Implementation and Evaluation of a Classroom Synchronous Participation System

by

Peter Beshai

B.Math., The University of Waterloo, 2010

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

The Faculty of Graduate and Postdoctoral Studies
(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA
(Vancouver)

August 2014

© Peter Beshai, 2014
Abstract

In large university classes it can be difficult to provide an engaging environment for effective student learning. Many instructors have turned to using student response systems (e.g., clickers) to mitigate this problem, with the *i>clicker* brand common in North America. However, usage of such systems is typically limited to administering multiple choice questions to the class, with real-time feedback displaying only an aggregate representation of the result distribution.

We developed an architecture for Classroom Synchronous Participation Systems (CSPS) to extend the use of such systems to support a wider variety of activities. We implemented a working system called *Rhombus CSPS*, which allows clickers to be treated as generic five-button controllers that can be used as inputs for interactive, multi-player applications.

Several game-theoretic exercises were implemented using the system and tested in a classroom setting with students enrolled in a third-year university cognitive systems course. The evaluation took place across two consecutive terms, with the researcher assisting with using the system in the first, and the instructor using it alone in the second. The results indicate both students and the instructor had a positive experience using the system. Students reported high levels of engagement and valued the activities’ effect on their learning, although there were differences between the two terms. The instructor praised the system for enabling him to teach the curriculum of activities he desired to, where he was previously limited by lack of technological support.

Displaying individual feedback to users in a CSPS can be challenging
when there is only a single shared display all students are viewing, especially if it is desirable to keep responses private. To tackle this problem, we developed a display technique for providing semi-private feedback to users based on exploiting limitations in visual perception. We ran two experiments to test the efficacy of our technique, with results indicating that our technique provides a high degree of accuracy for interpreting one’s own feedback, while limiting the ability to simultaneously interpret another’s feedback to near random chance.
Preface

The research presented in this thesis was carried out under the supervision of Dr. Kellogg S. Booth. I was the primary researcher in all work presented. Junhao Shi provided the initial code for the i>clicker driver that was used when creating the Clicker Server.

Ethics approval for the classroom evaluation and the experimental study with human participants was provided by the Behavioural Research Ethics Board at UBC under IDs H13-02138 and H14-01384 respectively.

The experimental work presented in Chapter 9 also appears in a manuscript intended for submission as a conference paper for which I am the primary author, with collaboration from Dr. Kellogg S. Booth.

The main body of the thesis will appear in a condensed form in a manuscript intended for submission as a journal paper for which I am the primary author, with collaboration from Dr. Kellogg S. Booth.
Table of Contents

Abstract ................................................. ii

Preface ..................................................... iv

Table of Contents .......................................... v

List of Tables .............................................. x

List of Figures ............................................. xi

Acknowledgements .......................................... xv

1 Introduction ............................................. 1
  1.1 Student Response Systems .............................. 4
  1.2 The \textit{i>clicker} SRS Device ....................... 7
  1.3 Thesis Overview and Contributions ................... 9

2 Game-Theoretic Exercises ............................... 12
  2.1 Coin Matching ........................................ 13
  2.2 Coordination ......................................... 14
  2.3 Stag Hunt ............................................ 14
  2.4 Prisoner’s Dilemma ................................... 15
  2.5 Ultimatum ............................................ 16
  2.6 Playing Prisoner’s Dilemma with Clickers ............ 17
  2.7 Grid Application .................................... 21
### 3 The Classroom Synchronous Participation System Architecture

- 3.1 The Participant Input Component .................................................. 24
- 3.2 The Identity Manager Component .................................................... 26
- 3.3 The Session Manager Component .................................................... 27
- 3.4 The Instructor Controller Component .............................................. 28
- 3.5 The Application Views Component ................................................... 29
- 3.6 A Simple Example of the CSPS Architecture ................................. 30
- 3.7 A More Complex Example of the CSPS Architecture ...................... 31
- 3.8 Prior Work That Informed CSPS ...................................................... 34

### 4 Rhombus Classroom Synchronous Participation System

- 4.1 Clicker Server ............................................................................... 37
- 4.2 ID Server .................................................................................... 40
- 4.3 Web Server .................................................................................. 41
  - 4.3.1 Application Managers, Viewers, and Controllers ................. 43
- 4.4 Web Framework ............................................................................ 45
  - 4.4.1 Controller ............................................................................. 45
  - 4.4.2 Viewers ................................................................................. 53
  - 4.4.3 State Applications ................................................................. 53
  - 4.4.4 User Representation .............................................................. 56

### 5 Sequence Aliaser ............................................................................. 57

### 6 Term 1 Evaluation and Results ......................................................... 61

- 6.1 Pilot Deployment ............................................................................ 61
- 6.2 Method ......................................................................................... 62
  - 6.2.1 Participants ............................................................................ 63
  - 6.2.2 Environment and Apparatus ................................................. 63
  - 6.2.3 Student Representation ......................................................... 63
  - 6.2.4 Procedure .............................................................................. 64
- 6.3 Results .......................................................................................... 66
  - 6.3.1 Student Questionnaire ........................................................... 66
  - 6.3.2 Student Short Answer Responses .......................................... 70
List of Tables

Table 2.1  The Coin Matching game pay-off matrix . . . . . . . . . . 13
Table 2.2  The Coordination game pay-off matrix . . . . . . . . . . 14
Table 2.3  The Stag Hunt pay-off matrix . . . . . . . . . . . . . . . 15
Table 2.4  The Prisoner’s Dilemma pay-off matrix . . . . . . . . . . 15
Table 2.5  The colours and animations of avatars in the Grid application 22

Table 4.1  The commands accepted by the Clicker Server . . . . . . . 38

Table 5.1  The colours and animations of avatars in the Sequence Aliaser 59

Table 9.1  The visual angles, and pixel and physical dimensions for the elements of the avatars used in the experiment. . . . . 106

Table B.1  Sequence-alias mapping used in Sequence Aliaser . . . . . 160
List of Figures

Figure 1.1 The i>clicker remote control and base station . . . . . . . 8
Figure 2.1 The giver and receiver pairing in the Ultimatum game . . 17
Figure 2.2 The check-in screen for playing Prisoner’s Dilemma with
clickers . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
Figure 2.3 Prisoner’s Dilemma being played. . . . . . . . . . . . . . 19
Figure 2.4 Prisoner’s Dilemma results screen. . . . . . . . . . . . . 20
Figure 2.5 The Grid application . . . . . . . . . . . . . . . . . . . . 23
Figure 3.1 Classroom Synchronous Participation System Architecture 25
Figure 4.1 Overall architecture of Rhombus Classroom Synchronous
Participation System . . . . . . . . . . . . . . . . . . . . . . . . 37
Figure 4.2 The relationship between the Manager, Controller, and
Viewers in the Web Server . . . . . . . . . . . . . . . . . . . . 43
Figure 4.3 The Controller interface in Rhombus . . . . . . . . . . . 46
Figure 4.4 The status bar in the Controller interface. . . . . . . . . 47
Figure 4.5 Configuration Panel for team-based Prisoner’s Dilemma
in Rhombus . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 50
Figure 4.6 Virtual web clickers in Rhombus debug mode . . . . . . 52
Figure 4.7 The Prisoner’s Dilemma basic state machine . . . . . . . 55
Figure 4.8 Avatar representation of users in Rhombus . . . . . . . . 56
Figure 5.1 An example slip of paper used to provide a sequence to a
participant . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 59
Figure 5.2 Sequence Aliaser in use ........................................... 60
Figure 6.1 Avatar representation of users in Rhombus ............... 64
Figure 6.2 Combined questionnaire results in term 1 .................. 67
Figure 6.3 Charts comparing helpfulness with learning between quizzes and Rhombus .................................................. 69
Figure 6.4 Charts comparing time value for completing quizzes and using Rhombus ....................................................... 70
Figure 7.1 Combined questionnaire results in term 2 ................. 82
Figure 7.2 Charts comparing helpfulness with learning between quizzes and Rhombus in term 2 ........................................ 84
Figure 7.3 Charts comparing time value for completing quizzes and using Rhombus in term 2 ............................................ 85
Figure 9.1 A sample user’s avatar .............................................. 100
Figure 9.2 The letters A to E as shown in the 7-segment display. . 106
Figure 9.3 A sample 5x5 grid of avatars. Here the target letter is A and the distractor, “bruce”, reveals letter E. ....................... 107
Figure 9.4 The sequence the target avatar moves through in each trial: fade in, reveal letter, hide letter, accuracy feedback, fade out. ............................................................... 108
Figure 9.5 The locations where distractors can show up in. Dark gray signifies a distractor location in the “outer ring” and light gray signifies a distractor location in the “inner ring”. The black square in the center is where the target avatar was located in all trials. ................................................. 110
Figure 9.6 The mean accuracy percentage for the target and distractor based on the duration of the target and distractor respectively. The error bars represent 95% confidence interval. 115
Figure 9.7 The sequence the target avatar moves through in each trial in Experiment 2: fade in, reveal letter, hide letter, accuracy feedback, fade out. ......................................................... 121
Figure 9.8 The mean accuracy percentage for the target and distractor based on the duration of the target and distractor respectively for Experiment 2. The error bars represent 95% confidence interval.

Figure B.1 The Question app in Rhombus.

Figure C.1 Rhombus check-in screen.
Figure C.2 Prisoner’s Dilemma play screen in round 1.
Figure C.3 Prisoner’s Dilemma results screen in round 1.
Figure C.4 Prisoner’s Dilemma play screen in round 2.
Figure C.5 Prisoner’s Dilemma results screen in round 2.
Figure C.6 Prisoner’s Dilemma play screen in round 5.
Figure C.7 Prisoner’s Dilemma total results screen.
Figure C.8 Iterated Prisoner’s Dilemma play screen in round 1.
Figure C.9 Iterated Prisoner’s Dilemma round results screen.
Figure C.10 Iterated Prisoner’s Dilemma play screen in round 2.
Figure C.11 Iterated Prisoner’s Dilemma phase 1 results.
Figure C.12 Iterated Prisoner’s Dilemma cumulative phase results.
Figure C.13 Ultimatum Game giver play screen.
Figure C.14 Ultimatum Game receiver play screen.
Figure C.15 Ultimatum Game giver results screen.
Figure C.16 Ultimatum Game receiver results screen.
Figure C.17 Ultimatum Game combined results screen.
Figure C.18 Ultimatum Game phase 1 results.
Figure C.19 Ultimatum Game cumulative phase results.
Figure C.20 Stag Hunt play screen in round 1.
Figure C.21 Stag Hunt results screen in round 1.
Figure C.22 Stag Hunt phase 1 results.
Figure C.23 Stag Hunt cumulative phase results.
Figure C.24 Coin Matching play screen in round 1.
Figure C.25 Coin Matching round results screen.
Figure C.26 Coin Matching play screen in round 2.
Acknowledgements

I thank my supervisor Dr. Kellogg S. Booth for his endless supply of interesting ideas, thoughtful discussion, and expert guidance.

I also thank Dr. Peter Danielson for his fearlessness in integrating Rhombus into his classroom.

Thanks to Dr. Ron Rensink for providing guidance on visual perception, and for inspiring the display technique used in the experiments.

Thanks to Dr. Steve Wolfman for being my second reader.

Thank you to the all members of the MUX lab, in particular to Matei Negulescu and Matt Brehmer for their sage writing and statistics advice, and to Ben Janzen and Francisco Escalona for assisting in running the no-onset experiment.

And lastly, I thank my mother for her amazing, unwavering support, my late father for inspiring me to achieve my best, and Vivian Chen for putting up with me through it all.
Chapter 1

Introduction

As classrooms and lecture halls have grown in size, instructors have sought out ways to keep lectures engaging by including in-class learning exercises in which students can participate rather than simply listen and take notes. This has given rise to a number of technological supports being introduced into the classroom, a recent one being student response systems (SRS). Our research uses the i>clicker SRS, one that is common in North American universities. An i>clicker is a simple remote with five buttons that allows students to submit answers (“votes”) to multiple choice questions during a lecture while the instructor receives real-time feedback about the distribution of responses.

Unlike some SRS devices, the i>clicker lacks a display; it has only a single LED that turns green or red to acknowledge a vote being received by the system. Consequently, students know only whether their response has been collected, but not whether it is “right” or “wrong”. If the instructor chooses, a histogram showing the distribution of responses can be displayed, enabling students to see the aggregate behaviour, but not their individual performance, which they must infer by recalling the button they pressed, interpreting the histogram, and listening to the instructor’s comments. A further restriction is that the vendor-provided software does not allow subsequent activity to be guided by previous responses. That must be done manually by the instructor. Despite these limitations, use of i>clickers and
similar SRS devices, their use in classrooms has been widespread and studies have shown they can have a positive impact on student attendance, motivation, engagement, and understanding [37, 40, 70].

Previous research by Shi [65] investigated using clickers to provide group-based feedback that allows the instructor and students to see the relative performance of each section of students in large-lecture courses rather than just the aggregate performance of the entire class. Shi also extended the interactivity of clicker exercises so students can guide interactive simulations of algorithms for common data structures, such as linked lists and trees, select areas of interest on a projected slide, and control the forward-backward progress of slide presentations, which facilitates referring to previous slides when they ask questions during lecture. All of these capabilities took advantage of the fact that every student in the classroom had a controller with five buttons. Encouraged by Shi’s earlier work, we developed an architecture for what we call Classroom Synchronous Participation Systems (CSPS), systems that allow a sophisticated level of interaction between students and the instructor with real-time feedback provided to both students and the instructor during the activities.

We implemented Rhombus CSPS (or Rhombus for short), a prototype system that realizes the new architecture. Our goal is to facilitate the creation of interactive applications that use clicker input (and input from other devices that may be present in the classroom) to drive innovative instructional activities that go well beyond the simple stimulus-response paradigms of basic SRS technology. We hope to support improved student learning by providing infrastructure that allows a wide range of pedagogically relevant activities to be conducted in the classroom that extend the capabilities of i>clickers and similar SRS devices. To accomplish this, our infrastructure is designed to support fully interactive applications where the progress of the activity depends not just on the instructor’s actions but also on those of individual students or groups of students working together. This approach enables the choice of what is displayed on a screen and what will happen next in an application to be based on the clicks gathered from students in class, both the aggregate collection of clicks and also the identities of the
specific students who made the clicks and their in-class relationships with each other.

We see the opportunity to deploy highly interactive applications using SRS technology as a way to extend their usefulness beyond limited stimulus-response activities. Answering multiple choice questions with an SRS requires students to select a single answer, typically testing knowledge or polling student opinion; our system provides a platform that will allow students opportunities to move beyond these simple use cases by providing a way to operationalize their knowledge and apply it in a relevant context. This form of teaching requires a deeper level of engagement from the student, an important factor in student success [12, 39]. Our system allows instructors to extend the benefits of thoughtful, engaging in-lecture activities to large classes, where running complex activities is currently logistically challenging.

The initial applications developed using Rhombus were game-theoretic exercises: the Prisoner’s Dilemma, the Ultimatum Game, and variants of these that can be used to demonstrate game-theoretic strategies. Wikipedia points to definitions of game theory as “the study of mathematical models of conflict and cooperation between intelligent rational decision-makers” and notes that it is also been called “interactive decision theory” in the literature [8]. These games provided a convenient testbed for our ideas because the very nature of game theory is that the outcome of actions taken by one person depend in some way on the actions taken by others. The games that we used were aligned with the curriculum for a Cognitive Systems course at our university in which we deployed the system during the first (September-December) Winter 2013 academic term. Students in the course provided us with feedback about the system through questionnaires and the instructor was interviewed at the end of the term. Based on the success of this initial trial, the same instructor agreed to use the system again during the second (January-April) term, this time without in-class support from the developers. We again surveyed the students, just once at the end of the term, and we conducted a second interview with the instructor soon thereafter.

In addition to the classroom deployment of Rhombus, we conducted two
experiments to evaluate a novel display technique for sharing semi-private feedback with users of a shared display that we developed for Rhombus. Our motivation, informed by results from our classroom evaluation and by our personal experience, for developing the new technique was to support students using clickers in classrooms who lacked confidence that their clicks had been received by the system or that what was received by the system was what the student had intended be sent. The display technique makes use of perceptual limitations of human vision to provide a strong capability for a student to interpret feedback intended for herself, while limiting the ability to interpret feedback intended for other students.

1.1 Student Response Systems

Student Response Systems (SRS) have been in use since the 1960s; early systems were typically mounted at students’ seats, with wiring connecting them to an instructor’s panel at the front of the class [49]. These were relatively expensive and were specific to the classrooms in which they were installed. When SRSs began using wireless technology, such as infrared or radio frequency, to transmit student responses to the instructor, the infrastructure costs of using SRSs greatly reduced, and consequently SRSs gained a boost in popularity at many institutions. Today, SRSs still use infrared and radio frequency, however, some are moving towards integrating web technologies and wireless networking (WiFi) to allow users to vote via other mobile devices, such as smart phones, tablets, or laptops [1, 2, 9, 10].

Despite the many years of SRS usage and changes in the technology, there has been very little difference in the way they have been used within classrooms to support learning.

Just as in the 1960s, today instructors still commonly use SRSs solely to pose multiple choice questions to their class and have students vote on which answer they think is correct [49]. The one notable difference is that instructors can now immediately display a histogram or other visualization that shows the aggregate distribution of responses to a question by the students in class. This is a useful way of allowing students to compare them-
selves with the rest of the class, but is still essentially a stimulus-response paradigm. When SRSs are used in such a way, research has shown that they have no significant correlation to subsequent student academic achievement [49].

Nevertheless, even limiting SRS usage to only administering multiple-choice questions in class, SRS technology has been shown to have many benefits for students, and are consistently perceived with positive attitudes by both students and teachers [24, 29, 36, 45, 49, 68]. They have been shown to have positive impacts on student attendance [23, 24, 31, 38, 62], attentiveness [17, 23, 24, 27, 28, 32, 44, 46, 55, 67, 69], and engagement [17, 45, 62, 68]. Furthermore, students appreciate the anonymity provided by SRSs, because it reduces the intimidation that students sometimes feel that is associated with participating in class [15, 29].

It is therefore puzzling that multiple studies have found that when SRS technology is used in a stimulus-response paradigm, it provides no significant benefit in terms of the final grades students receive [49]. However, in Kay and LeSage’s extensive literature review, they suggest a strong argument can be made for modern SRS use improving learning performance [51], citing numerous studies that demonstrate that classes with SRS use outperform traditional lectures [22, 31, 34, 50, 52, 61, 62, 64, 69].

Beyond student performance, there are several studies that point to other distinct learning benefits gained from using SRSs. There is increased interaction in the classroom when SRSs are used effectively, increasing valuable peer-to-peer discussion [16, 19, 28, 47, 59] and enabling active learning practices in large classrooms [32, 53, 69, 71]. The increased class discussion and two-way interaction between students and the instructor allows for contingent teaching practices to take place, where the instructor can modify the lesson on-the-fly to clear up material students are having trouble grasping [19, 24, 28, 32, 38, 46, 52, 71]. Furthermore, students report that they learn more when using SRS technology [32, 38, 41, 59, 61, 62, 67, 69, 71, 76] because it forces them to think more [28, 38] and to discover and correct their misconceptions [27].

It seems likely, however, that simply using SRS systems will not auto-
matically produce the benefits listed above, as suggested by Kristine [54]. Indeed, one would not expect immediate learning benefits simply from pushing buttons on a remote control. What seems to bring the biggest benefit out of using SRSs is leveraging peer-to-peer communication and other active learning principles to ensure that students are engaged with the material. The discussion and interaction that happens around the SRS usage is where perhaps the primary opportunity to affect learning takes place.

While by far the majority of SRS usage involves posing a multiple-choice question and receiving answers from students, some instructors have explored using SRS technology in different ways, or with slight twists to further increase student engagement.

