A position detection method of devices on conductive clothes by controlling LED blinking

Isoyama, Naoya

Terada, Tsutomu

Akita, Junichi

Tsukamoto, Masahiko

International Journal of Wavelets, Multiresolution and Information Processing
© World Scientific Publishing Company

A Position Detection Method of Devices on Conductive Clothes by Controlling LED Blinking

NAOYA ISOYAMA

Graduate School of Engineering, Kobe University
1-1 Rokkodai-cho, Kobe Nada-ku, Hyogo 657-8501, JAPAN
isoyama@stu.kobe-u.ac.jp

TSUTOMU TERADA

Graduate School of Engineering, Kobe University / JST PRESTO
1-1 Rokkodai-cho, Kobe Nada-ku, Hyogo 657-8501, JAPAN
tsutomu@eedept.kobe-u.ac.jp

JUNICHI AKITA

College of Science and Engineering, Kanazawa University
Kakuma-machi, Kanazawa, Ishikawa 920-1192, JAPAN
akita@is.t.kanazawa-u.ac.jp

MASAHIKO TSUKAMOTO

Graduate School of Engineering, Kobe University
1-1 Rokkodai-cho, Kobe Nada-ku, Hyogo 657-8501, JAPAN
tuka@kobe-u.ac.jp

Received (Day Month Year)
Revised (Day Month Year)
Communicated by (xxxxxxxxxx)

Various wearable computing devices face problems with their power supplies, communication channels, and placement. Conductive clothes can resolve these problems, but it is still difficult to know the positions of devices on the conductive fabric. Therefore, we propose a method to detect the positions of such devices by using a camera. To detect the positions, our method blinks the LEDs on the devices according to their ID. Additionally, we propose several methods to shorten the time for detection. An experimental evaluation confirmed that compared with conventional method our methods reduce the time to detect the positions of the devices.

Keywords: position detection; conductive fabric; LED.

AMS Subject Classification: 94A08, 94A13, 68U10

1. Introduction

In wearable computing environments, users wear a computer and several wearable devices such as sensors, actuators, and displays to acquire various services in daily life such as healthcare, navigation, and performances.

Users of wearable computing devices would like to be able to install them simply by attaching them to their clothes because the combination of necessary devices changes everyday according to a user's plans for the day. To provide an environment where users can place and use devices freely on their body, one faces certain problems restricting devices: power supply, networking ability, and positions.

Clothes made of conductive fabric have been studied as a way to solve the problems. Conductive clothes enable devices to be installed freely because they supply electric power and a communication channel without the need for cables or other wiring. This paper supposes a network system using conductive fabric called *TextileNet*[23]. Users can install adequate devices according to applications, and sufficient electric power for operation as well as communication channels can be supplied to the devices using *TextileNet*. However, *TextileNet* by itself cannot detect the positions of devices because its conductive cloth acts as a uniform solid electrode, while knowing the position of a device is important information for a system to assign the functions to the devices automatically since there is a strong relationship between the position and the function of each device.

Therefore, we propose a method for detecting the positions of worn devices by using a camera. Our method blinks LEDs on the devices according to their ID so that the camera can detect their positions efficiently. Our method shortens the time for detection by adding several different blinking patterns.

The remainder of this paper is organized as follows. Section 2 explains the background of this research, and Section 3 details our method. Section 4 describes the implementation, and Section 5 evaluates our method by comparing it with a conventional method. Section 6 presents our conclusions and outlines future work.

2. Related Work

In wearable computing environments, there are many applications using the actuators and sensors on clothes[20,14]. The left of Figure 1 shows an example in which many devices are installed on a clothing, and the right of the figure shows an example of a dancer wearing many LED devices[16]. Moreover, *Musical Jacket*[27] enables a user to play music by using sensors or buttons that are installed on the wear, but their functions are fixed on the position where they are installed beforehand. *Memory Rich Clothing*[8] acquires the physical touch to the clothing, and expresses the memory of the touch by using LEDs sewn on the clothing. When the LEDs are sewn, it is difficult that they are sewn on other position again. If the user can reposition LEDs according to time and circumstances, the expression can extend farther.

Conductive clothes have attracted a great deal of attention as a way to permit such a flexible placement of wearable devices. Conductive clothes are made from conductive fabric or conductive thread, and a user can install devices such as buttons, sliders, and LEDs on a wear, moreover operates them with electric power supplied from the wear. These characteristics of conductive fabrics solve the problems of wiring, communication, flexible placement, and power supply. Here, the like of Figure 1(right) is affected with expression by the positions of devices (LED), however the system is not able to detect the

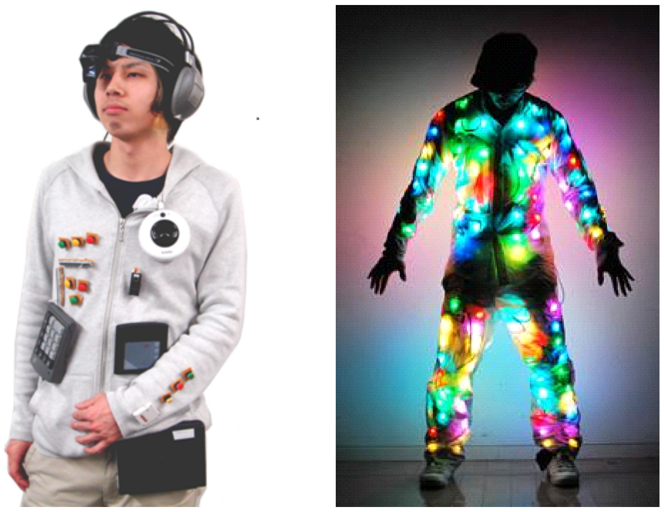
Fig. 1. Wearing devices

positions of devices if a user merely install the devices on the conductive clothes. Therefore it is necessary to detect the positions of devices in some way.

As researches on systems using conductive fabrics, *Networked Vest*[5] uses conductive fabric on both sides of the wear, and the devices attached to the vest have DC-PLC (Power Line Communication) modems that provide DC power from a single power supply as well as modulated analog signal communication. *Electronic Textiles*[18] uses a conductive thread with an insulator coating for horizontal and vertical directions as well as insulator thread. *Communication-Wear*[21] is a clothing concept that augments a mobile phone by enabling expressive messages to be remotely exchanged between people. It conveys the sense/experience of touch, and presence through sensations delivered by e–textiles. Mattmann has developed a way for body postures to be recognized using strain sensors in textiles[3]. In our study in this paper, we employ *TextileNet*[23] on which the user can freely install devices and easily arrange their layout (the conductive fabric and devices are shown in Figure 2, and the system structure is shown in Figure 3). There is an electrode of conductive fabric on both sides of the clothing, and we can install pin-shaped devices on the fabric. It is possible to supply electric power by connecting a battery to the fabric and to have devices communicate among themselves by using broadcast. Since the conductive cloth is a uniform solid electrode, *TextileNet* cannot detect the positions of worn devices.

