User Preference Learning System for Tangible User Interfaces

Kazumi Matsui
Osaka University
1-5 Yamadaoka,
Suita, Osaka, Japan
matsui.kazumi@ist.osaka-u.ac.jp

Tsutomu Terada
Kobe University
1-1 Rokkodai-cho, Nada-ku,
Kobe, Hyogo, Japan
tsutomu@eedept.kobe-u.ac.jp

Shojiro Nishio
Osaka University
1-5 Yamadaoka,
Suita, Osaka, Japan
nishio@ist.osaka-u.ac.jp

Abstract

There is increasing demand for tangible user interfaces (TUIs), for easy and intuitive control of applications. In order to allow use of TUI devices for controlling applications, a system needs to associate functions with the TUI devices. Although several TUI applications and toolkits have been proposed, it is difficult to use them for various applications since they require the user to perform several inconvenient operations. Therefore, we propose a preference learning algorithm for an automatic matching between TUI devices and application functions. A user can freely place and easily operate the TUI devices, and our system creates and learns user's preference data. Our system does this by determining three concrete types of user preferences specialized for each application and six abstract types of those common to all applications. Our proposed system has achieved over 80% accuracy in automatic matching.

1 INTRODUCTION

Recently developed computer applications usually have advanced functions and provide convenient computing environments. However, controlling these applications have become increasingly complicated, and it takes a long time to learn to use such functions. Tangible user interfaces (TUIs) is a promising solution to this problem. TUIs enable us to intuitively operate computer applications by manipulating physical objects with our hands. For example, Musicbottle is a bottle-shaped device that plays a melody when it is opened [3], and ActiveCube is a block-shaped input device for three-dimensional computer graphics [5]. Users can operate applications easily and intuitively with a TUI as if they were manipulating real objects. However, these systems are usually specialized for a particular application, and there is a need for TUIs that are more widely applicable, such as using TUIs for controlling PC applications.

TUI enables users to operate applications intuitively, for example, scrolling a screen by moving a real slider up and down is more intuitive than clicking on a button on a screen via a mouse. Moreover, allocating frequently used functions to real buttons increases an application's intuitiveness and ease of use for beginners. Several types of these products are available, and research is being done to develop new ones. However, conventional TUI toolkits have certain problems that prevent them being easy to use for various applications. For example, Phidget [2] requires users to have technical knowledge of programming, and VoodooIO [7] requires the user to manually associate devices with functions each time they are used.

A system was proposed for creating TUIs by automatically associating devices with PC application functions for which a TUI is used [6]. However, this system requires the users to manually create user preference information. Additionally, since the created preferences are specialized for each application, they cannot be used for new applications.

We have designed and implemented a system for creating and updating user preference information that can be applied to existing and new applications without requiring the user to perform annoying tasks. We discuss a pilot study for extracting user characteristics of physical device layout and the assignment of functions for each device. We propose an automatic learning algorithm based on the results of this pilot study, which enables users to develop a TUI without requiring them to perform any annoying tasks. Thus, our system enables users to easily construct, customize, and reuse TUIs.

2 ENVIRONMENTAL ASSUMPTIONS

Figure 1 shows the assumed environment. Users use a TUI to control PC applications when they want to intuitively operate an application. In our system, we use the Pin & Play TUI devices [1], which enable users to allocate devices freely on a board according to the application functions and each user's device preferences. The proposed system automatically associates the functions with the TUI devices

