Proc. of 7th International Workshop on Smart Appliances and Wearable Computing (IWSAWC 2007), DVD-ROM (June 2007).

CLAD: a Sensor Management Device for Wearable Computing

Kazuya Murao, Yoshinari Takegawa, Tsutomu Terada, and Shojiro Nishio
Graduate School of Information Science and Technology, Osaka University
Yamadaoka 1-5, Suita-shi, Osaka 565-0871, Japan
{murao.kazuya, takegawa, tsutomu, nishio}@ist.osaka-u.ac.jp

Abstract

There has been increasing interest in wearable computing. In wearable computing environments, a wearable computer runs various applications with various sensors (wearable sensors). Since conventional wearable systems do not manage the power supply flexibly, they consume excess power resources for unused sensors. Additionally, sensors frequently become unstable for several reasons such as a breakdown. This instability is hard to detect simply from the sensed data. To solve these problems, we propose a new sensor management device CLAD (cross-linkage for assembled devices) that has various functions for power management and sensed-data management. CLAD improves power saving, data accuracy, and operational reliability.

1. Introduction

The downsizing of computers has led to wearable computing attracting a great deal of attention. Wearable computing is different from conventional computing in three ways[1]. (1) Hands operation free : information can be obtained without manual operation because the computer is worn. (2) Power always on : the computer is always available because the power is always on. (3) Daily-life support : daily activities can be supported because the computer is worn all the time.

In particular, wearable sensors, which are sensors a user wears, enable the provision of various services such as navigation[2] and health care[3]. Although a user might wear several sensors to use multiple services, as shown in Table 1, some sensors may not always be used, e.g., health care services when battery power is low and navigation services when the user is indoors.

Since conventional systems using multiple sensors cannot control the power supplied to the sensors individually, unused sensors also consume power. Reducing the power consumption is important because of the limited battery life in wearable computing environments. Moreover, sensors can become unstable due to a malfunction, a power shortage, overcurrent, and so on, and it is difficult to detect unstable operation from only sensed data.

Table 1. Services using wearable sensors

Services	Sensors
Health care	Heat sensor, GSR sensor Accelerometer, Electric sphygmograph
Navigation	GPS, Geomagnetic sensor, Gyro

To solve these problems, we developed a sensor management device that has various functions for power management and sensed-data management. This CLAD (cross-linkage for assembled devices) device functions as a relay device between wearable sensors and a wearable computer and saves energy by managing the power supply to the sensors on the basis of the given circumstances. Moreover, CLAD detects errors from changes in the characteristics of the sensors and generates pseudo data for malfunctioning sensors. CLAD thereby improves data accuracy and system operational reliability.

This paper is organized as follows. Section 2 describes the system design of CLAD in terms of the structure, the functions, and pseudo data generation. Section 3 presents a practical implementation of CLAD. The performance of CLAD is discussed in Section 4. Finally, Section 5 concludes with a summary and a look at future work.

2. System Design

We designed CLAD assuming that a user wears various types and numbers of sensors that continually consume power and that sensing is not always required. CLAD is positioned between a wearable computer and these sensors, and it manages the sensors to achieve (1) flexible power supply control for energy saving, and (2) flexible error control for achieving sufficient sensed-data accuracy.

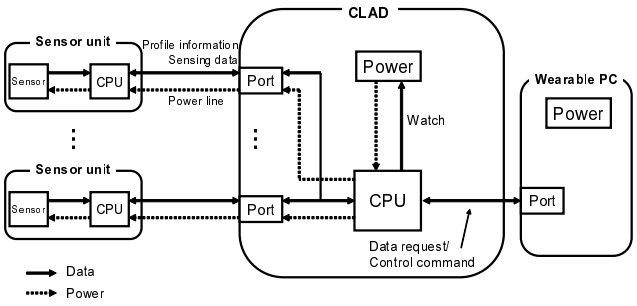

Figure 1. System structure of CLAD (not to scale)

