PitA-Board PDA

QUARTERLY REPORT

Henry Kyle Bygott
Henry.Bygott@colorado.edu
Undergraduate Research Apprentice
L^{3}D

Second Quarter, 2002

SCOPE

The PitA-Board (Participate In The Action Board) is a human-computer interface with primary goals of increasing simultaneous interaction and group collaboration. It does this in a system utilizing simulations and dynamic information spaces to provide transparent access to problem solving resources in order to increase user understanding of the problem being dealt with.

INTRODUCTION

The PitA-Board is designed to be a human-computer interface that is much more transparent and easy to use than a typical keyboard and mouse (or even touch-screen) system. It is an improvement to the action space in the Envisionment and Discovery Collaboratory (EDC). The PitA-Board is, at its core, a chessboard sensor net designed by DGT Projects. This sensor has the ability to detect the type of any chess piece (plus 3 unreleased pieces – a total of 15) placed in any one of the squares (64 per board). Using these sensor nets and the tags from within the chess pieces, the PitA-Board system was designed using the Smalltalk programming language in Squeak.

Using the PitA-Board allows information to be seemingly embedded into specific pieces. In the earlier EDC design, a user had to understand interface issues in the system to use the system effectively. In order to place a house, the user has to click draw, click house, and click where to put the house. With the PitA-Board system all one has to do to is pick up the house model and place the house wherever it is needed. Although the house model actually has no ability to actively store information, it does have one very important characteristic—a resonant frequency identified by the system as #8. This allows the PitA-Board to determine the piece's identity as a house, and take appropriate action.

The PitA-Board also allows for simultaneous interaction much better than the earlier version of the EDC. The previous EDC utilized a touch screen that only allowed one user to interact at a single time; if two tried to use the screen simultaneously it would place the touched point somewhere between the two users. The PitA-Board scans one square at a time, identifying pieces in that square. This means that any number of users can be using the PitA-Board at once, even using same pieces in certain cases. This is a great stride forward in simultaneous interaction.

PROGRESS SUMMARY

The second quarter of 2002 saw great leaps in PDA interactions within the PitA-Board system. The Pebbles class in Squeak was expanded in tandem with the PDA application, allowing a greater range of interactions between the PDA and the PitA simulation. "Wand" pieces were introduced, allowing PDA users to be associated with a specific piece and interact with spaces on the PitA-Board. Pebbles was then abandoned in favor of an entirely Squeak-based communication, and Nebraska was investigated to this end. Nebraska allows multiple users connected over sockets to share a single workspace with multiple cursors. However, Nebraska was found to be inappropriate for the PitA-Board due to a lack of interaction techniques and the inability to share sub-worlds. However based on knowledge gained through experimentation with Nebraska a PitaServer class was created that communicates with PitaClient objects across socket connections. This system was expanded to include much of the functionality found in the previous Pebbles applications. With both server and client operating in Squeak, coding interactions should be more straightforward and work should proceed quicker.

PROGRESS SPECIFICS

- Expand Pebbles class in Squeak
    - Receive messages
    - Send messages
    - Track users and associated “wands”
- Expand PDA application
    - Message window
    - House dialog
- Expanded “wand” functionality
    - Empty lots
    - Stores
    - Bus stops
- Abandon Pebbles/Investigate Nebraska
- Use of Squeak independently for PDA communication
- Improved PitaController interface

Expanded Pebbles class

The previously created Pebbles class in Squeak was expanded to have much more functionality in April. Firstly, it was modified such that it could properly receive and deal with Pebbles messages. This allowed Squeak to actually add a house to the simulation when one was selected on the PDA program. After this fairly simple modification the class was modified to send messages to the PDA. Beginning as a simple text message displayed on the PDA when a piece was placed on the PitA-Board, the functionality was expanded to send pertinent information about a house to the PDA. Please refer to the following section to read more about this functionality.

In order to prepare for future multi-PDA interactions, the Squeak Pebbles class was modified to track individual users and associate each on with a “wand” piece. This wand is a piece that users can place on the PitA-Board to access pertinent information about that space on their PDA, and represents the start of a personalized reflection and information space.

Expanded PDA application

With the ability to send messages from the PitA simulation to the PDA, a small box was created to display received messages, much like the previous PitA plugin used to test Pebbles. This allows the sent message to be confirmed on the receiving end, regardless of the PDA properly handling the message.

