Keio University Bachelor's Thesis Academic Year 2011

Vib-Connect : A Device Selecting Interface Using Vibration

Hiroshi Nakahara

Faculty of Environment and Information Studies

Keio University

5322 Endo Fujisawa Kanagawa 252-8520 JAPAN

Submitted in partial fulfillment of the requirements for the degree of Bachelor

Advisors:

Professor Hideyuki Tokuda

Professor Jun Murai

Associate Professor Hiroyuki Kusumoto

Professor Osamu Nakamura

Associate Professor Kazunori Takashio

Assistant Professor Noriyuki Shigechika

Assistant Professor Rodney D. Van Meter III

Associate Professor Keisuke Uehara

Associate Professor Jin Mitsugi

Lecturer Jin Nakazawa

Professor Keiji Takeda

Abstract of Bachelor's Thesis Academic Year 2010

Vib-Connect : A Device Selecting Interface Using Vibration

Summary

This thesis introduces a vibration based interface that enables end-users to select device easily. Due to the recent progress in Information Technology, various devices such as wireless speaker, wireless projector and wireless photoframe, have joined the wireless network. Those wireless-connectible devices enable end-users to enjoy flexible and convenient services, but however, there is no intuitive interface to select those devices to collaborate. Current device selecting interfaces usually make users to input or select the device with information such as IP addresses or model name of the device, and that is a great burden for end-users in using device collaboration. Vib-connect solves these problem by implementing these device information as a unique vibration pattern. By keeping smart-phones or any kind of vibration generating device within touch, end-users can easily select devices to collaborate. We have implemented a prototype, and got a great accuracy in vibration pattern recognition, and a high satisfaction by non-expert users in usability evaluation. With Vib-connect, even users without any kind of technical expertize can select devices to collaborate easily, and therefore it will greatly contribute to the ubiquitous computing environment.

Keywords:

1 Interaction 2 Interface 3 Vibrate Communication

4 Accelerometer 5 Ubiquitous Computing

Keio University Faculty of Environment and Information Studies

Hiroshi Nakahara

学士論文要旨　　2011 年度 (平成 23 年度)

Vib-Connect
振動を用いた機器指定インタフェース

論文要旨

本論文は，振動を用いた機器指定インタフェースを提案する．近年の情報技術の発達により，無線スピーカ，無線プロジェクタ，無線フォトフレームなどをはじめとして様々な機器が無線ネットワークの一員となった．これらの無線対応機器によって柔軟で便利な機器連携サービスが利用可能となったが，エンドユーザにとっても直観的な，機器指定インタフェースは存在しない．現状の多くの機器指定インタフェースは接続に必要な機器情報をユーザに入力させたり，型名リストから選択させるようにしており，エンドユーザが機器連携を利用する際に多大な負荷をかけている．Vib-connect は機器の連携に必要な情報を固有の振動パターンとして埋め込む事で，この問題を解決する．ユーザは，スマートフォンや振動を発生させるデバイスを連携したい機器に触れさせる事で，容易に連携させたい機器を指定する事ができる．我々は，Vib-connect のプロトタイプを実装し，振動パターンにおいて非常に高い精度を得たほか，機器連携の経験がない初心者も対象としたユーザビリティ評価において高い満足度を得た．Vib-connect は，技術的な専門知識のないユーザでも容易に連携させたい機器を指定する事を可能にし，ユビキタス環境の実現に貢献する事を目的とする．

キーワード：

1 インタラクション　2 インタフェース　3 振動通信
4 加速度センサ　5 ユビキタスコンピューティング

慶應義塾大学大学院　環境情報学部
中原 洋志

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Ubiquitous computing is the method of enhancing computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user. " - Mark Weiser [1]

1.1 Background

Due to the recent progress in Information Technology, many devices around us have become a member of wireless network. WirelessHD[2] and TransferJet[3], for example, are typical techniques supporting wireless device collaboriton today. Thanks to these technologies, nowadays we can access and transfer heavy contents such as videos and photos, over wireless networks.

Companies and electronic equipment manufacturers too, have contributed to enable devices to collaborate, by deciding standards and ways for connecting devices, such as DLNA[4]. They also provide us many smart devices such as cellular phones that has connectivity in both Wi-fi and bluetooth, printers that speak IP, TV which are able to access youtube just as we access to HDD recorders. Even photoframes can be linux-based and able to transfer photos over wireless networks. All of these smart devices and smart furniture can connect, communicate, collaborate, and control each other over wireless networks.

They not only provide those devices, but also propose various device collaborating styles and services. For example, Panasonic provides device collaborating style called Viera-link[5]. In Viera-link, users can use various devices over network by TV. For example, users can recept visitor, and access videos in HDD recorders and cameras through a TV in the living room. Not only Panasonic, but many electronic equipment manufacturers proposes various device collaborating style.

With these technologies, devices, and proposals, various ways of device collaboration and services have become possible and became familiar to users. Today, users watch the state of home using web-cams via smartphones outdoors, print documents over wireless network at their office, use and control speakers throuh TV in their living room, and transfer photos taken by digital camera to photoframes over wi-fi network in their bedroom.

These tendencies are expected to progress, and more and more devices would join the wireless, and various device collaborating styles and services would appear. These device collaborating styles are estimated to spread even to non-expert users.

1.2 Motivation and Objective

Even though there are many wireless-connectible devices and device collaborating styles, interfaces for selecting devices are far from being intuitive, and they are being a great burden for users in using device collaboration. In this thesis, we will aim to propose an intuitive, vibration-based interface to select devices, which enables non-expert users to select devices easily and simply.

Before the appearance of wireless network technology, cables were used to collaborate devices. Cables not only had a function of physical route for communication, but also had an function of device selecting interface. Plugging cables to devices meant to select the device as a member of collaboration, and the existence of the cable was an unmistakable indicator of relationship between devices.

After the appearance of wireless network technologies, cables disappered and users were released from limits of space and obstacles. Though, this also meant the disapperance of an intuitive device selecting interfaces. Interfaces such as having users to input the IP address, or having users to select device from a list of device, became popular, and took place of the cables.

In these on-screen interfaces, it is difficult for non-expert users to match the device relationship described in the interface, with the actual relationship in the virtual space. For example, suppose that there are three printers, which were made by the same company, have the same stock number, and look the same. When user wants to use the printer in the right hand, which is the only printer that has enough stock or papers and ink, it is difficult for him or her to select the correct printer from a list of devices such as " EPSON LP-S2000 ", " EPSON PX-5500 ", and " EPSON PX-G900 ". It will be easier for users if they can make relationship of devices, just as they do in real world, such as touching each devices they want to connect.

Therefore, we would like to propose an intuitive, vibration-based device selecting interface in this thesis. In this interface, user can make relationships and connection between device by simply pasting or putting on a small vibration-device, called Vib-connector. Users will be released from the complicated task of matching virtual connection of devices with the relationship in the real world. In addition, since this interfaces uses Vibration, it will be tough against the environmental change, such as lights and sounds.

1.3 Organization

This thesis is organized as following. Chapter 2 explains briefly about the background and the problems of the existing device selecting interfaces. In chapter 3, we will explain the concept of Vib-connect. Actual design will be described in chapter 4, and details about the prototype implementation will be provided in chapter 5. In chapter 6, we will explain about the evaluation and their results. At chapter 7, we will describe about future works and conclude this thesis.

Chapter 2

Device Selecting Interfaces

First, this chapter gives a brief overview of device collaboration and interfaces to select devices. We will go over the background of device collaboration, and explain about the device selecting interface for ubiquitous computing environment. Then existing methods for device selecting interface will be shown and their limitations will also be discussed. At the end, we will state the problems which this paper focuses.

2.1 Background

We will describe briefly about the background of device selecting interfaces in this section. Requirements of device selecting interfaces for ubiquitous computing environment will be described in the next section.

2.1.1 Progress in Ubiquitous Computing

Recent progress in information technology enabled various devices around us to compute and communicate. These are not only devices such as computers and smart phones, but also devices such as TV, photoframes, and speakers, which were not albe to communicate each other. This meant that users got access to those new services over network. Naturally, desire of collaborating devices and using those services became larger and larger.

Companies and electronic equipment manufactures have answered to these desires by deciding standards and ways for connecting devices, such as DLNA[4], and provided various devices supporting these standards. Also, they propose new device collaborating styles everyday. For example, Panasonic proposes a device collaborating style called Viera-link[5]. Not only Panasonic, but many electronic equipment manufacturers proposes various device collaborating style.

With the preparation and expansion of environment for device collaboration, more and more types of services and device collaborating style will appear and spread over users. Also, with those proposals and promotion of new device collaborating styles, even non-expert users such as elderly and childr, will be expected to use device collaboration.

2.1.2 Necessity of Intuitive Device Collaborating Interface

As mentioned in 2.1.1, more and more collaborative devices are expected to appear, and their services are also expected to increase. Also, by the proposals and promotion by companies and electronic equipment manufacturers, various styles of device collaboration will appear and spread to users. These users are not only expert users who are used to connect and handle devices; non-expert users who are not used to connect and handle devices, such as elderly and child, will use those device collaboration and enjoy their services.

Since the number of collaborative devices and services are increasing, and more non-expert users are expected to enjoy those services, device collaboration is expected to be more intuitive. As doing so, users will be able to collaborate those increasing devices more efficiently, and non-expert users will be able to connect devices more easily.

In order to make device collaboration more intuitive, an intuitive device collaborating interface that matches to ubiquitous computing environment, is needed. In the next section, we would like to explain about device selecting interface, which is necessary for intuitive device collaborating interface.

2.2 Device Selecting Interface in Ubiquitous Computing Environment

In this section, we would like to discuss about device selecting interfaces in ubiquitous computing environment. First, we will explain about selecting devices in device collaboration, and then describe four requirements for device selecting interfaces.

2.2.1 Selecting Devices to Collaborate

To connect and collaborate devices, users must first select the correct device to connect and collaborate. Since the number of collaborative device and service is increasing, and more non-expert users are expected to use device collaboration, an intuitive and simple device selecting interface is needed. This thesis focuses on proposing an interface for "selecting" these service devices and user devices.

Before discussing about the requirements for device selecting interface in ubiquitous computing environment, we define two types of devices that constitutes device collaboration; 1)Service Device, and 2) User Device.