The position of a device on the conductive fabric is important information for the system because there is a strong relationship between the positions of I/O devices and the functions to be assigned to them. This fact is discussed in [11], which proposes a method to automatically allocate functions to devices on a board including conductive fabric, called *Pin&Play*[13]. This method makes allocations in accordance with user profiles, functions to be assigned, device types, and device positions. If our system can recognize the positions

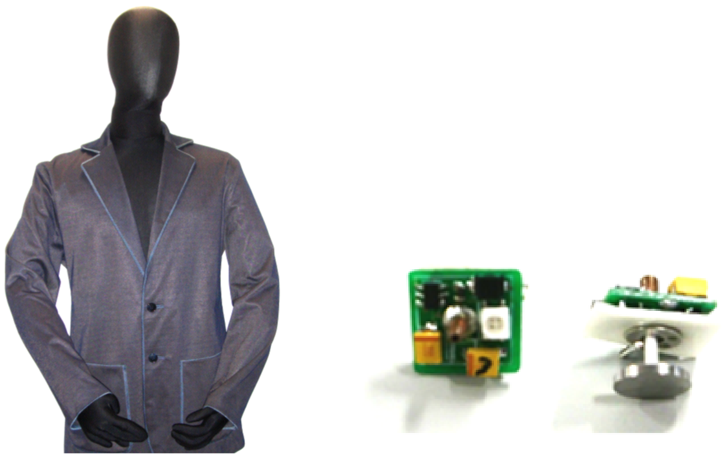

Fig. 2. TextileNet's facbric and devices

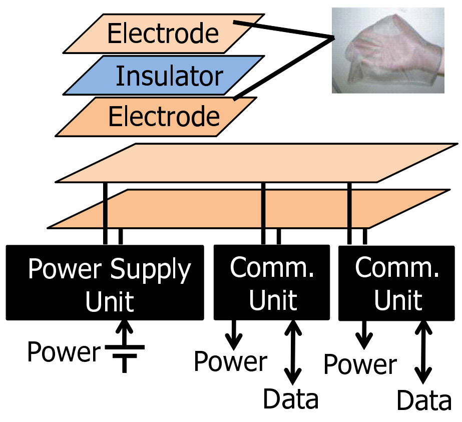

Fig. 3. TextileNet's system structure

of devices, it seems that the system can allocate the functions by such method even when a user utilizes the conductive clothes.

There have been several methods to detect the positions of many distributed devices. Helin detected the rough positions of devices by dividing a conductive board into several parts[6]. However, since we envision the possibility of several devices being on the same area of the cloth, it is difficult for us to apply this method to our system. *PushPin*[9] that uses pin-shaped interfaces detects the positions of the pins by calculating the difference of the time that has taken for receiving the lights and the ultrasonic waves from the peculiar device called *Pinger*[15]. Although this method is extremely precise, it needs to make an alteration to the pins and needs the device for position detection in addition to the pins. RSSI on wireless

LANs can be used[24,2]. The method has enough robustness but low accuracy. Nakata has proposed a method that a projector controls a large number of devices equipped with LED and optical sensor by irradiating the light patterns[17]. Moreover, Lee also has proposed a method to detect the positions of device with optical sensor in the same way[7]. Though these methods can manage a large number of devices, they cannot know each position of device. In addition, when the system of TextileNet uses this method, each device needs to be equipped with an optical sensor. Shinoda proposed a method to measure the positions of devices on thin sheets[12]. However, this method cannot be applied to our system since our clothing supposes flexible conductive fabrics. Our method does not require us to make changes to the *TextileNet* system, using only a web camera.

There is a number of methods for detecting the positions of devices by using a camera. *Papier-Mache*[22] is the research that aims at the materialization of tangible interface, having detected the positions of objects by using both the image processing and the barcode. If the system tries to detect the objects by using only image processing, it is needed to process complex calculations and has the problem that is hard to distinguish the similar shaped objects. *ARToolKit*[26], *CyberCode*[10], *TRIP*[4] employ visual markers that aim at detection of three-dimensional positions. Although it is easy to distinguish the indiidual markers, it is required each devices are attached the markers whose size is from a few to ten-odd centimeters square. Our proposed method uses only LEDs that are equipped with on respective devices for the position detection.

There are several methods for detecting the LED blinking by using the image processing, such as those using high-speed cameras[29]. These systems, however, need many costly cameras. In contrast, our method requires only a cheap web camera. For the *Pin&Play* that is the system using the conductive fabric, it has been proposed that the system detects the positions of devices by calculating the difference picture of before-and-after picture that the system had turned on the LEDs possessed on the devices such as buttons and sliders[25]. This method has been proposed on the assumption that the system is able to recognize the IDs of the installed devices beforehand, and does not consider the recognition error of blinking. *ID Cam*[19] detects positions with a camera by sending the ID information of the devices based on lighting patterns in a time series. This system can detect the positions regardless of the distance between the camera and the devices. However, it is not suited to wearable computing since it needs a relatively long time to synchronize the beacon and camera. *firefly*[1] efficiently detects the position of each LED by taking images of the LED lights. The system allocates 8-bit local addresses to each lighting elements, and it turns on the LEDs according to their ID in parallel. However, this method is prone to misdetections because it cannot always distinguish the positions of multiple devices when their LEDs are blinking at the same time. Our method solves this problem through its integrative use of parallel and serial blinking of LEDs.

3. Proposed Method

Our method detects the positions of devices worn on conductive clothes. In the TextileNet system, all devices have an LED for checking operation status. Our method uses this LED

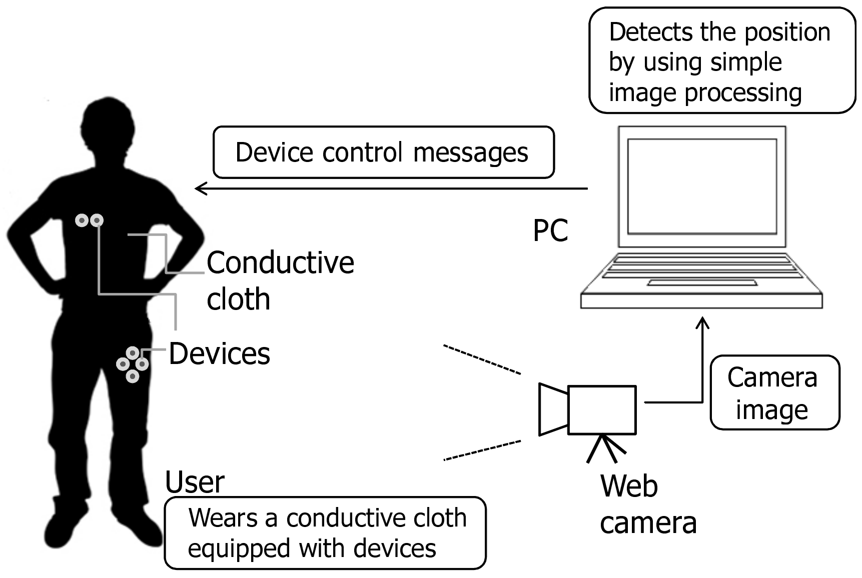
Detects the position by using simple image processing
Device control messages
PC
Conductive cloth
Devices
Camera image
Web camera
User
Wears a conductive cloth equipped with devices

Fig. 4. System structure

to detect the device's position by using simple image processing with a web camera. We suppose that each device has its own ID and a user, who wears conductive clothing with devices attached to it, stands in front of a camera while the system detects the positions of the devices from their blinking LEDs. Figure 4 illustrates the structure of our system. In our assumed environments, everyday a user decides the application to be used in that day and wears the devices for the intended applications. The user would then stand in front of a mirror equipped with a camera, and the system would detect the positions of devices and automatically allocate functions to the wearing devices.

3.1. *Blinking Algorithm*

The simplest method of detecting positions with a camera turns on LEDs corresponding to the IDs one by one. Because TextileNet cannot know in advance of the existence of installed devices, this method must turn on all possible IDs. Therefore, it cannot be used efficiently if there are many devices since it would take a lot of time to detect all of them. On the other hand, there is a method that turns on LEDs in parallel according to IDs. It takes only a little time, but it has a certain probability of misdetection. Therefore, we propose a new algorithm, two-phase LED blinking, to detect position efficiently and accurately.