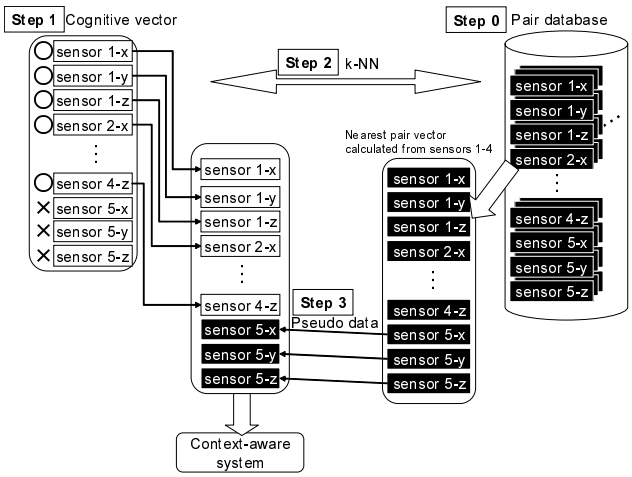

Figure 2. Pseudo data generation

2.1. System structure

As shown in Fig. 1, CLAD has its own power source and manages sensors connected to it. It monitors the voltage and current to detect power shortages and overcurrents. Each sensor has a microcomputer (CPU) to process commands from CLAD. Information about the sensor (type, accuracy, output range, start-up time, operating voltage, and operating current) is stored in the CPU.

2.2. CLAD function

The functions of CLAD are divided into the power management and the data management. Talbe 2 shows the features that contribute to each functions.

2.2.1 Power management

Alternative device retrieval and changeover

CLAD detects sensor anomalies from consecutive outlying data points, sensor data interruptions, etc. In such cases, CLAD identifies an alternative device by referring to the sensor profile information, and if one exists, it activates it.

Power saving

CLAD always monitors its internal power source. If it detects a power shortage, it reduces power consumption by stopping power supply to some of the sensors on the basis of a refusal policy.

Overcurrent detection

If overcurrent is detected, CLAD stops all power supply for safety.

2.2.2 Data management

Error detection

CLAD detects problems such as outlying data and a dying battery. Since CLAD notifies the PC such problems, applications can deal with them individually such as by displaying a message recommending a battery change.

Pseudo data generation

When there is no alternative device, CLAD can generate pseudo data from learned data and the correlation to other sensors. This function improves operational reliability.

An example of pseudo data generation is shown in Fig. 2 for a context-aware system with five acclerometers. Sensor 5 is malfunctioning. The pseudo data is generated as follows.

Step 0. Constructs pair database

CLAD has already collected sensed data (pair vectors) for all contexts and constructed a pair database.

Step 1. Acquire cognitive vector

When a sensor malfunction is detected, the system creates a cognitive vector, which is a set of sensed data without data from the malfunctioning sensors.

Step 2. Extract pair vector from pair database

The system finds the pair vector in the database that is nearest the cognitive vector by using the k-NN (k-nearest neighbor) method.

Step 3. Extract pseudo data from pair vector

The data for sensor 5 is replaced with that of the extracted pair vector.

Since this pseudo data generation does not depend on the sensor type or context-aware algorithm, it is applicable to any system.

2.3 Pair vector extraction

We developed and tested three methods for extracting the pair vector from the pair database in Step 2.

Table 2. Functions of CLAD

	Power managemet			Data management	
	Alternative device	Power saving	Overcurrent detection	Error detection	Pseudo data
Power control	○	○	○		
Profile information management	○	○		○	○
Voltage check for self		○		○	
Current check for self			○	○	
Voltage check for sensors				○	
Current check for sensors			○	○	
Sensed-data management				○	○

2.3.1 All Alive (AA) method

This method targets all working sensors for the k-NN calculation. From cognitive vector $X = (x_{1x}, x_{1y}, x_{1z}, \cdots, x_j, \cdots, x_{5x}, x_{5y}, x_{5z})$ and pair vector $P_i = (p_{i_{1x}}, p_{i_{1y}}, p_{i_{1z}}, \cdots, p_{i_j}, \cdots, p_{i_{5x}}, p_{i_{5y}}, p_{i_{5z}})$ $(i = 1, \cdots, N)$, euclidean distance d_{AA_i} is calculated. The N is the number of samples in the pair database.