1. **Service Device:**

   Service devices are devices which users want to control and use their service in remote. These devices are devices which provide service, such as TV, projectors, speakers, and photoframes. Usually, these devices are the target which users want to use, in device collaboration.

2. **User Device:**

   User devices are devices which users use to control the service devices. These are devices which are usually used to control service devices remotely, such as computers, smart phones, and tablet PCs.

These are the two types of devices this thesis assumes. Next, we would like to discuss about the requirements for device selecting interface in ubiquitous computing environment.

2.2.2 Requirements for Device Selecting Interface In Ubiquitous Computing Environment

Before showing some existing methods of device selecting interface, we would like to introduce four requirements for device selecting device selecting interface in ubiquitous computing environment ; 1) Less burden in recognizing and according devices, 2)more multimodal interface, 3) robustness against environmental influence, and 4) easiness in recognizing connections and relationships between devices.

1. **Less Burden in Recognizing and According Devices:**

   On selecting devices to collaborate, users must first recognize the correct device to collaborate. This step is usually taken unconsciously in real world, by visual information such as the appearance and location of the devices. Though, if these devices are expressed in a different way in the virtual world by the interface, such as list of device names, users usally get the wrong impression about the devices. To accord the impression which user get from the interface, with recognition in real world, users must associate and imagine what device is actually expressed in the interface.

   These recognizing and according steps are generally difficult for non-expert users. Since more and more non-expert users are expected to collaborate device, these associating steps would be better the less, or none.

2. **More Multimodal Interface:**

   Previous devices that uses devices, requires specific I/O ports such as USB or memory stick. Though, more and more types of collaborative devices, including thin client such as tablet PC and smart phones which don't have those I/O is increasing. Device selecting interface for ubiquitous computing environment must correspond to those devices too.

Since these thin clients are expected to increase, it will be better for device selecting interfaces, to be able to use other ways except of those existing I/O ports.

3. **Robustness against environmental influence:**

   As described in the previous section, more and more collaborative devices and services are increasing, and spreading around us. Since these device collaboration take place in various environment around us, device selecting interface must be robust against those environmental influence, such as noise and illumination.

   It will be more convenient if device selecting interface in ubiquitous computing environment is more robust against environmental influences.

4. **Easiness in Recognizing Connections and Relationships between Devices:**

   This is the requirement after the devices are being selected once. In case user connected to wrong device, it will be better for him/she to recognize the connection and relationship between devices easily. When user connects to wrong device, there is no way for him/she to know the error, since wireless connections cannnot be seen to person's eye. They have to use the device once to notice that he/she connected to the wrong device, and take time and trouble in error handling.

   Since collaborative devices and services are increasing, it will be better if these recognition of connection and relationship between devices, is easier.

2.3 Existing Methods for Device Selecting Interfaces

All of existing methods for device selecting interfaces can be classified into four types, Name Selecting Interface, Motion Selecting Interface, AR Selecting Interface, and Device Based Interface. After describing some representative of those methods, we will discuss about the problem and limitation of them.

Name Selecting Interface

First is name selecting interface. These interfaces are seen in Bluetooth[6] applications. These interfaces show a list of device names or device ID on the screen. In these interfaces, users will select the device that he or she wants to connect, from the provided list. Users don't have to know about the network information about the devices, such as IP addresses or MAC addresses. Device names can be fixed, if the user has permission.

Motion Selecting Interface

Second is motion selecting interface. These are more intuitive interfaces that uses user's everyday-motion. In these interfaces, users can recognize the devices just as they look at them with eyes in the real world. Users don't have to take those difficult associating steps in recognizing the device.

We will go through the representative researches briefly. Ubi-finger[7], by wearing on special gloves, users can select devices by "pointing" at them. Gaze-Link[8] enabled users to select devices by "gazing" at them. In Snappy[9], users can select devices by "swinging" his mobile phone according to a specific motion. SyncTap[10] enables users to select devices by simply "tapping" both devices at the same time, and Smart-Its Friends[11] do this by "swinging" them at the same time.

AR Selecting Interface

Third is AR (Augmented Reality) Selecting interface. These interfaces use visual markers, and enables users to select devices through virtual sights. In this way, users are able to recognize and select the devices, just as they look at them with eyes in the real world. Also, as they use augmented reality, it has possibility to easily show the connection and relationships between devices, to users.

These are some representative researches of these types. For example, in u-photo[12], users were able to select and recognize the devices and services in virtual sights, by looking through cameras. EVANS[13] also uses visual markers, to visualize the wireless connection.

Device Based Interface

Last is Device Based Interface. These interfaces use a specific device that matches I/O ports of PC, such as USB or memory sticks, as an indicator in the real world. They can easily recognize and select the device by only plugging those specific device. Also, as the specific device can be indicator in the real world, and user can easily recognize which device is being connected.

These are some representative researches of these types. In wivia[14] and P-stick[15], users can select PC that shares the screen over network, by plugging in a small USB-based device. TranStick[16] used memory stick instead of USB. In CVNC[17], they used a felica card which has specific function, and required felica port. Touch and Connect[18] puts a button nearby the device and simply let the user push the buttons.

2.4 Problem Statements of Existing Device Selecting Interface

In this section, we will describe problem statements of existing device selecting interface. This thesis focuses on solving four problems in device selecting interface : 1) Burden in recognizing and according devices, 2) narrowness of correspondence, 3) frailness against environmental influence, and 4) difficulty in recognizing device relationships. These are important problems to solve in making intuitive device selecting interface, and each of them will be described in following.

1. **Burden in recognizing and according devices:**

In most of the previous device selecting interfaces, users have to recognize and accord the correct device from the scarce information given from the interface. These steps are nothing but burden for end users, and usually make them mistake in selecting devices or make them give up collaborating devices.

For example, in Bluetooth Software Suite[6], users still have to associate the correct devices from those list of device names, such as "STB2819" and "DR-BT50". Also, it is too difficult and would be burden for non-expert users to recognize the connection and relationships between devices, from those expressions.

If there are interfaces that user can recognize and select device just as they watch in their eyes, users won't have to take the burden step of associating devices.

2. **Narrowness of correspondence:**

   Device based interfaces use specific I/O ports such as USB and memory stick for identifying devices. Users can select device intuitively and certainly by plugging in those specific device. Though, thin clients which don't have those I/O ports, such as tablet PC and smart phones, are increasing.

   For example, existing device based interfaces[14][15][16], plugged devices into specific I/O ports, such as USB and memory sticks, to select devices. As these devices were both indicator in the real world and identifier in the virtual world, it was easy to recognize the devices and their relationships.Though, as these interfaces used specific I/O ports, it was impossible to correspondence with thin clients such as tablet PC and smart phones.

   Since these thin clients are expected to increase and spread more and more, device selecting interface would be better if they are more multimodal.

3. **Frailness against environmental influence**

Since device and device collaboration are spreading around us, these device selecting interfaces must not be frail against environmental influence, such as noise and illuminations.

For example, in existing AR interfaces[12][13], users were able to select devices just as they look in their own eyes. Though, as they used visual markers and cameras to let system recognize the device, they were easily influenced by illuminations, and obstacles.

Since device collaboration is going to take place in various situations, such as noisy office, bright living room, and dark bedroom, they must be robust against those environmental influence.

4. **Difficulty in recognizing device relationships:**

   In most of the previous device selecting interfaces, users couldn't easily recognize the connection and relationship between the collaborating devices.

   Several works[7][8][9][10][11], can be gave as an example. These interfaces enabled users to select devices intuitively. Users don't have to associate to recognize devices, and can select them by everyday-motions such as "pointing" , "gazing", "swing", and "tapping". Though, as there is no indicator of connection and relationship in the real world, users cannot recognize the connection and relationship of collaborating devices easily.

   Without recognizing the connection and relationship of collaborating device, it will take more time and trouble in error handling. For example, suppose that there are color printer and monochrome printer, and the user is trying to use the monochrome printer. If user connects to the color printer in an accident, he wouldn't know that he connected to the wrong device until he wastes times, and also, some papers and valuable color inks. In order to make those error handling more easier, device selecting interface must be easy to recognize the connection and relationship between the collaborating devices.

2.5 Summary

This chapter described briefly about the background of device selecting interface, and gave some examples of device selecting interfaces. Limits of those device selecting interfaces were discussed, and the problem this thesis is trying to solve, was described. In the next chapter, we will describe the view and the concept of my proposal, vib-connect.

Chapter 3

Vib-Connect

This chapter describes about the aim of this research. We will first describe the aim of this thesis, and next, the aim and concept of Vib-connect.

3.1 Aim of this research

As mentioned in Chapter 1 and 2, recent progress in information technology enabled various devices to join the wireless network. The number of services that can be used over wireless network, is increasing and expected to increase more. In the future, more and more device collaboration will be expected in our everyday life such as in office, living room, and bed room.

Also, with the promotion and proposal made by the electronic equipment manufacturers, many device collaboration styles had appeared, and spread to users. Not only the users who are familiar with connecting and handling device, but also non-expert users such as elderly and children are going to collaborate devices in their daily life.

Therefore, a new intuitive, easy device selecting interface that better matches these ubiquitous computing environment, is needed. The aim of this research is to propose a better matching interface to "select" devices, for these ubiquitous computing environment. Proposing a better matching intuitive device selecting interface, it will be easier for non-expert users to select device and collaborate devices. By preparing an comfortable environment for device collaboration, we would like to contribute for the ubiquitous computing society.

In the next section, we will describe the aim and concept of Vib-connect.

3.2 Concept of Vib-Connect

Vib-connect focuses on solving the four problem statements described in Chapter 2 ; 1) Burden in recognizing devices, 2) Narrowness of correspondence, 3) Frailness against environmental influence, and 4) Difficulty in recognizing device relationships. We aim to solve these problems, by proposing Vib-connect, a device selecting interface using vibration.

In Vib-connect, we use a small device called Vib-connector, which mediates the relationship and connection between the service device and user device. Vib-connector has function as an indicator in the real world by glowing in different colors, and also has function as an identifier in the virtual world by vibrating in a specific pattern. Vib-connector is best if it is small enough to hold in hands.

By using Vib-connect, users will be able to select device with simple touch & paste interaction. User will be free from those burden of associating devices to recognize, and will be able to recognize the connection and relationships between devices, by the existence of Vib-connector. Also, as Vib-connect uses vibration to identify devices, it will be robust against environmental influence such as illuminations and noise.

