Wiinteraction: A study on smart spaces interaction using theWiimote

Hugo Seixas, Nuno Salgado, Rui José

University of Minho, Portugal

Abstract. This paper describes a study on the use of the Wiimote as a generic interaction device for smart spaces. We have identified the range of interaction possibilities that can be explored when creating Wii-based interfaces for smart spaces and we have explored the application of those interactive features in the context of a typical museum guide. While there are many features that can be explored creatively to sustain the use of the Wiimote as a generic interaction device, we also found that there is at least one critical requirement that is not supported. More specifically, we have identified the need to include access to rich information, such as the one provided by digital displays. We thus propose a shared control mechanism for public displays that enables a user equipped with a Wiimote to first gain control and then browse information in a public display. The results of our study show no major limitation in the proposed approach, but identify device discovery as one major technical flaw that still needs to be overcome before the wiimotemay realistically become a generic interaction alternative for smart spaces.

Keywords: Wiimote, interaction, public displays, museum guides

1 Introduction

Exploring new interaction models has always been a very active research topic in ubiquitous computing, the overall goal being to enable interaction with computers in the multiple situations of everyday life in which the traditional desktop metaphors no longer apply. Small mobile devices, such as mobile phones and PDAs, have been extensively explored for this purpose, but in their essence they are still very close to the desktop metaphor. Being designed as universal tools they are too complex and distracting for many scenarios in which physical exploration is the core of the user experience. On a completely different path, there has also been considerable research in conceiving and experimenting with multiple types of dedicated sensor and interaction devices, but to date we still do not have any alternatives with widespread use and easy deployment.

In this work we explore the potential of the Wii Remote as an alternative interaction device for situated services. The Wii Remote, which has unofficially been nicknamed "Wiimote", is the wireless controller for Nintendo's Wii console. The Wiimote is equipped with accelerometers, enabling motion sensing and gesture recognition, an optical sensor (Infrared), numerous buttons and also basic output through four blue LED lights, a rumble and a speaker. In addition to its technical features, there are some important points that can make it very attractive as an interaction device for smart spaces scenarios: firstly, the Wiimote has been reversed engineered and there are now numerous libraries enabling it to be used as generic

controller for any computer and using multiple programming languages; secondly, Wiimotes are an off-the-shelf product that is robust, widely available, and can be bought separately at a price range that is comparable to the price of a wireless mouse; thirdly, with the expansion of the game console market, these devices are increasingly part of the digital culture, with many people being already familiar with its commands and interaction modalities. This means that many people will be able to achieve a reasonable level of intuitive and easy handling, without the need for any long, descriptive learning processes.

Despite this obvious potential, the Wiimoteoriginates from the game consoles world and was conceived for a rather specific scenario, being far from obvious how its features can be leveraged as enablers for interaction in smart spaces.In this work, we take the almost canonicalexample of the museum guide as the background for a study on the interaction design space of the Wiimote. Our purpose with this study is to explore the range of alternatives for Wiimote-based interactions and provide a framework that can be useful to smart space designers. We have also implemented a shared control mechanism that enables Wiimotes to gain access and control a public display for information access. This has enabled us to test the overall technical feasibility of the proposed system and to evaluate some of the interaction concepts with users.

2 RelatedWork

There has been considerable work in exploringnew ways to use Wiimotes through some type of hacking.The most well-known and influent work has been done by John Chung Lee who published numerous projects of new applications for the Wiimote:

- Tracking Your Fingers with the Wiimote. "*Using an LED array and some reflective tape, you can use the infrared camera in the Wii remote to track objects, like your fingers, in 2D space."* [1]
- Low-Cost Multi-point Interactive Whiteboards Using the Wiimote. "*Since the Wiimote can track sources of infrared (IR) light, you can track pens that have an IR led in the tip*." [1]
- Head Tracking for Desktop VR Displays using the WiiRemote. "*Using the infrared camera in the Wii remote and a head mounted sensor bar (two IR LEDs), you can accurately track the location of your head and render view dependent images on the screen.*" [1]

There are many other similar projects, and in particular many projects focusing on the use of the accelerometers for gesture recognition:

- Gesture Recognition with a Wii Controller is a project that studied "*recognizing gestures to interact with an application and present the design and evaluation of our sensor-based gesture recognition*". [2]
- Wii Remotes as Tangible Exertion Interfaces for Exploring Action-Representation Relationships is a project that based their study in: "*Sensor technologies and exertion interfaces offer new opportunities for interaction with digital data. Technologies such as Wii Remotes (…) "*. [3]

- Wave Like an Egyptian — Accelerometer Based Gesture Recognition for Culture Specific Interactions. The authors of this project studied "*how the Wiimote can be utilized to uncover the user's cultural background by analyzing his patterns of gestural expressivity in a model based on cultural dimensions*". [4]

Another important type of work regarding the Wiimote, are the Wiimote libraries, through which Wiimotes can be integrated as interaction devices for any computer program.