$$d_{AA_i} = \sqrt{\sum_{j \in working} \{x_j - p_{i_j}\}^2}$$

By calculating the euclidean distance for all pair vectors in the pair database, we find the nearest pair vector, $P_{I\{I|r_{AA_I}=min(d_{AA_1},\cdots,d_{AA_N})\}}$. The system outputs the complemented cognitive vector, $C = (c_{1x}, c_{1y}, c_{1z}, \cdots, c_j, \cdots, c_{5x}, c_{5y}, c_{5z})$.

$$c_j = \begin{cases} x_j & (j \in working) \\ p_{I_j} & (j \in malfunctioning) \end{cases}$$

2.3.2 Correlative Alive with Threshold (CA-Th) method

The AA method uses the k-NN method for all working sensors, and if the data of working sensors for two contexts are nearly equal, the distance for these contexts is nearly equal, making it difficult to extract the correct pair vector. There is some correlation between sensor values, so the k-NN method achieves more accurate data complementing. We use the Pearson product-moment correlation coefficient:

$$correlation = \begin{cases} \left| \frac{\sum_{i=1}^{N}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \overline{x})^2 \sum_{i=1}^{N}(y_i - \overline{y})^2}} \right| & (x \neq y) \\ 0 & (x = y). \end{cases}$$

Generally speaking, an absolute value of 0.0-0.2 for the correlation coefficient means scarcely any correlation, 0.2-0.4 means some correlation, 0.4-0.7 means good correlation, and 0.7-1.0 means strong correlation. Since the correlation coefficient between the same sensed data is always 1, the system sets it to zero in the calculation. In the CA-Th method, the eculidean distance revised using the correlation coefficient is called correlated distance d_{Th}. This method sets a threshold for the correlation coefficient and applies the k-NN method only to sensors with good or strong correlation.

$$d_{Th_i} = \sqrt{\sum_{j \in working} \{x_j - p_{i_j}\}^2} \cdot \alpha$$

$$\alpha = \begin{cases} 1 & (if\ correlation \geq threshold) \\ 0 & (otherwise) \end{cases}$$

2.3.3 Correlative Alive with Distance/Correlation (CA-DistCor) method

The CA-DistCor method applies the k-NN method to all working sensors, as does the AA method, and it uses the sum of the euclidean distance divided by correlation coefficient defined as correlated distance d_{CD}. The calculation of the correlation coefficient is the same as in the CA-Th method.

$$d_{CD_i} = \frac{\sqrt{\sum_{j \in working} \{x_j - p_{i_j}\}^2}}{correlation}$$

In this method, euclidean distances among strongly correlated sensors carry much weight and scarcely correlated sensors carry little weight.

3. Implementation

We implemented a prototype CLAD device using a Microchip PIC16F873A microcontroller as a processing unit. We implemented the system software on a Microchip MPLAB ICD with a CCS PIC C compiler. The prototype and wearable sensors are shown in Fig. 3. Table 3 shows its specifications. Each wearable sensor has a processing unit for communication control. CLAD measures the power source voltage using a zener diode. An LTS6-NP current transducer (LEM) is used for current monitoring. Table 4 lists the control commands. Table 4(a) shows the control

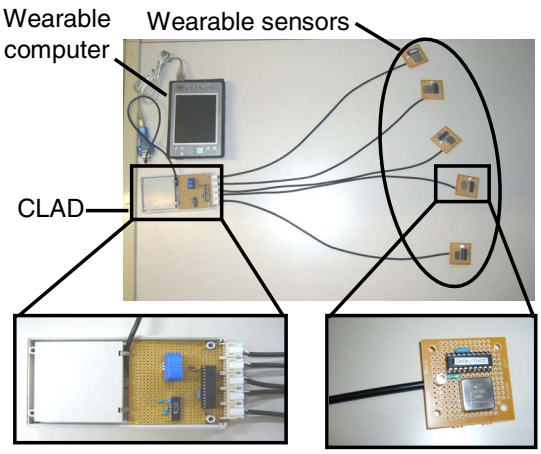

Figure 3. Prototype of CLAD

Table 3. Specifications of CLAD prototype

Connection method		RS232C
Power source		4 dry cell batteries (AA)
Communication speed		9600 bps (Max.)
Size (main body only)		W76 × H13 × D70 mm
Weight	Without battery and cable	130 g
	With battery, without cable	280 g
	With battery and cable	292 g
Power consumption (CLAD only)		0.05 W