A good example of how clickers can be used to “gamify” classroom teaching is introducing competition into in-class activities. Bruff reported on McCoy’s experience when she awarded bonus marks to the student who was first to answer a quiz question [21]. While this is only a small step from typical clicker quiz usage, it opens up the possibility of using clickers competitively. Unfortunately, the standard i>clicker software does not support doing this in real-time; there is no immediate feedback to the winning student unless the instructor runs additional software to immediately process the clicker data and display the results.

There are some examples in the literature of clickers being used for classroom activities that were not quiz-based. Salemi made use of clickers to auction off a T-shirt, helping to give students practical experience with the economics of auctions [63]. Bostian and Holt had students estimate the number of marshmallows in a jar by entering their estimates using clickers [18], reasoning that it was much faster than using pen and paper. They developed a system called Veconlab Clickers that was used to display aggregate results during class, and allowed students to sign on afterwards from their own computers to find out their individual outcomes. Both of these examples used more sophisticated remotes than i>clickers because students had to enter numeric information. In neither situation did students receive individual real-time feedback.

There are of course some challenges involved in even basic usage of SRS
technology in today’s classrooms. At times, students forget to bring them to class, resulting in those students being unable to participate during question-and-answer periods [24]. Some times, the devices simply do not work, so the clicks students are trying to send are not received by the system, resulting in unnecessary stress for students, especially if they are being evaluated based on their clicks [31, 41, 67]. Instructors may find they are unable to cover as much material as they would like, because more class time is devoted to discussion and impromptu explanations of misunderstood material compared to traditional lecture formats. Furthermore, good questions are required to get the most out of SRS usage, which can take time to develop and for which there are few repositories available for instructors to share. From the student point of view, the discussions of different perspectives or solutions may lead to confusion as to which one is correct. Some students may simply have difficulty adjusting to the new style of learning in which they are responsible for active in-lecture participation.

Our research does not deeply examine the many pedagogical questions that are obvious targets for on-going study. We instead focus on how to extend SRS usage beyond the simple stimulus-response paradigms that have been the dominant mode within classrooms. We do this in the context of the specific SRS provided by \textit{i>clicker} technology.

\subsection{The \textit{i>clicker} SRS Device}

The \textit{i>clicker} technology is a fairly basic SRS. It is comprised of a base station connected to a computer that communicates with the student’s remote controls, and vendor-provided software that manages the process and records the results for subsequent analysis. The two hardware components are shown in Figure 1.1. The remote control has six buttons, and is primarily used by students. Five of the buttons are used to provide responses (these are labelled \textbf{A} through \textbf{E}), which are transmitted over radio frequency, and the sixth is used to turn the device on and off and to initiate synchronization with the hardware base station. There are three lights located above the buttons that are the only method of feedback the remote control has.
The top light indicates whether the remote is on or off (labelled “POWER”), by emitting a blue light when the remote is on, and not emitting light when turned off. The middle light (labelled “LOW BATTERY”) is only lit when the batteries are low, blinking red if that is the case. The bottom light indicates whether the button response sent was successfully received by the base station (labeled “VOTE STATUS”). If one of the response buttons (A through E) is pressed, the vote status light will turn green for approximately half a second to indicate a successful transmission to the base station, and otherwise will turn red and flash four times to indicate a failure. When no buttons have been pressed, the vote status light is not lit.

The base station attaches to the instructor’s computer via a USB cable, and is configured to use one of 16 pre-defined pairs of frequencies through which it broadcasts to remote controls and listens for their responses. The pair of frequencies that are used are determined by configuration parameters
set via software on the instructor’s computer. Pairs of frequencies are coded to combinations of two letters between A and D (e.g., AA, BC, DB). In order for the clickers to communicate with the base station, they need to set their transmission frequency by pressing and holding the ON/OFF button on the remote followed by pressing the two buttons that code the frequency. The base station itself can be in one of two states: accepting votes, or not accepting votes. When accepting votes, all responses sent along its configured frequency by clickers will be received. When not accepting votes, only a specially configured instructor clicker will have its responses sent through the system; all other clicks are rejected, resulting in red vote status lights flashing on the student clickers. For more details on how the i>clicker hardware functions, refer to Shi’s master’s thesis [65].

1.3 Thesis Overview and Contributions

Several game-theoretic exercises, or games, were implemented in our system to aid students in understanding how the games work and provide experience in developing strategies for them firsthand. A general description of each of these games, as well as an example of how one of them is played using our system, is given in Chapter 2.

The chapters that follow cover the five contributions of the research reported in this thesis.

- An architecture for Classroom Synchronous Participation Systems is described in Chapter 3. This architecture is designed to support complex interaction between students and instructors, giving real-time feedback to both parties while being used in a classroom environment. It has been designed with institution-level support in mind, while also being able to function on a single user’s computer.

- The implementation of a fully-functional prototype CSPS, Rhombus, that is based on the architecture, is presented along with various lessons learned along the way in Chapter 4. Rhombus includes several components (Clicker Server, ID Server, Web Server, and Web Framework)
combined in ways that allow versatility for input device, anonymity, and synchronization across multiple displays. The web technologies are used in a novel way by leveraging the power of the combined HTTP and WebSocket server to allow multiple browser windows to synchronously communicate with each other.

- An efficient and simple method for concurrently registering multiple users in an SRS is described and demonstrated by the Sequence Aliaser in Chapter 5. Users are provided a sequence of buttons to press on their clicker along with an associated alias. Upon pressing the buttons in sequence, their alias appears on screen and can then be controlled by their clicker to assure them that they have correctly linked their clicker to the system. This system was successfully tested with 40 students as part of the first term classroom evaluation of Rhombus.

- The results of using Rhombus in a university classroom environment across two academic terms are presented. Chapter 6 describes the evaluation method used and the results from the first term field trial and Chapter 7 does the same for the second evaluation. Chapter 8 provides an interpretation and discussion that compares the results across the two terms. Student feedback and instructor feedback was largely positive, encouraging further use of the system in broader contexts. Notably, the effectiveness of the system appears to depend on the willingness of students to engage in discussion around the results shown while using the system, something that is impacted by the precedent set by course instructors.

- Lastly, a novel method of displaying semi-private feedback to users of a shared display was designed and tested in two experiments. The technique and the experiments are discussed in Chapter 9. Results from the two experiments indicate that the display technique allows users to interpret their own feedback with high accuracy, while simultaneously having difficulty interpreting another user’s feedback.

Chapter 10 provides a summary of the thesis, concluding remarks,
some ideas for future work that would extend the scope of the research reported in the thesis.
Chapter 2

Game-Theoretic Exercises

Game theory is the mathematical analysis of decision making [57]. We illustrate the complexity of classroom activity supported by Rhombus by explaining six game-theoretic exercises that were developed to assist students in understanding decision strategies for those games. These games were chosen because they are part of the curriculum for a third-year course in the Cognitive Systems program at our university. In all of the games that follow, the goal is to attain the highest score. The main pedagogical benefits of the games come from having students actually experience playing the games firsthand, as opposed to simply reading about them. Playing them firsthand gives students a direct experience of the divergence of actual behaviour from theoretical outcomes and provokes thoughtful discussion. For details on how the game worked in Rhombus, see Appendix C.

Each of the games discussed, besides the N-person Prisoner's Dilemma variant, require that students be secretly paired with one another and that they receive feedback from their actions after each round of play to help inform the next. Completing these two actions is infeasible with the vendor-provided i>clicker software, as there is no method for providing individual feedback and no concept of pairing student responses.
2.1 Coin Matching

The Coin Matching game (also known as Matching Pennies) [4] is a simple exercise to familiarize students with playing games without requiring much background knowledge. In this game, players are randomly partnered together for each round of play in which they must choose between playing heads or tails. In each pair of players, one player is the “matcher” and receives points when both players make the same choice, and the other is the “mismatcher”, receiving points when the choices differ, as indicated by the payoff matrix in Table 2.1.

<table>
<thead>
<tr>
<th></th>
<th>Heads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heads</td>
<td>1, 0</td>
</tr>
<tr>
<td>Tails</td>
<td>0, 1</td>
</tr>
</tbody>
</table>

Table 2.1: The Coin Matching game pay-off matrix. In each cell, the score for a matcher is given by the first value, and the score for a mismatcher by the second value.

This game is typically formulated as a zero-sum game, in which when one partner gains a point, the other player loses one, but in our formulation, informed by the instructor we worked with, no points are ever taken away. Still, this does not affect the overall strategy of the game: every combination of choices leaves one partner who could increase their score by changing their choice. Hence, there is no pure strategy Nash equilibrium, but there is however a mixed Nash equilibrium achieved by playing heads or tails with equal probability, giving each player an expected pay-off of 0.5.

In this game, the players have no incentive to cooperate with each other, since only one can benefit in any given round, and there is no obvious benefit to knowing what a player’s previous choice was since the equilibrium strategy involves randomly selecting a choice during each round of play.

Due to the random nature of play this game evokes, it may be found to be less engaging to students than games with more complex strategy if played for an extended period of time. Alternative games may provide a deeper level of interest to the students by allowing them to construct more complex strategies to play with. Explanations of these games follow. Still, the Coin
Matching game remains a good staple introductory exercise to playing games with clickers.

2.2 Coordination

The Coordination game [3] is very similar to the Coin Matching game, but instead of using heads and tails, the labels A and B are used, and instead of splitting the partners up as matchers and mismatches, both partners receive points if their choices are different (i.e., one partner chooses A and the other chooses B), and neither partner receives points if their choices are the same. The pay-off matrix for this game is given by Table 2.2. In our formulation, players are partnered up secretly with the same person for multiple rounds of play. A common example of a situation the coordination game models is the agreement to drive on the right side of the road to prevent collisions.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0, 0</td>
<td>1, 1</td>
</tr>
<tr>
<td>B</td>
<td>1, 1</td>
<td>0, 0</td>
</tr>
</tbody>
</table>

Table 2.2: The Coordination game pay-off matrix. Both partners only receive points if they coordinate their choices to be different.

In this game, there are two pure strategy Nash equilibria: in a partnership consisting of players X and Y, player X chooses A and player Y chooses B, or player X chooses B and player Y chooses A. In both of these cases, if either player changes their choice, they will reduce their score, and as such, they are Nash equilibria. This promotes cooperation amongst the players of the game, since they will both benefit by agreeing to play in a coordinated way. This is easy to manage if players can communicate with one another, but is more challenging if they cannot.

2.3 Stag Hunt

Stag Hunt [6] is a different type of coordination game, with varying pay-offs depending on the choices both partners make (see Table 2.3). Players are paired secretly and each must choose to either hunt a stag or hunt a hare.
If both hunt a stag, they each earn 3 points; if both hunt a hare, they each earn 1 point; if their choices differ, the stag hunter receives 0 points and the hare hunter receives 2 points.

<table>
<thead>
<tr>
<th></th>
<th>Stag</th>
<th>Hare</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stag</td>
<td>3, 3</td>
<td>0, 2</td>
</tr>
<tr>
<td>Hare</td>
<td>2, 0</td>
<td>1, 1</td>
</tr>
</tbody>
</table>

Table 2.3: The Stag Hunt pay-off matrix.

This game has two pure strategy Nash equilibria: both players hunt a stag, or both players hunt a hare. In these situations, neither player can improve their pay-off by changing their choice. However the two equilibria have different pay-offs: hunting a stag offers a higher pay-off, but has a higher risk, since it requires both players to decide to hunt a stag to get any points at all, while hunting a hare is safer, guaranteeing points regardless of what your partner does, but nets a lower pay-off.

2.4 Prisoner’s Dilemma

In the Prisoner’s Dilemma [5], players are secretly paired and each must choose to cooperate or defect. If both cooperate, they each earn 3 points; if both defect, they each earn 1 point; if one cooperates and the other defects, the cooperator earns 0 points and the defector earns 5 points (see Table 2.4).

<table>
<thead>
<tr>
<th></th>
<th>Cooperate</th>
<th>Defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cooperate</td>
<td>3, 3</td>
<td>0, 5</td>
</tr>
<tr>
<td>Defect</td>
<td>5, 0</td>
<td>1, 1</td>
</tr>
</tbody>
</table>

Table 2.4: The Prisoner’s Dilemma pay-off matrix.

This game has a single Nash equilibrium that is reached when both players defect. In this case, they each receive a low pay-off of 1 point, but cannot improve their scores by switching to cooperating unless they both do it together, in which case they improve to 3 points each. However, in the state when both players have cooperated, there is incentive to switch to defecting, which boosts a player’s score to 5 points, while reducing their partner’s score.
to 0. This leaves the only equilibrium as when both players defect, despite it providing the lowest total score.

Three variants of Prisoner's Dilemma are of interest. The most basic plays a single round with an anonymous partner; each subsequent round is played with a different partner. Different results are obtained when participants are partnered with the same person for multiple consecutive rounds, known as the Iterated Prisoner's Dilemma. The third variant is an N-Person game, where individuals are not partnered, but instead the group plays as a whole, with all cooperators receiving the same score and all defectors receiving the same score. Total social payoff is highest if all participants cooperate, but individual payoff is highest by defecting if everyone else cooperates.

N-Person Prisoner’s Dilemma is more complex than the individual variants, and thus is not ideal as an introduction to Prisoner’s Dilemma. However, it does not require providing individual feedback to participants, and so can be played on systems lacking this capability. For instance, one can make use of the histogram display from the vendor-provided i>clicker software to display the proportion of cooperators and defectors, and manually calculate the scores from them.

2.5 Ultimatum

In the Ultimatum game [7], players are secretly paired, with one the “giver” and the other the “receiver”. There is a sum of points that must be divided amongst the two players, with the giver deciding what fraction to offer the receiver. If the receiver accepts the offer, both players receive the designated fractions of the sum as points; if the receiver rejects the offer, both players receive no points.

This game functions in two stages: first the players acting as givers decide what amount they will offer to their partner, then receivers are presented with the offer their partner made them and must decide whether or not to accept it. In order to scale this game so that all students in the class can experience both roles, we used a directed cyclical partnering algorithm. In this case, each student in the class had a forward partner with whom they
Figure 2.1: The giver and receiver pairing in the Ultimatum game. Here player P1 acts as a giver to P2, who acts as a giver to P3, who acts as a giver to P1. This asymmetric pairing allows reactions made to offers to act independently of offers given since they correspond to different partners.

would act in the giver role to, and a backward partner with whom they would act as a receiver to, as indicated in Figure 2.1. This format allows all students to act as givers at the same time, and then all to act as receivers at the same time as well, without enforcing a symmetrical partnership.

This game is the most complex that we worked with. The five-button limitation of i>clickers means that our implementation provides at most five choices for a giver to offer a receiver.

2.6 Playing Prisoner’s Dilemma with Clickers

In this section, we will cover how to play the simplest form of the Prisoner’s Dilemma with clickers. This version of the game demonstrates the mechanisms used in the other games discussed without needing as much complexity because it does not involve multiple stages or teams.

To begin with, a check-in screen is presented to the class. As students press buttons on their clickers, an anonymized representation of themselves known as an avatar shows up on screen with a large checkmark superimposed, as shown in Figure 2.2. Their appearance on screen confirms that their clicker is working with the system and they are ready to play. The avatars themselves are inserted in lexicographic order as they appear on screen, causing a bit of shuffling around as users check in. The position they end up in before moving to the play state is the same position their avatar will be in for the rest of the game. Prior to checking in, users must be informed which alias is registered to their clicker. How this is done is left up
to the instructor, with one method described in Chapter 5.

Once the instructor is sure the students have all checked-in, he initiates the play state of the game. In doing so, the system secretly partners each student that was checked in with another student in the system. In the event of having an odd number of students checked in, a bot will be added to the
Figure 2.3: Prisoner’s Dilemma being played. Students press C to cooperate and D to defect. The scores that will be assigned to each user depending on the outcome of their match-up are indicated in the Pay-off Matrix. Those who have already pressed buttons on their clicker to play are dimmed and show the word “Played” on their avatar.
Figure 2.4: The Prisoner’s Dilemma results screen. The action taken by a given student represented by the hue of their avatar, with blue representing those who cooperated and orange those who defected. Student score is shown numerically and encoded in the lightness of the avatar. The two letters on the avatar represent the student’s action followed by their partner’s action. A histogram shows the average scores of cooperators (1.3) and defectors (2.3), as well as the overall average (2.0).
system to ensure everyone has a partner. The play state is displayed on screen by having each student’s avatar (and the bot’s if applicable) shown with instructions placed beneath them, as shown in Figure 2.3. Students have the option of pressing C on their clicker to cooperate or pressing D to defect, all other buttons are ignored. Upon pressing C or D, the corresponding avatar on screen will darken and display the word “Played”. Subsequent presses of either button will register the new action and provide feedback to students that the click was received having the word “Played” on their avatar flash once.

When all players have been marked as having played or the instructor has decided enough time has passed, the instructor can progress to the results state. If a student has yet to play and the instructor moves on, a default action of cooperating is assigned. In the results state, shown in Figure 2.4, students see how they fared in the game, as well as how the class behaved overall.

2.7 Grid Application

Prior to initial trials of playing games with clickers, we recommend using the Grid application to familiarize students with how the system works. In Grid, participants are represented as avatars on screen and placed in a grid, ordered lexicographically by alias. These avatars appear on screen upon the system receiving the first button press from a clicker, causing the grid to expand as more users participate. Pressing buttons on their clickers causes the user’s avatar to change colour, display the letter of the button pressed, and to animate as described in Table 2.5 and shown in Figure 2.5. The intent is that users can get used to seeing their avatar and gain confidence that indeed their clicker is connected to it since the feedback time between pressing a button on their clicker and seeing their avatar respond is short enough (less than 160ms) to give a sense of causation between the two actions [56].

The architecture we created to inform the development of systems that can support the applications described here follows in the next chapter.
<table>
<thead>
<tr>
<th>Button</th>
<th>Colour</th>
<th>Animation</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Green</td>
<td><em>Pulse.</em> The avatar cycles between getting larger and smaller.</td>
</tr>
<tr>
<td>B</td>
<td>Blue</td>
<td><em>Bounce.</em> The avatar moves up and down in a bouncing motion.</td>
</tr>
<tr>
<td>C</td>
<td>Purple</td>
<td><em>Shake.</em> The avatar moves left and right in a shaking motion.</td>
</tr>
<tr>
<td>D</td>
<td>Yellow</td>
<td><em>Swing.</em> The avatar swings left and right as if there was a pivot</td>
</tr>
<tr>
<td>E</td>
<td>Red-orange</td>
<td><em>Wobble.</em> The avatar moves left to right following a slightly circular path</td>
</tr>
</tbody>
</table>

**Table 2.5:** The colours and animations of avatars in the Grid application. The animations were taken from Eden’s Animate.css [30] and were selected for their distinguishability.
Figure 2.5: The Grid application. In this application, user avatars change colour and animate in response to the buttons pressed on their respective clickers. The letter of which button was pressed is also displayed. The application is typically used to re-acquaint users with the mechanics of the system.
Chapter 3

The Classroom Synchronous Participation System Architecture

We have developed *Classroom Synchronous Participation System* (CSPS), an architecture for supporting interactive activities in classrooms. This generic architecture can be implemented in various ways. One such implementation, *Rhombus*, is discussed in Chapter 4. The architecture, summarized in Figure 3.1, has five components: participant input, an identity manager, a session manager, the instructor’s controller, and views of the application.

3.1 The Participant Input Component

The participant input component of the CSPS architecture deals with the low-level details of all incoming activity from students in the class. This can be anything from clicks entered using SRS clickers, to SMS text messages, tweets, or even interactions happening remotely on a student’s laptop through a web-based interface or a custom application. The data from the various sources is streamed in a uniform manner to the Session Manager, optionally after being interpreted and transformed by the Identity Manager to conform to the security and privacy policies that are in place.
The Classroom Synchronous Participation System (CSPS) architecture isolates all of the device-dependent and communications details such as device drivers and networking or wireless protocols from other components so those components can be implemented without regard to the specific input devices that are being used. This means that pedagogical issues can be dealt with without regard to the specifics of the SRS hardware being used or the interaction techniques that students utilize. While this separation of concerns could potentially cause usability problems, the underlying assumption is that a particular SRS (such as the i>clicker SRS described in Chapter 1) will largely determine the low-level interaction techniques leaving only the high-level logic of the in-class activity to be defined by applications that are implemented within the architecture.