A new dialog box was created in the PDA to begin exploring the possibilities in a personalized reflection and information space. By placing a special piece (wand) on a house in the bus simulation, the PDA is triggered to display pertinent information about the house. From this dialog only the house type can be changed (small, medium, mansion, apartment complex), although future expansions will create more functionality. This dialog box is removed from the PDA if the wand piece is removed from the PitA-Board.

Expanded “wand” functionality

The PDA/wand system allows a user to place a “wand” piece linked to their PDA on the PitA-Board, which then allows the user to make appropriate modifications to the simulation space via their PDA. This functionality previously worked properly on house pieces, but was further expanded in May to operate on many other types. If the wand was placed on an empty square (figure 1), the PDA displayed a picture representing an empty lot and an option to build on the square (with options including the four residence types and a store). The wand was also programmed to properly handle stores, although little interaction was possible—the only option was to destroy the store.

Interaction with bus stops was by far the most difficult to implement, however allowed the most understanding of MFC and revealed weaknesses in the structure of the simulation code. When the wand is placed upon a bus stop (figure 2) a window appears allowing the user to select what routes should be present on the stop. The PDA only allowed two routes to be selected simultaneously. This interaction required a modification of the simulation's code, which previously only allowed stop modification through the on-board menu system. Once this minor flaw was repaired, stop modification via the PDA worked perfectly.

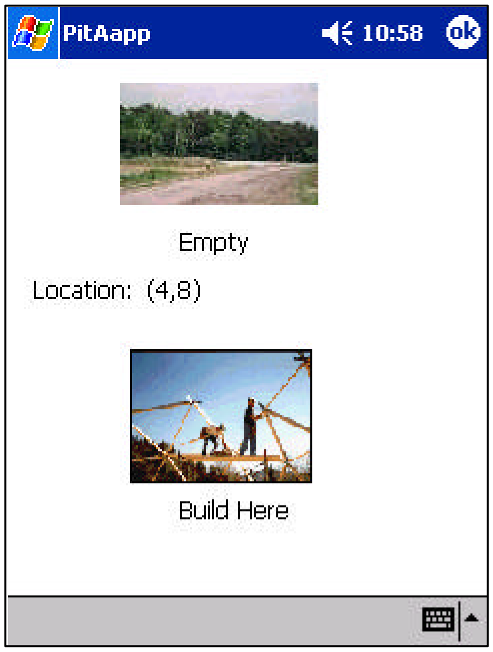
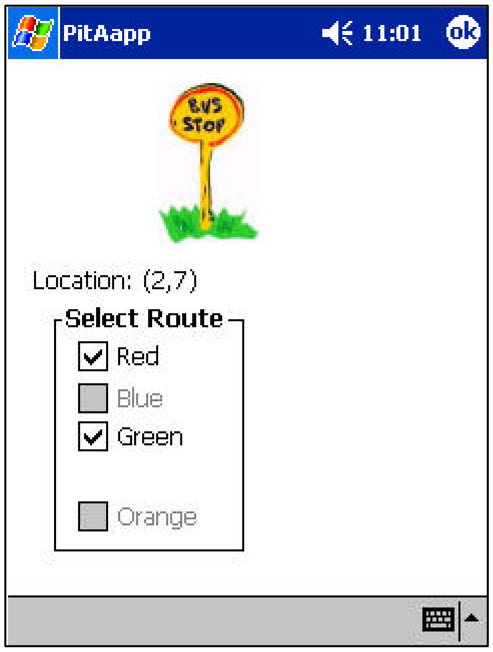


Figure 1—Empty Space



Location: (2,7)

Select Route

[✓ Red]
Blue
[✓ Green]
Orange

Figure 2—Bus stop

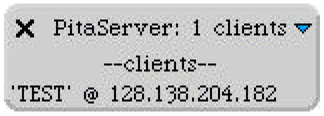
Abandon Pebbles/Investigate Nebraska

Using Squeak on the PDA in conjunction with Pebbles was never realized. Although documented protocol was followed exactly, Pebbles never recognized the PDA connection. Due to this problem, Pebbles was simply abandoned and an investigation began using Nebraska.

Nebraska is a system in Squeak to allow collaborative work in a single visual space. It creates another cursor for every participant in a session and all cursors are allowed to manipulate the shared space. Nebraska worked properly, but had little functionality that matched the PitA-Board's needs and displayed the host's entire World. Using Nebraska to serve a sub-world containing only the simulation space failed due to a difference deep within Squeak having to do with the way sub-worlds handle canvases. However through this experimentation, enough was learned to create a Squeak-based PDA interaction.