- WiiGLE - Wii-based Gesture Learning Environment is the implemented project that originates the paper referenced before. [5]
- WiimoteLib: It is written in C# and VB.NET and consists on using nintendo controllers in .NET Applications. [6]
- WiiUse: It is a library written in C that supports all features except gesture recognition. [7]
- WiiUseJ: It is a Java API to use Wiimote on the computer. Is bult on top of WiiUse.[8]
- WiiRemoteJ: It is another Java API to use Wiimote on the computer. [9]
- Wiigee: Implemented in Java, "is an open-source gesture recognition library for accelerometer-based gestures specifically developed for the Nintendo® Wii™ remote controller." [10]
- GlovePIE: Is the Glove Programmable Input Emulator. It can be understood like a script interpreter that allows a basic mapping by Wiimote (or other joystick device) to keyboard keys. [11]

For a more detailed list of projects and Wiimote based documentation, consult [12], [13], [14] and [15].

Our work takes a different approach, in that we are not trying to propose any new sophisticated Wiimote-based interaction, but instead trying to understand how the Wiimote could be leveraged as a generic, off-the-shelf device for mundane interaction in everyday life.

3 Exploring the interaction design space

The Wiimote provides a wide range of sensing and actuation mechanisms that were originally conceived to integrate game experiences. Our first task in studying the role of the Wiimote as a tool for smart spaces was to study those multiple interaction mechanisms from the perspective of their most basic affordances. The objective was to clarify the range of possibilities that can be explored when creating wii-based interfaces for smart spaces. We have identified the following categories of sensing and actuation mechanisms:

- **Detecting a Wiimote** – The most basic function of the Wiimote is as a presence detection device. Given the use of Bluetooth as the connection technology, a simple process of Bluetooth scanning is enough to obtain information on which Wiimotes are currently near each Bluetooth access point. This can be done even if the Wiimote is not connected with the scanning device. In a scenario of one

Wiimoteper user, this may be used as a process for detecting the presence of a particular person.

- **Basic Button Interaction**–The Wiimote is equipped with a diverse set of buttons. Any arbitrary action can be triggered either by pressing a single button or by using some preset combination. As long as some meaning is communicated to the use of those buttons they can easily be associated with any actions.

- **Accelerometers** – The accelerometers offer great potential as an interaction enabler, both for implicit and explicit forms of interaction. Based on the level of involvement required by users, we have identified the following main types of use for the accelerometers:

    1. Basic activity recognition: This corresponds to using accelerometer data for recognising possible activities by the owner, such as walking or seating. While the interpretation of the data generated may not be very robust in general, the interpretation within the context of a specific place and within the framework of the activities common to that place, may offer a simple and yet useful mechanism for basic activity recognition.
    2. Explicit actions through basic movements: This corresponds to basic movements that need to be communicated, but do not require any training. This may include simple gestures such as swinging the Wiimote, gently hitting with the Wiimote on a particular surface, or bumping two Wiimotes together (for example as a pairing mechanism). The main idea is that gestures should be simple enough not to require any training.
    3. Trained Gestures for Interaction: If we consider that people will have the opportunity to train more elaborate gestures, we can use many more types of movements as interactive actions. For example, drawing a square or a circle with the Wiimote to trigger some specific action. If the person trains the gestures, it is also possible to train “secret gestures” defined by each user, which can serve as signatures or authentication signals. That kind of signatures or authentications could be some mechanism of identification for the users in the scenario of shared Wiimotes. However, using trained gestures just to trigger a particular action may not be very efficient because it requires training, it is prone to errors, and it may be awkward for someone in a public place. This type of gestures is probably more suitable in scenarios similar to games in which the intensity and extension of the gestures is itself relevant.

- **IR LEDs in a grid seen by the Wiimote camera**–Given that the infrared camera in the Wiimote can very accurately detect the position of points of infrared light, it becomes simple to use infrared LEDs in a grid in order to allow pointing, moving or dragging actions. This corresponds to the standard approach used by many Wii games in which the distance and inclination of the Wiimoteis determined by measuring the relative position and relative distance of light points generated by the LEDs in a standard grid with known points placed at known distances. This feature can be understood as the movement of a mouse in a

standard desktop system. Alternatively, it should also be possible to have multiple grid patterns and use them to recognise specific locations associated with those patterns.