978-0-7695-3575-3/09 $25.00 © 2009 IEEE
DOI 10.1109/CISIS.2009.58

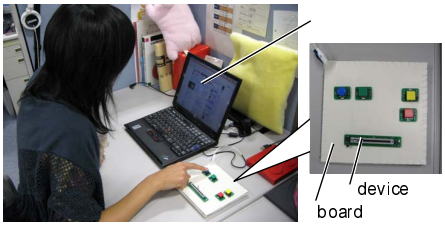


Figure 1. Snapshot of our system being used

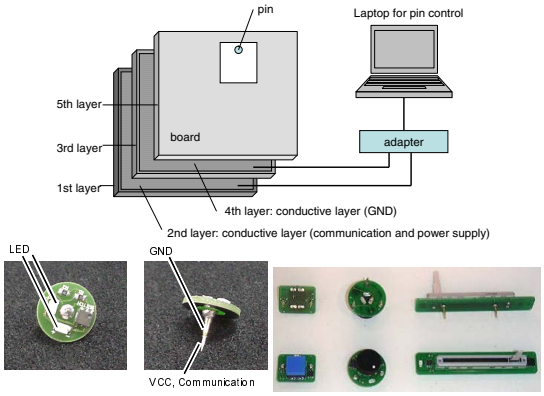


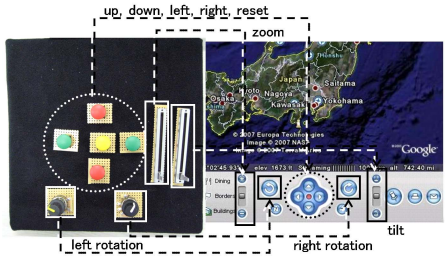
Figure 2. Pin & Play System and devices (button, dial, and slider)

and the user can then control the application with the TUI. When the user changes an application, the assignment is automatically recalculated.

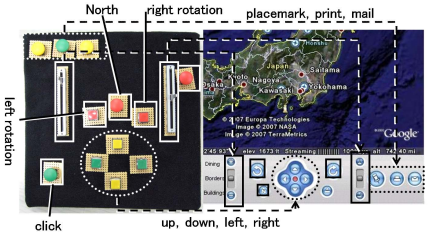
Figure 2 shows the structure of the Pin & Play system, which consists of a board for communication and power supply, and pin-shaped devices that are inserted into the board. The board consists of five layers. The second and fourth layers are made of a conductive fabric, and the others are made of an insulator. The second layer is connected to the communication and power supply ports of the adapter and the fourth layer is connected to the ground port. Each device has two ports, and each port is connected to each conductive layer when the device is inserted into the board. Each pin has its own ID and LEDs. The device types include pins, switches, buttons, and sliders. We previously proposed an image processing method to detect the position of inserted devices [4]. This is done by capturing images of the board. When a new pin is inserted, the pin blinks so that its position will be detected by an image processing system. Pin & Play devices enable flexible user interfaces since users can attach them anywhere on a board.

3 PROPOSED ALGORITHM

The purpose of our current research is to develop an environment where we can intuitively and easily operate various



(a) Participant A



(b) Participant B

Figure 3. Examples of layouts

PC applications through a TUI. Our system automatically develops user preference information. Using our method, the system learns a user's preferences concerning function assignment. Our method enables users to use applications with a variety of TUI devices without dealing with time-consuming operations.

3.1 Pilot Study

Preferences vary among users regarding the allocation of devices and the linkage between devices and functions, and the variety of these preferences strongly affects our matching algorithm. We performed a pilot study to determine the most important assignment decision factors. The participants were twelve university students in their twenties. Each participant played games using Pin & Play to get used to the TUI before the experiment. The procedure of the experiment was as follows. First, we showed the participants several commonly used PC applications such as a Web browser (e.g., Internet Explorer), a music player (e.g., Windows Media Player), and a mapping application (e.g., Google Earth). Each participant was then allowed to freely locate Pin & Play devices and assign the function for each device. The same procedure was then repeated with the most common functions's assigned to the devices.

From the results of the pilot study, we could see various patterns of device layout and function assignments. Figure 3 shows an example of device layout and function assignment for mapping software. Participant A located two slid-

Table 1. Participants' preferences extracted from pilot study

	Participants' Preferences	Precision
1	Locating devices according to the position in the GUI	7/9
2	Allocating devices in logical order/layout	9/9
3	Locating devices with similar purposes close together	9/9
4	Assigning relevant functions to same-shaped devices and irrelevant functions to different-shaped devices	9/9
5	Linking functions with similar purposes to same-shaped devices for every application	6/9
6	Aligning devices in the same direction if their purpose is related to direction	8/9
7	Locating devices based on personal preferences without considering their GUI position	4/9

ers, one for the zoom function and one for the tilt function, on the right side, while Participant B placed these sliders on the opposite side of the board as in the GUI layout. In other words, there was a difference between the device-layout strategies of the participants. In addition, the participants' placement of the buttons for the map-moving function in a cluster suggests a strategy of locating closely related devices within the same positional cluster.

Table 1 shows the layout preferences of all participants and the frequency for each preference. We found that when there is a clear relationship between functions, people locate devices according to the relationship. Moreover, although most participants located devices in positions similar to the corresponding positions in the GUI, some participants laid out the devices based on personal preferences without considering the GUI layout.

3.2 Matching Algorithm

We discuss our proposed system needed for use of a TUI. From the pilot study results, the matching system associates application with TUI devices by determining three factors: the characteristics of each function, the positional relationship among functions, and the semantic relationship among functions.