We will describe the details of the Vib-connect interaction in the next section.

3.3 Vib-Connect Interaction

This section will describe the detail of the Vib-connect interaction. First we will explain about Touching & Pasting action, and then describe how users will recognize the device relationship and connections.

3.3.1 Touching & Pasting

As used in [18], interactions such as touching and pasting is an intuitive, but also an extremely dependable way of selecting devices. Vib-connect also uses touching & pasting interaction to select devices.

In Vib-connect, users can select devices very simply. They only have to take two simple steps as illustrated in Figure 3.1: 1) Touch the Vib-connector to service device, and then 2) Paste the same Vib-connector to user device.

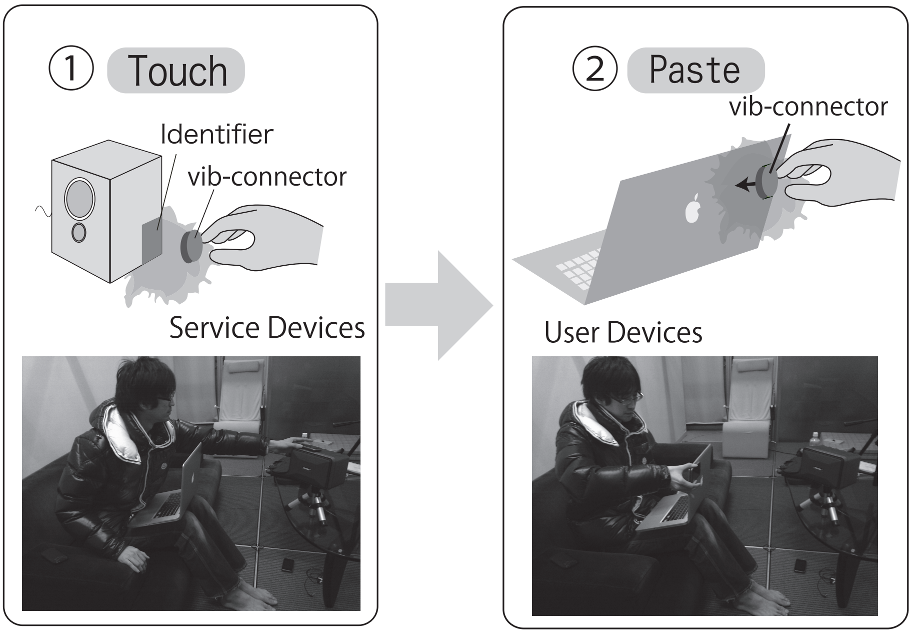

Figure 3.1: Two steps to select devices

1. **Touching Vib-connector to service device**

   Touching Vib-connector to service device is the first step taken by users. This is a step to bundle service device and Vib-connector. By touching Vib-connector to the identifier, Vib-connector will be bundled with the service device, and glow in the same color with the identifier.

2. **Pasting Vib-connector to user device**

   Pasting Vib-connector to user device is the second step taken by users. This is a step to bundle service device and user device by using Vib-connector. As illustrated in Figure 3.2, users can easily recognize the connections and relationships of devices by looking at the color of identifier and Vib-connector.

   When Vib-connector gets pasted on user device, it vibrates in specific vibration pattern, which is an index for service device the Vib-connector is being bundled with.

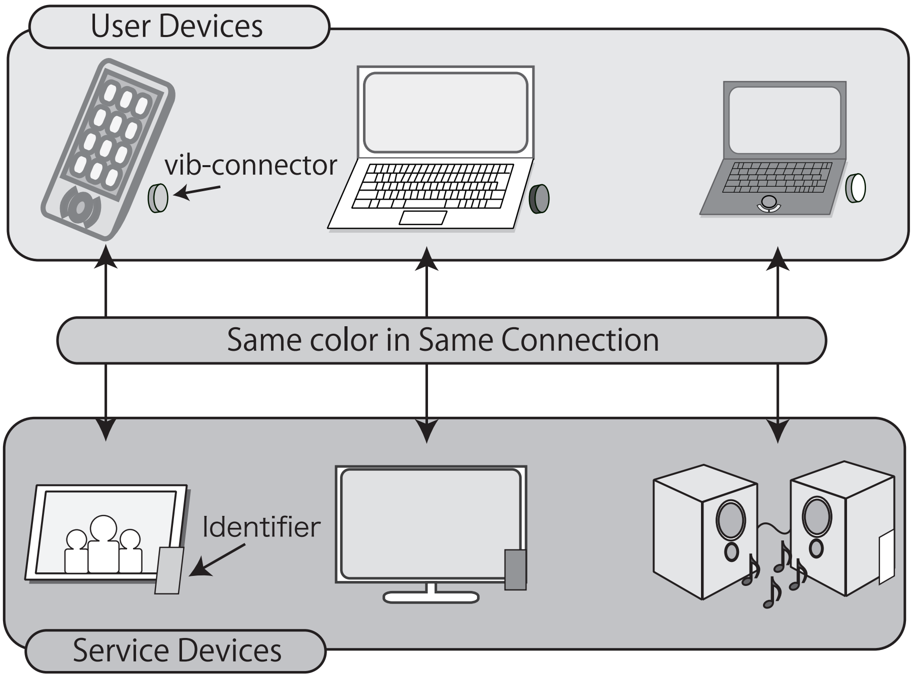

Figure 3.2: Expression of the Device Relationships and Connections

After those two simple steps, the two devices which were touched and pasted will be selected, and the connection will be created between each other.

3.3.2 Features

Vib-connector has the following features, and solves the four problem statements mentioned in Chapter2.

1. **Extremely little bundle in recognizing devices:**

   As this interface touches & pastes Vib-connect directly to the device, no assumption steps association steps will be needed. They can easily and intuitively recognize and select the correct device just as looking through their own eyes, in the real world. Since there is actually no assumption steps in recognizing the device, solving problem statement " 1) Burden in recognizing devices" is accomplished by this form of Vib-connect.

2. **More multimodal:**

   Since Vib-connect uses vibration and sense them by accelerometer, this system only requires accelerometer for sensing vibration. Since it can be used if there is an accelerometer, Vib-connect can be used to devices such as tablet PC and smart phones, which don't have I/O ports such as USB and memory sticks. As accelerometer will surely be implemented on those mobile from now on, solving problem statement " 2) Narrowness of correspondence" is accomplished by Vib-connector. Explanation about vibration communication will be provided in the next chapter.

3. **Robustness against environmental influence:**

   Since Vib-connector is based on vibration, it will not be influenced by environmental influences such as illumination and obstacles. Also, as a feature of vibration and accelerometer, Vib-connector is also robust against noises of the environment. Therefore, it can be said that Vib-connect also solves problem statement " 3) Frailness against environmental influence". Details about the feature of vibration and accelerometer will be given in the next chapter.

4. **Easy to recognize the connection and relationships between devices:**

   Users can easily recognize the relationships and connections of devices by looking at the color of the identifier pasted on service devices, and the Vib-connector pasted on user devices. Since there isn't any step of understanding linguistic expression from interfaces, even young children who cannot read letters can understand the relationships and connections between the devices. As the relationships and connections between the devices can be recognized easily and visually, solving problem statement " 4) Difficulty in recognizing device relationships " is accomplished by using Vib-connector.

In this section, we have explained about the detail of the vib-connect interaction. In the next section, we will describe the detail of vibrate communication, which is a method we use to handle the service device information to user device.

3.4 Vibrate Communication

In this section, we will explain about the vibration-based communication. First, the basic design and outline of vibrate communication is showed. Then we will explain about vibration patterns and the algorithm for detecting vibration. Last, we will show some features that vibration has.

3.4.1 Communicating by Vibration

As showed in Fig3.3 basic principle of vibrate communication, is to detect vibration by analyzing the acceleration caused by the vibration motor. As showed in Fig3.3, we will use the built-in accelerometer in Vib-connect.

Vibration causes acceleration to every axes, x-axis, y-axis, and z-axis. Vibration can be detected by adding the absolute value of those acceleration. Fig3.4 is a sample of acceleration caused by vibration.

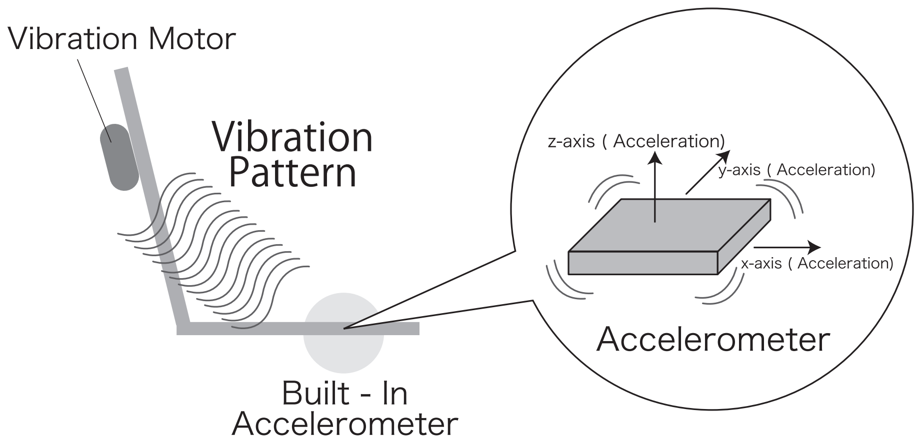

Figure 3.3: The basic principle of detecing vibration

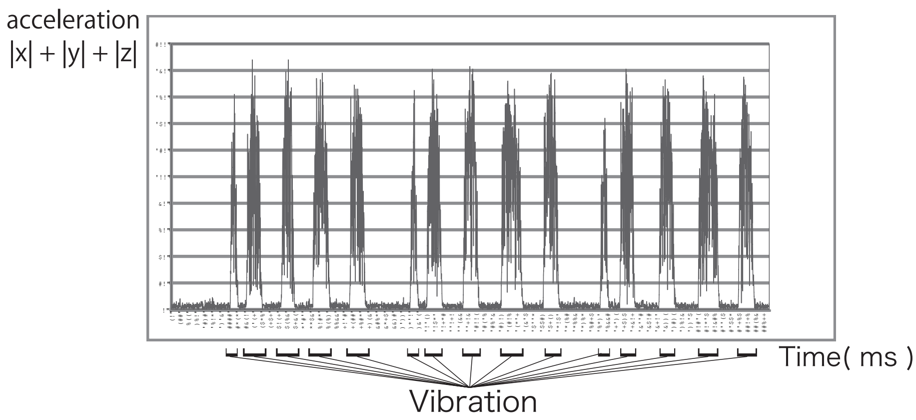

Figure 3.4: Acceleration caused by vibration

As can be seen in eyes, there is an obvious change in acceleration when vibration has occurred. Vibrate communication will analyze these acceleration, and use them as a tool for communication.