- **IR pointed to LEDs** - This feature is dual to the one stated above. In this scenario, Wiimotes are static and actively detecting the movement of the LEDs, which will be the ones performing actions. In our scenario, this is not applicable because we are considering that people are carrying the Wiimote.

- **Output** - Wiimotes come with three different ways of providing output:

    1. Audio: a built-in mono speaker can produce beeps to notify the user of any relevant events or some very relevant system state.
    2. Lights: The four LEDS can be used in combination to communicate multiple system states or alerting the user for some specific system state.
    3. Rumble: a small vibrating engine is incorporated in the device and can be used for notification or as simple feedback mechanism to communicate that a particular action has been correctly received, understood and executed.

    These three techniques are complimentary in providing output to the user. The audio and the rumble can be especially useful when there is a need to notify the user of some event. Pre-determined sounds our rumbles can be associated with a short number of relevant events, such as entering a new interaction area or being in proximity of something relevant. The lights are more useful to represent pre-defined states, allowing the user to glance them at any moment to get a quick update on the state of some system variable. This can also be combined with buttons to represent on request the state of specific variables.

4 A Wiimote-based museum visit

This work did not include any real-world deployment in which we could get more information on the practicalities of deploying and using the Wiimote in smart spaces. Instead, we looked at some common digital museum guides and studied how we could support similar functionality using the features available in the Wiimote. We specifically chose this very common scenario for smart space interaction because we thought it would make it easier to understand some of the base requirements and compare some of the results. Building on the interaction space outlined in the previous section, we have made a series of brainstorm sessions on how the Wiimote could be used as the basis for an interactive visit to a museum.For a complete understanding of a typical scenario we tried to find a story that could explain our intentions. We arrived at the following scenario:

Saturday. It's still very early in the morning, but family Molyneux is already in a hurry.

"Culture and Arts: Picappo paintings on exposure at Louvle Museum this weekend". The advertisement flooded newspapers and media the past days and lots of people are expected to come, so they want to arrive soon and avoid an endless waiting line at the entrance.

Louvle is huge. A complete tour would need at last two days to fully visit everything. Albert and Elvire, both art teachers at University, are happy for the opportunity to see one of the best art collections in the world. At the box office, the Molyneux are given a Wiimote that will accompany them during the whole day.

The twins, Enzo and Angeline, however, are not into boring cubism paintings and insist to go and see the science and technology interactive exhibit, on another wing of the Louvle, so another Wiimote is requested. Before departing, the family approaches a terminal to pair both devices. Now the museum system knows that these two Wiimotes are related to each other.

It's the first time at the museum, so they don't know their way around. Their Wiimote rumbles, warning them that there is interactive content nearby. They approach the terminal and look for information. A map shows the way for Picappo Gallery and they go straight there. All their movements are being tracked through the Wiimote and their interactive actions stored: now the museum system knows that the way to Picappo Gallery was searched for and it is the objective of those people.

Finally they reach the gallery. In the room, there are multiple displays where, by using the wiinteraction system, one can find more detailed information on the paintings. Albert and Elvire watched carefully and thrillingly all the masterpieces. The museum knows they didn't leave the gallery for the next three hours.

There was a painting that both loved particularly, so Albert "selects" it with the Wiimote and it is automatically sent to the souvenir's shop where a replica will be printed. At the end of the day, they can go pick it up and bring it home, where they will hang it by the fire pit.

As the lunch time approaches, they go to the restaurant zone. At a terminal they "ring" the other Wiimote, as they were previous paired. Now, as soon as the other Wiimote enters a Wiimote spot (there are hundreds over the museum) it will rumble and give a sonorous warning to their children. By consulting a map, the kids know where their parents' Wiimote is, and head up to the meeting place.

From a conceptual perspective, the use of the Wiimote as the interaction device for this scenario does not seem to offer any major problem. Depending on the back-office functionality, people would be able to trigger any arbitrary actions, coordinate their visit in a group, receive notifications, or bookmark any exhibit for future reference. The only crucial exception was rich information output, such as the one that can be obtained in the screen of a PDA or in a public display.