Characteristics of each function

when users operate applications, the devices used and their most suitable position depend on the function characteristics. This strategy is based on preferences 1, 5, 6, and 7 from Table 1. In the pilot study, we found that the scroll-bar function was always associated with a slider and placed on the right or at the bottom according to the corresponding GUI position, and functions normally operated by the left hand were placed on the left.

Positional relationship

devices are intuitively placed by considering the positional relationship. This is based on preferences 2 and 7. In the pilot study, four buttons for map movement were located on the top, bottom, left, and right relative to their directions, and the page-forward button of the Web browser was consistently located to the right of the page-back button.

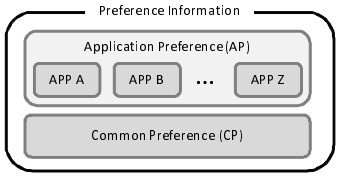


Figure 4. Structure of preference information

1. Position SimilarityToGUI (PS):
   Position similarity of devices to GUI position.
2. Order Compliance (OC):
   Order compliance of functions with meaningful order.
3. Relation Adjacency (RA):
   Position adjacency among related functions.
4. Relation Uniformity (RU):
   Device type uniformity among related functions.
5. Common Uniformity (CU):
   Device type uniformity among same functions between applications.
6. Direction Concordance (DC):
   Direction concordance between GUI shift and device operation.

Figure 5. Common preferences

Semantic relationship

users tend to cluster devices that have similar purposes or should be coupled semantically, as shown by preferences 3 and 4. In the pilot study, for example, participants tended to locate the mute button close to the volume control dial.

The above three factors include both application-dependent and application-independent preferences. In the pilot study, one user changed the placement of the stop and play buttons according to the type of music player.

By considering these results, the system has both an application preference (AP) of each application and a common preference (CP) for each user, as shown in Figure 4. Therefore, by using only the common preference, the system adapts the algorithm to new applications that the system does not have an application preference for yet. To be concrete, an application preference consists of detailed and concrete preferences specific to the application such as the characteristics of each function and the positional relationship among functions. A common preference consists of six general-purpose factors, shown in Figure 5.

The matching system associates application functions

with TUI devices by determining the above factors. The system grades the observance degree of the above factors and prepares the preference information in advance.

In keeping with Table 1, we decided these six factors as making up the common preference. The observance degree to each factor of users' preference is represented as a ten-grade score. The system determines the seventh factor in Table 1 (locating devices based on personal preferences) detailing the application preference. The application preference includes the preferences of the device type of each function, those of the positional relationship of functions and those of the distance relationship. The system expresses them as an open-ended score.

Using this preference information, the system lists all possible combinations between functions and devices, and calculates each evaluation value for the assignments. Then the system uses the combination that has the largest evaluation value, matches application functions to the TUI devices and graphically displays the matching result to the user.

3.3 Learning Algorithm

Based on the pilot study results, we designed a user preference learning system by considering both application and common preferences. When each application preference is updated, it is necessary to reflect that change to the common preference. Preference learning is done through the following steps.

STEP 1 Structure Application Preferences

The system first creates or updates the application preference information using the following two modes.

Error-correction mode:
When the assignment differs from the user's intention, the user can use the error-correction mode. This mode allows the user to inform the system of incorrect assignments by operating the TUI devices. Then the system revises the application preference by escalating the score that contributed incorrect assignment. As for the scores of position and distance, the revised value also has an effect on the scores of nearby grade, and the system avoids making the partial score from the strict position of only that time. After revising the score, the system associates the functions to the devices again, and repeats these steps until the system's assignment agrees with the user's intention. In this case, the system revises the score by determining the incorrect assignment history informed by the user.

Create-information mode:
In cases where there is no preference information, such as the first use, it takes time to develop preference information through the error-correction mode. Therefore, the system has a create-information mode. This mode presents a sample application to the user and allows the user to inform the system of the intended assignments. Based on the information, the system automatically creates an application preference. The system extracts the users' preferences from the function assignments, changes them to scores, and develops the application preference.

STEP 2 Structure Common Preferences

Based on the application preference created or updated in STEP 1, the system extracts a common preference. The system counts the number of application preferences that supports each common preference, converts the ratio into a ten-grade score, and updates the common preference as a score. The rate is not directly changed to a common preference. The system sets the rate threshold and views the subthreshold ratio as inadequate. A ratio larger than the threshold is converted into a ten-grade score through the following formula.

$$S_i^c = (\frac{P_i^s}{P_i^a} - P^t) \times \frac{(1 - P^t)}{9} \quad (1)$$

For the common preference i, S_i^c is the score of the common preference, i, P_i^a is the number of the application preferences that support and do not support the common preference, i, P_i^s is the number of those that supports the common preference, i, and P^t is the threshold of the support ratio. The system sets the steady score as the support score and views the application preferences that are over the support score as good enough to support. P_i^s is the number of these supported preferences.