3.4.2 Vibration Pattern

To embed information into these vibration as patterns, we have divided the acceleration data into time periods, which depends on hardware performance. As shown in Fig3.5, either “ 1 ”, “ 0 ”, or “ start point ” will be allotted to each of them as a pattern. Pattern “ 1 ” shows that there was acceleration caused by vibration, and “ 0 ” shows that there was none. “ start point ” is a indicator that shows where the embedded information starts from.

3.4.3 Vibration Detecting Algorithm

As shown in Fig3.6, we have divided 1 time period into two smaller time sections, to define “ start point ” from the normal patterns “ 1 ” and “ 0 ”. Since 1 time section is half of time period, normal patterns “ 1 ” or “ 0 ” is a double continuation of time section which vibration was detected or not.

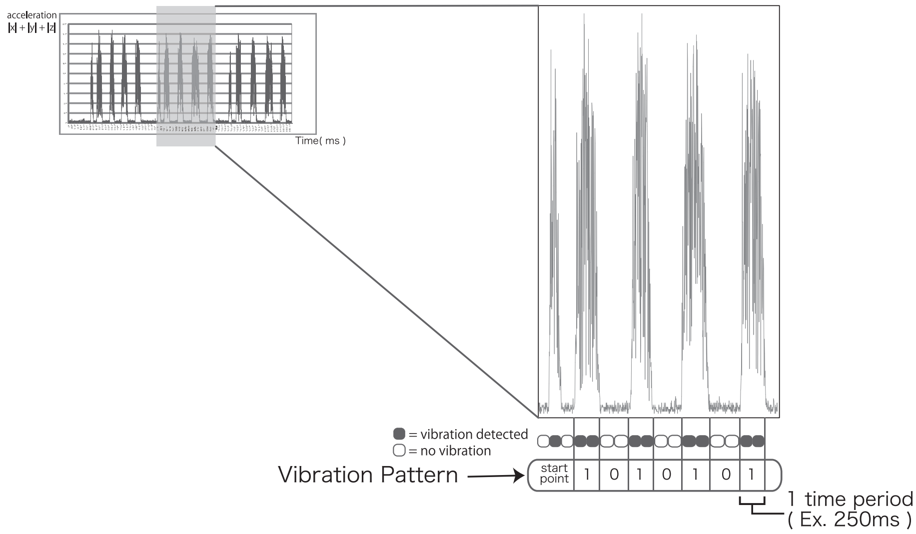

Figure 3.5: Vibration Patterns

Only when a single vibration-detected time section is sandwiched between time section which vibration was not detected, it will be defined as a start point.

In order to define if there was an vibration during a time section or not, we watched the total times of specific acceleration being detected, in one time section. As seen in Fig3.6, acceleration caused by vibration is obviously high than the acceleration when there is no vibration. So we made a border line to check if the acceleration was caused by vibration or not (Check Fig3.6). The value of this border line differs by the hardware you use.

Suppose A as the total count of grepping acceleration in a time section, and V as the total count of acceleration over border line being detected in a time section.If

$$V > \frac{T}{2}$$

is true, the time section will be defined that there was a vibration. If false, it will be defined that there was none vibration.

Also, in order to tell the acceleration caused by vibration from those of

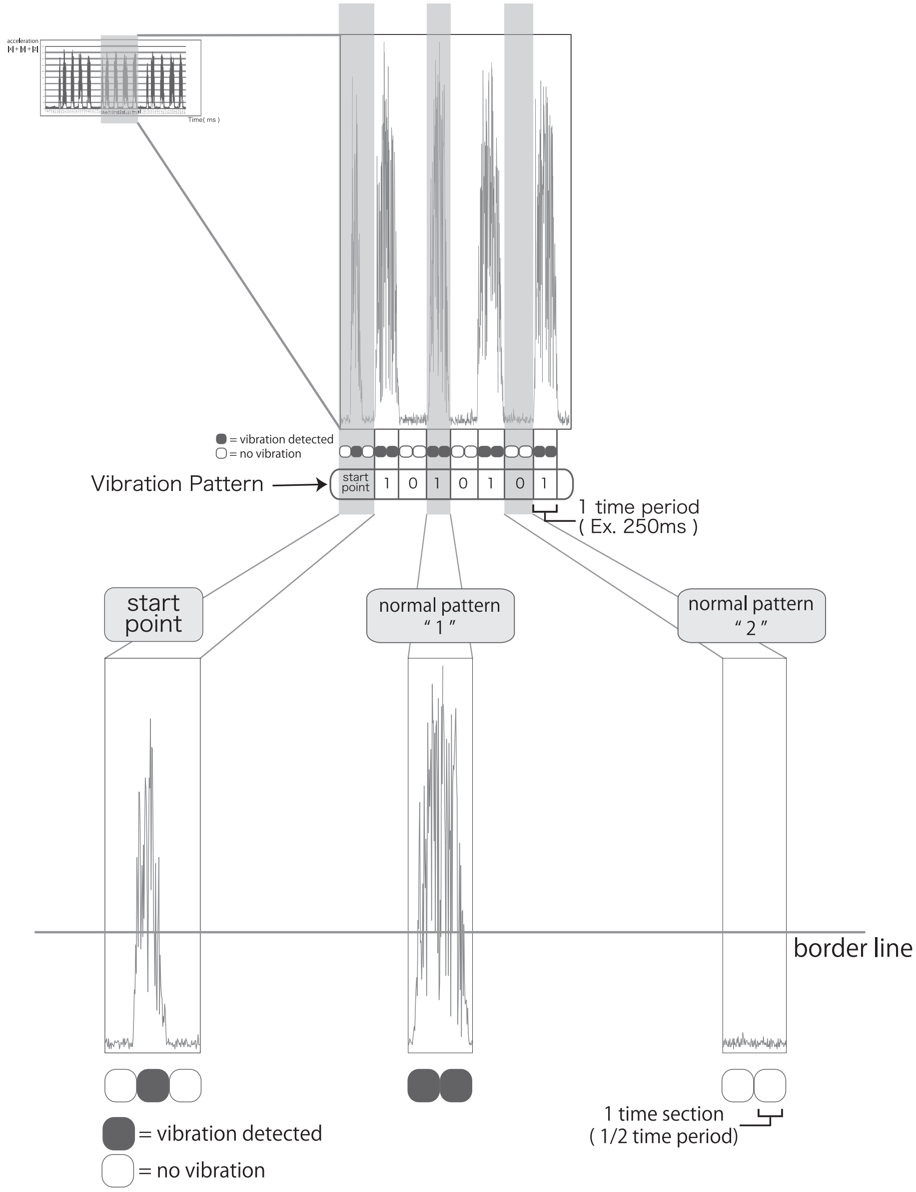

Figure 3.6: Difference of Vibration Patterns

daily action, such as moving or lifting lap-top computers, we will not count the rapid acceleration as acceleration caused by vibration.

3.4.4 Features of Vibrate Communication

We would like to explain some features and advantages of vibrate communication.

Multimodality

Since this method uses only vibration motor and accelerometer, it can support more various devices. Thin clients such as smart phones and tablet PCs, don't have previous I/O ports (I.E. USB, i-link) , but most of them have accelerometer loaded. Therefore, using this method will enable more devices to have an intuitive way of identifying devices.

Robustness against Environmental Influence

Accelerations are not easily influenced by loud noises in the environment. Therefore, this method can be used even in noisy places, such as offices and meeting rooms. Also, the vibration detecting algorithm have gave this method the robustness against vibration such as typing. We have already evaluated the robustness and the result will be provided in Chapter 6.

Range of Vibration

Vibration travels only to the object which is in direct contact, so other device won't detect those vibration in accident. This means that this method can make a tight bundle between the vibrating object and the contacting object.

On the other hand, those vibration can be generated and detected anywhere if they are in direct contact. Therefore, this methods enables free interaction, since user can paste the vibrating object where ever he likes to.

3.5 Summary

In this chapter, we explained how users will use Vib-connect, the features of Vib-connecting devices, and the detail of vibrate communication. In the next chapter, we will describe the design of Vib-connect, and show the implementation we done in Chapter 5.

Chapter 4

Design

We have described the concept of Vib-connect in the previous chapter. This chapter first describes the design of Vib-connect, and then show the prototype implementation we have made.

In this section, we will explain about the design of Vib-connect. First, we will describe the hardware components and software components. After the components were described, we would like to describe the flow of process inside the system.

4.1 Hardware Components

First of all, we would like to explain about the hardware components of Vib-connect. As illustrated in Fig4.1, there are four hardware components except of service device and hardware device in Vib-connect ; 1) Identifier, 2) Vib-connector, 3) accelerometer (Built In), and 4)Pattern Server. Each of them will be explained in following.

4.1.1 Identifier

Identifier is an object pasted to the service devices. Each identifier has their own specific color. These identifier holds two types of information ; 1) Color Information and 2) Vibration Pattern.

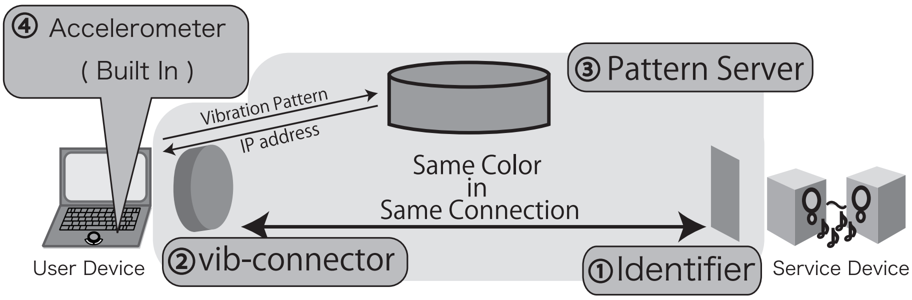

Figure 4.1: Hardware Components

1. Color Information

   Color information is the information about the color of identifier itself. For example, if the identifier is red, the color information is red, and if the identifier is blue, the color information is blue. These color information are going to be used by Vib-connector, to glow Vib-connector itself in the same color with the identifier.