The primary task of the Participant Input component is to encapsulate all of the low-level interactions with the devices that students use and present...
those interactions as a stream of virtualized “clicks” in a uniform way to the downstream components in the CSPS architecture. Virtual clicks may retain attributes that identify the particular hardware or software that originated them, but usually this will be ignored by later components in the system other than perhaps being collected for statistical purposes to assess usage patterns. In principle there would be no semantic differences between virtual clicks that come from an \textit{i>clicker}, an SMS text message, or a web-based interface running on a student’s laptop computer. We describe these inputs as clicks simply due to the posturing of using \textit{i>clickers} as the primary input, however the architecture supports any form of input being sent across the system, allowing implementations to enforce their own local restrictions as necessary.

3.2 The Identity Manager Component

The Identity Manager is a system that intercepts virtual clicks sent by the Participant Input component enroute to the Session Manager component and transforms the identity of the participant into something more usable by the Session Manager. For example, the Identity Manager might intercept clicks from an \textit{i>clicker} base station that contain clicker IDs and transform the clicker IDs into student names or student ID numbers, so that when they are displayed in the application they are more readable or so they can be used to interface with marking software that expects student ID numbers, not clicker IDs. The Identity Manager should be able to accept multiple streams of input from different instances of the Participant Input component and combine them into a single output stream that is sent to the Session Manager, enabling it to serve as an aggregator even in cases where identity mapping is not needed. In simple systems where there is only a single Participant Input component, and thus no need for aggregation, and when there is no need for identity mapping, the Identity Manager can be excluded. In these cases, the Participant Input component connects directly to the Session Manager.

This component of the CSPS architecture is a mechanism for a university
registrar to ensure that students receive consistent identifiers across multiple courses throughout their academic careers and to isolate the administrative aspects of this from all of the other components in the architecture. Having a single location where this mapping is done reduces the need for students to reconfigure their identities for each use of a CSPS while avoiding the need for a university to adopt a single approach to either the low-level input mechanisms or the high-level pedagogical approaches, which are the concern of other components in the architecture.

A second benefit of having the Identity Manager centrally located is that university information technology staff can be assured of the security and privacy capabilities of the system by inspecting only one trusted component. This could be especially important if in-class activities are not just pedagogically motivated, but are also part of research activities that require approval by behavioural research ethics boards. A suitably designed Identity Manager can allow researchers to see only anonymized identifiers for students who participate in research studies, without having any access at all to the actual identities of students. This would relieve researchers of the need to repeatedly convince a university ethics board that their software can be trusted.

3.3 The Session Manager Component

The Session Manager component oversees the coordination of participant input and instructor controls for specialized applications that support in-class activities. It receives the input from participants, possibly transformed via the Identity Manager, and passes it along to the relevant application controllers and viewers. A Session Manager could be designed to support only a single session, but the architecture is intended to scale up to simultaneously support multiple concurrent sessions across an entire university in an enterprise system.

In the enterprise setting, participant input and instructor controls would come bundled with information about which session they belong to and would be routed accordingly to the applications and viewers associated with
the session. There are multiple levels of granularity that could define a session, which is left up to individual implementations to decide. For example, a session could be an in-class lecture in a specific classroom at a specific time for a specific course, or it could be a distributed lecture across multiple locations, in which case only a course and a time would be specified. In all cases, the session manager coordinates the student and instructor input so that the applications that implement an in-class activity only receive data relevant to their session.

3.4 The Instructor Controller Component

The Instructor Controller component is a special type of input to the system, different from the typical participant input. It is where the instructor controls the applications active for a given session. Usually this will be done through a web-based interface or a dedicated application in order to provide rich functionality beyond the rather limited input capability provided by SRS technology. Using the Instructor Controller, an instructor should be able to select which application(s) will be loaded, set the configuration parameters for the application, and control the application and the various Application Views during the in-class activity. The Instructor Controller accomplishes this by sending commands through the Session Manager that are then routed to the appropriate applications and viewers. This automatically updates the Application Views appropriately without the Instructor Controller needing to deal with the low-level coordination of the various applications and viewers.

Each in-class activity is implemented by a mini-application that is specific to the activity, with the idea being that the mini-apps run inside of the overall active system (i.e., they do not require switching to new software to activate). The architecture does not fully define where the high-level logical flow of an application is located. It is possible to store application logic and state in the Instructor Controller component. In this case, the participant inputs will flow through from the Session Manager to the Instructor Controller, so the controller can determine what to do with them and then update the views
accordingly by communicating back via the Session Manager. This method reduces the complexity of the Session Manager by off-loading some work into the Instructor Controller. This choice is recommended when there is a single Session Manager and multiple Instructor Controllers (e.g., at the enterprise Session Manager level).

The other option is to store application logic and state in the Session Manager and have the Instructor Controller only provide input to the application. This choice is recommended when the system has most of its components bundled, for example on the instructor’s laptop.

We advise against having the application logic placed directly in the views because maintaining multiple instances of application state raises the possibility of inconsistent behaviour being displayed when randomness is involved. This can lead to duplicate views not exhibiting identical behaviour.

3.5 The Application Views Component

The Application Views component comprises the basic display modes of the system. Application views receive updates on what to display for an application via the Session Manager. They should contain as little application logic as possible to ensure consistency across multiple instances of the same or different viewers (as explained at the end of the previous section). Typically there may be only a single view available that is projected onto the shared display in a classroom, but in other use cases it may be that each student or each group of students has access to a custom display tailored to their particular role or perhaps their individual needs.

One example of an application view is a display of how voting is progressing for a basic SRS multiple-choice question. The display on the classroom projector would indicate the percentage of students who have voted and the time remaining to cast votes. Optionally, the display might also show a histogram of how many votes have been cast for each answer and (once voting is done) the correct answer(s) for the question. This is the extent of the functionality provided by the vendor-provided software for the i>clicker SRS.
A more sophisticated application view might be a web-based interface that provides feedback tailored to a specific student about his/her performance, such as the student’s cumulative “score” during an interactive question-and-answer session, the sequence of answers that the student provides to each question, and an option to see an explanation of why the correct answer is the correct answer. Each instance of this application view would be available only to the specific student for which it was created, with access control determined by the Session Manager perhaps using information provided by the Identity Manager.

3.6 A Simple Example of the CSPS Architecture

An implementation that provides the basic functionality of the i>clicker SRS would have a simple Participant Input component that had a software driver for the USB-connected base station where clicks were collected and passed on to the Identity Manager where the hexadecimal clicker IDs are converted to student ID numbers using a simple .csv file that has one column of clicker IDs and a second column that has the corresponding student IDs. The Session Manager would turn voting on and off and would save the votes in a second .csv file for subsequent processing by other software that might score the votes and then upload the results to the university’s enterprise learning management system. The Instructor Controller would be limited to starting and stopping votes, enabling and disabling the available application views, and controlling the forward and backward progress of the slide presentation. There might be three Application Views: a status display indicating whether voting is on or off and the time remaining to vote, a histogram of the votes cast so far, and information about the correctness of each of the possible answers. The application logic could allow the instructor to dynamically specify whether the histogram is displayed during voting, and to make a mode selection for whether the the correctness display is automatically revealed as soon as voting ends.

In this case the application logic and state would probably be within the Session Manager, which would be the main component of the system. Each
Application View would be isolated to a custom output-only GUI widget implemented using a standard view-controller paradigm, and the Instructor Controller would be a GUI widget resembling a standard dialogue box where options could be selected using on/off buttons, menu selection, or text entry. The instructor would interact with the Controller using a mouse and keyboard on her laptop. Auxiliary control via the instructor’s \textit{i>clicker} would be achieved by the Session Manager recognizing the unique ID hardware of the instructor’s (set using the GUI dialogue box) and forwarding only those clicks to the Instructor Controller which would then simulate the corresponding GUI actions to turn voting on or off, show the histogram, or advance to the next or previous slide by requesting that the Session Manager send the appropriate right-arrow or left-arrow keyboard event to the external software (such as PowerPoint) that is providing the slide presentation.

Shi \cite{65} provides an overview of how software similar to this example was implemented in a platform-independent manner for the \textit{i>clicker} hardware to achieve functionality equivalent to the vendor-provided platform-dependent software. His implementation does not fully conform to the CSPS architecture, but could be readily adapted to it by refactoring some of the code. It does already isolate the details of the \textit{i>clicker} hardware into a low-level driver module, but the other components are more integrated with each other and thus lack the full degree of modularity that the CSPS architecture envisions.

### 3.7 A More Complex Example of the CSPS Architecture

In this example, we consider a more complex implementation of the CSPS architecture where there are multiple student inputs and an enterprise-level Session Manager component. The example involves a system that can load various applications via the Instructor Controller, but focuses on a single application that gives students practice inserting into a binary search tree (BST), as inspired by Shi \cite{65}. In this application, students are split into groups based on lab section and have to work together to control the insertion
process. Each lab section has their own BST to work on, visualized on screen. Given a node to insert into the tree, their inputs allow them to choose whether to go to the root of the tree, go to the left child, go to the right child, insert the node as the new left child, or to insert the node as the new right child. Once enough students in the lab section select the same choice, passing a predefined threshold, that choice is made.

This implementation would use two Participant Input components: one for clickers and one for smartphone input. The clicker component would, similar to the previous section, use a software driver to receive input from the clicker base station. Additionally, it would encode with the clicks a course identifier, which would be passed to the Identity Manager component with the click data. The smartphone component would allow students to use a mobile app to interact with the system. In the app, they would register an authenticated account and select which courses they would be using the system in. When using the app, students would select the course they are currently in and then press buttons on screen representing choices to be made in the active application (the equivalent of clicks on a clicker). When a button is pressed on screen, the app would send the choice, the user ID, and the course ID to the Identity Manager component.

The Identity Manager component would convert the clicker and app user IDs to student IDs via database lookup in a table consisting of clicker ID, app user ID, and student ID columns. Using the student ID and the course identifier, the data would be augmented with which lab sections of the course the student was enrolled in via lookup in a different table in the database consisting of student ID, course ID, and lab section columns. The data passed on to the Session Manager component would include all three parts (the student ID, the course ID, and the lab sections) for the given student.

The Session Manager component would maintain a list of active sessions in memory that are identified by course ID. Each session would have a connection to a single Instructor Controller component. When data arrives from the Identity Manager component, the course IDs are examined and are used to route the data to the Instructor Controller component in the session with the corresponding course ID.
The Instructor Controller component would require the instructor to specify the course ID upon initializing. Once specified, the instructor would be able to select an application to load in the system. Beyond loading the application, the Instructor Controller component would allow the instructor to activate different controls depending on which application was loaded, a common one being opening Application Views that are associated with the session. In the BST example, the instructor would be able to enable or disable student input, navigate between different BST scenarios, and to reset the active scenario to its initial state. The participant input would flow into the Instructor Controller component from the Session Manager component, and would be used to update the state of the active application. In the BST example, it would threshold the inputs, updating the state of each lab section’s BST when enough students had entered the same choice.

The Application Views would depend on the active application in the session. In the BST example, there would be two types: one displaying a single shared scoreboard showing the time elapsed and number of errors made by each lab section, and one that indicated a current lab section’s BST state. There would be one of each of the latter type for each section in the class. These views would receive the data they used to draw from the Instructor Controller via the Session Manager.

In this implementation, the Instructor Controller contains the application logic, and the Session Manager is simply used to route data to the various components. The Instructor Controller would be a standard GUI interface with various widgets, such as buttons and text-input fields, for loading and configuring applications. It would primarily be interacted with by keyboard and mouse, but similar to the previous example, would also be able to be controlled by the instructor’s clicker. The Session Manager component would be a server hosted somewhere on the university’s network that was accessible for the Instructor Controller and Application Views to connect to. The Application Views would be a basic GUI interface that allowed connecting to a session and when connected, provided output-only displays of application state. By indirectly connecting the Application Views with the Instructor Controller via the Session Manager, students in the class would also be able
to load the active application’s views on their own devices.

3.8 Prior Work That Informed CSPS

Shi [65] created a standalone driver for the i>clicker base station that enabled flexible use of clickers as input devices. He constructed a system called WebClicker that allowed users to input clicker votes from various devices, aggregate them via the web, and then deliver them to client applications over a single network socket. This modularization of input informed our separation of the Participant Input component from the application logic areas of the CSPS architecture. In our approach, the WebClicker functionality is replaced by two components: the Participant Input and the Identity Manager. The Participant Input component maps to the various input modalities participants use to interact with the system (clickers, web, smartphones, etc.), while the Identity Manager component handles the aggregation of the input. However, the Identity Manager goes a step further, providing one location where users assume a single identity, despite accessing the system via different devices.

One of Shi’s initial extensions to i>clicker usage was to provide novel visualizations of the responses students provided in class [65]. Instead of using the default histograms, which showed bars representing the number of responses split by button, Shi created a histogram that displayed stacked bars for each lab section in the class. Each of these bars was split in two, with one half representing the proportion of users who answered correctly and the other side representing those who answered incorrectly. The purpose of this modification was to alleviate the issue of students switching to whichever answer has the majority of votes, while still being able to provide visual feedback about their responses. This work by Shi, along with his other novel views for binary search tree insertion, linked list algorithms, and selecting portions of the screen (all described in [65]) inspired us to be flexible in allowing varying views in CSPS architecture. In Shi’s work, the views were tied directly to the instructor’s laptop where the software was being run. Our architecture loosens this coupling by indirectly linking application logic
with the views, which allows us to support both having local views on an instructor’s laptop, but to also support remote access.

Newson’s Clic\textsuperscript{in} system [58], extended later by Shi [65], used clicker input to drive a variety of activities, but focused primarily on a presentation architecture that integrated clicker activities with lecture slides to mitigate the awkwardness of switching between the two.

Another aspect of Shi’s work that informed our architecture was the introduction of roles associated with clicker IDs. Shi used a simple external file to map clicker IDs to roles (e.g., Student, Instructor or Demonstrator) and to their lab section number if they were students [65]. This enabled support for having multiple users gain control of the applications by assigning each of their clickers the Instructor role, which may be useful for administering complex applications should there be additional course personnel (e.g., teaching assistants) available to assist. In our architecture, this can be achieved at a number of locations, but most naturally at the Identity Manager and Session Manager components. At these junctures, auxiliary information of any sort can be augmented to clicks or other user input prior to being interpreted by the applications.

An illustration of how the CPSC architecture described here can be employed is provided in the next chapter, which describes our design and implementation of the \textit{Rhombus CSPS}. 
Chapter 4

Rhombus Classroom Synchronous Participation System

Rhombus Classroom Synchronous Participation System, or Rhombus for short, is an implementation of the architecture that was presented in Chapter 3. A diagram of the system is shown in Figure 4.1. All of the activity-specific logic and visualization computations take place using a Web Framework level that allows students and instructors to use platform-independent web browsers to view and control the applications. These browser-based components implement the roles of the Instructor Controller and Application Views in the CSPS architecture. An instructor typically places one web browser window with the main application view on the shared classroom projected display for the students to see and another browser window on the instructor’s laptop is used to control the activity. The Web Framework is hosted on the Web Server, which is the Session Manager component in Rhombus. It facilitates communication between the web browser windows and the participant inputs. The inputs are streamed to the Web Server over a network socket by the Clicker Server, which connects to the i>clicker base station hardware in the classroom and receives clicks from student clickers. The ID Server is an optional intermediate server that sits between the Web Server and the
Figure 4.1: Overall architecture of *Rhombus Classroom Synchronous Participation System*. The Clicker Server receives *i-clicker* clicks from the base station and transmits them over a network socket to the ID Server. There, the clicker IDs are replaced with aliases, and the data is transmitted to the Web Server. The applications themselves are viewed in web browsers, which receive data from the Web Server via the Web Framework over WebSocket connections.

Clicker Server in order to translate clicker IDs into aliases if it is important to anonymize the identities of students or to make the IDs more salient.

### 4.1 Clicker Server

The Clicker Server is a multi-client Java server whose primary purpose is to broadcast *i-clicker* clicks to all connected clients. It implements the Participant Input component of the CSPS architecture: it transmits clicks received from the base station over a socket connection and forwards commands to the base station that are received as messages over the socket. This effectively isolates the *i-clicker* hardware from the rest of the system and opens the possibility of using other types of devices instead of, or in addition to, *i-clickers*. All messages sent over the socket to and from the Clicker Server are done in JavaScript Object Notation (JSON) format. This format was chosen due to its relative compactness (compared to Extensible Markup Language or XML), its natural integration with web technologies.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>choices</td>
<td>Send a collection of i&gt;clicker ID, choice, and timestamp tuples (outgoing only)</td>
</tr>
<tr>
<td>enable choices</td>
<td>Open voting on the base station</td>
</tr>
<tr>
<td>disable choices</td>
<td>Close voting on the base station</td>
</tr>
<tr>
<td>status</td>
<td>Send the instructor i&gt;clicker ID, whether voting is open or closed, the current time, and the number of servers connected over the socket.</td>
</tr>
<tr>
<td>ping</td>
<td>Send an empty response.</td>
</tr>
<tr>
<td>choose</td>
<td>Input to the server a choice tuple that is output as if it were from an i&gt;clicker</td>
</tr>
<tr>
<td>instructor</td>
<td>Set the ID(s) of the instructor clicker(s).</td>
</tr>
</tbody>
</table>

Table 4.1: The commands accepted by the Clicker Server. These can be sent over a network socket or directly via standard input.

(JSON format translates directly to JavaScript objects in browsers), and its robustness and flexibility (JSON supports many different types of data, including collections).

The Clicker Server makes use of a custom clicker driver, originally developed by Shi who reverse-engineered the vendor-provided i>clicker hardware protocol [65]. Shi’s driver has been extended to support live plug-and-play for i>clicker base stations, auto-detection of the i>clicker base station version, and automatic system detection (supporting Mac OS X, Windows, and Linux) for loading the correct native libraries.

The Clicker Server supports broadcasting clicks to multiple clients, commands to toggle the accepting of clicks from clickers, and a command to simulate clicks. Being able to simulate clicks via a message received over the socket allows the user to create test harnesses to simulate real clickers interacting with the system (e.g., a web page with a virtual representation of a clicker on it, or a script that generates click commands over the socket) without those listening for clicks knowing anything is different about them.

The commands the server accepts are shown in Table 4.1. These commands can be sent to the Clicker Server as messages over the socket, or directly input to the server via standard input.
Unlike the vendor-provided software, the Clicker Server supports having multiple instructor *clickers* being used simultaneously (e.g., an instructor and one or more teaching assistants or other instructors can have administrative control over the classroom activities). The base station hardware allows for only a single privileged clicker ID, typically the instructor’s clicker, that will always have its clicks communicated to connected software regardless of whether the voting window is open or closed. This design prevents errant student clicks from having any effect when votes are not being recorded, and allows the instructor’s clicker to function as an administration device that controls slide navigation and toggles the voting window. When multiple instructors are active on the Clicker Server, voting is always enabled on the base station to work around the hardware’s limitation.

Instead of relying on the base station hardware to be the gatekeeper of clicks, the Clicker Server enables or disables processing student votes based on the software’s knowledge of the current state of the system. This filtering allows the multiple instructors’ choices to be sent over the socket as commands to the higher layers in the software (e.g., to the Web Server), emulating what the hardware does for a single instructor. This means that student *clickers* work slightly differently when multiple instructors are enabled: clicks made when voting is disabled will show a green light even though the choices will not be accepted. This is an unfortunate side effect, but we considered this a good trade off because of the usefulness of allowing multiple instructors for large classes. When only a single instructor is configured, the *clickers* vote status light behaves the same as it does with the vendor provided software. This inconsistent experience from the student perspective is undesirable, and future versions of the software may default to always using the behaviour that supports multiple instructors.