STEP 3 Restructure Application Preferences

The matching algorithm adds the score of the application and common preferences, and the application preferences applicable to each common preference in STEP 1 are needed to reduced the common preference.

Then each application preference that was involved in structuring the common preference is subtracted from the total score of all the involved common preferences and updated as a new application preference.

4 IMPLEMENTATION

We have implemented a prototype system. In the system, when a user placed devices (sliders, dials, buttons, and other TUIs) freely, the system automatically calculated the assignment of functions to the foreground application by referring to the application function and the preference descriptions. The user then controlled the applications using the TUI devices. Additionally, the user can select the error-correction and create-information modes using hot keys.

```
<function>
  <v_scrollbar>
    <h_position value="same" score="5" />
    <v_slider num="1" color="" score="4" />
    <dial num="1" color="" score="3" />
  </v_scrollbar>
  <prev>
    <v_position value="same" score="2" />
    <button num="1" color="" score="2" />
  </prev>
</function>
```

Figure 6. Example application preference

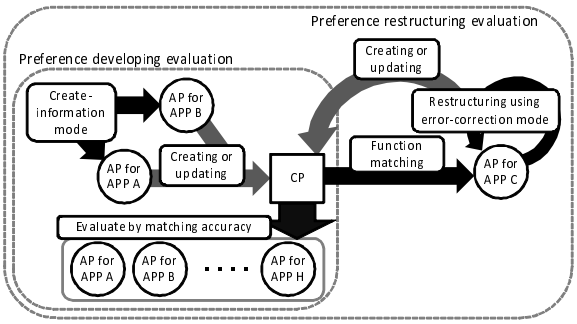


Figure 7. Pattern diagrams of evaluation

We also implemented a simulator using Microsoft Visual C#.NET on Microsoft Windows XP Professional which realizes functions as in actual use of TUI devices.

Figure 6 shows an example description of an application preference. It includes detailed preference information of the device types and positions for the functions.

5 EVALUATION EXPERIMENT

We evaluated our system using three types of applications: three Web browsers, three music players, and two mapping applications. Participants in this experiment were eleven university students in their twenties who were familiar with using a PC. In this experiment, the participants could freely locate TUI devices on a board for several specified functions. We evaluated the preference development algorithm using the create-information mode and the accuracy of the updating preference using the error-correction mode. These evaluations are described in detail below.

5.1 Preference developing evaluation

We evaluated the threshold and its accuracy in creating preferences. We prepared the preference information including no preference and applied the correct assignment information of two applications to the algorithm using the create-information mode, as shown in Figure 7. Using the created information, the system matches devices and functions of eight applications, including two applications that were used in creating the information, and we evaluated the function matching accuracy.

Table 2. Threshold evaluation results

Confidence threshold	0.9	0.8	0.7	0.6
Matching rate	0.6155	0.6651	0.6881	0.7321

0.5	0.4	0.3	0.2	0.1	0
0.7575	0.7569	0.7571	0.7562	0.7564	0.7509

5.1.1 Threshold evaluation

The threshold (P^t) has an effect on creating common preferences. To adequately set the threshold, we evaluated the accuracy of the development of common preferences by changing the threshold.

The evaluation results are shown in Table 2. We found that there were few differences in the confidence threshold and the stable matching rate in the threshold range from 0.2 to 0.5. Therefore, it is acceptable to set the threshold within that range. In a high threshold, the useful ratio also may be lower than the threshold. Therefore, the ratio was viewed as unuseful and then the useful ratio does not lead to a proper common preference score.

5.1.2 Accuracy evaluation

To evaluate the accuracy of creating-preference information, we examined the function matching accuracy of eight applications per participant.

Based on the above evaluation results, we used the appropriate threshold, and the overall matching accuracy was 76%.

We evaluated the matching accuracy of eight applications including two applications that were also used for creating preference information. For these applications used for creating information, our system achieved a 96% success rate, and for the another six applications, our system achieved a 65% success rate. Since the system has no application preference and uses only the common preferences for matching for the six applications, the success rate was lower than that with manually prepared information. However, we found that this success rate indicates the effectiveness of the algorithm if we use this together with the error-correction mode described in the next section. Additionally, we confirmed that the matching system adapts easily to the applications even if the system has no application preference for the applications.