2. Vibration Pattern

   Vibration pattern is the information of vibration pattern, which is used as an index to get the information of the service device. Each identifier have their own specific vibration pattern as same as the color information. This vibration pattern are used by user device to inquiry the pattern server about the information of service device, such as IP addresses and MAC addresses.

   Details of vibration patterns will be described in Chapter 5.

These two information are supposed to be already inputted by manufacturers, but also users themselves can manage them if he wants to. Identifier can be any kind of methods if it can realize intuitive touch interaction. We are thinking of RF-ID technology for the present, but it can be something different if there is another method that better matches the situation.

4.1.2 Vib-Connector

Vib-connector are devices which are used to mediate the connection and relationships between user device and service devices. Vib-connector has vibration motor, color LED, switch to detect itself being pasted to user device, and RF-ID reader, if RF-ID tag is going to be used. Vib-connector has three states, according to three step user takes ; 1) Default state, 2) Standby state, and 3) Vibrating state.

1. Default State

   Default state is state which no action is taken by the user. There will be no information held in Vib-connector, and the Vib-connector itself, will be white in the default state. As soon as user takes touching step, two information, color information and vibration pattern are read to Vib-connector, and Vib-connector changes into standby state.

2. Standby State (User Step : Touching)

   Standby state is state in which user has done the touching step. In standby state, Vib-connector glows in the same color with the color information. As soon as detecting itself being pasted on user device, Vib-connector changes into vibrating state.

3. Vibrating State (User Step : Pasting)

   Vibrating State is state in which Vib-connector is vibrating in a specific pattern, according to the vibration pattern information. Vib-connect glows in the same color with the color information, in this state too. As soon as detecting itself being unpasted, it will stop vibrating and go back to standby state.

Vib-connector can also be placed with device that has controllable vibrating function, such as smart phones.

4.1.3 Accelerometer (Built In)

Some types of laptop computers today, such as MacBook series and ThinkPad series, uses SMS (Sudden Motion Sensor) to protect their hard disk. SMS are built-in accelerometer which is used to detect sudden motions such as dropping, and to prevent the hard disk being broken in those motions. The number of laptop computers which have these accelerometer are expected to increase more and more, due to the improvement in MEMS (Micro Electro Mechanical Systems) technology.

Also, accelerometer is not loaded in only laptop computers, but also to those thin client, such as tablet PC and smart phones. Number of devices which have accelerometer built in will increase more and more, in the future.

In Vib-connect, these built-in accelerometer in user devices, will be used to detect the vibration from Vib-connector.

4.1.4 Pattern Server

Pattern server is a server which has the map of vibration pattern and service device information. Service device information are information of service device, which is necessary for user device to open connection and collaborate with them, such as IP addresses and MAC addresses.

On receiving the request and vibration pattern from the user device, pattern server returns back the device information that conforms with the received vibration pattern.

Though, as we will mention in the next section, any kind of binary data can be sent in vibrate communication, by increasing their bps. Since these vibration pattern are just an index for service device information, these service device information can be directly embedded into identifier, if the bps of vibrate communication is enough. In that case, pattern server doesn't have to be prepared.

These four hardware are the hardware components of Vib-connect. Software components will be described in the next section.

4.2 Software Components

In this section, software components of Vib-connect will be explained. Modules working in Vib-connect will be shown, and we will also explain the details of those modules in this section.

There are seven modules working in Vib-connect. Firstly, three modules are working in Vib-connector ; 1) Information Reader Module, 2) Color Generator Module, and 3)Vibration Generator Module. Secondly, three modules are working in the user device ; 4) vibration detecting module, 5) pattern analyzing module, and 6)pattern inquire module. Last one, 7) Pattern Matching Module, is working in the pattern server. The details of the modules are described in following.

4.2.1 Information Reader Module

Information reader module is an module working inside the vib-connector. This module receives two information from the identifier ; color information and vibration pattern. On receiving these two information, color information will be passed to color generator module, and vibration pattern will be passed to vibration generator module.

4.2.2 Color Generator Module

Color generator Module is also an module working inside the Vib-connector. When color information is received from the information reader module, this module generates the color according to the received color information, and makes the Vib-connector itself glow in that color. In the default state, color generator module generates white color.

4.2.3 Vibration Generator Module

Vibration generator is also working inside the Vib-connector. When vibration pattern is received from the information reader module, this module generates vibration according to the received vibration pattern, and make

Vib-connector itself vibrate in that pattern. In case that service device information can be embedded directly, this module will encode those information into vibration patterns, and generates vibration according to them.

4.2.4 Vibration Detecting Module

Vibration Detecting Module is an module working in the user device. This module receives acceleration from the accelerometer built in the user device, and detects vibration by analyzing those acceleration data. The vibration detect information will be passed to pattern analyzing module.

The details of the algorithm working in this module, are explained in Chapter 3.

4.2.5 Pattern Analyzing Module

Pattern Analyzing Module is also an module working in the user device. This module receives and analyzes vibration detect information, and decodes those vibration to vibration patterns. Once recognizing a pattern analyzing module, pattern analyzing module will pass the vibration pattern to pattern inquire module.

The details of the algorithm working in this module, are explained in Chapter 3.

4.2.6 Pattern Inquire Module

Pattern inquire module also works inside the vib-connector. This module receives the recognized vibration pattern from pattern analyzing module, and sents these vibration pattern to the pattern server. Once receiving the service device information from the pattern analyzing module, this module passes those information to device collaboration applications. Device collaboration applications are those applications that realizes device connection and collaboration, which is out of Vib-connect's focus.

In case the service device information is embedded in the detected vibration pattern, this module doesn't have to be prepared.

4.2.7 Pattern Matching Module

Pattern matching module is an module working inside the pattern server. This module receives the vibration pattern from the user device, searches the map of vibration patterns, and returns back the service device information that matches to the vibration pattern, to the user device.

This module also doesn't need to be prepared in case the service device information is embedded in the detected vibration pattern.

These are the modules working in Vib-connect. In the next section, we would like to explain about the flow of the process taken in the system.

4.3 Flow of the Process

In this section, we will explain about the flow of process taken inside the Vib-connect. Before explaining the flow of process taken inside the Vib-connect, we would like to show all of the hardware components and software components in Fig4.2.

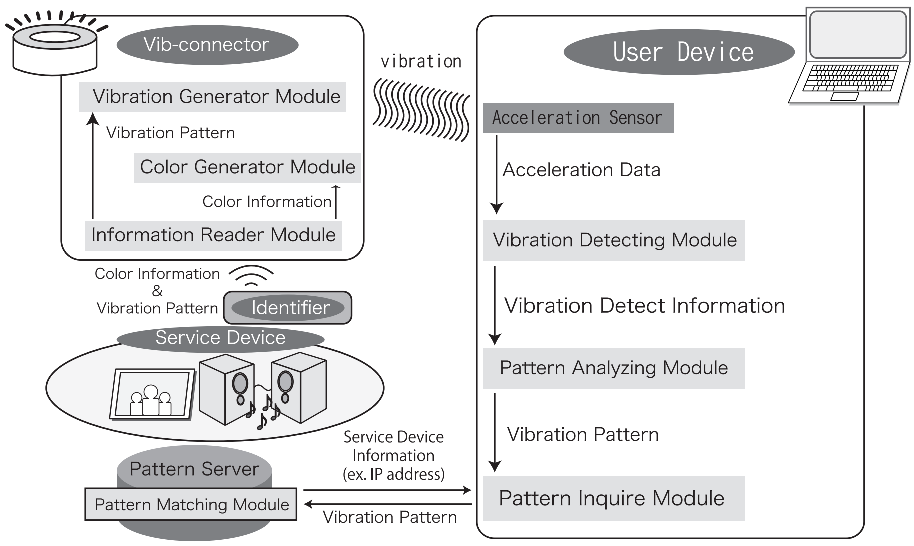

Figure 4.2: Components of Vib-connect

Next, we would like to explain the flow of process taken inside the Vib-connect. Fig4.3 is the sequence of the process taken inside the Vib-connect.