The Clicker Server accepts filters for affecting the inputs and outputs of the server that are added via Java’s service plugin system, with the multiple instructor feature being supported by one of them. This architecture decision allows the expansion of new features to the server without modifying the code itself. The filters are implemented as Java classes that conform to a simple interface consisting of three methods: initialize, input, and output. The
initialize method is called when the Clicker Server is initializing itself, the input method is called every time the Clicker Server receives input (either over the socket or from standard input), and the output method is called every time the Clicker Server sends output, allowing the filter to modify the data the Clicker Server sends and receives. An example of another filter would add additional roles to certain clicker IDs besides the instructor role, which may be useful for designating lesser administrative remotes such as those of teaching assistants.

Beyond filters, the server is configured by a properties text file `config.properties`, in which the port the server listens on, the ID of the instructor’s `i>clicker`, and the channel the `i>clicker` base station listens on are specified. Multiple instructors can be configured in this file by specifying multiple instructor clicker IDs separated by commas for the `instructorId` property. Future versions of the software may switch to a more verbose configuration requiring explicit toggling of the multiple instructors option to ensure users are aware of the changes to the student experience that take place when multiple instructors are enabled.

All clicks received by the server and all data transmitted over the network are logged. The main log, `server.log`, preserves the exact JSON messages that are sent over the network, including all student clicks. The clicks are also separately logged in an abbreviated format in their own file, `clicks.log`, because they are typically the most interesting part of the log and are often used to record student performance. Both log files are automatically archived and compressed after each day.

### 4.2 ID Server

The ID Server is an optional intermediate server between the Clicker Server and the Web Server, filling in the role of Participant Identifier in the CSPS architecture. When present, all clicks received from the Clicker Server are intercepted, having their `i>clicker` IDs (eight-hexadecimal digit numbers such as 371BA68F) changed to aliases (typically more readable identifiers such as names or student IDs) through a mapping determined by a database pro-
vided by the instructor or the institution. If no mapping from a given clicker ID to an alias is found in the database, the clicker ID remains untouched in the data. After the aliases have been swapped in, the clicks are then sent to the Web Server. All others messages from the Clicker Server to the Web Server, as well as all messages from the Web Server to the Clicker Server pass through without any modification.

The reason for having the ID Server separate from the Clicker Server is to provide a clean separation of functionality to ensure security and privacy for users. Theoretically, the ID Server can be hosted by a trusted third party that handles the translation from raw participant IDs (e.g., clicker IDs) to the appropriate alias (e.g., student ID, pre-assigned anonymous name, or the “celebrity” aliases that we describe later). This feature makes it possible to use Rhombus for behavioural research applications if the ID server is approved for use by a research ethics board.

Similar to the Clicker Server, the ID Server is also configured in a `config.properties` file. Administrators can configure the port the ID server listens on, and the host and port of the Clicker Server that the ID Server connects to. Logging is done in the same fashion described for the Clicker Server.

### 4.3 Web Server

The Web Server is a combined HTTP (or HTTPS) server and WebSocket server that connects to the Clicker Server for input. It is a Node.js implementation of a Session Manager from the CSPS architecture. When initialized, the Web Server continuously attempts to connect to the Clicker Server specified in the configuration until a connection is made, allowing the two servers to start and stop independently. The main views and controls of the system are in windows in a web browser, hence the need for it to be an HTTP server. The system needs to support multiple views all being managed by a single controller, but the views are shown in web browser windows, which typically operate in complete isolation. This presented some challenges, because we needed them to communicate with each other. The eventual solution was to have the Web Server mediate communication be-
tween the main controller window and the multiple viewing windows through WebSockets. For instance, if the controller needed to tell the views to show the Play state of a game, it would communicate with web browsers showing the views by sending a message to the Web Server over a WebSocket, which would then broadcast the message to the connected viewers over their own WebSocket connections to the Web Server. Upon receiving the message from the Web Server, the viewers would update accordingly.

Our initial design used URLs to map to different applications, so if you went to http://localhost:8000/apps/pd, the Prisoner’s Dilemma would be loaded and clicks would be interpreted with respect to whatever state the application was in in that browser window. While this design was simple, it did not support the usage of multiple browsers to display the same application. You could open up multiple browsers to the same URL, and each would receive clicks from the Web Server, but they would run their applications independently. This meant that if any randomization occurred in the application at all, the two windows would not be in sync. We saw this as a major flaw, since we commonly wanted to have multiple views for the same application, and so we re-designed our solution.

The new design required a clear distinction between a Controller, the browser that handles all application logic, and a Viewer, a browser that simply receives data to display on screen using a pre-defined view. With this distinction, we could have a single Controller communicate the same data to multiple Viewers in order to have multiple windows all showing the same application state. It also lets us show specific views of the application state in different windows, all corresponding to the same active application. Since web browser windows cannot directly communicate with one another, infrastructure at the Web Server was added to support this, as well as to supply a connection from the Clicker Server to the Controller. The component that manages all of this communication is called the Manager (Figure 4.2).
4.3.1 Application Managers, Viewers, and Controllers

Managers mediate all communication between Viewers and a Controller, using WebSocket connections to exchange messages between them. When a Viewer or Controller initializes in a browser, it sends a “register” message over the WebSocket connection to the Web Server, specifying the Manager ID to use, the type of browser it is (“viewer” or “controller”), and its name if it’s a Viewer. With this information, the WebServer looks up the Manager by ID and adds the new WebSocket connection to it if it exists. If no Manager is found by the specified ID, a new Manager is created with the appropriate association to the WebSocket connection. This is akin to the idea of creating a session, discussed briefly in the CSPS architecture description in Chapter 3.

When a Manager is first created, it establishes a connection to a pre-specified Participant Server, which is either the ID Server or the Clicker Server, but the Web Server does not know the difference. It simply connects
to a network socket and communicates following the interface specified in Table 4.1. Because all the ID Server does is change clicker IDs into aliases, it can be connected to by the Web Server in place of a Clicker Server.

If the connection fails, it is retried every 5 seconds, allowing the servers to initialize independently (i.e., you can start the Web Server before the Clicker Server or vice versa). Once a connection is made, pings are sent to the Participant Server every 5 seconds to ensure the connection is still alive. If a ping fails, a disconnection must have occurred, so the Controller is notified and attempts to reconnect every 5 seconds until the connection is restored.

Viewers use the socket message type “app-message” to send messages to the Controller. This can be useful when there are inputs to individual views required to send updates to the main application state. There are no restrictions on what JSON data can be sent with this message type to allow as much flexibility as possible in communication between Viewers and Controllers. Viewers primarily are used to simply receive data and update what they are displaying on screen, so it is common to only use their WebSocket connection to the Manager to receive updates from the related Controller.

Controllers can also send messages to Viewers with the “app-message” type. If the message specifies a “viewer” property, then only the Viewer that has that name will be sent the message, otherwise the Manager broadcasts the message to all Viewers associated with the Controller. Additionally, the Controller has more functionality it can use to communicate with the connected Participant Server. It can send “enable-choices” or “disable-choices” messages to enable and disable voting at the server level, can ask for a server status update with “status”, and can submit choices (i.e., the equivalent of clicks on an i>clicker) via the “submit-choice” message. It can also request a list of all connected Viewers with “viewer-list”, and will receive updates when new Viewers connect and disconnect with “viewer-connect” and “viewer-disconnect” messages.
4.4 Web Framework

The Web Framework provides infrastructure to build Rhombus applications that operate in web browsers. It provides many visual assets and base classes from which custom applications can be built, and also provides the general routing capabilities required to get applications loaded and in communication with the Web Server (and consequently, the Clicker Server).

Most applications require a Controller and a Viewer, which are easily registered with a Manager on the Web Server by loading carefully formatted URLs. For example, to load a Controller in Rhombus, you go to the URL http://host/managerId/controller (e.g., http://localhost:8000/m1/controller), and to load a Viewer, you go to http://host/managerId/viewer/viewerName (e.g., http://localhost:8000/m1/viewer/main). In these examples, the Manager is identified by “m1” and the Viewer is using the typical name “main”. The Web Framework interprets the URL to send the appropriate WebSocket “register” message to the Web Server to indicate a new Viewer or Controller needs to be registered. If a Controller is already registered for the Manager specified in the URL, it is replaced.

4.4.1 Controller

The main Controller interface, shown in Figure 4.3, offers the instructor control over Rhombus by providing the following:

- the ability to load an application from a list
- review a list of current participants
- navigate and review the state machine of the active application
- configure the active application
- open new and refresh connected Viewers
- control global configuration
- simulate clicks via virtual clickers in the web browser
Figure 4.3: The Controller interface in *Rhombus*. This is the main view the instructor interacts with to load and configure applications. It provides information about the current state of the system and offers controls for navigating the active application’s state machine. Global system state can be modified, and in debug mode, the instructor can simulate clicker input via virtual web clickers.
**Status Bar**

There is a status bar at the very top of the Controller with three areas: zoom controls, the Controller ID, and Participant Server status (see Figure 4.4). The zoom controls allow the instructor to adjust the zoom level of the Controller window independently of the other browser windows open in *Rhombus*. Because *Rhombus* runs in a web browser, the screens can also be zoomed in with the built-in browser zoom, but all windows open in *Rhombus* will be affected, which is often not the desired behaviour, since Viewer windows are commonly zoomed to different levels than the instructor controller. The Controller ID is primarily for debug purposes, and displays the manager ID (e.g., m1.controller), which is also retrievable from the URL. The Participant Server status area indicates whether or not the Web Server is connected to the configured Participant Server, and whether or not the Participant Server is accepting choices from participants.

![Figure 4.4: The status bar in the Controller interface.](image)

**App Selector**

In the App Selector, all available *Rhombus* applications are shown as large buttons that can be clicked to have the corresponding application loaded into the system. Doing so will activate the initial state of the application and cause all connected Viewers to display the first screen of the selected application. *Rhombus* currently has two utility applications: Grid for warming up (see Section 2.7) and Question for asking multiple choice questions. It also has a suite of Game Theory applications, such as Prisoner’s Dilemma, Stag Hunt, and the Ultimatum Game. It is important to note that the actual state of the application is stored in the browser window the Controller is running in. Navigating away from the page, refreshing the page, or closing the window will cause the current state to be erased. To prevent users from accidentally doing this, a warning dialog pops up asking them to confirm their desire to leave the page. The applications that exist are automatically
detected by the system by having directories in the standard location (e.g., \texttt{web/app/apps/\{appName\}/App.js}). The Controller runs a web service call to the Web Server at \texttt{/api/apps} to get the list.

\section*{Participants List and Latecomers}

The Participants area of the Controller displays the aliases of all currently active participants, along with their count, and if applicable, a list of queued participants who have been recognized by the system, but are not yet active in the application. The reason for this separation is that sometimes it does not make sense to allow latecomers to join in applications midway through. For example, if all teams have been balanced and partnered already, adding in a new user and re-balancing the teams can have undesirable side-effects, depending on the application. To prevent this from happening, users that click in when not already active within the application may end up in a queue, waiting for the next opportunity to be added in by the system.

Each state of an application has the ability to automatically add in latecomers, but it is optional. If the current state does not support adding latecomers, they get queued until a state of the application is reached where latecomers are added in on load. Should the instructor wish to forcibly add in the latecomers, as may be necessary if the state only adds those queued upon loading and others join in later at a safe time, a button below the list of queued participants can be clicked to add them to the currently active participants.

Due to this behaviour, we have discovered that when secret partnering is required in an application, that it be done at the latest possible time. In our current versions of the Prisoner’s Dilemma, we partner prior to the play phase, disabling our ability to add participants while the play phase is active. We could have just as easily partnered after the play phase had completed, but prior to scoring, to allow latecomers to seamlessly join the experience.
State Machine

Most Rhombus applications are created using a state machine architecture (see 4.4.3). In the State Machine area of the Controller interface, each state of the active application is shown as a pill-shape with arrows pointing in the direction that they flow in. Currently, Rhombus only supports a single next and previous relationship, although it is possible to develop states that determine their next state on the fly. The active state is shown in green. Should a state contain multiple states nested within it (e.g., a phase state contains multiple round states), they are represented that way geometrically. Applications can move from state to state automatically or manually, by pressing the Next State or Previous State buttons below the state machine, or by pressing the button \textbf{C} for next state and \textbf{D} for previous state on the instructor’s remote. This mapping was decided upon because it matches what the default i>clicker software uses for advancing and backing up slides. When the next state is activated, if necessary, the state machine area will automatically scroll horizontally to make it visible on screen.

Configuration

Many applications support configuration, which can be done in the Controller once the application has been loaded (Figure 4.5). The Configuration panel is initially collapsed, as it can be quite long, but can be toggled by clicking anywhere on the header. Each configuration panel contains a message field to allow the instructor to write a reminder message as to why this configuration was set, which will show up in the log files of the application. Otherwise, configuration fields can be manually specified by the application or automatically generated from a configuration object used by the application already. Updating the configuration does not change previous application state. All views that are currently active will update as soon as the “Update Configuration” button is clicked.
Figure 4.5: The configuration panel for team-based Prisoner’s Dilemma in Rhombus. The instructor can configure the names of the teams, the scoring matrix and the number of rounds for each of the three phases of the game. This panel was constructed automatically based on the properties of a configuration object, but can be modified to support custom styling and inputs (e.g., placing the matrix inputs in a matrix format).

Viewers List

The Viewers section shows all of the Viewers that are actively connected to the same Manager that the Controller is using. Typically, each Viewer listed represents an open web browser that is displaying a view of the application. They are listed by name with the plans to allow differently named views to receive different information from the Controller, or at least to display different things. Currently this is not implemented, so messages from the Controller are broadcast to all connected Viewers. A refresh icon is shown next to each Viewer, which can be clicked to send a message to the Viewers that causes them to redraw. Two convenience buttons are provided to create new browser windows that open to a URL that maps to the main Viewer for this Controller, or to an instructions Viewer. The instructions Viewer is
a specially named Viewer that was added to quickly create a window that only displays the application’s instructions and nothing else.

**Global Configuration and Tools**

The “Controls” area is a very important part of the Controller interface. Most importantly, it allows the instructor to enable choices on the Clicker Server by clicking the “Not Accepting Choices” button. The current state of the Clicker Server is shown in the status bar (whether it is Connected or not, and whether choices are being accepted or not), and is partially duplicated on this button. A red circle indicates a negative state, and a green circle a positive state. After attempting to enable choices on the server, if it is successful, the button will have a green circle and read “Accepting Choices”, and the status bar will have updated correspondingly. This is necessary for any clicks to come into the system besides those from the instructor’s remote.

Another useful tool is the “Instructor Controller Enabled” button. When enabled, the instructor’s controller is treated as a controller to the interface: A toggles accepting choices, C goes to the next state of an application, and D goes to the previous state. Sometimes, however, it is useful to use the instructor’s clicker as a normal clicker, especially when demoing an application, since the instructor’s clicks will come through even when clicks from students cannot. This means that an instructor can leave accepting choices disabled, disable the “instructor controller” via the button, and then demonstrate using an application.

The only utility currently provided by *Rhombus* is a countdown timer that allows the instructor to display a timer on all open Viewers in the upper-right corner. This can be useful when an instructor wants to let students know how much time is left for them to enter their responses before moving on to the next state, causing those who have not yet responded to assume the default choice.
When activating the instructor controller in Rhombus debug mode, virtual web clickers become available at the bottom of the page. The IDs of these clickers are of the form WebXX where XX is a number. Clicking the buttons on the clickers sends the clicks to the attached Clicker Server, which then sends them back up the chain to the Web Server just as if a real clicker had submitted a vote.

Figure 4.6: When activating the instructor controller in Rhombus debug mode, virtual web clickers become available at the bottom of the page. The IDs of these clickers are of the form WebXX where XX is a number. Clicking the buttons on the clickers sends the clicks to the attached Clicker Server, which then sends them back up the chain to the Web Server just as if a real clicker had submitted a vote.

Debug Mode

When testing out applications in Rhombus, it is typically inefficient to continuously test with actual clicker devices. To make the debugging process more expedient, a debug mode for the controller exists and can be accessed by going to a URL of the form: http://host/managerId/controller/debug (e.g., http://localhost:8000/m1/controller/debug). At the bottom of this page, a section entitled “Web Clickers” is visible, with convenience buttons for adding web clickers and causing them all to vote one way or another (Figure 4.6). There are also keyboard shortcuts for accessing these buttons as well as the next and previous state buttons, which expedite the process further. The Web Clickers themselves work by submitting choices all the way to the Clicker Server via the Session Manager, which then transmits them as if they are normal i>clicker inputs, ensuring an accurate debugging experience.
4.4.2 Viewers

As described previously, a Viewer can be loaded by going to the URL http://host/managerId/viewer/viewerName (e.g., http://localhost:8000/m1/viewer/main), which is typically done by clicking the “Open New Main Viewer” button in the Controller. Upon doing so, the browser will display the view for the current state of the active application. There is no set limit to the number of windows that you can open to create new Viewers. Typical usage is to have the Controller open in a browser on the instructor’s screen, while a single Viewer window is opened and placed on an external display (e.g., a projector). If multiple external displays are available, they can each have their own Viewer window, possibly showing different parts of the application. Closing Viewers at any time does not have any impact on the state of the application.

If there is no active application when a Viewer is loaded, the browser window will display the name of the Viewer and an “awaiting view” message. Once an application has been selected, the Viewer will be automatically notified via the WebSocket connection and the new view will be loaded. The reason why we display the name of the Viewer when no application is selected is to facilitate placement of the different windows. When Rhombus is upgraded to support sending specific messages to specific views, it will be helpful to configure them accurately prior to loading the application. The only current situation where this is necessary is the distinction between the “main” viewer and the occasionally used “instructions” viewer.

Similar to Controllers, Viewers also all display a status bar at the top with a zoom control and the ID of the window (e.g., m1.viewer.main). However, the Viewer’s status bar does not show the Connected status to the Participant Server, it only shows whether or not the system is currently accepting choices from participants.

4.4.3 State Applications

The primary structure used in Rhombus to construct applications is a state machine. Each application consists of a series of states connected to one
another in a linear way. States are intended to handle all the logic of the application, and are only known to the Controller. It is common for states to have an associated view, which Viewers use to show relevant application views, but not all states have them. If a state solely exists to process data, it automatically moves on to the next state when it is complete, until a state with a view, a ViewState, is reached. ViewStates wait until certain conditions are met before moving to the next state in the application, typically the instructor clicks the Next State button (or presses C on the instructor clicker), but it could also be programmatically determined.

For example, in the simplest form of Prisoner’s Dilemma, the state machine consists of the following states (Figure 4.7):

1. **Attendance (ViewState):** Participants use their clickers to check-in to play the game
2. **Botcheck:** If there are an odd number of users, a bot is added to the list of participants to ensure everyone has a partner
3. **Partner:** All participants are matched with randomly selected partner
4. **Play (ViewState):** Participants can now play the Prisoner’s Dilemma, clicking C to cooperate, and D to defect.
5. **Score:** The results from the play phase are calculated for each participant
6. **Stats:** Overall statistics for the game are computed
7. **Results (ViewState):** The results from the game are displayed on screen, and are logged to the server.

Note that there are many ways to divide the states up, but we have found having simple states that do a single operation allows for the greatest reusability.

*Rhombus* provides a number of basic states that can be used to build complex applications:
Figure 4.7: The state machine of the basic version of Prisoner’s Dilemma. This version of the game only has three primary states: attendance, play, and results. During these states, the state machine pauses and allows participants to interact with the system. In between them, logic states, botcheck, partner, score, and stats, are run to maintain the proper internal state of the application.