Phase 1. *Parallel blinking*: Detect the positions of LEDs by turning on all LED in parallel.
Phase 2. *Sequential blinking*: Sequentially turn on LEDs for IDs that have been not decided on *Parallel blinking*.

Table 1. Example of Parallel blinking

Decimal	Binary	1st	2nd	3rd	4th
1	0001	-	-	-	ON
2	0010	-	-	ON	-
3	0011	-	-	ON	ON
4	0100	-	ON	-	-
8	1000	ON	-	-	-
15	1111	ON	ON	ON	ON

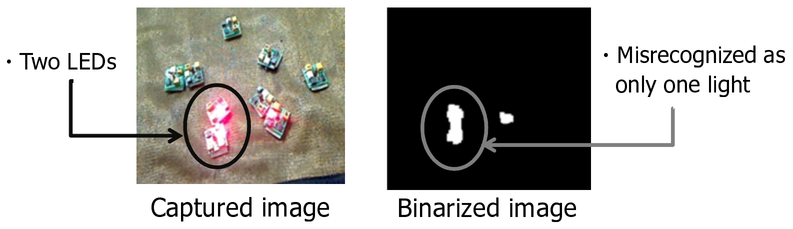

Fig. 5. Example of misrecognized image

Parallel blinking detects positions in parallel by switching multiple LEDs on and off based on the devices' IDs. On the N'th blink, the system turns on the LEDs whose the Nth bit of ID is 1. For example, when the ID is 4 bits long, a device whose ID is 1 blinks only at the 4th blink period, a device whose ID is 3 blinks in the 3rd and 4th blink periods as shown in Table 1. In this phase, when ID is expressed as N bits, the system can detect the positions of devices in N blink period, thereby dramatically reducing the detection time. However, this method is still prone to misdetections, since there are the cases that blinks from the multiple LEDs are misrecognized as being from one LED when devices are placed close to each as shown in Figure 5. For example, Figure 6 shows a case of parallel blinking with closely placed devices whose IDs are 1 and 3. In this case, the system misrecognizes blinks from the two LEDs as being from one LED, and detects only one device whose ID is 3. In addition, the system must have a positional margin (we call this margin the *Misdetection radius*) for the LED to be detected since the user may move during the detection phases. Thus detections within the *Misdetection radius* are treated as being from the same LED, and it causes a number of misdetections.

The Sequential blinking solves this problem by checking for detection errors with additional blinks after the Parallel blinking. In the following explanation, we suppose that the *Detected ID* refers to IDs that have been detected with Parallel blinking, and the *Candidate ID* means IDs that may be attributed to certain ID. For example, the *Candidate IDs* of a detected device whose ID is 3 are 1, 2, and 3 since there are possibilities of device combinations whose IDs are 1 and 2, 1 and 3, 2 and 3, and 1, 2, and 3. The procedure of Sequential blinking is as follows:

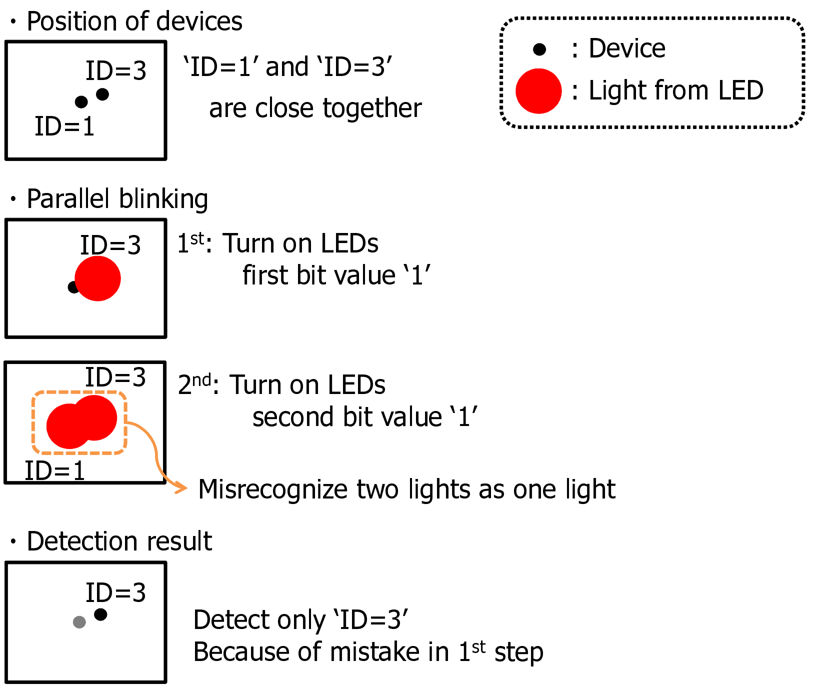
・Position of devices
ID=3
ID=1
'ID=1' and 'ID=3' are close together
● : Device
● : Light from LED
・Parallel blinking
ID=3
1st: Turn on LEDs first bit value '1'
ID=3
ID=1
2nd: Turn on LEDs second bit value '1'
Misrecognize two lights as one light
・Detection result
ID=3
Detect only 'ID=3'
Because of mistake in 1st step

Fig. 6. Example of misdetection

Step 0 Calculate the Candidate IDs for each Detected ID.
Step 1 Determine Detected IDs that are exclusively decided.
Step 2 If there is an ID that may be contained in only a certain Detected ID, turn on it and return to *Step 1*.
Step 3 If there are IDs that have not been determined yet, turn on one of them and return to *Step 1*.

Figure 7 and 8 show the flow of Sequential blinking and the pseudo code for it. In the following, we explain the procedure for each step using the example in Figure 9, which shows a situation in which a user installed devices for operating a music player application. In this example, each device has a 4-bit ID, as shown on the right of Figure 9. The IDs detected by Parallel blinking are 1, 3, 5, 11, 14, and 14 including detection errors, while the correct IDs are 1, 2, 3, 5, 6, 9, 10, 12, and 14.

Step 0: Calculate the Candidate IDs

Our method makes a list of the Candidate IDs for each Detected ID. Table 2 shows Candidate IDs for each Detected ID in the example. If an ID is determined, the ID is removed from all Candidate IDs.