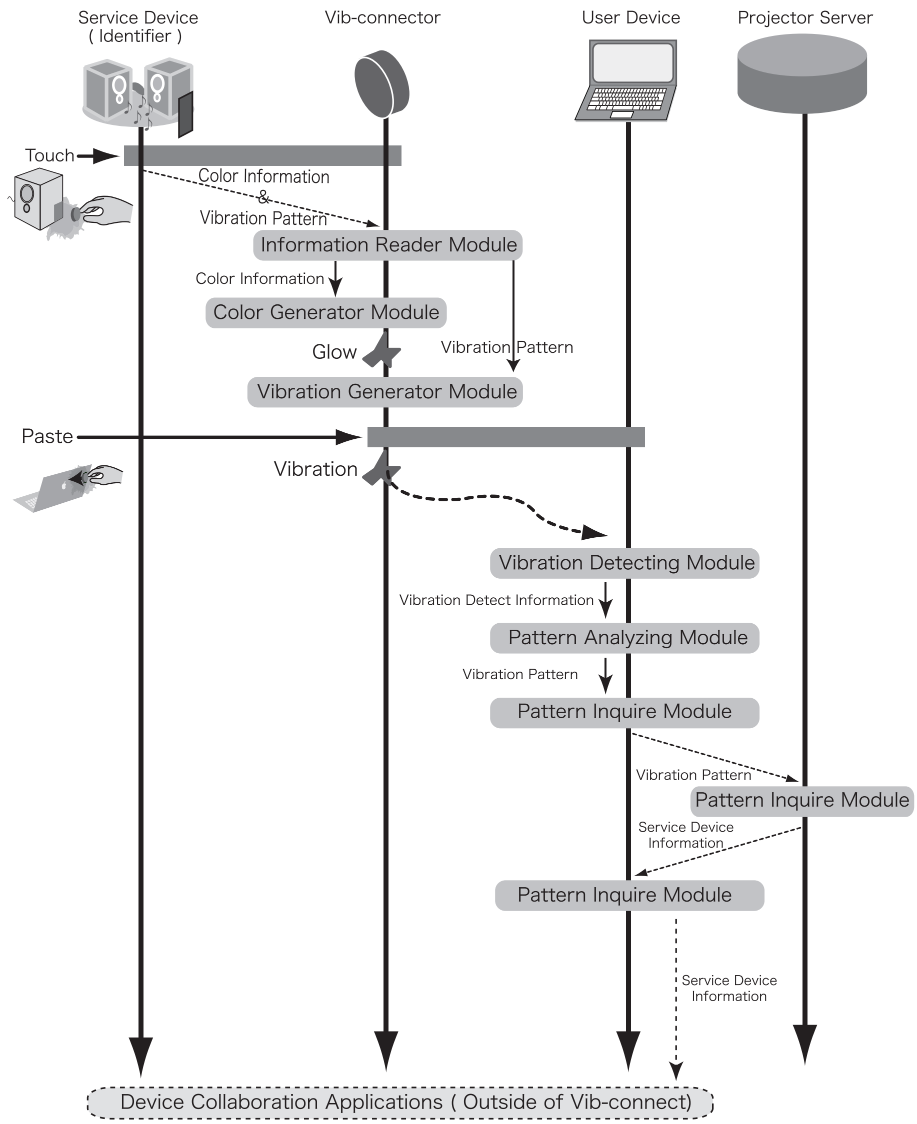

Figure 4.3: Sequence of Process Taken Inside Vib-connect

First, when users touches Vib-connector to identifier which is pasted on service device, color information and vibration pattern will be passed to information reader module inside the user device. Those two information, color information and vibration pattern, would be passed to color generator module and vibration generator module. These two modules, color generator module and vibration generator module, will make Vib-connector to glow in the color according to the color information, and make Vib-connector to vibrate in the received vibration pattern.

Secondly, when user pastes Vib-connector to the user device, vibration detecting module in user device, detects the vibration from Vib-connector, using acceleration data of the built-in accelerometer. Vibration detect information will be passed to pattern analyzing module, and this module will analyze those Vibration detect information and recognizes the vibration pattern.

Finally, as soon as pattern analyzing module recognizes the vibration pattern, this vibration pattern will be passed to pattern inquire module. Pattern inquire module gets the service device information with exchange of vibration pattern. Then it passes the service device information to applications which manages connection and collaboration.

4.4 Summary

In this chapter, we explained about the design of Vib-connect. Details of the hardware components, software component, and the flow of process taken in Vib-connector. In the next chapter, we would like to show the prototype of Vib-connect we have implemented.

Chapter 5

Implementation

In this chapter, we will show the prototype of Vib-connect we have implemented. First, we will explain the implementation environment of the prototype, and next, we will some applications we have made to show examples.

As implementation for prototype of Vib-connect, we have implemented the modules and applications for vibrate communication, which is a very important part of Vib-connect. Also, we have implemented some device collaborate applications to simulate and demonstrate the pasting action.

5.1 Implementation Environment

In this section, we will describe the devices and environments we have used to implement the prototype of Vib-connect. Details of implementation environment used to implement user device, service device, and Vib-connector will be described in the following.

5.1.1 User device

For user device, we have used MacBook Air (late 2008)5.1, MacBook Pro 15inch (Early 2008)5.2, and Nexus One5.3.

We will show the technical specification of each three devices in following.

Table 5.1: Technical Specification of MacBook Air

H/W/D	1.94cm/32.5cm/22.7cm
Weight	1.36kg
CPU	1.8GHz Intel Core 2 Duo
Memory	2GB 667MHz DDR2 SDRAM

Table 5.2: Technical Specification of Macbook Pro 15 inch (Early 2008)

H/W/D	2.59cm/35.7cm/24.3cm
Weight	2.45kg
CPU	2.5GHz Intel Core 2 Duo
Memory	4GB 667MHz DDR2 SDRAM

Table 5.3: Technical Specification of Nexus one

H/W/D	11.9cm/5.98cm/1.15cm
Weight	130g
CPU	1GHz Qualcomm QSD 8250 Snapdragon
Memory	512MB

Also, I used java-based Processing Sudden Motion Sensor Library[19] to get the acceleration data from MacBook Air and MacBook Pro 15 inch.

5.1.2 Service device

For service device, we have used MacBook Pro 15 inch (late 2008)5.4, and Thinkpad X60s5.5, as a virtual speaker and virtual projector. We will show the technical specification of them below.

Table 5.4: Technical Specification of MacBook Pro 15 inch (late 2008)

H/W/D	2.41cm/36.4cm/24.9cm
Weight	2.49kg
CPU	2.53GHz Intel Core 2 Duo
Memory	4GB 1066MHz DDR3 SDRAM

Table 5.5: Technical Specification of ThinkPad X60s

H/W/D	3.5cm/2.68cm/21.1cm
Weight	1.27kg
CPU	1.06GHz Intel Core Solo
Memory	512MB

5.1.3 Vib-Connector

We have used Nexus One5.3 as deputy for Vib-connector.

5.2 Prototype

We have implemented a prototype of Vib-connect. In this section, we will show VibMaker, VibDetector, and VibAnalyzer, the prototype of system to realize vibrate communications.

5.2.1 VibMaker

We developed VibMaker, an android application that generates vibration with Nexus One. The time period of the vibration pattern is 250ms, and can send information by vibration with 4bps speed. Vibration patterns for VibSpeaker and VibProjector are already embedded. This application realizes the vibration generator module inside the Vib-connector. Fig 5.1 is the screen shot of VibMaker.

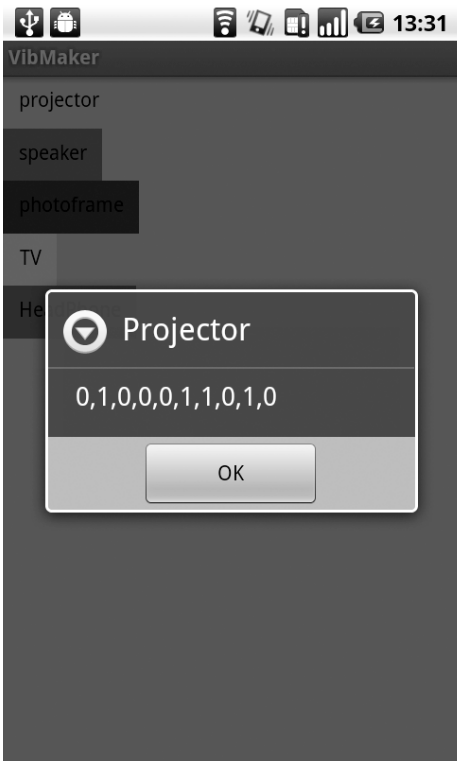

Figure 5.1: Screen shot of VibMaker

5.2.2 VibDetector

VibDetector is an java-based application, that gets acceleration data from the SMS library[19], and send them to VibAnalyzer with the UNIX timestamps. These UNIX timestamps will be used to sync the detection time period with the vibration pattern. This application realizes the vibration detecting module inside the user device.

We have also implemented this application with Nexus One, in order to prove the multimordality of vib-connect.

5.2.3 VibAnalyzer

VibAnalyzer is an java-based application, that analyzes the acceleration data sent from VibDetector, to recognize the vibration pattern. This application syncs it time period automatically with the VibMaker's vibration, when it first detects the vibration. This application realizes the pattern analyzing module inside the user device.

Screen shot of VibAnalyzer is shown in Fig5.2. Once it detects vibration pattern, VibSpeaker or VibProjector (applications which are both explained in the next section) starts, and will change to screen such as shown in Fig5.3.

Figure 5.2: Screen shot of VibAnalyzer

5.3 Application

We have also implemented some applications in order to simulate and demonstrate device collaboration. In this section, we will show VibSpeaker and VibProjector, applications to simulate service devices.

5.3.1 VibSpeaker

VibSpeaker is an application that simulates a network speaker. On detecting vibration pattern of VibSpeaker, user devices can stream and play music and use the VibSpeaker over wireless network. JLayer[20], a mp3 library for java was used in order to implement the function to play mp3 files. Fig5.3 is the screen shot of VibSpeaker. On clicking the "connect" button, the music playing in the user device will be streamed to the service device.

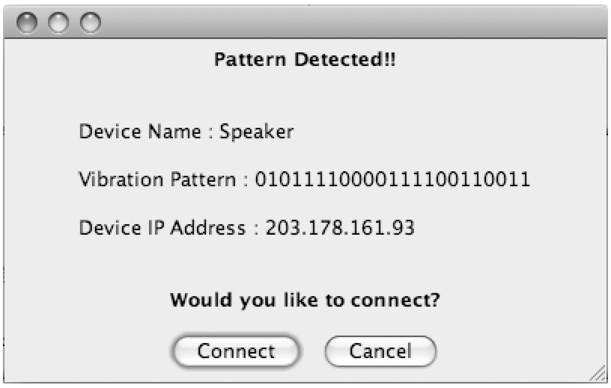

Figure 5.3: Screen shot of VibSpeaker

5.3.2 VibProjector

VibProjector is an application that simulates a network projector. This application will be activated when user device detects vibration pater of VibProjector. This sends a screen shot of user device in a high speed, and works as an projector emulator over wireless network. Fig5.4 is the look of how VibProjector works.