- **State**: The basic State object that provides the bare bones structure needed. This object is typically extended for use with states that only do processing, and do not have a view (e.g., botcheck or partner).

- **ViewState**: An extension of State that adds in the correct behaviour of rendering a view on connected Viewers and does not automatically move to the next state in the machine (e.g., play or results).

- **MultiState**: A state that essentially has its own internal state machine nested inside of it, which is useful when you need to reuse blocks of states (e.g., a round state that contains play, score, and results states).

- **RepeatState**: A modification of the MultiState with conveniences implemented to allow repeating a single state a specified number of times. This is intended to be used with repeat occurrences of the same state (e.g., trials in an experiment, or a phase state that contains a number of round states nested within it).

While attempting to support being able to move back to previously seen states, we ran into issues with the participants being modified by future states and thus not properly repeating the previous states. To fix this, we created a StateMessage object, which is used now as the standard vessel of input and output between states. Each state holds on to the input it previously received as a copy, ensuring that when it is loaded again by the
instructor moving back to a previous state, it can reload the input it had received initially and be in the same state as it was in the first time.

4.4.4 User Representation

In *Rhombus*, one of the most common ways to depict users on screen is to use an *avatar*. Avatars, shown in Figure 4.8, are widgets that consist of a photograph as a backdrop, the user’s alias, and any relevant feedback the user may need for the given application (e.g., whether or not they have played or their score). In many of the apps developed for *Rhombus*, avatars of all active participants are displayed in a grid on screen. Users find their place on the screen by locating their alias and photograph and can direct their attention to this smaller region of the screen to interpret their own individual feedback.

The following chapter provides a description of an application we created, *Sequence Aliaser*, that allows a large number of users of *Rhombus* to register their clickers with aliases in parallel.
Chapter 5

Sequence Aliaser

The ID Server comes packaged with an application called Sequence Aliaser, which allows users to have their *i>clicker*-alias mapping created interactively by entering a predefined sequence on the clicker itself. The goal was to provide a way to have clickers that are not known to the system prior to usage quickly assigned an identity to make using the system more enjoyable. It is easier to find a name on screen than it is to find a hexadecimal clicker ID, and with the Sequence Aliaser, we can associate clickers with names without gathering all the clicker IDs in advance.

To use the Sequence Aliaser, each user must be given a pre-defined sequence that has been mapped to an identity. The user then enters the sequence of buttons on their clicker, which effectively associates the clicker with the identity in the system.

We decided on using celebrities as the identities that are provided by Sequence Aliaser, since there are so many of them, and they are a very diverse group. In our implementation, we had a pool of 64 celebrities, with half of them being female, half male. A large number of the celebrities came from movies, music, or television, but there were some other popular figures, such as Steve Jobs, Bill Gates, and Michael Jordan. We did our best to avoid controversial figures and political figures, since we would be randomly assigning them to participants and did not want them to feel uncomfortable. After selecting 64 celebrities and finding iconic photographs of them to use
in the system as avatars, we tested the collection for recognizability with seven graduate students of varying background at our university. Along with photographs, each celebrity was given a brief nickname or alias that either matched their actual name, nickname, or character they were famous for. The goal in assigning names was to keep them somewhat different from the actual celebrity’s identity and to add in humour (e.g., Queen Elizabeth II had alias “liz”, Leonard Nemoy was “spock”).

The sequences used for the 64 celebrities in our implementation consisted of four characters, from a selection of the letters A through D, mapping to four of the five $i>clicker$ buttons. For example, the sequence BDAC mapped to “arnie” (Arnold Schwarzenegger), and CACC mapped to “hova” (Jay-Z). A typical slip of paper indicating the sequence and celebrity identity associated with it that is handed to users is shown in Figure 5.1. The button E was reserved for resetting input, allowing users to start over in case an error was made either by entering the sequence incorrectly, or by accidentally entering another user’s sequence. The sequences themselves were carefully designed to have a Hamming distance of 2 with every other valid sequence (i.e., any two sequences differ in at least two locations) to reduce the chance of the latter issue occurring. Sequences of length 4 with Hamming distance 2 were chosen to limit the number of buttons required to have an identity assigned, but it may be better to increase the sequence length and have a higher Hamming distance because with a Hamming distance of 2, two valid sequences may only differ by a swapping of two characters (e.g., BADC is “cera”, and BDAC is “arnie”).

As buttons are pressed, unidentified gray boxes show up on the screen representing each clicker that is being recognized, with white circles inside the boxes representing the number of characters entered (Figure 5.2). When a sequence is successfully entered, the gray box becomes an avatar, and the clicker buttons, instead of being used to enter a sequence, can now be used to verify that the alias was registered to their clicker by making it change colours and animate in various ways as described in Table 5.1. These changes are similar to those used in the Grid application (Section 2.7).

After all clickers have successfully entered their sequences and the screen
Figure 5.1: An example slip of paper used with Sequence Aliaser that indicates the celebrity identity and sequence required to associate a user’s clicker with it.

<table>
<thead>
<tr>
<th>Button</th>
<th>Colour</th>
<th>Animation</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Green</td>
<td>Pulse. The avatar cycles between getting larger and smaller.</td>
</tr>
<tr>
<td>B</td>
<td>Blue</td>
<td>Bounce. The avatar moves up and down in a bouncing motion.</td>
</tr>
<tr>
<td>C</td>
<td>Purple</td>
<td>Shake. The avatar moves left and right in a shaking motion.</td>
</tr>
<tr>
<td>D</td>
<td>Yellow</td>
<td>Swing. The avatar swings left and right as if there was a pivot</td>
</tr>
</tbody>
</table>

Table 5.1: The colours and animations of avatars in the Sequence Aliaser. As with the Grid application, the animations were provided by Eden’s Animate.css [30] and were selected for their distinguishability.

now displays celebrity avatars, the administrator can register the mappings into the database by clicking the Next State button on the control panel, or by pressing C on their i>clicker.

The Sequence Aliaser and other applications were tested in a live university classroom environment over two consecutive terms. The method and results of the first evaluation are covered in the following chapter.
Figure 5.2: Sequence Aliaser in use. Some users have entered between 1 and 3 letters of their sequence, indicated by the white circles on unidentified gray boxes, while others have completed their sequence and now see a celebrity avatar that they can animate with clicker button presses. Contrary to other uses in the system where avatars are shorted lexicographically by alias, the position of the avatars on screen is determined by the order in which their first button press was received. This behaviour prevents locations shuffling while aliases change.
Chapter 6

Term 1 Evaluation and Results

In this chapter, we describe a field trial that was conducted to test the Rhombus CSPS in a university classroom environment. The evaluation used both quantitative and qualitative methods to gain insights into operational aspects of using the system and to understand the experiences of the instructor and the students who used the system. The evaluation was conducted during the iterative development cycle for Rhombus. The developers were present in the classroom and during classroom sessions and provided technical support to the instructor and assistance to students, beyond what would be expected for normal usage. This was done because we wanted to obtain concrete in-lecture experience with the system as part of the design process. A second field trial in which there was less intervention by the research team is reported in the next chapter, and the results of the two studies are discussed in Chapter 8.

After explaining the method that was followed for the first field trial, we present the results of the study and some of the implications that we used to improve the system.

6.1 Pilot Deployment

Prior to using the system in a classroom, it was deployed during a keynote presentation at the CompArch 2013 conference held in June in Vancouver,
British Columbia, where 58 attendees played Prisoner’s Dilemma, some having no prior experience using clickers. Everyone managed to play the game successfully. Informal observations and followup comments from participants led us to believe they understood the results of the exercises and the underlying game-theoretic principles that were being demonstrated. The audience seemed engaged and the system worked without any issues, demonstrating its robustness in handling over 50 clickers at once.

Based on this, we conducted a field trial in an actual classroom in the Fall.

6.2 Method
To test the efficacy of using the interactions Rhombus provides, we evaluated the system in a classroom environment. The system was used as an integral part of lectures in a single term offering of a third-year Cognitive Systems course at our university. The class makes regular use of clickers for answering quizzes throughout lecture and historically has used the histograms provided by i>clicker software to play the N-Person Prisoner’s Dilemma. The following games were played with Rhombus: Coin Matching, Stag Hunt, Prisoner’s Dilemma (single round per partner), Iterated Prisoner’s Dilemma (5 rounds per partner), and the Ultimatum Game. These five games were played one per lecture over a series of non-sequential lectures. The system was used an additional two times to familiarize the students with it and the game they would play prior to the lectures where the actual full games were played. These two warm-ups took place before the Coin Matching game and before the Stag Hunt game, and are described in the results as session 1 warm-up and session 2 warm-up, respectively. Each session took between 15 and 20 minutes and typically involved 15 rounds of play.

The class in which the system was evaluated had two instructors who alternated teaching every two lectures. Our system was used solely by one of the instructors, henceforth referred to as “the instructor”. We refer to the other instructor in the course as “the co-instructor”. Prior to running the course evaluation, the game-theoretic exercises described in Chapter 2 were
developed in consultation with the instructor to ensure they met his needs for the course.

6.2.1 Participants

The study had prior approval from the Behavioural Research Ethics Board. Students were required to opt-in to the study or their data was not included. There were 40 third-year students enrolled in the class, with 34 consenting to have their data used in the study. No compensation was given for participating. While all students in the class were expected to take part in the games as they were part of the class curriculum, students faced no consequence for not consenting to have their data used for research purposes.

6.2.2 Environment and Apparatus

The room had capacity for approximately 50 students and was arranged with several rows of tables split into two columns with 4 students per column. The classroom projector was directed towards the front-centre of the class and was used for the main display of the games. An additional projector was brought by the researcher and projected on a makeshift screen directly adjacent to the main screen to display the instructions of the games. Both projectors used a 1024x768 resolution. Rhombus was run on the researcher’s laptop, a MacBook Air computer (2 GHz Intel Core i7, 8GB 1333 MHz DDR3 RAM, 11" screen) running Mac OS X 10.8.5 for the first 4 sessions and Mac OS X 10.9 for the final. The main controller interface of the games was displayed on the 11” laptop screen and controlled via both trackpad and i>clicker. The i>clicker base station model TMX14 was used.

6.2.3 Student Representation

Showing 40 students individual feedback on a projected display greatly constrains what can be placed on the screen. Each game represented individual students with the standard avatar representation discussed in Section 4.4.4: squares with their alias at the top, current score or action in the middle, and photo of their celebrity as the background (see Figure 6.1).
Figure 6.1: The avatar representation of user with alias “jobs” in various states during Prisoner’s Dilemma. From left to right, the avatar is successfully checked-in during attendance, has yet to play during the play state, has played during the play state, and has its score shown during the results state. (N.B. This is a duplicate of Figure 4.8.)

6.2.4 Procedure

When playing the games, the researcher would enter the classroom and set up the apparatus. The instructor brought his own i-clicker base station to the class with him for quizzes, which was unplugged for the duration of the game, and replaced by the researcher’s own base station. It ran on the BB frequency, the same channel used for the class.

We decided to assign each student a celebrity as their alias in Rhombus, largely due to their sheer quantity and recognizability. The celebrities used were vetted for recognition by seven graduate students. On the first warm-up session, students were handed slips of paper with their assigned celebrity alias and a sequence of buttons to press for use in the Sequence Aliaser (see Chapter 5). After the students had all entered their sequences, the aliases were saved in the system for future use.

In all other sessions, after setting everything up, the researcher initiated the Grid application as a warm-up activity (see Chapter 4 for details). The purpose for doing this was to give the students a chance to acclimatize to the system and recall how everything works.

Once everyone was confident that their clicker was working, the game of the day would begin. An attendance screen would show up and students would see their squares with large check marks to confirm that they were ready to play the game. The first play state would follow and each student would be given a chance to make an action. For instance, in the Prisoner’s
Dilemma, the students could choose between pressing C to cooperate or D to defect. When all students had played, the instructor would move to the results state and the students would be able to see their scores in the middle of their avatars.

Most games played in this field trial had 3 phases of 5 rounds each after which their total accumulated scores would be displayed, with top scores highlighted. At the end of each game a log was produced that contained all the actions and scores in CSV format. To preserve the identities of the student participants, the logs were transformed, replacing aliases with clicker IDs, before giving them to the instructor.

After each session, a digital questionnaire was administered via the system. Each questionnaire contained the same seven questions relating to the experience they just had playing a game (see Appendix B for details). The questionnaires on the first and last session were augmented with questions about their thoughts on the system overall. All questions used a five-point semantic difference scale (e.g., from Strongly Agree to Strongly Disagree). Beginning on the third session, a 20 second countdown timer was used during the questionnaire period. This done in reply to a student comment that they wanted an option to not answer a question. There are only 5 buttons on the i>clicker, and each question had 5 possible answers, so there were no buttons available to assign to those who wish to abstain. In previous sessions, we had waited until the number of responses matched the maximum number we had seen playing games that day, which forced all students to answer so we could move on. With the countdown timer, we could show everyone on screen when we would move on to the next question, so they would not feel unnecessary pressure to answer if they did not want to.

After the final session, a written questionnaire was distributed to students with several open-ended short answer questions and the instructor was interviewed to learn his original intentions for using the system, how it supported him, and where it could be improved. This interview was audio recorded and transcribed.
6.3 Results

We report the results of the evaluation in this section, beginning with those from the student questionnaires administered via the system after each session, followed by the open-ended student questionnaires, the interview with the professor and any observations made by the researcher. Students who did not consent to have their data used were pruned from the results before beginning analysis.

6.3.1 Student Questionnaire

We ran several statistical tests to detect if there were any differences in responses as students gained more experience with the system, and similarly, to see if the various games were received in different ways by the students. The following results use the effect size measurement $r$ where Cohen suggests a value of .1 is small, .3 is medium, and .5 is large [26]. For all post-hoc comparisons, the Bonferroni method was used to adjust p-values. All responses are to five-point semantic difference scale questions, typically ranging from strongly disagree (1) to strongly agree (5). Results for many of these questions are summarized in Figure 6.2.

Q1 – It was easy to find myself on the screen. A Friedman test revealed a significant effect of session on ease of finding oneself on screen ($\chi^2(4) = 16.816, p < .01$). Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between session 1 and 4 ($p < .05, r = .51$), and session 1 and 5 ($p < .05, r = .55$), but not between the other sessions. Of the 34 possible participants considered for this analysis, only 16 were included due to missing data from various sessions. Using all participant data for a given session, the medians were 4, 4, 5, 5, 5 for sessions 1 through 5 respectively, with a 1 being strongly disagree, and 5 strongly agree.

Q2 – I understood the controls of the game. A Friedman test revealed a significant effect of session on understanding controls of the game ($\chi^2(4) = 10.405, p < .05$). Post-hoc pairwise Wilcoxon Signed-Rank tests showed no significant difference between pairs of sessions. Of the 34 possible participants considered for this analysis, only 17 were included due to missing data.
Figure 6.2: The percentage of responses to various questions in the digital questionnaire, accumulated across each session. The questions are as follows: Q1 – It was easy to find myself on the screen; Q2 – I understood the controls of the game; Q3 – I understood the results of the game; Q4 – I liked playing the game; Q5 – I felt engaged during the game; Q6 – I would like to use this system in other classes; Q7 – It was satisfying to use this system to play the game.

from various sessions. Using all participant data for a given session, the medians were 4, 5, 5, 5, 5 for sessions 1 through 5 respectively, with a 1 being strongly disagree, and 5 strongly agree.

Q3 – I understood the results of the game. A Friedman test revealed no significant effect of session on understanding the results of the game ($\chi^2(4) = 8.28, p = 0.082$). Of the 34 possible participants considered for this analysis, only 15 were included due to missing data from various sessions. Using all participant data for a given session, the medians were 4, 4, 5, 5, 5 for sessions 1 through 5 respectively, with a 1 being strongly disagree, and 5 strongly agree.

Q4 – I liked playing the game. A Friedman test revealed no significant effect of session on liking playing the game ($\chi^2(4) = 3.624, p = 0.459$). Of the 34 possible participants considered for this analysis, only 19 were included due to missing data from various sessions. Using all participant data for a given session, the medians were 4, 5, 4, 5, 4 for sessions 1 through 5 respectively, with a 1 being strongly disagree, and 5 strongly agree.
Q5 – I felt engaged during the game. A Friedman test revealed no significant effect of session on feeling engaged during the game ($\chi^2(4) = 3.117, p = 0.538$). Of the 34 possible participants considered for this analysis, only 14 were included due to missing data from various sessions. Using all participant data for a given session, the medians were 4.5, 4, 4, 5, 5 for sessions 1 through 5 respectively, with a 1 being strongly disagree, and 5 strongly agree.

Q6 – I would like to use this system in other classes. This question was only asked on the first and last (fifth) session. A Wilcoxon Signed-Rank test revealed no significant effect of session ($W = 64, Z = -0.0971, p = 1$). Of the 34 possible participants considered for this analysis, only 28 were included due to missing data from various sessions. Using all participant data for a given session, the medians were 4.5, and 5 for sessions 1 and 5 respectively, with a 1 being strongly disagree, and 5 strongly agree.

Q7 – It was satisfying to use this system to play the game. This question was only asked on the first and last (fifth) session. A Wilcoxon Signed-Rank test revealed no significant effect of session ($W = 44, Z = 0.90, p = 0.388$). Of the 34 possible participants considered for this analysis, only 24 were included due to missing data from various sessions. Using all participant data for a given session, the medians were 4.5, and 4 for sessions 1 and 5 respectively, with a 1 being strongly disagree, and 5 strongly agree.

Q8 – How would you rate the system compared to typical iClicker usage? This question was only asked on the first and last (fifth) session. A Wilcoxon Signed-Rank test revealed no significant effect of session ($W = 20, Z = -1.0142, p = 0.35$). Of the 34 possible participants considered for this analysis, only 26 were included due to missing data from various sessions. Using all participant data for a given session, the median was 4 for both sessions 1 and 5, with a 1 being much worse, and 5 much better.

Q9 – How helpful are the in-class multiple choice questions with regards to learning? This question was only asked on the last session, and had a scale ranging from very harmful (1) to very helpful (5). The median answer was 4 (helpful), with 3.6% answering very harmful, 3.6% harmful, 10.7% neutral, 60.7% helpful, 21.4% very helpful. The results are summarized in
Figure 6.3: The percentage of responses in the last session to questions Q9 and Q10, which asked how helpful the in-class multiple choice questions (Q9) and playing the games (Q10) was with regards to learning.

Q10 – How helpful was playing the games with this system with regards to learning? This question was only asked on the last session, and had a scale ranging from very harmful (1) to very helpful (5). The median answer was 4 (helpful), with 0% answering very harmful, 0% harmful, 16.0% neutral, 64.0% helpful, 20.0% very helpful. The results are summarized in Figure 6.3.

Q11 – It is worth taking class time to do multiple choice questions. This question was only asked on the last session, and had a scale ranging from strongly disagree (1) to strongly agree (5). The median answer was 4 (agree), with 3.7% answering strongly disagree, 7.4% disagree, 7.4% neutral, 66.7% agree, 14.8% strongly agree. The results are summarized in Figure 6.4.

Q12 – It was worth taking class time to play games with this system. This question was only asked on the last session, and had a scale ranging from strongly disagree (1) to strongly agree (5). The median answer was 5 (strongly agree), with 3.7% answering strongly disagree, 0% disagree, 18.5% neutral, 18.5% agree, 59.3% strongly agree. The results are summarized in Figure 6.4.
6.3.2 Student Short Answer Responses

This section reports on the results from analyzing the short answer surveys filled in by participants at the end of the final session.

**R1 – Which input device(s) would you prefer to use with an interactive classroom response system?** There were 31 responses to this question, which allowed participants to circle multiple choices from the set: clicker, mobile phone, tablet, laptop, other. The responses were as follows: clicker 83.9% (26), mobile phone 22.6% (7), tablet 9.7% (3), laptop 25.8% (8), other 0%.