Step 1: Determine Detected IDs that are exclusively decided

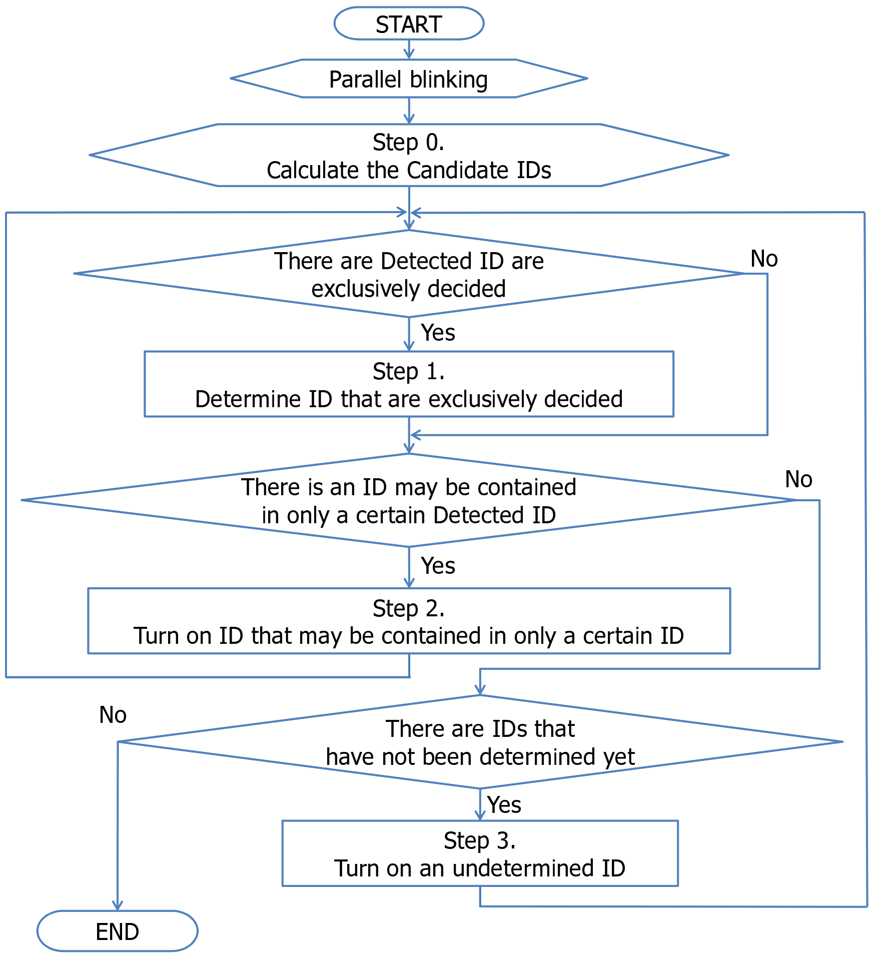
START
Parallel blinking
Step 0.
Calculate the Candidate IDs
There are Detected ID are exclusively decided
No
Yes
Step 1.
Determine ID that are exclusively decided
There is an ID may be contained in only a certain Detected ID
No
Yes
Step 2.
Turn on ID that may be contained in only a certain ID
No
There are IDs that have not been determined yet
Yes
Step 3.
Turn on an undetermined ID
END

Fig. 7. Flow of Sequential blinking

Our method turns on LEDs of devices that have the possibility of being misdetected. For efficient detection of these LEDs, our method employs a procedure to determine IDs without re-blinking. Concretely, for each $n(1, 2, .., bit\ length\ of\ ID)$, if there is no Candidate ID whose nth bit is 1 when the nth bit in its Detected ID is 1, the Detected ID can be deleted from Candidate IDs of all Detected IDs. This is because this case means there is no other blink without the Detected ID in the nth blink. This procedure is executed again until no ID can be determined.

In the example, the LED whose ID is 1 is determined first. Then, the LED whose ID is 3 is determined because the LED whose ID is 1 has already been determined. In the same way, the LED whose ID is 5 is determined as shown in Table 3.

```
Algorithm  Sequential blinking
  Blinking_Times = Device_Bit_Length;
  Dictionary Candidate_Group;    // List of Candidate ID
  List Bit_List[];   // Numbers of each Bit

  Step0
    Candidate_Group.Add(Candidate-ID);
  Step1
    for i = 1 to Candidate_Group[i].Count{
      for j = 1 to Bit_List[j].Count{
        if (Bit_List[j].Contain(Candidate_Group[i]))
           contain_times++;}
      if  (contain_times == 1){
        Delete(Candidate_Group[i][k], Candidate_Group);
  Step2
    if (Candidate_Group.Contain(Candidate_Group[i][j])){
      contain_times++;}
      if  (contain_times == 1)
        Temp_Step2_Blinking_List.Add(Candidate_Group[i][j]);
      Candidate_Group[i].minimun(Temp_Step2_Blinking_List);
        Step2_Blinking_List.Add(Candidate_Group[i][j]);
        Delete(Candidate_Group[i][j], Candidate_Group);}}
    if(Step2_Blinking_List.Count > 0){
      Turn_On(Step2_Blinking_List);
      Blinking_Times++;
      to Step1;}
    else{
      to Step3;}
 Step3
    if(Candidate_Group.Count > 0 ){
      Turn_On(Candidate_Group[i][j]);
      Blinking_Times++;
      to Step1;
      Delete(Candidate_Group[i][j], Candidate_Group);}
    else{return(0);}
```

Fig. 8. Pseudo code for Sequential blinking

Step 2: Turn on Candidate IDs that may be contained in only a certain ID

Our method turns on multiple LEDs simultaneously according to the following procedure.

For each Candidate ID, our method counts the number of Detected IDs that have the ID as the Candidate ID. Then, for each Detected ID, if it has Candidate IDs whose count is 1, our method chooses one of the Candidate IDs as the blinking ID. All blinking IDs can be turned on in the same time since there is no possibility they are at the same place. If there is blinking ID(s), the method goes back to Step 1.

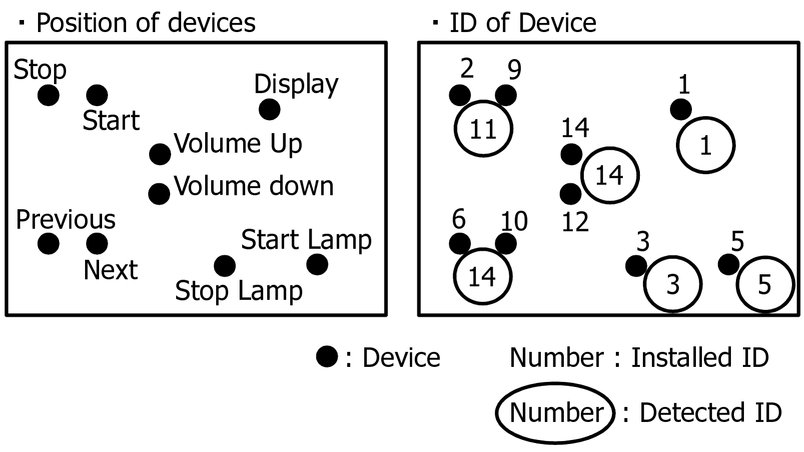

Fig. 9. Example of device layout and detection result

Table 2. State after Step 0

Detected	Candidate	Fixed
1	1	
3	1, 2, 3	
5	1, 4, 5	
11	1, 2, 3, 8, 9, 10, 11	
14	2, 4, 6, 8, 10, 12, 14	
14	2, 4, 6, 8, 10, 12, 14	

Table 3. State after Step 1

Detected	Candidate	Fixed
1		1
3	2	3
5	4	5
11	2, 8, 9, 10, 11	
14	2, 4, 6, 8, 10, 12, 14	
14	2, 4, 6, 8, 10, 12, 14	

In the example, the blinking ID candidates are 6, 9, 11, 12, and 14. IDs of 6, 12, and 14 have the same Detected ID 14, and IDs of 9 and 11 have the same Detected ID 11. Therefore, LEDs whose IDs are 6 and 9 can be turned on simultaneously, and both of them are determined. After returning to Step 1, this step turns on LEDs whose IDs are 11 and 12, and the system knows the position of LED whose ID is 12 and the fact there is no LED whose ID is 11 (see Table 4).

Step 3: Turning on an unfixed ID

The remaining LEDs must be turned on one by one. As such, the system turns on one of the undetermined ID and returns to Step 1. In the example, it turns on the LEDs whose IDs are 2, 4, 8, and 10 in order. After turning on the LED whose ID is 10, Step 1 determines the LED whose ID is 14, as shown in Table 5. To complete the detection, our method takes only 10 blinks, whereas 15 blinks are required if IDs were determined one by one.