commands from CLAD to the sensors for managing power control and sensed-data requests, Table 4(b) shows the commands from CLAD to the PC for sending status reports, and Table 4(c) shows the commands from the PC to CLAD for controlling the CLAD functions. When a new sensor is attached to CLAD, it sends its profile information to CLAD. CLAD then sends commands to the sensor for controlling it on the basis of the commands from the PC. When CLAD detects a sensor error or other problem, it sends an alert to PC and displays them.

4. Evaluation

4.1 Performance of energy-saving function

Figure 4 shows the performance of CLAD's power saving function when there are six accelerometers. Power consumption did not vary when CLAD was not used because the sensors were always active. When CLAD was used, power consumption could be varied by changing the number of active sensors. Power consumption with CLAD was higher than without CLAD for six sensors due to the operation cost of CLAD itself.

Table 4. Command tables

(a) Sensor control commands
Power off
Sensing data request
Profile request
Power on

(b) PC control commands
Overvoltage
Overcurrent
Data annomaly
Sensor annomaly
CLAD start-up
CLAD end

(c) CLAD control commands	
Pseudo data generation	On
	Off
Filtering	On
	Off
Data mergence	On
	Off
Change importance	High
	Low
Change criteria	Importance
	Rareness
	Power consumption
	Accuracy
	Start-up time
Power supply	Start
	Stop
Sensing	Start
	Stop
Profile information request	
CLAD end	

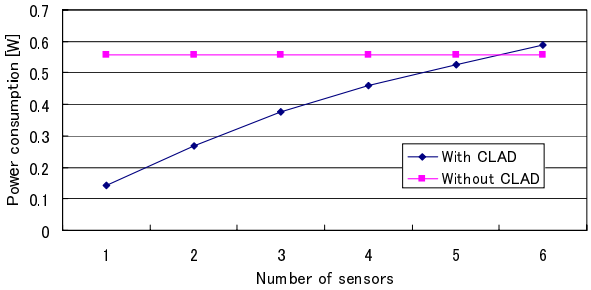

Figure 4. Power consumption vs. no. of active sensors

Figure 5 shows example daily activities for a wearable user. The user uses health care, agricultural support, and navigation services. Their power consumptions are 0.5, 1.5, and 0.1 W respectively. Working from 9 a.m. to 0 a.m. without CLAD results in total power consumption of 31.5 Wh, while with CLAD it is 13.75 Wh. CLAD achieves a 56.3% reduction.

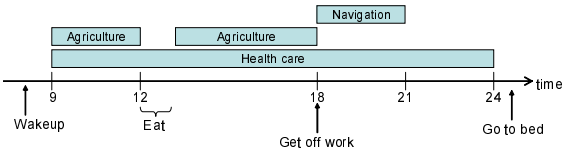

Figure 5. Daily activities for wearable user

4.2 Performance of pseudo data generation function

For this evaluation, a user wore five sensors (both wrists, both ankles, and one hip). We evaluated the recognition accuracy of the three proposed methods for 10 contexts: walk, run, climb steps, descend steps, jump, bicycle, lie, kneel, sit, and stand[4] on all combinations other than all sensors are malfunctioning (31 patterns). The sensors were compact wireless accelerometers (WAA-001, Wireless Technologies, Inc.)[5]. The sampling frequency was 20 Hz. The algorithm for a context awareness incorporated memory-based reasoning (MBR), euclidean distance, and the k-nearest neighbor (k-NN) method. The constructed context-aware system used mean $\mu_{i,T}$ and variance $\sigma_{i,T}$ for 20 samples of each component $c_{i,T}$ of 15-dimensional sensed data (cognitive vector) retraced from time $t = T$.