One of the most common scenarios in any sort of digital guide is the possibility to obtain more information about exhibited items. This is not naturally supported by the Wiimote, which does not have any type of screen. To overcome this limitation, we assumed that the museum would occasionally have some sort of public display and that the Wiimote would be used as an interaction device for those displays.Our implementation is focused on this particular feature and our study specifically addresses to what extent this specific use of the Wiimote could be efficiently supported. Ourscenario is based essentially in basic button interaction and also the output features, coupled with the use of the Wiimote for interaction with public displays.

4.1Implementation

Our implementation is focused on addressing the specific scenario of a public display in a room in a museum that is available for interaction by holders of Wiimotes. As represented in Fig. 1, the system consists in a set of displays (WiiSpots – computer display with a bluetooth dongle) with which Wiimotes are able to connect.

Figure 1: Wiimote interaction with public displays

The system can be seen by two distinct points of view. First, we have the back office that handles the access to the display, following a set of rules that define who has permission to use the application.Secondly, we have the system interface that allows users interaction. This is a web-based interface and should be independent of the implemented back-office. For this project we chose to use theWiiUseJ [14] Wiimote libraries, mainly because we wanted to implement the system in Java. We built an interface based on common menu navigation by binding the directional buttons on the Wiimote to the respective arrow keys that can be used in common menu navigation, as shown in Fig.2.

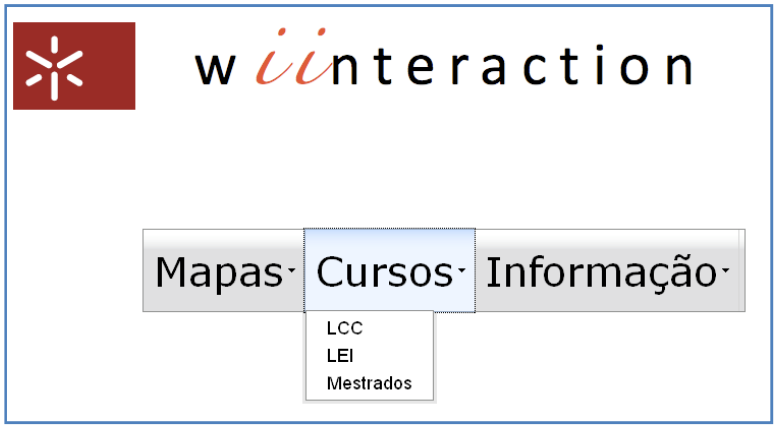

Figure 2: The winteraction display interface.

The front-end of our application is a common web-browser interface. We use the typical HTML and CSS to make the final proposal. The main menu is constructed using Yahoo UI Library. [20]. The most important reason to use Yahoo UI Libraries that this Library allows keyboard navigation. So, the idea is mapping keyboard keys into Wiimote buttons and use this solution to navigate the interface. This enabled the user to easily browse the menus as if they were using the arrows of a keyboard. This kind of solution makes our solution interface independent.

Given our multi-user scenario, we also needed to find a method not to allow a second user from interfering with the one "in charge" of the interaction at one particular moment. It was necessary then, to come up with some sort of protocol able to prevent that from happening, but at the same time, giving some fairness to the usage and avoiding monopolization of the display. So, as soon as a second user approaches a terminal already in use, he or she automatically enters a queue, giving the current user a maximum of three minutes to finish his activities, automatically giving the control to the second in the queue after that time. As soon as a new user is given priority, his Wiimote will rumble, giving him or her a light warning. A visual status of the queue is shown at all times in the screen. In the Wiimote that is currently in charge the left LED is also lightened up to signal control. The application of this access protocol is governed by the state diagram represented in Fig.3.

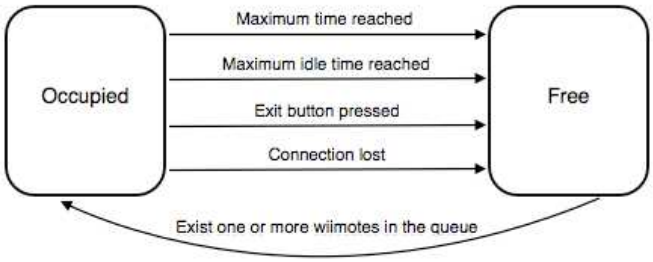

Figure 3: Transactions between display states.