Table 4. State after Step 2

Detected	Candidate	Fixed
1		1
3	2	3
5	4	5
11	2, 8, 10	9
14	2, 4, 8, 10, 14	6
14	2, 4, 8, 10, 14	12

Table 5. Results of procedure

Detected	Candidate	Fixed
1		1
3		3
5		5
11		2, 9
14		6, 10
14		12, 14

3.2. *Enhancement using Parity*

Our method takes fewer blinks to detect the positions of LEDs through its integrative use of parallel and sequential blinking. There is another way to reduce the blinks by using additional parity-like bits. It works because sequential blinking takes longer than parallel blinking. This enhancement enlarges the time period of parallel blinking but shortens that of sequential blinking.

We call the method described in Section 3.1 the *Basic method*. Here, we describe four enhanced methods: the *Reverse method*, *Search method*, *Partition method*, and *Pair method*.

Reverse method

The Reverse method adds bits that are the reverse in value of those of the original ID, which means that the length of the encoded ID is twice as long as that of the original ID. During parallel blinking, the method turns on LEDs on the basis of the encoded ID. For example, if the ID is 0010 (= 2), the encoded ID is 00101101, which has additional 4 bits of parity.

During parallel blinking, there is a characteristic that the detection of 0 means all Candidate IDs have 0 in that bit, while the detection of 1 means at least one of the Candidate IDs has 1 in that bit. The Reverse method increases the chance of detecting 0 and thereby dramatically decreases the Candidate IDs. In addition, if the Detected ID is not consistent with its reverse part, our method knows that it has misdetected.

Table 6 shows an example of a misdetection happening because there are two LEDs whose IDs are 9 and 10 within the Misdetection radius. The 2nd and 5th bits did not blink in this case, so the Candidate IDs are $10{*}{*}$ (i.e. 8, 9, 10, or 11), instead of ${*}0{*}{*}$ (i.e. 1, 2, 3, 8, 9, 10, or 11) in the Basic method.

Search method

The Search method adds bits that depended on the value of the low-order n bit to the Reverse method. For example, when the n is two, the method adds four bits according to the patterns of the low-order two bit; 00, 01, 10, and 11. Table 7 is an example that is the same case as the Table 6. On the Reverse method, Candidate IDs were 8, 9, 10, and 11 in this example. This method detects all IDs that have either **01 or **10 in the 10th and 11th blinks. 9 has 01, and 10 has 01 in the low-order two bit. Since there are over two LEDs on the detected point, it can detect two LEDs whose IDs are 9 and 10 directly without turning

Table 6. Example of lighting pattern in the Reverse method

	ID	Binary	Reverse	1st	2nd	3rd	4th	5th	6th	7th	8th
Position	9	1001	0110	ON	-	-	ON	-	ON	ON	-
	10	1010	0101	ON	-	ON	-	-	ON	-	ON
Detected	11	1011	0111	ON	-	ON	ON	-	ON	ON	ON
Candidate		*0**	1***		Fixed			Fixed			

Table 7. Example of lighting pattern in the Search method (2 bit)

	ID	Binary	Reverse	1st	2nd	3rd	4th	5th	6th	7th	8th	9th 00	10th 01	11th 10	12th 11
Position	9	1001	0110	ON	-	-	ON	-	ON	ON	-	-	ON	-	-
	10	1010	0101	ON	-	ON	-	-	ON	-	ON	-	-	ON	-
Detected	11	1011	0111	ON	-	ON	ON	-	ON	ON	ON	-	ON	ON	-
Candidate		*0**	1***		Fixed			Fixed					**01	**10	

on any LED again.

Moreover, Table 8 shows an example of the case where the n is three in the same pattern.

Partition method

The Partition method divides LEDs into two or more groups and parallel blinking is performed per group to decrease the chance of misdetection. For example of dividing LEDs into two groups as to whether the 1st bit of the ID is 1 or not, it uses the Reverse method to detect each unit's position. Table 9 shows an example where LED whose IDs are 2, 3, 10, and 14 are within the Misdetection radius. Our method blinks IDs whose 1st bit is 0, and then blinks the other IDs. Since the 1st bit for all IDs do not have to be blinked, it takes 12 blinkings for parallel blinking, and in total 12 blinks are required to detect all IDs. The Basic method needs 19 blinkings in this case.

Table 8. Example of lighting pattern in the Search method (3 bit)

	ID	Binary	Reverse	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th	13th	14th	15th	16th
												000	001	010	011	100	101	110	111
Position	9	1001	0110	ON	-	-	ON	-	ON	ON	-	-	ON	-	-	-	-	-	-
	10	1010	0101	ON	-	ON	-	-	ON	-	ON	-	-	ON	-	-	-	-	-
Detected	11	1011	0111	ON	-	ON	ON	-	ON	ON	ON	-	ON	ON	-	-	-	-	-
Candidate		*0**	1***		Fixed			Fixed					*001	*010					

Table 9. Example of lighting pattern in the Partition method (1 bit)

	ID	Binary	Reverse	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th
Position	2	0010	1101	-	ON	-	ON	-	ON	-	-	-	-	-	-
	3	0011	1100	-	ON	ON	ON	-	-	-	-	-	-	-	-
	10	1010	0101	-	-	-	-	-	-	-	ON	-	ON	-	ON
	14	1110	0001	-	-	-	-	-	-	ON	ON	-	-	-	ON
Detected (MSB ‘0’)	3	011	101	-	ON	ON	ON	-	ON	-	-	-	-	-	-
Detected (MSB ‘1’)	14	110	101	-	-	-	-	-	-	ON	ON	-	ON	-	ON

When there are four groups, the method divides LEDs according to the patterns of the 1st bit and 2nd bit; 00, 01, 10, and 11. Table 10 shows an example of this case.

Pair method

The Pair method divides an ID in steps of 2 bits, and adds encoded bits according to the patterns of the pair; 00, 01, 10, and 11, after performing the basic method. Table 11 shows an example where LEDs whose IDs are 9 and 10 are nearby each other. This method detects all IDs that have 10** in the 5th, 7th, 9th, and 11th blinks, and detects all IDs that have either **01 or **10. In this case, it can detect two LED whose IDs are 9 and 10 directly without turning on any LED again.

Table 10. Example of lighting pattern in the Partition method (2 bit)

	ID	Binary	Reverse	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th	13th	14th	15th	16th
Position	2	0010	1101	ON	-	-	ON	-	-	-	-	-	-	-	-	-	-	-	-
	3	0011	1100	ON	ON	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	10	1010	0101	-	-	-	-	-	-	-	-	ON	-	-	ON	-	-	-	-
	14	1110	0001	-	-	-	-	-	-	-	-	-	-	-	-	ON	-	-	ON
Detected (‘00’)	3	11	01	ON	ON	-	ON	-	-	-	-	-	-	-	-	-	-	-	-
Detected (‘01’)	-	00	00	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Detected (‘10’)	10	10	01	-	-	-	-	-	-	-	-	ON	-	-	ON	-	-	-	-
Detected (‘11’)	14	10		-	-	-	-	-	-	-	-	-	-	-	-	ON	-	-	ON

Table 11. Example of lighting pattern in the Pair method

	ID	Binary					00		01				11	
			1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th
Position	9	1001	ON	-	-	ON	-	-	-	ON	ON	-	-	-
	10	1010	ON	-	ON	-	-	-	-	-	ON	ON	-	-
Detected	11	1011	ON	-	ON	ON	-	-	-	ON	ON	ON	-	-