$$\mu_{i,T} = \frac{1}{20} \sum_{t=T-19}^{T} c_{i,T}$$

$$\sigma_{i,T} = \frac{1}{20} \sum_{t=T-19}^{T} \left\{ c_{i,t} - \mu_{i,T} \right\}^2$$

To trim down the dimension, we calculated the average of x-, y-, and z-axis of $\mu_{i,T}$ and $\sigma_{i,T}$. Six-dimensional vector X_T was generalized using the following equation for characteristic vector Z_T, where M_T and S_T are the mean and variance of X_T, respectively.

$$Z_T = \frac{X_T - M_T}{S_T}$$

The system calculated the euclidean distance between the cognitive vector and a training vector and recognized context using the k-NN method (k=1).

The results of the AA and CA-Th methods are plotted in Figure 6. The horizontal axis shows the 31 combinations of working and malfunctioning sensors (◯ means working, a blank means malfunctioning). The vertical axis shows the accuracy of context recognition. We used the correlation coefficient of the variance and set the threshold (Th) to 0.2, 0.4, or 1.0. The results for AA and CA-Th with $Th = 0.2$, or 0.4 were the same except for some combinations. The

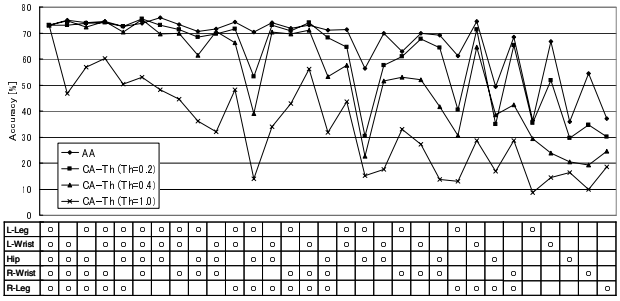

Figure 6. AA method vs. CA-Th method

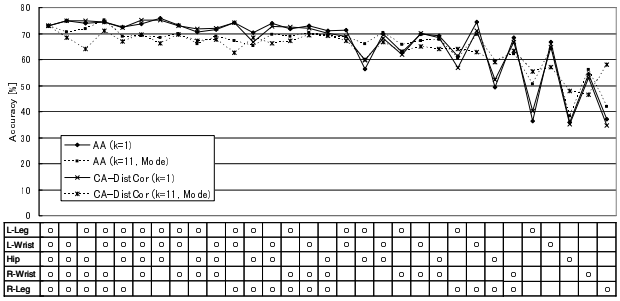

Figure 7. AA method vs. CA-DistCor method

AA method was better when there were one or two working sensors. $Th = 1.0$ was obviously inferior.

The results of the AA and CA-DistCor methods for two setting (k=1 and k=11) and using their mode are shown in Figure 7. The AA method is had slightly better results. Comparing k=1 and k=11, we see that k=11 was better when there were fewer sensors (right half of Fig. 7) and that k=1 was better otherwise (left half of Fig. 7).

Since the AA and CA-DistCor methods had almost equivalent performance, we can complement data with high reliability by changing the number of nearest neighbors.

Although the AA and CA-DistCor methods with k=1 had high accuracy, since both of them select the nearest pair vector, the probability of extracting a correct pair vector drops with the number of working sensors. However, the drops for k=11 were not deeper. In other words, with many malfunctioning sensors, we should select a pseudo data not from the nearest neighbor but from multiple nearest neighbors.

This experiment did not evaluate the performance of methods using the correlation coefficient because there may be evenly high relationships among the contexts. Therefore, we did an additional experiment to evaluate their performance. Figure 8 shows recognition accuracy against squat with and without arm rotation (Fig. 9). As indicated by the arrows, the CA-DistCor method performed better. This is because the correlations between the hips and same-side