The system also maintains a state for each Wiimote, which is represented in Fig.4

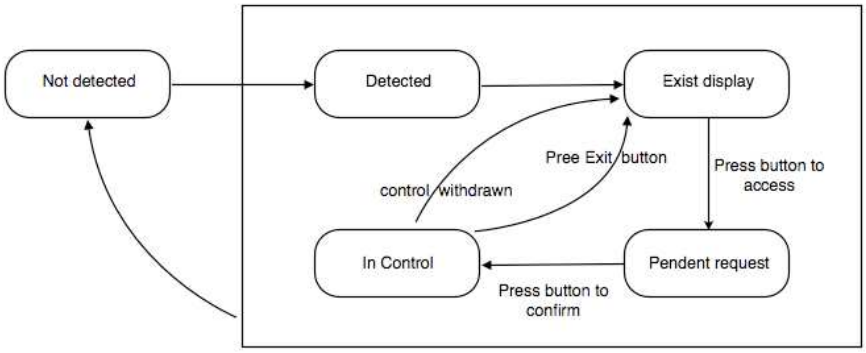

Figure 4: State diagram representing Wiimote states.

The initial state is the case where a particular Wiimote is not being detected. When it is detected, it changes to state detected, and if a public display exists it again changes to Exist Display. In this case, the Wiimotewill rumble and turn on one the LEDs. The user can now press a button to indicate the intention of using the display, causing another shift of the state. Another light will be on indicating the specific transition of the state. At this moment, the user is in the queue waiting for his turn.When his turn arrives, all the lights will blink repeatedly and the rumble will be activated once indicating the shift of the state. At this moment, the user is in control of the display.

Once in control, that control can be withdrawn, for example by exceeding the time limit, or, by pressing the exit button, with which the user can indicate his intention of dropping the control. The user also looses the control after a period of inactivity.

5 Results

To gain a better understanding of the feasibility of the proposed system, we created an implementation of the display control software and set an experimental deployment in which we could conduct some experiences with users. In this section we will now described our main results.

5.1Lessons learned from the implementation

A major effort in this work was devoted to dealing with the Wiimote libraries, and especially with the discovery process that needs to take place when Wiimotes are first detected. For example, using WiiUseJ, discovery is not possible and the only way to connect with Wiimotes is paring the device with wincom windows stack. We tried to use WiiRemoteJ for development. However, although the specification indicates that discovery is possible, the reality has been very different. First, the information related to this API is very limited. Secondly, even if we found information to solve some problems, the reality shows that very often it was not reliable.

Wiimote discovery using WiiRemoteJ seems to be possible in Windows, however, it is necessary to use bluecove [16] for discovery and bluesoleil [17] or widcomm [18] drivers switching the traditional wincom drivers. The problem seems to be solved, but again, we concluded that bluesoleil does not work under windows platform. We also find a big number of references for this in web forums. [19]

So, the last opportunity to solve the problem was to use bluecove with widcomm drivers. One more time we found problems. Widcomm drivers are a Broadcom solution for bluetooth devices and the problem is that these drivers do not support many number of bluetooth devices. In fact, none of our bluetooth devices was supported by this driver, so, we hadn’t opportunity to conclude if it works or not.

Another important conclusion is related with the number of Wiimotes connected at the same time. Using WiiUseJ, by API limitation, the limit is four Wiimotes. However, if we shift our code to WiiRemoteJ API (hoping that the discovery problems will be solved), theoretically, there isn’t any limit to the simultaneous

Wiimote connections. But, in practice, the Windows bluetooth stack imposes a limit for bluetooth devices connected at the same time, and that should be the limit.

In the interface we also found a technological limitation. Using yahoo UI we need to have the menu selected and clicked by the mouse to begin the navigation. There isn't any kind of solution for this problem in this library because, it is impossible to make the menu selected in its initial state. One possible solution is to use one button of the Wiimote as a mouse click and informing the user to click that specific button. The strategy is having the mouse pointer under the menu surface and after that click the menu should be able to be browsed. We solve this problem using Robot solution of J2SE API [21]. The tests indicate that this technical insufficiency is the major usability problem. Once in the application, almost all the users forgot that every time they enter the home page they needed to "click" in the mouse (button plus of Wiimote).

5.2 Users Tests

The target audience for this kind of solution is typically visitors of public spaces. In our specific case, we pretend to reach the visitor of DI (Informatics Department). Thus, we tested our system with five different people. In each case we explain to the person in what consist and what was the purpose of our system. We tried to explain that it would be important to the experience if they imagine that they were actually visiting a museum, or, in our particular case, imagine that they are visiting our informatics department and that there would be several WiiSpots around. We explain all the crucial issues and give to them a guide with all necessary explanations. We then gave them a set of tasks related with specific information available in the system and let them explore the system. At the end, we conducted structured interviews with the participants to collect information about their experience with the system.