4. Implementation

We implemented a prototype system incorporating our method. The prototype consisted of a camera, multiple devices, and conductive cloth (TextileNet). Figure 10 shows a snapshot of using the prototype. We used a Lenovo ThinkPad X200 (CPU 2.4 GHz, RAM 3.0 GB), with Microsoft Windows 7 operating system, Microsoft Visual C++.NET 2005 and OpenCV[28] to implement our method and the application. We used a Logicool Qcam Orbit AF web camera to capture images. The prototype turned on LEDs installed on TextileNet's conductive fabric via Bluetooth, and detected the position of the LEDs automatically. We confirmed that the proposed method correctly worked. The prototype turns on LEDs two

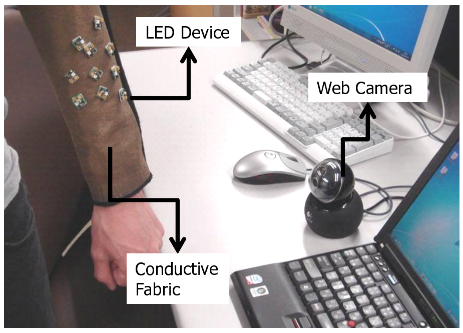

Fig. 10. Snapshot of a user wearing prototype

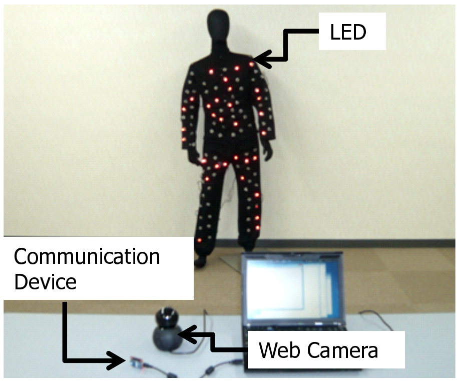

Fig. 11. Snapshot of user wearing LEDs

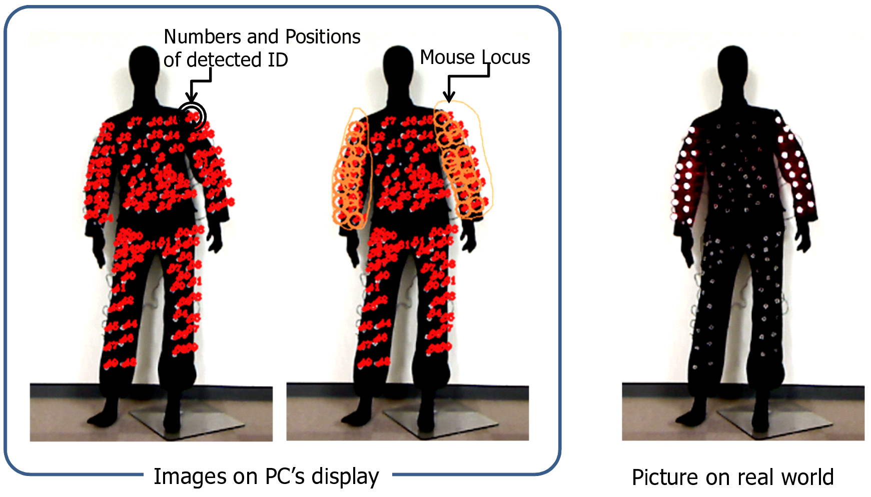

Fig. 12. Snapshots of using the application

times per second because the web camera needed a period of time to detect light.

In addition, we implemented an application for controlling a lot of LEDs on the cloth flexibly and intuitively by GUI. Figure 11 and 12 show snapshots of using the prototype. The number of worn devices was 100, and each device had 7-bit IDs. Since our method automatically detects the positions of all LEDs, we can control the LED states by drawing a circle to decide the area in which to turn on the LEDs.

5. Evaluation

We implemented a simulator in which we could set the number of devices, detection methods, sizes of the misdetection radius, ID length, and resolution of the camera. We evaluated our methods from the viewpoint of blinking time of LEDs, which represents the time to

Table 12. Parameters for evaluation

Number of devices	0∼250
Resolution of camera	640×480 [pixel]
Misdetection radius	10, 20, 30 [pixel]
Bit length of ID	7∼12 [bit]
Trial times	1000

complete the detection. This is because the user of our system stands still in front of the camera, and people cannot stand still for a long time. The comparative method, called the *Conventional method* turned on LEDs one by one. Table 12 shows the parameters of the evaluation.

5.1. *Effect on Bit Length*

If there are a lot of different possible devices, the bit lengths for identifying the devices would be long. Since all devices should have own IDs, the length of the ID is determined by the application model, and the length has a significant effect on the performance of position detection. Therefore, we evaluated the blinking times by varying the ID length. Figure 13 shows the result. When there were a few misdetections because few devices were placed on the conductive fabric, the Reverse method performed best since it had fewer parallel blinkings. When Misdetection radius was large, the ID length was long, or the number of devices were large, the Pair method or the Partition method (2 bit) were good since it could reduce sequential blinking by adding bits in parallel blinking procedure. The Partition method (2 bit) is better than the Pair method on the case that the number of misdetection is large.

In most cases, the enhanced methods that had several types of parity checking had much higher performance compared with the Conventional method. These results confirmed the effectiveness of the proposed method and the enhancements.

5.2. *Effect on Number of Devices*

The number of installed devices changes according to the number of applications to be used and the type of applications. Therefore, we evaluated the blinking times while varying the number of worn devices. Figure 14 shows the results of the evaluation. If a user can stand still for one minute, the limit of blinking times is 120 in our setting. If the length of the ID is 8 bits and the Misdetection radius is 10 pixels, every method can detect more than 250 IDs within one minute. If the Misdetection radius is 20 pixels and 30 pixels, then the Partition method (2 bit) can detect all IDs and 160 IDs within one minute, respectively. On the other hand, if the length of the ID is 12 bits and the Misdetection radius is 10 pixels, 20 pixels and 30 pixels, then the Pair method can detect 100 IDs, 40 IDs and 20 IDs within one minute, respectively. When the length of the ID is 12 bits and the Misdetection radius is 30 pixels, the number of blinks increases. Nevertheless, while the Conventional method