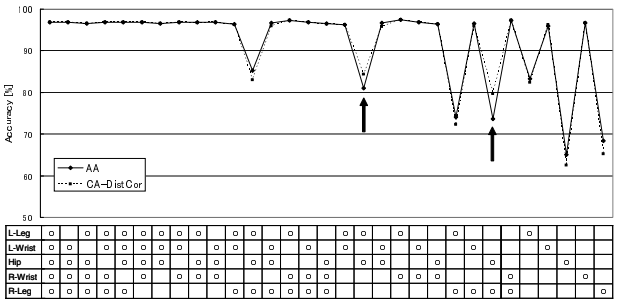

Figure 8. AA method vs. CA-DistCor in additional experiment

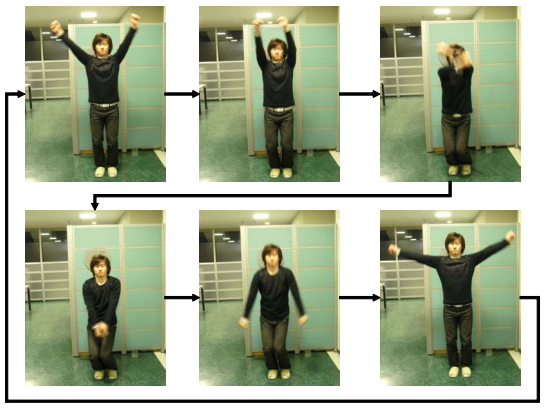

Figure 9. Action for additional experiment

wrists were high, so the value for the wrist sensor was complemented by that of the hip sensor. This means it is better to use correlation for similar contexts.

Table 5 shows the processing times for the three methods using a SONY VGN-U50 ultra-mobile PC (Celeron M, 900 MHz) as a wearable computer. Since the processing times for all the methods were less than 50 msec, which the interval for data input (20 Hz), the system achieves real-time processing. The greater the number of malfunctioning sensors, the greater the number of sensors that have to be complemented. However, since the number of the working sensors used in the retrieval comes from pair data deteriorates, the calculated amount is less. Therefore the processing time does not linearly increase with the number of malfunctioning sensors.

5. Conclusion

We have designed and implemented CLAD, a sensor management device for wearable sensors. CLAD achieves power saving by managing sensor power on the basis of the circumstances and high data reliability by managing the sensed data. Three methods were tested for generating pseudo data for malfunctioning sensors. They can be applied to other algorithm such as SVM (support vector machine) and HMM (hidden markov models) because our approach is independent of the context-aware algorithm. An evaluation experiment showed that CLAD does save power and does complement data effectively. We plan to reduce the size of CLAD, implement applications using CLAD, and develop another method for generating data with higher accuracy. Moreover, by changing the sensor combinations to match the circumstances, we can construct a context-aware system that performs with high accuracy and low power consumption.

Table 5. Processing times (msec)

No. of malfunctioning sensors \ Method	AA	CA-Th	CA-DistCor
1	8.008	21.936	23.329
2	6.964	35.167	37.256
3	5.571	40.738	42.479
4	3.830	38.301	39.345

Acknowledgements

This research was supported in part by a Grant-in-Aid for Scientific Research (A) (17200006) from the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References

[1] M. Miyamae, T. Terada, M. Tsukamoto, and S. Nishio: "Design and Implementation of an Extensible Rule Processing System for Wearable Computing," in *Proc. MobiQuitous 2004*, pp. 392–400 (Aug. 2004).

[2] M. Kanbara, R. Tenmoku, T. Ogawa, T. Machida, M. Koeda, Y. Matsumoto, and K. Kiyokawa: "Nara Palace Site Navigator: A Wearable Tour Guide System Based on Augmented Reality," in *Proc. of the 3rd CREST/ISWC Workshop*, pp. 7–14 (Oct. 2004).

[3] K. Ouchi, T. Suzuki, and M. Doi: "LifeMinder: A wearable Healthcare Assistant," in *Proc. of IWSAWC2002*, pp. 791–792 (July 2002).

[4] K. V. Laerhoven and H. W. Gellersen: "Spine versus Porcupine: a Study in Distributed Wearable Activity Recognition," in *Proc. of ISWC2004*, pp. 142–149 (Oct. 2004).

[5] Wirelss Technologies, Inc.: http://www.wireless-t.jp/.