Participants were asked to explain their choice, revealing that clickers were selected for the following reasons: simple and easy to use (10), they are already in use (7), affordability (6), everyone uses the same device (5), like using clickers (4). Reasons against using clickers included: forgetting to bring them (2), expensive (1), unreliable (1), would prefer not having to buy another device (1). There were not enough answers for the other categories to draw clear trends, but participants did mention a benefit to using a mobile phone was that they always had it on them (3), and a benefit of using a laptop being that they already own the device (2).

**R2 – Describe any issues or problems you had with RPS.** There were 25 responses to this question. The most common response was about the
i>clicker devices not working, mentioned by 5 participants, which was no fault of Rhombus. These responses ranged from “sometimes clicker would turn off” to “low battery light doesn’t seem to work”. Others mentioned problems finding themselves on screen (3), for example, “Sometimes it was a bit difficult to find myself on screen - during check in because there weren’t always faces with the pseudonyms, also the tiles kept shifting so it was hard to track.” Another problem was with interpreting the results screen of the various games (3), for example, “Difficult to understand results/rules of the game; relies on instructor heavily”. Two participants mentioned bugs in the system that were corrected on the fly or for the next session.

R3 – Describe what you liked most about RPS. There were 31 responses to this question. The most common response was praising the design of the interface (12), with participants saying, for example, “A neat way to represent responses,” “The results are easy to read,” and “The pictures, colours, real-time feedback... everything about it is pretty nice.” Individual feedback was mentioned by 8 participants, for example, “It’s nice to know that you 110% sure you have inputted an answer”, and “You could keep track of your votes, you could see yourself respond, and you could track your “points” for the session.” Individual engagement was mentioned by 7 participants, for example, “It is interactive and forces students to pay attention.” We distinguish individual engagement from classroom engagement (5), where students mentioned the social engagement of the whole class, for example, “I liked how we interacted together as a class, like an in class activity.” Other categories included ease of use (4), having fun (4), faces of avatars (2), and comparing to others (2). One student mentioned the applicability of the games played to the course material, saying “Mostly the application to game theory - if it were just [multiple choice] questions, it wouldn’t be useful.”

R4 – Describe any suggestions you have for new features or improvements to the system. There were 21 responses to this question, with little convergence into categories. The most common answers were that there was nothing to improve (3), the order of names on screen could be “less arbitrary” and be made “easier to find” (2), the results display could be sorted differently or made more “accurate and clear” (2), and identification of users
could be modified to support user customization via web or otherwise (2).

R5 – Do you have any suggestions for other applications of the system? There were 11 responses to this question, with the most common answer being to use the system for polls (4), the same way the typical i>clicker software is used. There were no other converging categories, but responses covered the following topics: cooperative applications, reaction time games, competitions, music rhythm practice, forming discussion groups, use in labs to “test possibilities with other groups”, and attendance.

6.3.3 Instructor Interview
A semi-structured interview with the COGS 300 instructor provided insights into the value of playing games in class, the shortcomings of his previous efforts, and benefits of using our system.

P1 – Pedagogical Value. The instructor described the primary pedagogical outcome of playing the games was to overcome the limitations of only teaching theory to students, stating, “I think teaching it just as theory you get a very small number who really get it.” He followed up by mentioning having direct experience as a subject in the games made it easier to understand psychological experiments that were discussed in class that involved the games. Beyond these reasons, he suggested a large portion of the learning comes from experiencing a social environment in which not all agents act the same way, and then reasoning about what is happening in reality and how it differs from the theory.

I would think the problem with university students in this kind of a class is, why would you come to class and not be looking at something else on your screen? Why would you be engaged? One way you’re engaged is you’ve invested in this see-no-light process and invested in and suddenly it’s like the expectations you invested in that didn’t work out in ways you don’t understand and now you’ve got to decide what resources to use. How do you explain that? Are they all wrong? Are you wrong? Was the theory misapplied? And I think that’s where lots of the learning
happens. The more that you can do more rounds, you can do more variations, there’s more chance for that to happen. I mean ideally if we change this in this way, would this still happen?

P2 – Multiple Games. His reasoning for using the system for multiple games was that he felt it necessary to gradually bring more advanced concepts to students over time, saying,

I doubt you could go in in one day, one lecture, and do “here’s all the cool things about game theory.” So it’s a little bit like going in and saying “here’s all the cool things about computers, or programming, or anything.” You have to have enough of a track record so you can see why this thing is important.

He went on to describe the succession of games as a “mini curriculum based on games getting harder in one way, or bringing out different features of the social environment.”

P3 – Games before clickers. Prior to using clickers for playing the games in class, the instructor had tried out doing a paper version of the Prisoner’s Dilemma, where they wrote out a program on paper and exchanged with another person who then ran their program against another to determine their scores. The information was collected in aggregate on the blackboard, photographed, and entered into a spreadsheet. This process had to be completed between lecture sessions. Furthermore, there was “no comparative analysis at the individual level”. The instructor simply collected the papers and nobody ever saw the results of others. He noted that there were further problems with the paper-based method, saying, “the papers go missing, the papers aren’t always legible and the instructions are a little bit too complicated and so the failure rate was quite a bit higher.” He also mentioned that it was confusing for students dealing with all the paper, and that “a lot of time was spent on administrative stuff.” He said that “the bookkeeping got in the way” of the actual goal of playing the game.

P4 – Initial clicker attempt. The instructor had tried using the default i>clicker software in the past to play the N-person variant of the Prisoner’s
Dilemma, using the histogram to show the distribution of results. This level of feedback was suitable for this game, as all cooperators received the same score, and similarly for all distractors. He said, “the clickers let us both do it and talk about results right away.”. He described a multi-round process where students would vote with clickers in round 1, review the histogram of results, then vote again in for round 2, and so on. He noted that the built in software was “quite deficient” for supporting gaming purposes, but it did effectively allow them to play the N-person game.

P5 – Problems with initial clicker system. The instructor lamented that they had only done the N-person variant of the Prisoner’s Dilemma because it was what they could do with the system they had in place. He described it as “playing some weird game we could jigger into the classroom format,” instead of what the instructor desired to teach.

So the problem there, it meant that, knowing that that worked, we tended to use that game and actually started doing readings around that game. But it’s a really hard game to analyze, so what you’re seeing now is an artifact of a difficulty in classroom procedure. In a stadium we can do this, so I guess we’re going to do a lot of this card flipping. Why are we doing that? Well because that’s the thing we can do with a stadium full of people, well this is the N-person game we can do easily in the class, but that’s not a good reason to choose the game. We should be able, I mean ideally, the instructor should have a palette of games they can choose from and say “No, I’m really interested in this, I want to use this game that has this shape.”

He mentioned that one major limitation preventing using the default i>clicker software to run the games was that they needed to pair students anonymously in a group in order for the games to work. He described the individual feedback mechanism as “crucial” for playing the games, which he further explained saying, “You can’t say you have a well-informed agent if [they are thinking] ‘I chose some things over a few times, I don’t know what happened’.”
P6 – Clicker benefits and issues. The instructor speculated that one benefit of using the clickers as the input device for the games was that “the clickers have a seriousness about them because they’re used for quizzes”.

He admitted, however, that there are issues with using clickers as well, stating “you know you have to give people feedback, but it’s difficult to know how to give a room full of people feedback working with extremely limited device with no individual level feedback on it.”

P7 – Benefits of new system. The instructor described one benefit of using the system was that since playing the games was “easier and easier”, it freed him to think more liberally about what the “ideal curriculum” for the course would be. He was already able to adjust the selection of games since what was easy and hard to do had changed as a result of using the system. He described having a freedom to experiment with different games since they were “cheap” to run.

He valued highly the ability to anonymously pair individuals, saying, “I think by not being forced to do N-person games, with actually being able to now, crucially, pair people up individually, then you can play the games you want to play.” He went on to describe the benefits was not only in the ability to pair people individually, but to be able to quickly and easily play games a repeated number of times, as they were intended to be done. He was now able to play “standard games”, “the games that actually drove game theory.”

Since we’re linking up with the literature, we’re not doing this weird thing of reading papers of what you don’t want to read because they’re linked to the game that we’re stuck playing and now we’re trying to explain complicated papers. We’re reading classics.

He described the increase in the amount of games they can complete in a session as “an order of magnitude more” than he previously could. He described completing the games in a third of a session, when previously with paper it took the entire session. He said, “the administrative stuff was a lot less and the actual pedagogy was more.”
The anonymization ability of the system was seen as an important benefit to the instructor; he described it as being “clear to everyone,” and that it “puts the games in another space with separate pseudonyms.” This gave him the confidence that they were “meeting the conditions of the literature.”

He also found the comma-separated values (CSV) format the data was provided in was conducive to extra analysis after the games were over, saying the data was “already excel friendly and ready”, and “you have a finer level of analysis available because all the data is already in a CSV, standard form.”

He noted that playing the games with the students “made a definite difference” with student engagement, and the real-time individual feedback was a “huge improvement”. He elaborated, “you’re basically making available huge amount of information for them to do what they like with.”

P8 – On registering clickers. The instructor noted that he was “very sceptical” that using celebrity aliases would work with the students for playing the games, but described the experience as having “exceeded expectations” and that it was a “really good solution to the problem.” He went on to state that the way we registered clickers (using Sequence Aliaser) had a fun engagement factor. He said, “it wasn’t a chore” and described it as akin to “little half-time exercises” where people have fun instead of becoming disengaged and bored with an otherwise administrative task.

6.3.4 Observations
This section covers observations made by the researcher throughout each session.

Registration with Sequence Aliaser. The class understood what to do, with the majority getting it right immediately. Around five or so participants took longer than the rest, with 41 participating in total. One person accidentally entered someone else’s sequence, but the problem was corrected after the other person declared somebody had taken her pseudonym. I asked the class to press E if you did not see your avatar on screen, which freed up the accidentally taken avatar and the problem was resolved.

Towards the end of registration, there was a single participant everyone
was waiting for who was having a hard time entering the sequence. It was not obvious who the person was, however, as many participants were still pressing buttons to make their avatar wiggle on screen, and others were jokingly faking having a hard time entering the sequence. Eventually the final person registered and I moved to save all the associated aliases in the system by pressing the C button on my (the instructor’s) clicker, causing the application to move to the next state. However, I had disabled instructor’s remote mode and so in pressing a button on my clicker, a new gray box was displayed on screen. This caused people to cry out in dismay that somebody new had joined that they would now have to wait for. I assured them it was me and saved the aliases.

The last participant to register was under alias “spears”, but the slip for spears was not handed out. It later became clear that this participant was an outlier in the class, having very little experience with computers and technology in general, and often needed assistance in using the clicker.

Game Playing. In their first time using the system to play Coin Matching, participants seemed to understand how to use the system with ease. People groaned, cheered, and laughed as results were displayed after each round. A bot was automatically added to one of the teams to balance the numbers and nobody said anything about it.

During the second session warm-up, a student approached me and asked for her sequence again. At this point, the clickers had been registered with the system already and the sequences were no longer needed, but it was not clear to her. At the end of the session, the instructor noted there had been “really great engagement” in the class, and that the system had been received very well. He said there was the “right balance of comical and educational aspects” to the system, referring to the natural comedy that follows playing a celebrity in a game.

When it came time to actually play Stag Hunt, there were a number of bugs in the system with calculating results. This caused great confusion to the students who cried out “Hey, that’s not right!” when scores did not accumulate correctly, and “Why do I have a previous score?” when it was the first round of a new phase.
One student had forgotten their clicker, so I provided them another and updated the system to use the new clicker ID for their alias, allowing them to enjoy the same experience they would have had, had they used their regular clicker.

Another person forgot her clicker on the third session and was loaned a clicker for use in the class that day.

The channel the clickers used in the classroom was BB, which required each clicker to be setup prior to use. In the third session, I was delayed in setting my clicker to the correct channel prior to starting the game, and ended up setting the channel during the first round. It appears that when changing the frequency on an i>clicker, it sends the button press A when it is done. Since my clicker was in instructor mode, the A signal was interpreted as a command to disable voting for the other students. For a moment, we all waited while nothing changed on screen until a student called out that it was frozen. At that point, I realized what had happened and re-opened voting.

In the fourth session, the game was over in roughly 16 minutes, and the questionnaire in another 2, returning the podium to the instructor 20 minutes after class had started. During the game play, a participant came in late, joined a team (replacing a bot), and nobody seemed to notice the change. A student forgot their clicker again, and this time I had brought some clickers with aliases already registered to them to expedite the process of giving them a replacement. This student ended up with alias “liz” and the clicker worked without issue. Some students noticed the new alias “liz” on screen and commented on how they had never seen her before, saying things like, “Is that a transfer student?” The avatar “spears” had become known for typically being the last person to play in each round, and the class began making comments about it, but did not seem to direct them at any student in the class. It appeared they did not know who spears was, and instead directed their attention to the representation on screen.

On the final session, the class played the Ultimatum Game, where they were partnered randomly with different people in each round. One student expressed feelings that he was not being randomly partnered, since he was
repeatedly getting the low offer of 1 (instead of 9), which was rare. In actuality, he was being partnered differently each round. One student used a different clicker than was registered in the system and showed up on screen with no photograph, and with alias being a 8 digit hexadecimal $i>cl\text{icke}r$ ID. He did not seem to have any difficult playing despite this irregularity.

6.4 Summary

In this chapter, the evaluation method and results from the first term field trial of Rhombus CSPS have been described. The system was used to play five games in a third-year cognitive systems course and was well-received by both the students and the instructor. Students rated it to be on a similar level to clicker quizzes with regards to being worth taking class time and its helpfulness with learning. They described themselves as being very engaged while using the system in class and noted that the individual real-time feedback was a key feature. The instructor was pleased with the system as it allowed him to run the games in class that he had always wanted to, but could not previously due to lack of technological support. Given the positive response from the field trial, the instructor decided to use this system again during the following term, this time without the in-lecture assistance of the researchers. The method and results of this second trial are covered in Chapter 7 and an interpretation and discussion of the results from both trials is covered in Chapter 8.
Chapter 7

Term 2 Evaluation and Results

In this chapter, we describe the second of two field trials conducted to test Rhombus CSPS in a university classroom environment. In this trial, the instructor used the system on his own, with the researchers only attending lecture on the final usage of the system to collect survey data from the students. The evaluation was similar to the first trial described in the previous chapter, using both quantitative and qualitative methods to better understand the experiences the instructor and the students had using the system. We first describe the method we used, then present the results. An interpretation and discussion of the results from both field trials is given in Chapter 8.

7.1 Method

The following term after our initial evaluation, the same course was taught again. This time, the instructor that was using our system was teaching once more, but the co-instructor was different and had little experience with clickers. In this term, the instructor used the system on his own without any additional support from the researchers beyond an initial hour-long training session. We provided no rigorous procedure for the instructor to follow when using the system, allowing him to integrate it into the lectures as he saw fit. As such, we report on his procedure in the interview results (Section 7.2.3).
In this term, the system was used to play four games: Coordination, Prisoner’s Dilemma (single round per partner), Iterated Prisoner’s Dilemma (5 rounds per partner), and the Ultimatum Game.

7.1.1 Participants
The same ethics approval and procedure were used as in the earlier study. We had 23 students consent to have their survey data used, while there were 31 students enrolled in the course. No compensation was given for participating in the study.

7.1.2 Environment and Apparatus
The room was nearly identical in setup to that described in our initial evaluation. The instructor did not make use of a makeshift screen to project the instructions, and instead simply used the single projector display available. The instructor used his own laptop, which was also an 11” MacBook Air, to use Rhombus in the same way as the researcher did in the first evaluation.

7.1.3 Procedure
In this term, the instructor had knowledge of the students’ clicker IDs a priori, something the researcher did not have in the first term. This negated the need for using the Sequence Aliaser to register clickers with the system. Instead, the instructor ran a script prior to first usage that registered the clickers and used the Grid application to familiarize them with who they were. He simply told them to press buttons on their clicker and due to the synchronization between button presses and avatar feedback (animation, colour changing, letter displaying), students were able to figure out who their alias was.

The researcher only showed up on the final day of use of the system where the class played the Ultimatum Game. After the game had completed, the researcher administered the same digital and short answer questionnaires that were used on the final day of the first term evaluation. The instructor was interviewed after the final session to learn about his experience using
the system on his own and with a different class of students.

7.2 Results

We report the results of the second term evaluation in this section, beginning with those from the student questionnaires administered via the system after the final session, followed by the open-ended student questionnaires and the interview with the professor held after the evaluation was completed. Students who did not consent to have their data used were pruned from the results before beginning analysis.

7.2.1 Student Questionnaire

The same as in the first term evaluation, all responses are to five-point semantic difference scale questions, typically ranging from strongly disagree (1) to strongly agree (5). Results for many of these questions are summarized in Figure 7.1.

Q1 – It was easy to find myself on the screen. The median of 23 responses
was 5, with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 0%, disagree 4.3%, neutral 8.7%, agree 30.4%, strongly agree 56.5%.

Q2 – I understood the controls of the game. The median of 21 responses was 5, with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 0%, disagree 0%, neutral 14.3%, agree 23.8%, strongly agree 61.9%.

Q3 – I understood the results of the game. The median of 22 responses was 4 (agree), with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 0%, disagree 4.5%, neutral 4.5%, agree 45.5%, strongly agree 45.5%.

Q4 – I liked playing the game. The median of 22 responses was 4 (agree), with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 4.5%, disagree 4.5%, neutral 31.8%, agree 36.4%, strongly agree 22.7%.

Q5 – I felt engaged during the game. The median of 20 responses was 3.5 (neutral/agree), with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 0%, disagree 15%, neutral 35%, agree 30%, strongly agree 20%.

Q6 – I would like to use this system in other classes. The median of 22 responses was 4 (agree), with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 4.5%, disagree 0%, neutral 22.7%, agree 36.4%, strongly agree 36.4%.

Q7 – It was satisfying to use this system to play the game. The median of 21 responses was 4 (agree), with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 0%, disagree 0%, neutral 42.9%, agree 23.8%, strongly agree 38.1%.

Q8 – How would you rate the system compared to typical iClicker usage? The median of 19 responses was 4 (better), with a 1 being much worse, and 5 much better. The response breakdown was much worse 0%, worse 10.5%, neutral 21.1%, better 57.9%, much better 21.1%.

Q9 – How helpful are the in-class multiple choice questions with regards to learning? The median of 19 responses was 4 (helpful), with a 1 being
very harmful, and 5 very helpful. The response breakdown was very harmful 5.3%, harmful 10.5%, not helpful or harmful 15.8%, helpful 47.4%, very helpful 21.1%. This result is summarized in Figure 7.2.

**Q10 – How helpful was playing the games with this system with regards to learning?** The median of 22 responses was 4 (agree), with a 1 being very harmful, and 5 very helpful. The response breakdown was very harmful 0%, harmful 0%, not helpful or harmful 45.5%, helpful 41.0%, very helpful 13.6%. This result is summarized in Figure 7.2.

**Q11 – It is worth taking class time to do multiple choice questions.** This The median of 22 responses was 4 (agree), with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 4.5%, disagree 13.6%, neutral 9.1%, agree 36.4%, strongly agree 22.7%. This result is summarized in Figure 7.3.

**Q12 – It was worth taking class time to play games with this system.** This The median of 22 responses was 4 (agree), with a 1 being strongly disagree, and 5 strongly agree. The response breakdown was strongly disagree 4.5%, disagree 0%, neutral 27.3%, agree 36.4%, strongly agree 18.2%. This result is summarized in Figure 7.3.
Figure 7.3: The percentage of responses to questions Q11 and Q12, which asked students if it was worth taking class time for in-class multiple choice questions (Q11) and for playing games with Rhombus (Q12).

7.2.2 Student Short Answer

This section reports on the results from analyzing the short answer surveys filled in by participants.