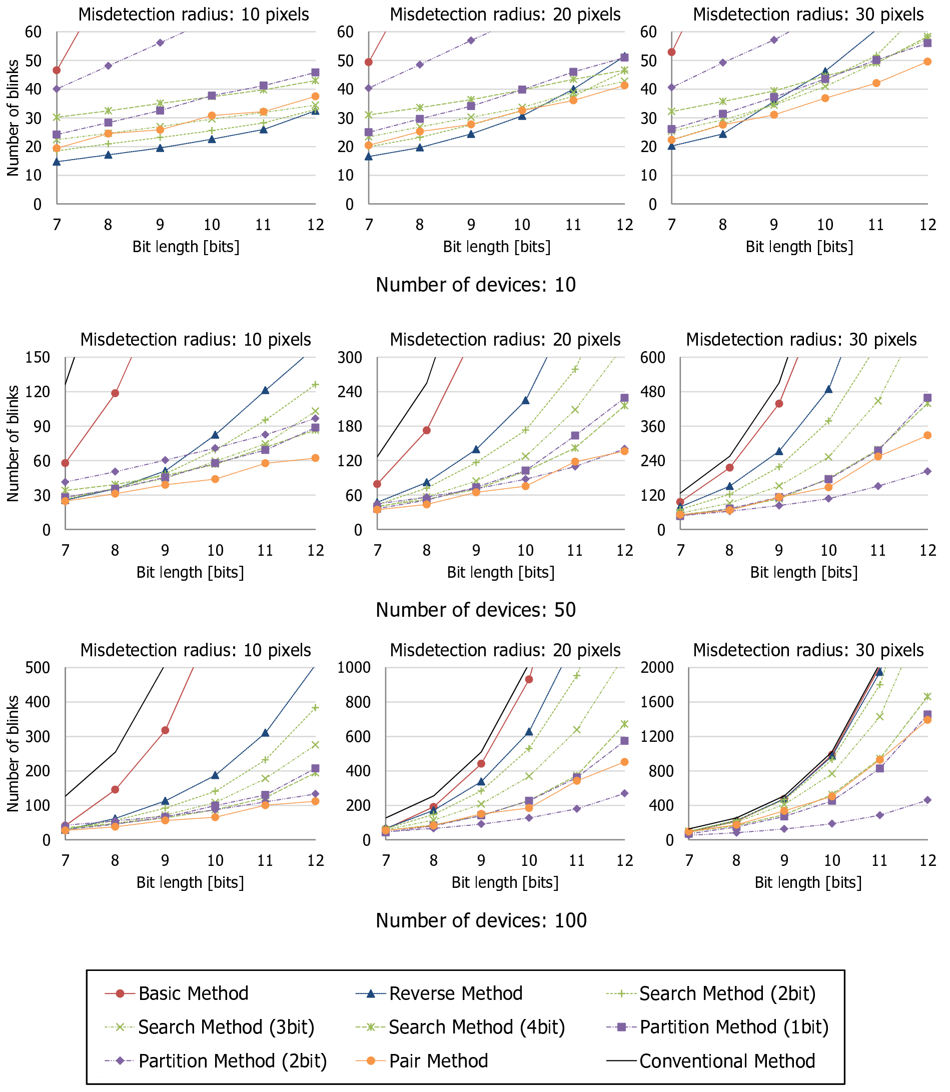

Fig. 13. Effect of varying bit length

needs more than 4000 blinks, the Partition method (2 bit) can detect 100 IDs in only 10% on it.

In addition, when the ID length is 8 bits, the number of blinks converges to a certain number. This is because too many Candidate IDs destroys the advantage of parallel blinking.

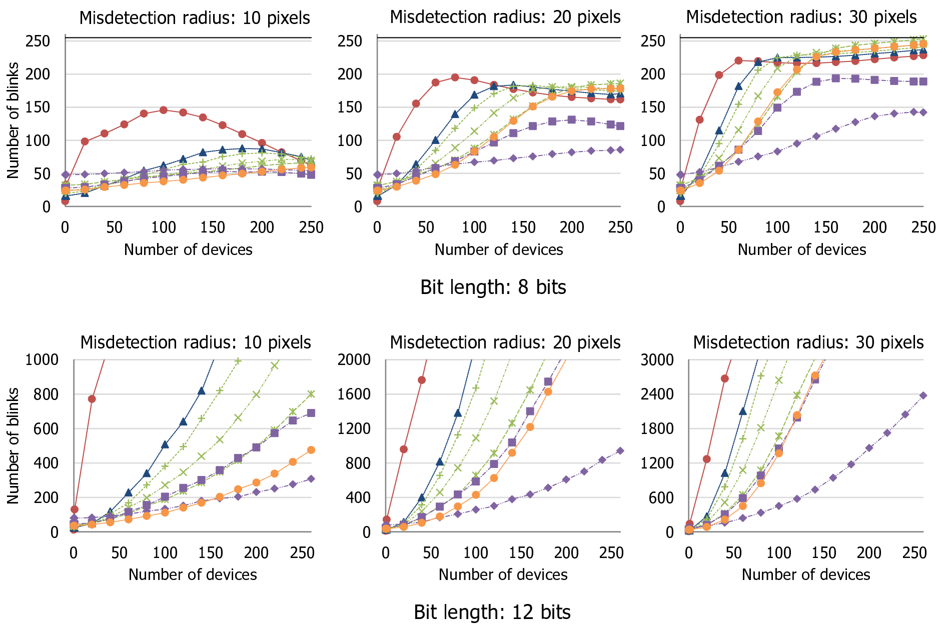

Fig. 14. Effect of varying number of devices

5.3. *Effect on Misdetection radius*

Depending on various environmental conditions such as the surrounding intensity of illumination and the capability of camera, the system needs to change the Misdetection radius. Therefore, we evaluated the blinking times while varying the Misdetection radius. Figure 15 shows the results. When the ID length is 8 bits and the number of devices is 10, all method except for the Basic method can detect all devices within one minute even though Misdetection radius is 50 pixels. If the number of devices is 50 and 100, then the Partition method (2 bit) can detect all IDs within one minute when the Misdetection radius is more than 50 pixels and less than 40 pixels, respectively. Here, the setting whose Misdetection radius is 40 pixels can cope with most detection environments. When the ID length is 12 bits and the number of devices is 10, most methods detect all devices within one minute even though Misdetection radius is 50 pixels. If the number of devices is 50 and 100, the Partition method (2 bit) can detect all IDs within one minute when the Misdetection radius is less than 10 pixels.

6. Conclusions

In this paper, we proposed a method for detecting the positions of devices on a conductive cloth by using a camera. Our method blinks the LEDs on the devices according to their IDs and detects their positions. We proposed several detection algorithms for reducing the

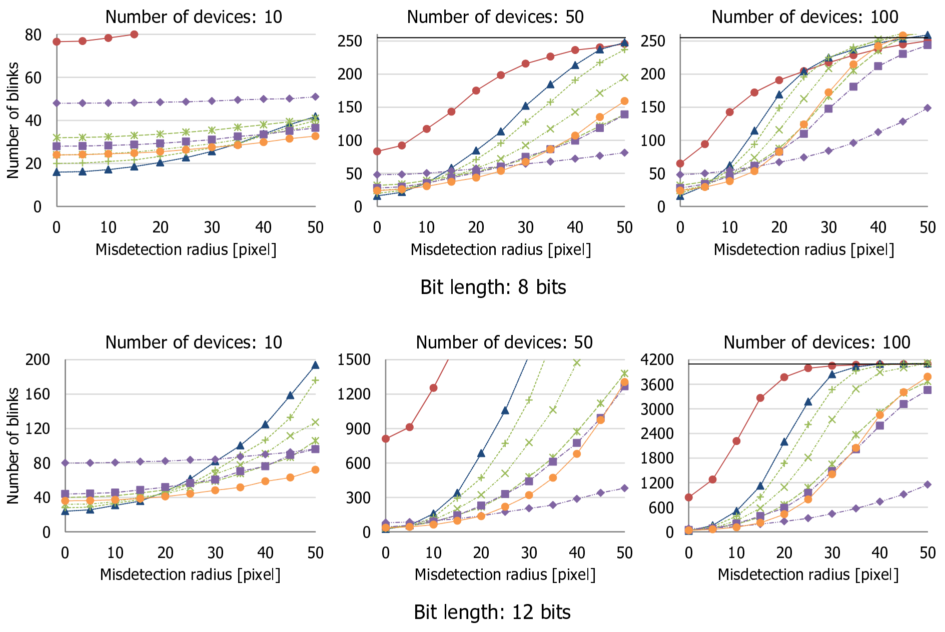

Fig. 15. Effect of varying Misdetection radius

total number of blinks. Evaluation result using an implemented simulator confirmed that our methods are more efficient than a conventional serial blinking method. By using the proposed method, it allows to construct applications that utilize the positions of devices on the wear.

