the user to scroll through lists of items and select entries to activate them. The menu structure is organized hierarchically - entries can be modified, added, or removed during runtime. Each menu entry is linked to a shell command that is executed when the entry is selected. The eWatch GUI library supports TrueType fonts, drawing of geometric shapes, and displaying bitmaps.

4.2. Applications

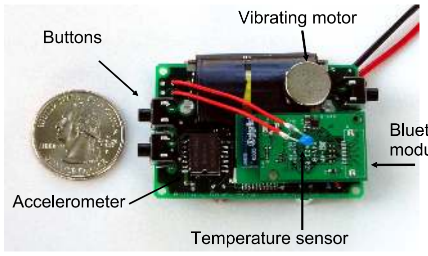
Several applications make use eWatch's capabilities. Figure 3 shows screenshots of the eWatch user interface and applications.

Like a standard wrist watch, eWatch provides the user with the current time (Figure 3(b)). In addition, it shows information about incoming text messages and upcoming calendar events. Information about the sensor logging and available memory is shown while logging is in progress. Figure 3(c) shows a real-time plot of sensor data.

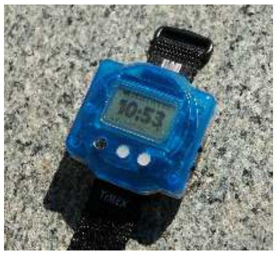
The calendar application stores and notifies the user of events (Figure 3(d)). The calendar can be synchronized with



(a) Top view of eWatch board

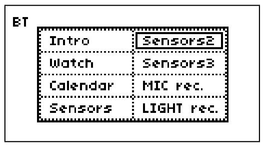


(b) Bottom view of eWatch board

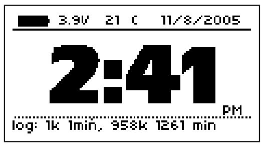


(c) eWatch with housing

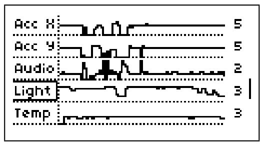
Figure 2. eWatch board and housing



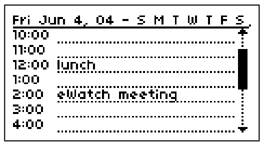
(a) First page of menu



(b) Watch screen



(c) Sensor overview



(d) The eWatch calendar

Figure 3. The eWatch GUI

a personal computer using the standard iCal format. The data from the calendar can be used for context aware applications that recognize the user's location and activity.

4.3. Sensor Sampling and Recording

Recording sensor data is a core system functionality. The sensor sampling and recording system is designed to consume minimal power and make efficient use of memory. Sensor sampling is interrupt-based to minimize sampling jitter. Every sensor has a timer interrupt with a configurable sampling frequency. In the interrupt handler, the ADC value is read and then stored in a memory buffer. Between interrupts the system remains primarily in the idle state to conserve energy, occasionally powering up to compress the collected data and write it to flash.

We wanted a lossless compressor to allow for 24 hours of data recording. We chose an algorithm described in [4] that performs compression using four linear predictors and efficient coding of residuals. Our experiments showed that it reduced the memory consumption of the sensor data to 20% - 50% of the original size.

4.4. Power Management

The ARM7TDMI microcontroller supports two power-saving modes and frequency scaling. An event-based architecture that waits in idle mode for incoming events was selected. When an event occurs, the processor wakes up to service it. After the application completes, it relinquishes control to the scheduler that can then return the processor to idle mode.

5. Location Recognition

Knowing about the user's location is an important aspect of a context aware system. Using eWatch we developed a system that identifies previously visited locations. Our method uses information from the audio and light sensor to learn and distinguish different environments.

We recorded and analyzed the audio environment and the light conditions at several different locations. Experiments showed that locations have unique background noises such as car traffic, talking, noise of computers, air conditioning and television. The light sensor sampled at a high frequency can also provide additional information beyond the brightness of the location. Frequency components generated by electric lights (at 120Hz and 240Hz) and displays (television, computer screen at 60Hz) can be observed. We observed that the frequency characteristics of light conditions tend to remain constant in most locations. For our study, audio data was recorded with the built-in microphone at a sample rate of 8kHz and the light sensor at a frequency of 2048Hz. At every location five consecutive recordings of audio and light were taken, separated by 10 second pauses. For every recording, we sampled the microphone for four seconds (32000 samples) and the light sensor for 0.5 seconds (1024 samples).

The recorded data was then compressed and stored into flash memory. Locations frequently visited by the user were recorded; the rooms of the user's apartment (living room, kitchen, bedroom, bathroom), their office, the lab, different street locations on the way to university, the interior of a bus, and several restaurants and supermarket. Each location was visited multiple times on different days. In total, we collected 600 recordings at 18 different locations.

5.1. Feature extraction

We estimated the power spectral density of the recorded sensor data using Welch's method. A 128-point FFT was calculated for a sliding window over the complete recording and averaged over frequency domain coefficients for all windows. The result is a smoothed estimation of the power spectral density. To reduce the number of feature components, the Principal Component Analysis was used. The dimensionality of the feature vector was reduced to its first five principal components. To visualize the feature space, Figure 4 shows the first three components of the feature vectors after a Linear Discriminant Analysis (LDA) transformation.