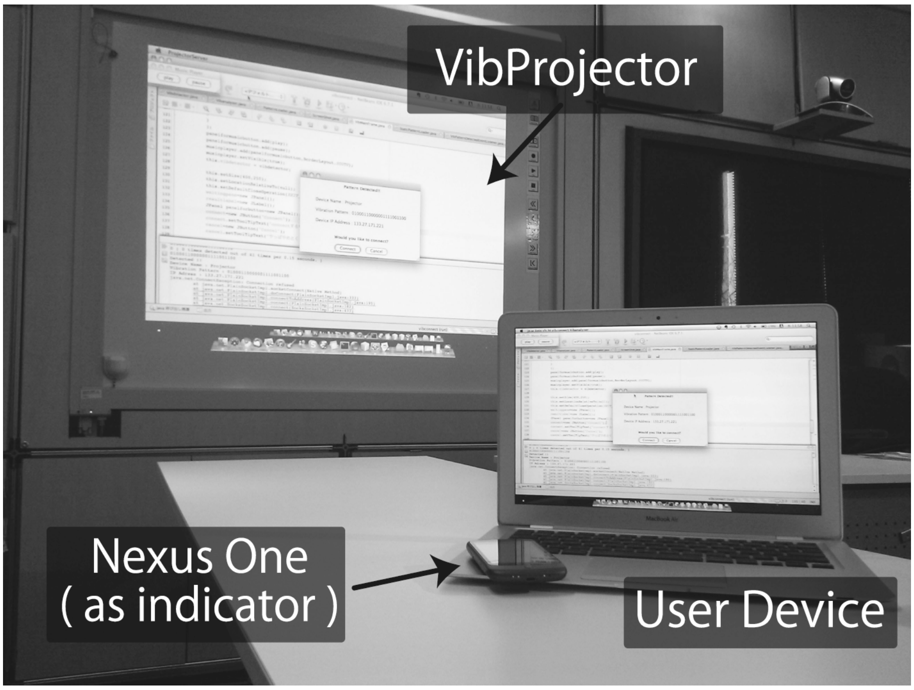

Figure 5.4: Look of VibProjector Working

5.4 Summary

In this chapter, we have showed the prototype of Vib-connect we have implemented, and introduced some sample applications. In the next chapter, we will describe the evaluation we have done, and show their results.

Chapter 6

Evaluation

This chapter evaluates accuracy and usability of Vib-Connect. Accuracy evaluation is an experiment to test the accuracy of recognition in vibration pattern. Usability Evaluation is an experiment to test the usability, by having few examinees fill the questionnaire after using Vib-connect. After that, we would like to evaluate if we have accomplished the requirements mentioned in chapter 2. First of all, we would like to explain about the accuracy evaluation.

6.1 Accuracy Evaluation

In this section, we will show the result of the accuracy evaluation test in recognizing vibration pattern. In the test, we have tried the pattern recognition in two cases, when Vib-connector is 1) put on the base of the PC, or 2) pasted at the top of the PC. In addition, to test the robustness against environmental influence, we have also tested them while typing on the service device itself. Two patterns, "Speaker (1100110101)" and "Photoframe (1010101010)", were tested on four types of devices ; MacBook Air (late 2008), MacBook Pro 15inch (Early 2008), MacBook Pro 15 inch (Late 2008), and Nexus One. Each of the environment was tested 20 times.

The following tables6.16.26.36.4 are the result of the accuracy evaluation test using MacBook Series in each environment.

Table 6.1: Accuracy in Normal Environment with Vib-connector on base

	Speaker (1100110101)	Photoframe (1010101010)
MacBook Air (Late 2008)	(20 / 20) 100%	(20 / 20) 100%
MacBook Pro 15 inch (Early 2008)	(20 / 20) 100%	(20 / 20) 100%
MacBook Pro 15 inch (Late 2008)	(20 / 20) 100%	(20 / 20) 100%

Table 6.2: Accuracy in Typing Environment with Vib-connector on base

	Speaker (1100110101)	Photoframe (1010101010)
MacBook Air (Late 2008)	(20 / 20) 100%	(20 / 20) 100%
MacBook Pro 15 inch (Early 2008)	(20 / 20) 100%	(19 / 20) 95%
MacBook Pro 15 inch (Late 2008)	(20 / 20) 100%	(20 / 20) 100%

Table 6.3: Accuracy in Normal Environment with Vib-connector on top

	Speaker (1100110101)	Photoframe (1010101010)
MacBook Air (Late 2008)	(20 / 20) 100%	(20 / 20) 100%
MacBook Pro 15 inch (Early 2008)	(20 / 20) 100%	(20 / 20) 100%
MacBook Pro 15 inch (Late 2008)	(20 / 20) 100%	(20 / 20) 100%

Table 6.4: Accuracy in Typing Environment with Vib-connector on top

	Speaker (1100110101)	Photoframe (1010101010)
MacBook Air (Late 2008)	(20 / 20) 100%	(20 / 20) 100%
MacBook Pro 15 inch (Early 2008)	(20 / 20) 100%	(19 / 20) 95%
MacBook Pro 15 inch (Late 2008)	(20 / 20) 100%	(19 / 20) 95%

We have also tested the accuracy with two Nexus One, being stacked on together as Vib-connector and user device. The result of the accuracy is shown in table6.5.

Table 6.5: Accuracy with Nexus One

	Speaker (1100110101)	Photoframe (1010101010)
Nexus One	(17 / 20) 85%	(16 / 20) 80%

As a result, we have got an extremely high accuracy in MacBook series. Even when typing, they have got accuracy over 95%. The reason of these failure in MacBook series, is the attunation of vibration through the joint between the top part and the base part. If this joint is tight and secure, vibration from the top part will reach accurately to the accelerometer in the base part.

In Nexus One, accuracy was about 80% to 85%. These failure in Nexus One seems to come from the preciseness of the accelerometer and the processing speed. If they rise, the accuracy is expected to rise too. In the next section, we will show the usability evaluation and the result.

6.2 Usability Evaluation

In this section we will evaluate usability of Vib-connect by 5 indicators[21] - *Learnability*, *Efficiency*, *Memorability*, *Error Handling* and *User Satisfaction*.

6.2.1 Experiment

We have tested the usability of Vib-connect. The participants are 12 students who have never used Vib-connect. The 12 students (participants A - L) consists of (participants A - G) non-expert users, who have never used wireless device collaboration, and (participants H - L) expert users, who are using wireless device collaboration in everyday life.

As an setup for experiment, we have pasted an color tag on few devices as an identifier. We had the participants select these devices randomly with our prototype, and measured the time they took to select the devices.

6.2.2 Result

The result of time for selecting devices is showed in figure6.1. After the test, we had them fill up a questionnaire about the 5 indicators[21], based on the Likert scale[22], in the range of 1 to 5 (1 as lowest, and 5 as highest). The answers by the participants are showed in table6.6.

Table 6.6: Result of questionnaire

Usability Indicator	1	2	3	4	5	Average
Learnability	0	0	1	9	2	4.08
Efficiency	0	0	5	4	3	3.83
Memorability	0	0	0	2	10	4.83
Error Handling	1	1	1	6	3	3.75
User Satisfaction	0	1	1	9	1	3.83

In the next section, we would like to discuss about the result of evaluation experiment.

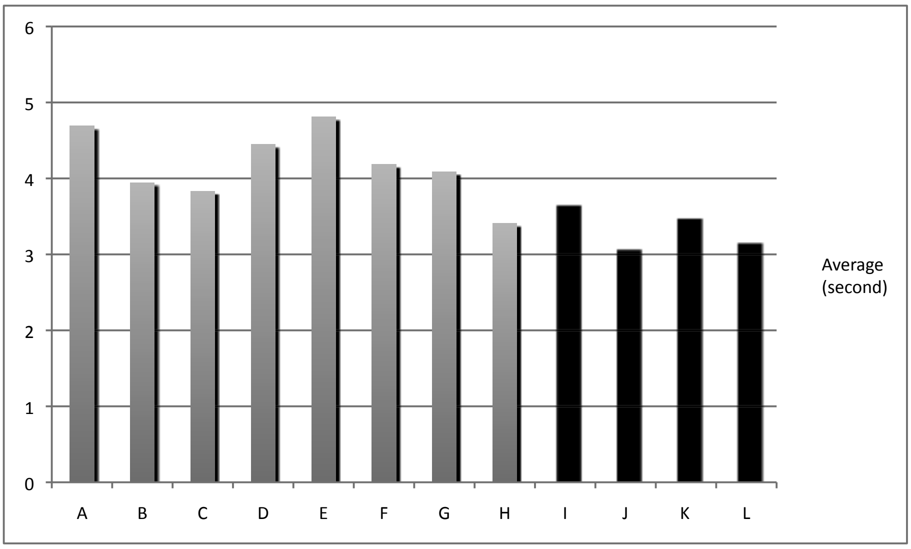

Figure 6.1: Average of times for device selecting by participants

6.3 Discussion

In this section, we will discuss about the 5 usability indicators of Vib-connect; Learnability, Efficiency, Memorability, Error Handling, and User Satisfaction.

6.3.1 Learnability

Regardless of experience in device collaboration, every participant was able to use Vib-connect in short explanation. The average rate in the questionnaire about this indicator was 4.08. From this result, I consider that non-expert users can select device in their actual life, without requiring expert users' help.

6.3.2 Efficiency

Regardless of the number of trial, the participants were able to select devices within 5 seconds (see Figure6.1). Thinking of those device-recognizing time in name selecting interfaces, this result shows high efficiency of Vib-connect in recognizing and selecting device. Also, there were comments from participant that it is physically troublesome to select devices which is farther than their hand reach. Though, in case the device is nearby, it can be said that Vib-connect is more efficient than those existing device selecting interfaces.

6.3.3 Memorability

Vib-connect requires users only to touch & paste, and they don't need any knowledge about computers and networks. As a result, we got a high average rate, 4.83 about this indicator (see table6.6). This shows that Vib-connect is a simple, highly memorable interface.

6.3.4 Error Handling

During this evaluation experiment, none of the participants made wrong operation. As substitute, we had them imagine the case to re-select the devices, when they selected the wrong device accidentally (I.E. Selecting wrong device in accidental movement , such as turning around). About this question, most of them answered it was easy to re-select. Though, there were also comments that it is difficult to tell if the error was caused by the human or the system.

6.3.5 User Satisfaction

The average rate of this indicator was 3.83. There were two types of comments about difficult point of Vib-connect ; 1) existence of Identifier, and 2) physical troublesome in selecting far devices. 1) Existence of Identifier, is about the visual change and interior influence against the service device. 2) Physical troublesome in selecting far devices are points mentioned in the efficiency section.

Though, every participants were enjoying using Vib-connect, and most of them had comments such as "Easy and Simple", "I want this system come true". Also, 10 out of 12 participants estimated this system better than either "Very good" and "Great", in the questionnaire.

6.3.6 Comments by participants

As the last question of the questionnaire, we had the participants to write comments freely. First, we will show the negative comments which some of them are mentioned in the previous discussions.

“ I think the color tag doesn’t have to be colors, but photographs of the devices. If the photograph is shown in the screen, that’s enough for me. ”

- Participant A (non-expert user), and Participant I (expert user)

“ I’m sure touching is easy and reliable for us, but I think it’s physically troublesome. ”

- Participant B, E, F (non-expert user), and Participant J (expert user)

“ I don’t want to paste those color tag on my devices. That thing will be bad for the interior of the devices in my room. ”

- Participant K (expert user)

“I think I might lose the Vib-connector.”