Applying this method to the wearable computing system that uses the conductive fabric, the users can operate the devices only by placing the devices on a cloth and standing in front of a mirror.

In the future, we will consider a method that can cope effectively with errors in which LEDs do not turn on in spite of having received a signal to turn on. Additionally, we will construct applications that utilize the positions of devices.

References

1. A. Chandler, J. Finney, C. Lewis, and A. Dix: Toward Emergent Technology for Blended Public Displays, *Proc. of the 11th International Conference on Ubiquitous Computing (UbiComp 2009)*, pp. 101–104 (Sep./Oct. 2009).
2. A. Hatami and K. Pahlavan: Comparative Statistical Analysis of Indoor Positioning Using Empirical Data and Indoor Radio Channel Models, *Proc. of the 3rd Consumer communications and networking conference (CCNC 2006)*, pp. 1018–1022 (Jan. 2006).
3. C. Mattmann, F. Clemens, and G. Tröster: Sensor for Measuring Strain in Textile: *Sensors*, vol. 8, no. 6, pp. 3719–3732 (June 2008).
4. D. L. Ipina, P. Mendonca, and A. Hopper: TRIP: a Low-Cost Vision-Based Location System for

Ubiquitous Computing, *Personal and Ubiquitous Computing*, Vol. 6, No. 3, pp. 206–219 (May 2002).

5. E. Wade and H. Asada: Conductive Fabric Garment for a Cable-Free Body Area Network, *Proc. of the 5th International Conference on Pervasive Computing (Pervasive 2007)*, Vol. 6, No. 1, pp. 52–58 (May 2007).
6. F. Helin, T. Hoglund, R. Zackaroff, M. Hakansson, S. Ljungblad, and L. E. Holmquist: Supporting Collaborative Scheduling with Interactive Pushpins and Networking Surfaces, *Proc. of the 6th International Conference on Ubiquitous Computing (UbiComp 2004)*, demo (Sep. 2004).
7. G. Lee, Y. Ahn: An LED display Using Active Reflectors and Free-Space Optical Transmission, *Proc. of the 35th International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2008)*, Posters, Article No. 46 (Aug. 2008).
8. J. Berzowska and M. Coelho: Memory-Rich Clothing, *Proc. of the 24th International Conference on Human Factors in Computing Systems (CHI 2006)*, pp. 275–278 (Apr. 2006).
9. J. Lifton, D. Seetharam, M. Broxton, and J. Paradiso: Pushpin Computing System Overview: A Platform for Distributed, Embedded, Ubiquitous Sensor Networks, *Proc. of the 1st International Conference on Pervasive Computing (Pervasive 2002)*, pp. 139–151 (Aug. 2002).
10. J. Rekimoto and Y. Ayatsuka: CyberCode: Designing Augmented Reality Environments with Visual Tags, *Proc. of ACM Designing Augmented Reality Environments (DARE 2000)*, pp. 1–10 (Apr.2000).
11. K. Matsui, T. Terada, and S. Nishio: User Preference Learning System for Tangible User Interfaces, *Proc. of the 3rd International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS 2009)*, pp. 766–771 (Mar. 2009).
12. K. Nakatsuma and H. Shinoda: High Accuracy Position and Orientation Detection in Two - Dimensional Communication Network, *Proc. of the 28th International Conference on Human Factors in Computing Systems (CHI 2010)*, pp. 2297–2306 (Apr. 2010).
13. K. V. Laerhoven, N. Villar, A. Schmidt, H. W. Gellersen, M. Hakansson, and L. E. Holmquist: Pin&Play: The Surface as Network Medium, *IEEE Communications Magazine*, Vol. 41, No. 4, pp. 90–96 (Apr. 2003).
14. L. Buechley, M. Eisenberg, J. Catchen, and A. Crockett: The LilyPad Arduino: Using Computational Textiles to Investigate Engagement, Aesthetics, and Diversity in Computer Science Education, *Proc. of the 26th International Conference on Human Factors in Computing Systems (CHI 2008)*, pp. 423–432 (Apr. 2008).
15. M. Broxton, J. Lifton, and J. A. Paradiso: Localizing a Sensor Network via Collaborative Processing of Global Stimuli, *Proc. of the 2nd European Worlshop on Wireless Sensor Metworks (EWSN 2005)*, pp. 321–332 (Jan./Feb. 2005).
16. M. Fujimoto, N. Fujita, T. Terada, and M. Tsukamoto: Lighting Choreographer: an LED Control System for Dance Performances, *Proc. of the 13th ACM international Conference Adjunct Papers on Ubiquitous Computing (Ubicomp2011)*, (Sep. 2011).
17. M. Nakata, K. Kodama, N. Fujita, Y. Takegawa, T. Terada, M. Tsukamoto, S. Hosomi, and S. Nishio: Design and lmplementation of a Ubiquitous Optical Device Controlled with a Projector, *Proc. of the 6th International Conference on Advances in Mobile Computing and Multimedia (MoMM 2008)*, pp. 130–135 (Nov. 2008).
18. N. B. Bharatula, P. Lukowicz, and G. Tröster: Functionality-Power-Packaging Considerations in Context Aware Wearable Systems, *Personal and Ubiquitous Computing*, Vol. 12, No. 2, pp. 123–141 (Jan. 2008).
19. N. Matsushita, D. Hihara, T. Ushiro, S.Yoshimura, J. Rekimoto, and Y. Yamamoto: ID CAM: a Smart Camera for Scene Capturing and ID Recognition, *Proc. of the 2nd International Symposium on Mixed and Augmented Reality (ISMAR 2003)*, pp. 227–236 (Oct. 2003).
20. R. Ueoka, H. Kobayashi, and M. Hirose: SoundTag: RFID Based Wearable Computer Play Tool for Children, *Journal of Transactions on Edutainment III*, Vol. 5940, pp. 36–47 (2009).

21. S. Baurley, P. Brock, E. Geelhoed, and A. Moore: Communication-Wear: User Feedback as Part of a Co-Design Process, *Proc. of the 2nd Haptic and Audio Interaction Design (HAID 2007)*, Vol. 4813, pp. 56–68 (Nov. 2007).
22. S. R. Klemmer, J. Li, J. Lin, and J. A. Landay: Papier-Mache: Toolkit Support for Tangible Input, *Proc. of Conference on Human Factors in Computing Systems (CHI2004)*, pp. 399–406 (Apr. 2004).
23. T. Murakami, J. Akita, and M. Toda: Power Line Communication Transceiver on Conductive Wear for Wearable Computing, *International Transactions on Systems Science and Applications (ITSSA)*, Vol. 4, No. 3, pp. 287–291 (Oct. 2008).
24. T. Roos, P. Myllymaki, H. Tirri, P. Miskangas, and J. Sievanen: A Probabilistic Approach to WLAN User Location Estimation, *International Journal of Wireless Information Networks (IJWIN)*, Vol. 9, No. 3, pp. 155–164 (July 2002).
25. Y. Kishino, T. Terada, N. Villar, H. W. Gellersen, and S. Nishio: A Position Detection Mechanism enabling Location-aware Pin&Play, *International Journal of Smart Home (IJSH)*, vol. 1, No. 1, pp. 31–39 (Apr. 2007).
26. ARToolKit, `http://www.hitl.washington.edu/artoolkit/`.
27. Musical Jacket, `http://opera.media.mit.edu/levis/`.
28. OpenCV, `http://opencv.jp/`.
29. PhoeniX Technologies: The Visualeyes System, `http://ptiphoenix.com/`.