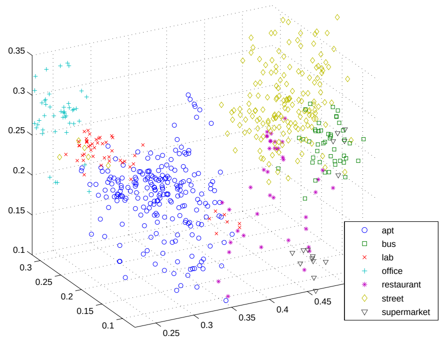


Figure 4. Features after LDA transformation

Location	Sensors		
	Light only	Audio only	Audio & Light
apartment	98.2%	90.9%	95.9%
bus	35.6%	84.4%	77.8%
lab	64.0%	90.0%	98.0%
office	84.6%	76.9%	89.2%
restaurant	62.5%	77.5%	90.0%
street	93.5%	91.2%	90.6%
supermarket	73.3%	66.7%	66.7%
Class average	73.1%	82.5%	86.9%
Overall	84.9%	87.4%	91.4%

Table 2. Location recognition accuracy

5.2. Location Recognition Results

The nearest neighbor method with a 5-fold cross validation was used for classification. Three different feature sets were evaluated: features from the light sensor only, microphone only and both sensors combined. As expected, the combination of both sensors gave the best results in identifying the location. The classification with the light sensor alone gave an overall result of 84.9% correctly classified samples. The classifier confused the *lab* and *office* location and also the *bus* with the *street*. This occurred because both location pairs can have similiar light conditions. Using only the audio sensor the overall recognition accuracy was 87.4%. The *office* and *apartment* location were confused in this case. Both sensors combined gave the best result of 91.4%. Locations that could not be distinguished well with only one sensor were classified more accurately with both sensors combined. Table 2 shows an overview of the classification results for the individual locations.

5.3. Online Classification and Performance

The 1-NN classification method was implemented on the eWatch to allow online classification in realtime. The sensor recording uses 4.5 seconds of data (4 seconds for audio, 0.5 seconds for light), the computing time for the classification is about 1.4 seconds. 98.5% of the classification time is spent performing the feature extraction. The PCA and nearest neighbor search take less than 20ms to compute. In order to reduce the time spent in the feature extraction, other features are being investigated, such as time domain characteristics.

6. Conclusions and Future Work

This paper describes eWatch, a wearable sensor and notification computing platform for context aware research. The hardware design focused on providing enough computational resources to perform machine learning algorithms locally, while still allowing a comfortable form factor and a battery capacity sufficient for extended user studies. Likewise, the software environment was designed to facilitate easy development while automatically managing resources such as power and sensor data. We also described a system that uses the eWatch and its sensors to categorize its environment in real-time.

Future work will focus on activity recognition and combining activity data with location information. Integration of an 802.15.4 radio is planned to allow the eWatch to function as a mobile node in a sensor network. This added flexibility will further integrate the eWatch into its environment by allowing a larger area of network coverage.

6.1. Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010, the National Science Foundation under Grant Nos. 0205266 and 0203448, a grant from Intel Corporation, and PA Infrastructure (PITA).

References

[1] N. B. Bharatula, M. Stäger, P. Lukowicz, and G. Tröster. Empirical Study of Design Choices in Multi-Sensor Context Recognition Systems. In *IFAWC: 2nd International Forum on Applied Wearable Computing*, pages 79–93, Mar. 2005.

[2] R. W. DeVaul and S. Pentland. The MIThril Real-Time Context Engine and Activity Classification. Technical report, MIT Media Lab, 2003.

[3] C. Gould. Dick Tracy. *Comic, Chicago Tribune*, Oct 1931.

[4] M. Hans and R. Schafer. Lossless Compression of Digital Audio. *IEEE Signal Processing*, 18:21–32, July 2001.

[5] K. Lorincz and M. Welsh. Motetrack: A robust, decentralized approach to rf-based location tracking. In *Proceedings of the International Workshop on Location and Context-Awareness (LoCA 2005) at Pervasive 2005*, May 2005.

[6] C. Narayanaswami and M. T. Raghunath. Application design for a smart watch with a high resolution display. In *ISWC '00: Proceedings of the 4th IEEE International Symposium on Wearable Computers*, page 7, Washington, DC, USA, 2000.

[7] N. Noury. A smart sensor for the remote follow up of activity and fall detection of the elderly. In *Proc. of the IEEE Special Topic Conference on Microtechnologies in Medicine & Biology May*, 2002.

[8] A. Smailagic and D. P. Siewiorek. Wearable and Context Aware Computers: Application Design. In *IEEE Pervasive Computing, Vol. 1, No. 4*, pages 20–29, Dec 2002.

[9] A. Smailagic, D. P. Siewiorek, U. Maurer, A. Rowe, and K. Tang. eWatch: Context-Sensitive Design Case Study. In *In Proc. of the IEEE Annual VLSI Symposium*, pages 98–103. IEEE Computer Society Press, May 2005.

[10] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. Technical Report 92.1, ORL, 24a Trumpington Street, Cambridge CB2 1QA, 1992.