R1 – Which input device(s) would you prefer to use with an interactive classroom response system? There were 23 responses to this question, which allowed participants to circle multiple choices from the set: clicker, mobile phone, tablet, laptop, other. The responses were as follows: clicker 69.6% (16), mobile phone 34.8% (8), tablet 8.7% (2), laptop 26.1% (6), other 4.3% (1).

Participants were asked to explain their choice, revealing that clickers were selected for the following reasons: simple and easy to use (4), clickers are academic, not personal tools (3), they can be anonymous (2), they’re accessible to everyone (2), and they’re portable (2). However, the most common sentiments were that students did not want to buy clickers (5), at least not at their current price, and that phones were convenient since they always had them on their person (5).

R2 – Describe any issues or problems you had with RPS. There were 21 responses to this question. The most common response was that there were no issues with the system (12, note that this does not include the 2
blank responses). The next most common sentiment in the responses was that the issues revolved around the instructor’s usage and explanation of the system (6), including student difficulty understanding game instructions (3), for example, “It would be good to have a trial run, before actual. Otherwise mistakes are made. It might be not the problem with the game, but with how it was explained!” This also included difficulty the instructors had using the system and setting it up (3), for example, “Hard for the instructor to set up/run the first couple times. The first time playing the games took forever, and one of our 2 profs had a hard time with the standard software all term.” Two students mentioned having difficulty discovering their alias during the first session of the term, while another student mentioned forgetting their pseudonym between classes.

R3 – Describe what you liked most about RPS. There were 20 responses to this question. The most common answer had to do with individual engagement (9), with participants saying, “It is far more interactive than the other iclicker system and funner to use”, and, “Interactivity helps me from falling asleep or making bad clip art drawings in my notes.” The next most common responses were about real-time feedback (6) and ease of use (6). These were followed by anonymity (4), for example, “Being able to play games real-time and having the alias to remain anonymous,” and use of celebrities (4), for example, “The celeb alter-egos are fun anonymizers, it was engaging way to experiment with game theory.” The rest of the responses fell into having fun (3), individual feedback (2), visual design (2), easy setup (1), and game theory (1).

R4 – Describe any suggestions you have for new features or improvements to the system. There were 17 responses to this question. The most common answer was that there was nothing to improve (6). There was little convergence beyond that category, with the following responses covering visualizations of results (2), initial alias assignment (2), computer players (2), instructor proficiency (2), cartoon avatars (1), including instructions before the game (1), persistent scoring across sessions (1), and a way to “view scores afterward”.

R5 – Do you have any suggestions for other applications of the system?
There were 11 responses to this question, with the most common answer being “none” (4), followed by using the system for polls (3), the same way the typical i>clicker software is used. One student in that group mentioned “it could be used for Q and As where history of answers matter (socratic method).” There were no other converging categories, but responses covered the following topics: entertainment (e.g., “I would totally play this with my friends at a house party”), strategic games, neuroscience, music rhythm games, game theory games.

7.2.3 Instructor Interview

A semi-structured interview was conducted with the instructor where he described his preparation and procedure for using the system on his own in the classroom, and his experience compared to the previous term, including issues and ideas for improvement.

P1 – Using system on his own. The instructor described the experience as “a bit of plus and minus”: on one hand, being in complete control granted greater flexibility over his use of the system, freeing him to take breaks and go on tangents as needed, while on the other hand, he felt more distracted since he also had to pay attention to the operation of the system. He noted however that one factor that made the use of the system feel “a little looser” was that he had more experience with it, and that the system was well tested after many bugs were corrected during the initial term.

He suggested that it may have been a less intimidating experience for the students since it was just the professor at the front of the class, not the researcher. However, he mentioned that the class’ second professor this term was much less interested in game theory than the previous term, which may have impacted student engagement.

The instructor noted that it was a “fairly fun” system to use, contrasting it with the “nerve-wracking” use of animation or videos in slideshow presentation software, where he felt he often worried about having a connection or needing a local copy of the video. The room he was in had a poor internet connection, which further bolstered his contentment with the fact that the
system ran locally with “obvious connections”.

He described initially having difficulty in controlling the system, finding it awkward to navigate between states using the keyboard and mouse on his small laptop. He said it was difficult to know what keys would affect which software when he was running slideshow software, the \textit{i>clicker} software, a video player, and \textit{Rhombus}. However, once he recalled he could control the games with the clicker, it became much easier for him. He said, \textit{“The point is that with \{the clicker\} in hand, it’s very easy to click through it and you can walk around.”} He noted that walking around allowed him engage better with the students and notice if they were ignoring or misinterpreting any portions of the screen.

He had a positive attitude about the system after having used it on his own for the term, describing his thoughts as \textit{“What new things could we do with it?”} rather than \textit{“How can we get it finally to work next term?”}

\textbf{P2 – Preparation.} Prior to the start of the term, the instructor ensured there were enough aliases to match to students in the class. When he received the list of clicker IDs, he assigned the aliases to the students via a script provided by the researcher.

Before each of the times it was used in class, the instructor reviewed the game in the system with the built-in debug mode, which allowed him to simulate clickers via the web browser. He said the rehearsals typically took 10 minutes.

\textbf{P3 – Classroom Procedure.} The classroom procedure was similar to the previous term in that students were given instructions at least one lecture in advance of playing the game. While in the first term, a warmup session was done with the system, administered by the researcher, this time the instructor did things differently. He said they ran warmups for two of the games using the regular clicker software to save having to switch back and forth, something that was not perceived as being easy to do. He also said that for the purposes of the warmup, since the students were already comfortable with using the system, the extra features the system provided were not necessary; he said they could have had people raise their hands instead of clickers to accomplish the same result.
On the actual day of gameplay, he typically started with the game first to minimize switching between the i>clicker software and the system. This was not ideal as sometimes students would “trickle in” during the game, which he described as “not the section of the class you want people drifting in.” To mitigate this problem, he often would place administrative announcements at the beginning to provide a buffer. He did not re-acquaint participants with the system by having them play around in the Grid program prior to playing the games as we had done in the first term. Instead, he began with the games immediately.

P4 – Communicating Aliases. In this term, the students were assigned aliases prior to their first usage of the system, but the instructor was charged with communicating these aliases to the students. To do so, he tried to tie it in to the topic of interactive robotics they were discussing at the time in the class. The idea was to try to use the feedback provided in the Grid application to determine which alias you were assigned. He was confident this would work given that the class only had 31 students. He said that people successfully “figured it out” just by interpreting the movements and changes to avatars that took place in sync with button presses on their clickers. He said, “I don’t think I sent out anything or announced it in any [way], and there didn’t seem to be anyone saying I’m forgetting who my alias is.”

P5 – Student Motivation. While last term students were under the impression that their scores in the games would translate to marks in some way, there was no such impression this term. The instructor said “there was some vague discussion that maybe there would be some bragging points or something,” but nothing concrete and nothing relating to marks. He admitted that this may have had an adverse effect on student motivation and in ensuring the proper methodology needed for experimental game theory, but felt constrained given that the system was still being used in a semi-experimental way as part of research. He was wary of violating research ethics in this case, and so decided to avoid using marks, but he said he is motivated now to figure out a way to tie in marks in the future.

P6 – Student Engagement. As mentioned earlier, there was a different co-instructor this term than previous, and he was less interested in game theory
than his predecessor. The instructor described his previous co-instructor as “a much more jump off and have an objection and raise a problem in every class” kind of instructor. He said, “other people pick up from that and think ‘oh that’s okay to do’ and this term was much less of that. And so I think it carries over in all aspects of the course. So I think the games played out well, and we got some really interesting results, but there was maybe less of that interrupting and debating and that.” He said this “big difference in style between the two terms” was likely the largest reason for the change in student engagement, which he felt was lower than the previous term. While he thought students may be less intimidated since there was no researcher controlling the system this term, and so engagement might have increased, he thought the lack of interest in game theory from his current co-instructor may have overshadowed the other changes.

P7 – Celebrity Avatars. The instructor believed that the use of celebrities as avatars in the system had a large impact in the students enjoyment of using the system. He said that I solved the random assignment to unknown other players in the classroom in “a fun way,” and that the avatars “were personified enough that people had the sense that they were actually playing a definite other person.”

He elaborated further by describing the celebrities as “a prop in an interactive performance.” He clarified, saying, “It’s not just collecting data, it’s not putting people in a room and collecting data from them, it’s actually a part of interactive stage performance, a classroom.” The varied naming schemes and general quirkiness of the avatars provided material for the instructor to use to smooth over times in rounds when the class is waiting for the last few users to make their moves. He said, “There were things you could play with a bit without really nagging how come this person [who] for all I know for very good reasons takes always 35 seconds rather than 10.” He continued, “You’re talking at versions of people that aren’t [them themselves]. So I think that was a good thing, and numbers wouldn’t do it.” He was adamant that neutral alternatives to celebrities, such as shapes, would be “really different” after having this experience with celebrities. He clarified further as follows:
It had a little bit of edge to it, and the thing is that’s good, it’s a performance. It’s good to have a little weird thing that you can focus on because partly what you’re doing is, between rounds and staff, giving people other things to come on folks. So think of whatever, it’s Craig Ferguson, and it’s weird chicken jokes. I mean, you’ve got a crowd that you’re trying to involve and I think the interface worked really well for that.

P8 – Issues. The primary issue the instructor had was in switching back to the i>clicker software after having used the system. He said that in doing so, the i>clicker software was not responding. He figured out that he needed to unplug and plug back in the i>clicker base station after stopping using the system before the i>clicker software would work again, but the initial time he ran into this problem he ended up doing a paper quiz instead of one with clickers.

He said it was “not totally easy to drop in and out of [the i>clicker software]” since he would end up with two clicker files for the day. He felt the vendor-provided software was “unforgiving” and given that he was already frustrated by having two instructors use it which he said “it’s not designed for”, he didn’t want to try it.

Another issue he had with the system was a lack of visibility with regards to whether or not the data had been saved from the active game. He described himself as “less than fully confident that work was saved.” He said he only exited too early once, resulting in him not receiving the overall summary data file, but was able to recover the phase results from the intermediate data files, which contained all the information he needed. He said “maybe if there had been some feedback that said ‘Stage 2 saved, Stage 3 saved, Final Result saved.’ I think that was the only lack of confidence.”

With regards to the data files themselves, while he said that they were “interpretable”, they “invited [him] to make mistakes” because the rows were non-homogenous, containing results for a single participant intermixed with that participant’s partner. This representation duplicates data in the file and makes it difficult (but not impossible) to run row-wise formulas on the
data. He described the process as “you have to do these weird skipping over things” to analyze the data.

He noted a shortcoming of the system is that you cannot easily give your clicker to another person and have them use it since they would also need to know your alias. He wanted to do this so that students could run scripts written by other students using the other student’s clicker.

P9 – Improvements. The instructor had a couple of suggestions for improvements to the system. First, he suggested having a suite of visualization options that the administrator can select to analyze the results of rounds or phases immediately in the system. He suggested having a dynamic way to customize which are shown while the system is in use, to help aid in answering student questions and ad-hoc analysis.

The second suggestion he had was about the results of the system. Instead of only providing the data in raw CSV format, there should also be production of reports at both an overall class level and an individual student level.

7.3 Summary
In this chapter the evaluation method and results in the second term field trial of Rhombus CSPS have been described. The system was used to play four games in class without the researcher present to assist the instructor, except for the final usage when the researcher came to administer a survey to the students. While students had little difficulty understanding how to use the system, their ratings of enjoyment and engagement, while positive, were lower than the previous term. The instructor maintained a positive attitude towards the system and provided some explanation for the difference in student behaviour across the two terms. He expressed his desire to continue using the system in the future and made suggestions on how it could be improved from an end-user standpoint. Interpretation and further discussion of the results found in both field trials can be found in the following chapter.
Chapter 8

Discussion

Our findings indicate that the usage of *Rhombus* in the classroom was a positive experience for both the students and the instructor. Across the two terms, students were easily able to understand the controls and results of the games, as well as find themselves on screen. While in both terms the majority of students answered they liked playing the games and that they felt engaged, there was a noticeable difference between the results of term 1 versus term 2.

Comparing the questionnaire results from the final session of term 1 to term 2, we see that 75.0% of students liked playing the game in term 1 versus 59.1% in term 2. Similarly, 71.4% felt engaged in the game in term 1 versus 50.0% in term 2. Note that these final session percentages from term 1 are lower than the average percentage across term 1 (82.0% for liking playing and 82.3% for feeling engaged), so the lower values may be a result of the Ultimatum Game itself, but this does not explain the difference between terms. The relative lack of enthusiasm for using *Rhombus* in term 2 shows up again when looking at the results of wanting to use the system in another class; in term 1, 92.9% of students wanted to, while in term 2 this number dropped to only 72.7%. This trend continues when comparing how students rated the helpfulness of learning the games had, 84.0% in term 1 to 54.5% in term 2, and if it was worth taking the time in class to play the games, with 78.0% agreeing in term 1 to 63.1% in term 2.
To explain these differences, we first note that the students also had lower ratings of the helpfulness of clicker quizzes in the class across the terms, with 82.1% finding them helpful in term 1 versus 68.4% in term 2. A similar difference is seen with whether the quizzes were worth taking class time: 81.5% in term one agreed they were, and only 68.4% in term 2. This can help explain the differences to some degree, as it seems the students had generally more negative opinions of the in-class activities, but there is more behind the 29.5% decrease seen between terms 1 and 2 on how many students thought the games were helpful with learning.

The instructor noted that one of the primary pedagogical outcomes of playing the games was the experience of using them in a real social environment, and the ensuing discussion around the results seen during gameplay. However, the engagement was lower in term two and there was less debate in the class compared to term 1, which may have had an effect on how helpful the students perceived the games to be to their learning. The instructor suggested that one possible reason for the stark difference between the two terms was the interaction of the second course instructor (the co-instructor). In the first term, the co-instructor stirred more discussion on a regular basis and was perceived to have set a tone in the class that fostered debate, which was missing in the second term. Seeing as this was a primary driver in the pedagogy of playing the games, it may partially explain the lower ratings in term 2.

Beyond the change in co-instructor, there was also the effect of the instructor having to get used to and gain confidence in running the system on his own. His first use of the system was somewhat awkward until he adopted using the clicker to control the games, which may have left a poor first impression on the students, compared to term 1 when I, the creator of the system, was there to ensure it ran smoothly. This may have had a negative impact on student engagement, and was mentioned by a number of students in the short answer questionnaire as one of the issues they had with the system.

A final possibility for why students perceived helpfulness of the games was lower in term 2 was that the motivation behind the games had changed
between terms. While in the first term, students may have been under the impression that their results in the games were going to translate into grades, during the second term there was no such misunderstanding. Consequently, students may not have had the proper motivation to care about the outcomes of the game and become involved more deeply in playing them. One student even said, “I do not find it to be an effective teaching tool. There is no reason for students to act rationally and this skews results.”

Despite these differences, in both terms students rated the usage of the system to be better than typical *i>clicker* usage.

Students in both terms preferred clickers as their input device of choice, although in term 2 there was 14.3% decrease in selecting clickers and a 12.2% increase in preferring mobile phones. Students had emphasized that it was important everyone in the class could afford a device to use, and it was good that they were all on the same device as reasons for selecting clickers, but in the second term there was a larger contingent of students who would have preferred not having to spend money on clickers at all. Interestingly, no student mentioned that *i>clicker* provides a way to simulate clicks via a mobile phone for their default software.

Clickers are still a viable input device, however, as they are relatively cheap and ubiquitous across university campuses. Students praised their use in this context as they associate them as being academic tools, and liked not having personal devices be involved in academic work. This sentiment is similar to the instructor’s speculation that clickers carry a seriousness about them since they are typically associated with quizzes and marks, which may impact how students interact with the games.

Across the terms, students consistently valued the engagement and interactivity the system provided, as seen in their responses to the freeform question asking what they liked about the system (38.7% of responses in term 1 and 45.0% in term 2). In term 1, the instructor noted that using the system for playing the games provided a substantial improvement in student engagement from his perception compared to playing the games in previous terms.

Another feature highly lauded by students across the terms was the in-
dividual and real-time feedback that was provided. Students valued being “110% sure” their clicks had been received by the system. They also liked not having to wait to see overall class results and to be able to compare the scores of other students in the class.

By integrating the system into his class, the instructor was able to play the games he felt were best for the education of the students, as opposed to the limited games he was previously restricted to. He was happy to have this freedom and eager to begin developing the “ideal curriculum” for his students. Along with this freedom, using the system also reduced the amount of administrative work involved with playing the games, another bonus for the instructor who described Rhombus as having “exceeded expectations”.

The usage of celebrities as avatars for students was received positively by both the instructor and the students. Several students described them as being a fun aspect of the system, and the instructor, despite his initial skepticism, lauded the celebrities for their secondary use as a prop in the classroom to facilitate passing time in what may otherwise be awkward moments waiting for stragglers to play. However, since these aliases are not what students naturally identify themselves with, at times they may forget who their assigned alias is. In term 1, we avoided any difficulties with this by beginning each session by using the Grid program, allowing students to reacquaint themselves with their avatar before gameplay began. In term 2, this was not the case, and at least one student complained of having forgotten their alias for the first couple of turns in a game.

There was a divergence in informing students of their celebrity alias for the term. In the first term, students were given slips of paper with a sequence on it for use with the Sequence Aliaser, while in the second term, they simply had to press buttons in the Grid application until they figured out which avatar was their own. Using the Grid application worked for the most part, but some students mentioned difficulties with it in the questionnaires. It may be better in the future to explicitly communicate to students their alias to ensure a smoother first experience with the system. Alternatively, even if clicker IDs are available in advance, as they were in term 2, using the Sequence Aliaser may be a suitable option to engage the students in
registering their clickers to an alias themselves.

An interesting observation took place when using the Sequence Aliaser in the first term: at the very end when my clicker accidentally caused a box to show up on the screen, there was an outcry from the class about somebody showing up at the last minute. This behaviour may suggest that social pressure from other students in the class will dissuade misbehaving or deliberately delays in gameplay by individual students.

The major issue that arose from the system use came from term 2 when the instructor was using both the default *i-clicker* software and *Rhombus* in the same lecture. The instructor’s experience indicates that once *Rhombus* establishes a connection with the clicker base station, the base station must be unplugged and plugged back in again before the *i-clicker* software will recognize it. This forces an undesirable user experience, which cannot be avoided unless an alternative to the *i-clicker* software that works with the Clicker Server is used instead.

In the following chapter, an experiment evaluating a novel display technique to use with *Rhombus* is described.
Chapter 9

No-onset Presentation of Semi-Private Feedback on a Shared Display

This chapter describes two experiments that were conducted to evaluate the effectiveness of a novel feedback technique for users of a shared display. This chapter duplicates some previously mentioned information to preserve its form as a manuscript for a conference paper submission and it concludes with acknowledgements specific to the research reported in this chapter.

9.1 Introduction

We explore the problem of providing individual visual feedback to users of a shared display without clearly alerting other users to what feedback was given. We are motivated by a classroom application, where the shared display is typically a projected screen in an auditorium or classroom and the individual feedback confirms the responses provided by individuals in a potentially competitive situation in which it is desirable that users not know how others have responded.

Large screen displays are often used in classroom environments, both for sharing course material and for sharing instructions for interactive activi-
ties. One common activity of this type is to have a quiz conducted in class with the use of a student response system (SRS), such as the $i>clicker$ [10], where a question is displayed on the shared display with a mapping of buttons to answers (commonly labelled A through E) and students press the corresponding buttons on their remotes to answer. SRS usage in classrooms has been shown to positively affect engagement, motivation, attendance, and understanding [37, 40, 70], encouraging further work to refine and enhance the user experience.

Older versions of the $i>clicker$, which still have wide use in institutions today, have very limited feedback capabilities, displaying only a single green or red light indicating whether a student’s click has been registered with the system or not; newer versions of the $i>clicker$ provide more sophisticated feedback with small LCD displays or LEDs for each button, but they are more expensive, not in widespread use, and still suffer from student doubt about whether the other end of the system properly received the input.