The results of our trials have been fairly satisfactory regarding the response of the Wiimote usage by the people, even when our interaction environment is a simple test web page. We believe that one of the major factors of its success is the natural bond and amusement people automatically feel when given a Wiimote to their hands: all users showed immediate mastery at using the device. The management protocol for multiple users was also well accepted and thought fair.

Regarding the interface, the process was very simple to implement: we simply bound the buttons of the Wiimote to specific keys of a keyboard, allowing the user to intuitively perform his actions, demonstrated by the fact that no user needed an explanation on handling the Wiimote and interacting with the system during the trials. We noticed, however, that users had problems with the mouse click. In fact, every time the web page is accessed, users need to press the plus button of the Wiimote to select the proper menu (solution for the technical problem already described in the previous section) in order to map the mouse click, and thus, navigate the menu. Although we explain this question in the guideline, usually the users had problems with this process.

Up until this point, we always tested our system with a Wiimote already paired with the computer where the system was running. In one of our series of trials, we used a second Wiimote not yet paired with the system, confirming one of our

suspects: our libraries do not support active Wiimote discovery, working only with the Wiimotes paired at boot time. With this technical limitation, we can only manage up to a maximum of four Wiimotes at the same time, due to a restriction on windows Bluetooth stack. We hope that in the future, some kind of discovery service may be developed, that would allow us to fully develop our system features. Some testers mentioned the lack of information in the interface itself.

It was also obvious from several interviews that many people were expecting different interaction types. More than half the participants expressed that gesture recognition should be a better solution for this kind of application than basic button interaction. Even though they may be much less reliable, as was also pointed by other participants, gestures may have some advantages: The first is facilitating interaction without forcing the user to look at the Wiimote. Even though this can also be partially achieved with buttons, it may be much simpler by tilting the Wiimote. Also, using gestures may correspond more naturally to the user expectation of using the Wiimote, and that is also how we interpret this result. Even though our sample is small, this is an important conclusion to retain.

6 Conclusions

In conclusion, a lot of features are yet to be added in order to increase our system usability, particularly, taking advantage of the gesture capabilities of the Wiimote. Still, from our results we found nothing absolutely fundamental to makes us think that this idea may not be viable. There are still, some technical issues regarding discovery that need to be solved before this type of system may be put into practice in realistic scenarios. Despite our research, we were not able to find any service capable of supporting Wiimote Bluetooth discovery in a way that matched the requirements posed by our scenario.

Regarding the interactive possibilities, we consider that a solution based on gestures recognition should be an important alternative to basic buttons directions and that given widespread use, a set of conventions for basic commands could emerge that would cover most common scenarios.

References

1. Johnny Chung Lee website: http://johnnylee.net/projects/wii/
2. Schlomer, T., Poppinga, B., Henze, N., and Boll, S. - Gesture Recognition with a Wii Controller. A multi-university project of University of Oldenburg and OFFIS - Institute for Information Technology
3. Sheridan, J. G., Price, S. and Pontual-Falcao, T. - Wii Remotes as Tangible Exertion Interfaces for Exploring Action-Representation Relationships
4. Rehm, M., Bee, N. and André, E. - Wave Like an Egyptian — Accelerometer Based Gesture Recognition for Culture Specific Interactions. (2007)
5. WiiGLE Project - http://mm-werkstatt.informatik.uni-augsburg.de/project_details.php?id=46
6. WiimoteLib - http://www.wiili.org/index.php/WiimoteLib

7. WiiUse - http://www.wiiuse.net/
8. WiiUseJ - http://code.google.com/p/wiiusej/Sadsad
9. WiiRemoteJ - http://www.wiili.org/WiiremoteJ
10. Wiigee - http://www.wiigee.org/
11. GlovePIE - http://www.wiili.org/GlovePIE
12. Wiimote Project - http://www.Wiimoteproject.com/
13. WiiForPc - http://wiiforpc.com/index.php?page_id=3
14. WiiBrew - http://www.wiibrew.org/wiki/Index.php
15. WiiLi - http://www.wiili.org/index.php/Wiimote
16. Bluecove - http://www.bluecove.org/
17. Bluesoleil - http://www.bluesoleil.com/
18. Widcomm - http://www.broadcom.com/
19. WiiRemoteJ Forums - http://www.wiili.org/forum/
20. Yahoo UI Library - http://developer.yahoo.com/yui/
21. J2SE API - http://java.sun.com/j2se/1.4.2/docs/api/java/awt/Robot.html