Table 3. Evaluation results of updated algorithm

update frequency	no	after 5 AP updates	after 10 AP updates
average correction times	8.9	9.11	8.7
average CP update times	0	0.67	0.23
matching rate	-	80.1%	81.3%
upgrade matching rate	-	9.2%	12.5%
updated sample rate	-	78/254	51/254
upgrading sample rate	-	78/87	51/51
downgrading sample rate	-	6/87	0/51

5.2 Preference restructuring evaluation

For evaluation the updating information algorithm, we evaluated the function matching accuracy with the information that is created through the create-information mode and updated through the error-correction mode, as shown in Fig. 7. First, we obtained the preference information through the create-information mode with two applications. If the system's assignment of the information differed from the participant's intention, the system was informed of the incorrect assignment through the error-correction mode and corrected the application preference score. Then the system periodically updated the common preference and restructured the application preference. Finally, using the preference information, we evaluated the matching accuracy between the system's assignment and the users' intention.

In this evaluation, since the updating frequency is affected by the matching accuracy, it is important to decide how many times the system updates the application preference before updating the common preference. In accord with the average correction time, 8.9 times, the evaluation was, therefore, conducted in two patterns: the common preference was restructuring after 5 or 10 updates of the application preference. Table 3 shows the results of this evaluation. In either case, the assignment accuracy was over 80%. We confirmed that users can easily create preference information, and the system restructures the preference information through learning users' preferences.

After updating the application preference 5 times, we found that the assignment accuracy of many samples were improved by restructuring since this pattern has a higher frequency. On the other hand, some samples needed more correction updates. After 10 updates, for less frequency, we found that assignment accuracy improved in some samples and no samples were negatively affected.

The improvement rate after restructuring was better in restructuring after 10 updates, and there was much improvement in samples with restructuring after 5 updates. Considering this result, improved efficiency is needed in restructuring after 5 updates. We should take measures, such as setting a threshold of the number of application preferences that are used to restructure common preferences and avoid low-trust preference restructuring. We also believe that an improvement plan should be available, such as restructuring against preferences that are not updated over 10 times in the case of restructuring after 10 updates.

6 CONCLUSION

We have designed and implemented a system for automatically developing and restructuring preference information used in a matching system between application functions and TUI devices. Based on pilot study results, our system managed six common preferences and three application preferences. The evaluation showed a matching accuracy of over 80%. When the system's assignment did not match those of a user's intention, the user could easily and effectively restructure his or her preferences using the error-correction mode. Users could, therefore, comfortably operate PC applications through TUIs using our system.

Our system can be adapted to any other TUI by using the preference information of position or device types for functions. Therefore, in our future work, we plan to adapt our algorithm to other TUIs and to evaluate the use of our system by beginner computer users, such as children, and the elderly.

References

[1] The pin & play project; http://ubicomp.lancs.ac.uk/pin & play/.

[2] S. Greenberg and C. Fitchett. Phidgets: easy development of physical interfaces through physical widgets. In *Proceedings of the ACM symposium on User interface software and technology (UIST 2001)*, pages 209–218, 2001.

[3] H. Ishii, M. A., and J. Lee. Bottles as a minimal interface to access digital information. In *in Extended Abstracts of Conference on Human Factors in Computing Systems (CHI 2001)*, pages 187–188, 2001.

[4] Y. Kishino, T. Terada, N. Villar, W. H. Gellersen, and S. Nishio. Position detection mechanism using camera images for pin & play. In *in Adjunct Proc. of 17th International Conference on Ubiquitous Computing (UbiComp 2005)*, 2005.

[5] Y. Kitamura, Y. Itoh, and F. Kishino. Real-time 3d interaction with activecube. In *in Extended Abstracts of Conference on Human Factors in Computing Systems (CHI '01)*, pages 355–356, 2001.

[6] K. Matsui, Y. Kishino, T. Terada, and S. Nishio. An automatic matching algorithm between devices and application functions for tangible user interfaces. In *Proceedings of the IADIS Interfaces and Human Computer Interaction (IHCI 2007)*, 2007.

[7] N. Villar, M. K. Gilleade, D. Ramduny-Ellis, and W. H. Gellersen. The voodooio gaming kit: a real-time adaptable gaming controller. In *Proceedings of Advances in Computer Entertainment 2006 (ACE 2006)*, 2006.