Use of Squeak independently for PDA communication

A technique similar to Nebraska's was used to create a class PitaServer, which accepts and tracks socket connections. It stores these connections as PitaSClientMorphs, a simple class that allows easy information storage on the server side. A PitaClient on a PDA can connect to this system through sockets, and has much of the same functionality as the predecessor Pebbles PDA interactions. The PitaClient can request a wand from the PitaServer, and if the request is granted, this wand may be used to interact with the PitA system. This allows the PitaServer to have control over wands and only allow one PDA to use a wand at any given time.



```
PitaServer: 1 clients
--clients--
'TEST' @ 128.138.204.182
```

Figure 3 - PitaServer

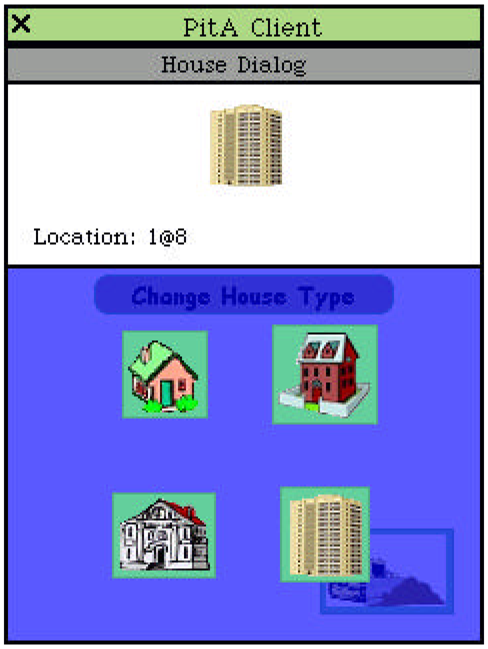
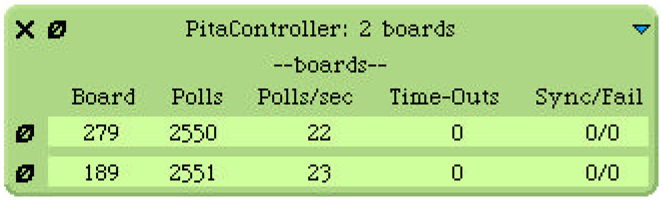


Figure 4 - PitaClient

Improved PitaController interface

Using techniques learned through investigating Nebraska and creating the PitaServer, a new PitaController interface was created which allows an expanded and compressed view. In expanded mode, the window displays all boards and their statistics, while in compressed view it displays only the number of boards. The new interface also has a close button that closes down the entire PitA system, eliminating the need for a workspace command for such a purpose. The only change to the PitaController's functionality was a modification allowing it to appropriately handle a "no" response when confirming the number of boards in the system.



PitaController: 2 boards

--boards--

Board	Polls	Polls/sec	Time-Outs	Sync/Fail
279	2550	22	0	0/0
189	2551	23	0	0/0

Figure 5 - PitaController

Future Work Plan

In the immediate future, transferring images over the socket connection will be investigated. This will allow icons and images to be downloaded to the PDA from the server, as opposed to the current technique in which images need to be stored on the PDA. This will allow the PDA to conserve memory and increase system adaptability. In this spirit, an approach will be investigated which would allow the simulation to have more control over the PDA (name dialog titles, create and place buttons, etc), rather than having all interaction hard-coded into the PDA. This would again increase the system's generality and usefulness.

The ability to transfer images from the simulation to the PDA will lead directly to the PDA gaining the ability to display the current simulation. This will allow a second type of PDA interaction to be investigated—interaction with no wand. In conjunction with Jack's work, this will allow for annotations to be made on the board from PDAs.

The concept of an "administrator" PDA will also be investigated. This PDA would attempt to put all of the functionality and needs of an administrator onto a PDA interface, allowing the administrator to become unattached from the computer running the simulation. As an example, changing phases in the WDSim could be done from the PDA. This may require an entirely different system if launching and deletion of simulations is needed from the PDA, as current interactions go through the simulation.

Throughout these tasks the PDA interaction system will be reworked to account for newfound flaws, and will be coded to remain as general and expandable as possible. In this way new programmers should be able to create custom PDA interactions without needing an intimate knowledge of the workings of PitaServer and PitaClient. This will allow other projects, such as MAPS, to utilize the system with a minimal learning curve.