- Participant L (expert user)

Next, we would like to show the positive comments by the participants.

“ It’s easy and quite fun to use. ”

- Participant B, C (non-expert user)

“ I think the idea of using color tag is original and interesting. ”

- Participant E (non-expert user)

“ I think it will be usefull if it is actually made as a product. ”

- Participant D, G (non-expert user)

“ I think it is very usefull in case that there are a lot of same devices, like office. ”

- participant H (non-expert user)

" Since I use many peripheral equipments of computer, I think it will be usefull in classifying them in purpose of each devices. "

- participant L (user)

We have discovered some problems from the negative comments, but most of them were positive comments. In the next section, we will evaluate and discuss about the accomplishment of requirements which are mentioned in chapter 2.

6.4 Requirement Evaluation

In this section, we will evaluate and discuss if we have accomplished the 4 requirements mentioned in chapter2; 1) Less burden in recognizing and according devices, 2) more multimodal interface, 3) robustness against environmental influence, and 4) easiness in recognizing connections and relationships between devices.

1. **Less Burden in Recognizing and According Devices:**

   Since users can make virtual connection by simply touching the device in real world, there is no assumpting and according step. In the usability evaluation test, there were no users who made mistake in selecting device.Vib-connecter can be said that it has obviously less burden in recognizing and according devices.

2. **More Multimodal Interface:**

   Vibrate communication requires only vibration motor and accelerometer. Accelerometers are usually built-in the lap-top computers to protect hard disks, and also in the thin clients such as tablet PCs and smart phones. These thin clients don't have physical I/O ports such as USB or memory stick, which are required in the present methods, but most of them still has accelerometer built-in. It can be said that

Vib-connect has accomplished this requirement. In addition, we have proved that this method can be used with smart phones by implementing on Nexus One.

3. **Robustness against environmental influence:**

   Since vibrate communication uses vibration, it will not be influenced by environments such as lights and music. In addition, as shown in the previous section6.26.4, vibrate communication has an extremely high accuracy, even when user is typing with the user device itself. It can be obviously said that this requirement has been accomplished.

4. **Easiness in Recognizing Connections and Relationships between Devices:**

   Users can recognize the connections and relationships between devices easily, by looking at the colors of identifier and Vib-connector. This is evaluated by participants better than "normal", 3.75, as showed in table6.6. Not the best, but Vib-connect can be said that it accomplished this requirement.

6.5 Summary

In this section, we have evaluated Vib-connect using the prototype we have made. We got an extremely high accuracy, and good result in usability evaluation. Though, we also discovered some problems about Vib-connect in this evaluation. In the next chapter, we would like to explain about the future work we are intending to do, to solve these problems. Then, at the last of next section, we would to conclude this thesis.

Chapter 7

Future Work and Conclusion

In this chapter, we will discuss about the future work and conclude this thesis.

7.1 Future Work

Even though, Vib-connect allowed non-expert users to select devices, they still have some problems. In this section, we would like to show them, and explain about the future works we are intending to solve them.

7.1.1 Improving Speed of Vibrate Communication

The speed of vibrate communication in the prototype was 4 b/s. With prototype, handling over raw device information by vibrate communication, will take over 8 seconds. This enables Vib-connect to work without pattern server, but 8 second is too long for users to select device without stress.

To realize more stress-less device selecting interface, we are intending to improve the speed of vibrate communication, by improving the pattern recognition algorithm.

7.1.2 Identifier of the Service Device

As some users mentioned in the previous chapter, existence of Identifier might cause visual change and interior influence against the service device. Using color tag as an identifier has an great advantage in making users recognize the devices and their relationships easier, but also has risk to influence the interior of the service device.

As a future work, we are intending to improve the identifier as a way that does not influence the interior of the service device.

7.1.3 Implementation on Smart Phones

Vibrate communication is robust against environmental influence such as sound and typing. Though, since it uses acceleration, the devices must be put down in a calm place, such as indoor. If we can distinguish the vibration from everyday motion, by using only acceleration, vibrate communication can be used in various situations. We are planning to accomplish this by improving the vibration detecting algorithm.

7.2 Conclusion

In this thesis, we have proposed Vib-connect, a vibration based device selecting interface. To select devices to collaborate, this thesis adopted Touch & Paste interaction which associates devices and set indicator to the real world, with a simple interaction. We have created a vibrate communication algorithm and implemented them as a prototype of Vib-connector on android smart phone. In addition, we have implemented several applications on MacBook series, which simulates network speaker and network projector. Then, I evaluated the accuracy of vibrate communication and the usability of our prototype. As a result, we presented the current implementation of Vib-connect which realized easy device selecting for non-expert users.

With Vib-connect, users can select devices for collaboration, without any kind of technical expertise. This alleviated burden of device selecting task, and should lead to development of device collaborating services that better suit the real-life needs of many end-users, and therefore greatly contribute to the field of ubiquitous computing research.

Acknowledgments

The process of writing this thesis made me realize that one can experience such a deep sense of gratitude. Now, I feel more grateful than I ever have before.

First of all, I would like to express my deepest gratitude to my advisor, Professor Hideyuki Tokuda. Professor Tokuda always took time out of his extremely busy schedule to support and guide me.

I would like to extend my gratitude to members of the HT & KT laboratory, especially to Professor Kazunori Takashio, Professor Jin Nakazawa, Tomotaka Ito, Naoya Namatame, Takahiro Nozawa, Takayoshi Araki, and Kenji Yonekawa for their advice and encouragement. Special thanks go to members of the move! group - especially to Kazuya Nishi, Ryota Niwa, Tetsurou Horikawa and Mao Ueda - for their friendship and support. My experience here at SFC has been all the more enriched by the candid comments we have exchanged over the years. Without their help, I would never have been able to enjoy such a productive life as a budding researcher.

I would like to express my sincere thanks to associate Professor Takuro Yonezawa for his patient revisions of this thesis. He led me through the writing process, which gave me a new perspective on my own work. I would also like to thank NICT (National Institute of Information and Communications Technology), who have supported this project as a part of the Dynamic Network Projcect.

Last but not least, I would like to take this opportunity to express my sincere gratitude to my family - especially to my parents, Kumiko and Toshihiko Nakahara - for always supporting me. Thinking of them and other family members and relatives often chased my blues away.

January 19, 2011

Hiroshi Nakahara

Bibliography

[1] Mark Weiser. Some computer science issues in ubiquitous computing. *Commun. ACM*, 36(7):75–84, 1993.

[2] Wireless hd consortium. http://www.wirelesshd.org/.

[3] Transfer jet consortium. http://www.transferjet.org/.

[4] Dlna (digital living network alliance). http://www2.dlna.org/.

[5] Viera link (panasonic corp.). http://panasonic.jp/viera/link/.

[6] Bluetooth. www.bluetooth.com/English/Pages/default.aspx.

[7] Tsukada Koji and Yasumura Michiaki. Ubi-finger : Gesture input device for mobile use. *Transactions of Information Processing Society of Japan*, 43(12):3675–3684, 2002-12-15.

[8] Yuji Ayatsuka, Nobuyuki Matsushita, and Jun Rekimoto. Gaze-link : A new metaphor or real-world oriented user interfaces. *Transactions of information Processing Society in Japan*, 42(6):1330–1337, 2001-06-15.

[9] Tomotaka Ito, Katsuya Hashizume, Kyohei Kawada, Naoki Nakagawa, Naoya Namatame, Masaki Ito, Jin Nakazawa, Kazunori Takashio, and Hideyuki Tokuda. Snappy: A snap-based human interaction for multiple device collaboration. *Pervasive*, pages ppNA–ppNA, 5 2009.

[10] Jun Rekimoto. Synctap: synchronous user operation for spontaneous network connection. *Personal Ubiquitous Comput.*, 8(2):126–134, 2004.

[11] Lars Erik Holmquist, Friedemann Mattern, Bernt Schiele, Petteri Alahuhta, Michael Beigl, and Hans-W. Gellersen. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *International Conference on Ubiquitous Computing(UbiComp)*, 2001.

[12] Genta Suzuki, Shun Aoki, Takeshi Iwamoto, Daisuke Maruyama, Takuya Koda, Naohiko Kohtake, Kazunori Takashio, and Hideyuki Tokuda. u-photo: Interacting with pervasive services using digital still images. In Hans-Werner Gellersen, Roy Want, and Albrecht Schmidt, editors, *Pervasive*, volume 3468 of *Lecture Notes in Computer Science*, pages 190–207. Springer, 2005.

[13] Hideki Shimada, Naoya Sakamoto, Masakazu Okada, Ryota Ayaki, and Kenya Sato. Evans : The visualization system for wireless network using augmented reality technology. *Multimedia , Distributed, Cooperative, and Mobile Symposium(DICOMO)*, 2010.

[14] Wivia. www.uchida.co.jp/wivia/index.html.

[15] p-stick. www.ubigraph.com/products/P-Stick/.

[16] Yuji Ayatsuka and Jun Rekimoto. transticks : Media virtually connected with each other. 46(7):1671–1681, 2005-07-15.

[17] Yuya Negishi and Nobuo Kawaguchi. A desktop screen sharing system based on various connection methods. *The transactions of the Institute of Electrical Engineers of Japan. C A publication of Electronics Information and System Society*, 125(12):1882–1890, 2005-12-01.

[18] IWASAKI YOHEI, KAWAGUCHI NOBUO, and INAGAKI YASUYOSHI. Touch-and-connect : A connection request framework for the ubiquitous computing environment(ubiquitous computing)(¡special issue¿next generation mobile communication systems). *Transactions of Information Processing Society of Japan*, 45(12):2642–2654, 2004-12-15.

[19] Daniel Shiffman. Processing sudden motion sensor library. www.shiffman.net/p5/sms/.

[20] Javazoom. Jlayer - mp3 library for the java platform. www.javazoom.net/javalayer/javalayer.html.

[21] Jakob Nielsen. *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995.

[22] Mary J. LaLomia and Joseph B. Sidowski. Measurements of computer satisfaction, literacy, and aptitudes: A review. *International Journal of Human-Computer Interaction*, 2, 1990.