In our experience, students often have low confidence that their click is being registered despite the green light flashing on their clicker. Evidence for this can be seen by observing students repeatedly clicking the same button multiple times to answer a question and by sometimes swapping their answers back and forth between the one they believe is correct and a different one when a histogram of results is shown because this will indicate to them that their clicker is in fact working properly. Without some form of confirmation from the system receiving the clicks that your individual click has been correctly received, doubt remains, leading to students having a focus towards the technology and away from the course material, which is an undesirable outcome pedagogically.

Reducing this uncertainty is the problem we addressed. Our previous work showed that students highly valued having individual, real-time feedback to their clicks displayed on screen, which seemed to alleviate the aforementioned issues (see Chapters 6 and 7). We focused on a solution using the basic $i>clicker$ functionality that would be suitable for in-class quizzes and similar activities where it was desirable to minimize the chances that students could interpret each other’s feedback to gain an advantage.
In our solution, we provide each user with a section of the shared display that functions as their individual feedback area. This approach was taken to support the sizable audiences found in large classes, where serial presentation of user feedback is not an option; feedback must be provided in parallel to many students at once. Each user’s feedback area contains an avatar, which is a widget that consists of a photographic portrait (typically of a celebrity), an alias or identifier, a colour overlay, and a 7-segment display (Figure 9.1). When a user presses a button on their clicker, we display the letter on the user’s avatar using the 7-segment display for a brief period of time before having it disappear so that other users do not have much time to look at it.

![Figure 9.1: A sample user’s avatar. This user has alias “you”, and the 7-segment display, which can show the letters A through E, is displaying the letter D. No colour overlay is active on the avatar in this figure.](image)

A simplistic approach to using this would be to have no letter displayed on the avatar until a button is pressed, then briefly display the letter, and finally return to displaying nothing. However, perceptual research has shown that objects that suddenly appear in view (said to have an abrupt onset presentation) tend to draw attention [77], so students may find this distracting. Furthermore, the attention drawing nature of this approach may make it too easy to see how other students in the class are responding with their clickers, which may not be desirable, especially if the activity underway is a quiz or other situation in which students are competing for marks.
To address this limitation, the letters displayed could be offset differently for each user’s feedback with a privately known Caesar cipher (e.g., A means C, B means D, C means E, etc). Another solution could be to assign each student a private set of symbols that maps to the buttons they have pressed, allowing only them to interpret what was shown on screen. We decided to avoid these solutions because they required extra setup and cognitive effort for the users, and the institution would need to manage a single mapping for each student, because students could not reasonably be expected to learn different mappings for each course that uses the system.

Instead, we leveraged the limitations of human visual perception to create a novel technique for semi-privately sharing feedback to users on a shared display. We describe the feedback as semi-private. Our intention is to allow only someone who is already focusing on the location at which the feedback will appear to be able to see the feedback. We expect that although others might notice something change, they will not be able to interpret what they have seen. We manage this by using a no-onset presentation, which describes a presentation where a symbol is at first camouflaged on screen and the camouflage later disappears to reveal the symbol. To ensure that the intended user is focusing at the right time, an example application could synchronize the display of feedback with a user action, such as providing the feedback immediately after the user presses a button on an i>clicker. We associate feedback with a user’s avatar to provide a distinct spatial focus for each user’s feedback.

In this chapter, we report on an evaluation of our technique in an experimental setting, designed to test the efficacy of being able to interpret letters shown for briefly visible durations. We measured the accuracy of identifying a target letter shown in a known position associated with a user’s avatar, as well as the accuracy identifying another distractor letter that was simultaneously shown in a randomly located position associated with a different user’s avatar. We ran two experiments, one using a no-onset presentation, while the other used an abrupt onset presentation. Our results suggest that the no-onset technique offers a better balance of target versus distractor accuracy, particularly when the visible letter duration is set at 80ms.
9.2 Related Work

We break down the related work into two parts: by applications and by perceptual research. In the first part we discuss specific applications that make efforts to provide private feedback to multiple users of a shared display. In the second part we discuss various results from the field of visual perception that have led to the design of our experiments and our display technique.

9.2.1 Applications

Shoemaker and Inkpen introduced the concept of Single Display Privacyware, where a shared display supports contextually placed private output along with publically shared information [66]. The prototype they created involved giving users stereoscopic glasses where each lens was synced to show either the even-numbered or odd-numbered frames, depending on the user. By doing this they could provide private information for one user on the even frames, and for a second user on the odd frames. This technique, however, does not scale for many more than two users and requires the purchase and setup of stereooscopic glasses for each user.

Cao, Olivier, and Jackson introduced techniques for using crossmodal feedback to provide private cues that augment public information on a shared display [25]. Relevant information would appear on the displays and users would be notified privately, possibly through a vibration on a mobile device, that what was displayed or highlighted on screen was related to their query. In this way, all information was publicly visible, but relevance of the information was private. The benefit of this approach was that by taking advantage of multiple modalities, users could remain focused on the screen and consequently have reduced cognitive load for their task. This mechanism could be adapted for our uses, but would require output synchronization between the devices, as well as devices sophisticated enough to provide auditory or haptic feedback. Currently i>clickers support neither of these capabilities.
9.2.2 Perception

Much prior research has been done to demonstrate the difference between abrupt onset presentation, gradual onset presentation, and abrupt no-onset presentation. Here, *abrupt* means that a sudden visual change takes place, *gradual* means that the visual change takes place over time, and *abrupt no-onset* means that there is a sudden visual change to the camouflage. Todd and Van Gelder showed that abrupt no-onset presentations had slower reaction times than abrupt onset presentations [73]. Yantis further refined these results by demonstrating there was no difference between gradual and abrupt no-onset presentations when compared to abrupt onset presentations, with results suggesting that abrupt onset automatically captured attention but neither no-onset presentations did [77]. This suggests that by making use of a no-onset technique similar to what Yantis used we can provide feedback without drawing the attention of onlookers, but those who are already paying attention will be able to see the change immediately.

Atkinson et al. showed that reaction times increase approximately linearly with the number of items being displayed [13], suggesting that the more avatars that are present on screen, the more difficult it will be to track down the one that has changed to displaying feedback. However, by directing attention to locations on screen in preparation of a display, processing time can be reduced [43], allowing a higher probability of reading the change in the target at high speeds.

Studies have shown that people cannot attend to multiple noncontiguous regions simultaneously [42, 60], furthering the expectation that it will be difficult to interpret multiple users’ feedback at the same time if they are spaced sufficiently far apart spatially, and that the difficulty increases with separation distance.

However, Treisman and Gelade showed that if the targets differ substantially from nontargets in the display, they will be easily detected regardless of the number of nontargets [75]. This suggests that users will be able to easily identify that multiple targets have been provided feedback, because they will differ clearly from those that have not, but they may not be able
to interpret the feedback shown across the different targets. Interpreting the feedback will be especially difficult if it is only shown briefly, because typical non-preattentive processing rates are more than 40ms per item [74].

Broadbent suggested that abrupt onsets may increase the perceptual intake of information from a sensory region while decreasing it elsewhere [20]. This may lead to a higher distraction level should other targets display feedback with abrupt onsets. However, Yantis later showed that if attention is focused on a location in advance, abrupt onsets do not capture attention [78]. That is, people can voluntarily control their attention despite abrupt onsets taking place elsewhere on screen.

Yantis also demonstrated evidence for a model of perception that requires that processing of single onset objects be done one at a time, handling the attended-to location first and then serially scanning the rest in a random order [77]. With only two active displays (target and distractor), the distractor should be found next, but may not be able to be seen at the same time as the target. Eriksen and Hoffman showed that the focus of attention is roughly 1 degree of visual angle [33]. Yantis’ work was done with a 5.7 degree viewing angle between the point of fixation and the targets of interest, which may limit how this finding applies in our context where the displays are between 1.7 and 4.8 degrees apart. If the targets are closer together, it’s not clear whether or not they will be able to be seen simultaneously.

9.3 Experiment 1: No-Onset Display

Our goal was to evaluate the effectiveness of a display technique that allows a user to successfully interpret feedback displayed on screen without being able to interpret the feedback shown simultaneously for other users. We wanted to find a display duration that provided these properties without being too difficult or uncomfortable to use. We chose to use the durations 400ms (long), 80ms (medium), and 16ms (short) after a period of piloting the technique.

We also wanted a technique that provided a high degree of target accuracy, allowing the user to correctly guess the target letter 95% of the time or
better, while limiting distractor accuracy to near random chance (20%). We expect that when users are mistaken in applied uses of this technique, the course of remediation will be simply to press the clicker button again and then interpret the feedback once again, a relatively low cost activity that justifies allowing a 5% error rate.

9.3.1 Participants

We recruited 24 participants (11 female) between 19 and 29 years of age, all of whom were compensated $20 for their efforts. All participants had normal or corrected to normal vision and were not colour-blind. The participants were recruited via mailing lists at our university and through in-lecture announcements of the study.

9.3.2 Apparatus

The experiment was conducted using a MacBook Pro computer (2GHz Intel Core i7, 8GB 1333MHz DDR3 RAM, AMD Radeon HD 6490M video card) running Mac OS X 10.9.3, with a 27” LCD monitor having a resolution of 2560x1440. The monitor had a 60Hz refresh rate, limiting the fastest possible reveal time to 16ms, which was used. The experiment was programmed in javascript as an application in the Rhombus Classroom Synchronous Participation System. The web browser Firefox 29 was used to run the experiment. Participants entered their responses for the task via an i>clicker remote. The i>clicker base station model TMX14 was used. The avatars were square, 1.51° of visual angle (134px, 3.0cm) in width, spaced at 0.15° (12px, 0.3cm) apart from one another. The 7-segment displays were 0.60° (52px, 1.2cm) tall and 0.40° (35px, 0.8cm) wide. Participants sat approximately 114cm away from the monitor. These are shown in Table 9.1.

9.3.3 Task

We used a 7-segment display to reveal the letters A to E to participants, using the formulations shown in Figure 9.2. We used these letter forms because they provided a high degree of distinguishability between the letters,
Table 9.1: The visual angles, and pixel and physical dimensions for the elements of the avatars used in the experiment.

<table>
<thead>
<tr>
<th>Element</th>
<th>Visual Angle</th>
<th>Pixels</th>
<th>Physical</th>
</tr>
</thead>
<tbody>
<tr>
<td>avatar height</td>
<td>1.51°</td>
<td>134px</td>
<td>3.0cm</td>
</tr>
<tr>
<td>avatar width</td>
<td>1.51°</td>
<td>134px</td>
<td>3.0cm</td>
</tr>
<tr>
<td>avatar spacing</td>
<td>0.15°</td>
<td>12px</td>
<td>0.3cm</td>
</tr>
<tr>
<td>7-segment height</td>
<td>0.60°</td>
<td>52px</td>
<td>1.2cm</td>
</tr>
<tr>
<td>7-segment width</td>
<td>0.40°</td>
<td>35px</td>
<td>0.8cm</td>
</tr>
</tbody>
</table>

and because displaying each required some change from the default state of the display that had all segments visible.

Figure 9.2: The letters A to E as shown in the 7-segment display.

The 7-segment displays were located in the lower right corner of an avatar, as described earlier (Figure 9.1). Each trial used 25 avatars arranged in a 5x5 grid (Figure 9.3), with the target avatar always located in the middle and labelled with the alias “you”.

At the start of each trial, the screen was blank and the avatars faded in over 800ms. The user’s avatar was always the same, but the other avatars’ aliases changed randomly for each trial. All fades or animations used jQuery’s swing easing functionality: they progress more slowly at the start and at the end than in the middle [48]. In this initial state, every avatar’s 7-segment display had all segments visible. Immediately after fading in completely, the target avatar’s 7-segment display briefly showed a single letter from A to E then returned to having all segments visible. Simultaneously, a single other avatar, known as the distractor, similarly showed a letter from A to E (Figure 9.3). When segments disappear and reappear from the 7-segment displays, that happens immediately, without animation, because prior research has shown that having a gradual offset of the segments made
Figure 9.3: A sample 5x5 grid of avatars. Here the target letter is A and the distractor, “bruce”, reveals letter E.

no difference in the attentional demand the change produced \[77\] compared to an abrupt offset, and we wanted to test our feedback technique using as fast a speed as possible.

After the letters had been revealed and then hidden, the system would wait for the participant to press a button on the \(i>\text{clicker}\) corresponding to the letter seen on the target avatar. There was no time limit on how long the user could delay before pressing the button. Upon pressing the button, visual feedback was provided with the target avatar animating to have a green overlay if they guessed correctly, and red otherwise. This colour feedback persisted for 750ms. The \(i>\text{clickers}\) used for inputting choices had physical limitations as to how quickly they could send new button presses, which required a delay of approximately 500ms. The feedback duration was set to
ensure that after having faded, the clicker would be able to send another button press.

At this point, the system again waits for the participant to press a button, this time indicating the letter seen on the distractor’s avatar. Again, the participants had no time limit within which to press the button so they could delay if they chose to. No visual feedback was given as to whether they got the distractor correct. After receiving the button press for the distractor’s letter, all avatars would fade out over 400ms and the screen would remain blank for 600ms before the fade in of the next trial would automatically begin. The overall process the target avatar goes through in a single trial is depicted in Figure 9.4.

![Figure 9.4: The sequence the target avatar moves through in each trial: fade in, reveal letter, hide letter, accuracy feedback, fade out.](image)

### 9.3.4 Procedure

The experiment was designed to take at most 90 minutes, with most participants finishing it in approximately one hour. Upon arriving, participants were verbally instructed from a script in how the experiment would proceed and asked to sign a consent form, which was provided 24 hours in advance.
The instructions asked participants to perform the tasks as accurately as they could and to prioritize getting the target correct over the distractor, but to try to get both correct when possible.

Next, participants filled in a brief demographic questionnaire and then completed a warm-up session where they had to complete at least ten trials of the experiment with both target duration and distractor duration at the long setting, and also confirm that they understood how the experiment would proceed before moving on.

After completing the warmup, participants confirmed their understanding of the experiment and then completed three blocks of trials. In each block, participants had to complete 144 trials with a fixed target duration, which took approximately 10 to 15 minutes. The first block used the long duration (400ms), the second the medium duration (80ms), and the third block used the short duration (16ms). The participant controlled starting a new block by pressing any button on the *clicker*, which would present the first trial. The button presses required to complete a trial automatically started the next trial until the block was complete. At this point, the participant was informed that the block was complete and all subsequent clicks were ignored for 1 minute. This served to prevent a participant from accidentally starting the next block and also to enforce a minimum break time between blocks. Between the blocks, a brief questionnaire was administered to collect difficulty ratings of detecting the target and distractor letters, confidence in interpreting the target, comfort during the block, as well as to note any strategies that may have been used. Participants had the opportunity to take a break between blocks.

After completing all blocks, participants filled in a final questionnaire that allowed them to provide additional open-ended comments about their experience.

### 9.3.5 Design

The experiment used a within-subjects 3 (target duration) × 3 (distractor duration) × 16 (distractor location) design. Each combination of these three
factors was seen 3 times by each participant, making for a total of 432 trials. The target duration was fixed during blocks of 144 trials, but the distractor duration and location varied from trial to trial.

The distractor durations and locations for each trial within sub-blocks of 16 trials were permuted in such a way that each location was only used once in the 16 trials and two adjacent trials always had different durations and locations, even across sub-blocks. The 16 distractor locations are depicted in Figure 9.5, where they are further classified as being in either the inner or outer ring to simplify analysis.

![Figure 9.5: The locations where distractors can show up in. Dark gray signifies a distractor location in the “outer ring” and light gray signifies a distractor location in the “inner ring”. The black square in the center is where the target avatar was located in all trials.](image)

Each trial involved two letters of five possible letters: one for the target, and one for the distractor, but there were not enough trials to allow for full balancing of these factors. We instead ensured that in each 144 trial block, four letters were used 29 times and the remaining letter was used 28 times for target, and again for the distractor. The letters were permuted with a constraint that no letter repeat itself more than two times in a row.

### 9.3.6 Measures

The primary dependent variables were target accuracy and distractor accuracy, determined by pressing first the i>clicker button that matched the letter shown on the target avatar and then the button that matched the
distractor avatar. We also assessed comfort, confidence, difficulty of target, difficulty of distractor, and the degree of distraction via questionnaires that were completed after each block. All ratings from the questionnaires were self-reported and scored on a 5-point Likert-style scale.

9.3.7 Hypotheses

The following hypotheses were formed prior to running the experiment:

H1. Target accuracy will be higher the longer the visible target duration, with greater than 95% accuracy on the long and medium durations, and worse than 75% on the short duration.

H2. Distractor accuracy will be higher the longer the visible distractor duration, with greater than 50% accuracy on the long duration, and worse than 25% accuracy on short and medium durations.

H3. The shorter the target duration, the less comfortable the participant will feel: comfortable during long and medium target durations, but uncomfortable during the short duration.

H4. Target duration will have no impact on the degree of distraction caused by the distractor.

H5. Distractor accuracy will be higher when located closer to the target than when located farther from the target.

9.3.8 Results

For all one-way repeated measures ANOVAs, if sphericity was violated we applied a Greenhouse-Geisser adjustment, signified by * on the p-value. For all post-hoc comparisons, the Bonferroni method was used to adjust p-values. We report effect size for ANOVAs using generalized eta-squared ($\eta_g^2$) [14], and Cohen’s criteria that .02 is a small effect, .13 is medium, and .26 is large [26].
Target accuracy

We used a 3 (target duration: long, medium, short) × 3 (distractor duration: long, medium, short) × 2 (distractor location: inner ring, outer ring) repeated-measures ANOVA to analyze the results on target accuracy. The ANOVA revealed both effects and two-way interactions.

The ANOVA revealed a main effect of target duration on target accuracy ($F(1,05,24.14) = 43.02, \ p < .001^*, \ \eta^2_g = .460$). The longer the target duration, the higher the target accuracy. Post-hoc pairwise comparison $t$-tests of target duration on target accuracy revealed significant differences between all durations ($p < .001$). There were a total of 144 trials per target duration with mean accuracy: long 95.6% ($m = 137.7, sd = 2.7\%$), medium 92.5% ($m = 133.2, sd = 4.6\%$), short 70.5% ($m = 101.5, sd = 18.2\%$).

The ANOVA revealed a main of distractor duration on target accuracy ($F(2,46) = 8.95, \ p < .001, \ \eta^2_g = .007$). When the distractor duration was long, target accuracy was worse than when the distractor duration was medium or short. Post-hoc pairwise comparison $t$-tests of distractor duration on target accuracy revealed significant differences between long and short durations ($p < .05$), and long and medium ($p < .001$), but not between short and medium ($p = .55$). There were a total of 144 trials per distractor duration with mean accuracy: long 84.9% ($m = 122.2, sd = 7.5\%$), medium 87.2% ($m = 125.7, sd = 7.3\%$), short 86.5% ($m = 124.6, sd = 7.1\%$).

The ANOVA revealed a main of distractor location on target accuracy ($F(1,23) = 15.49, \ p < .001, \ \eta^2_g = .012$). When distractors were located in the outer ring, target accuracy is better than when distractors are in the inner ring. There were a total of 216 trials per distractor location (inner/outer) with mean accuracy: inner 84.9% ($m = 183.4, sd = 7.7\%$), outer 87.5% ($m = 189.1, sd = 6.8\%$).

There was a two-way interaction of target duration and distractor duration on target accuracy ($F(4,92) = 4.329, \ p < .01, \ \eta^2_g = 0.008$), indicating that target accuracy was not affected by distractor duration when target duration was long.

There was also a two-way interaction of target duration and distractor
location on target accuracy \((F(2, 46) = 4.234, p < .05, \eta^2_p = 0.004)\), with target accuracy being better when target duration was short and distractor location was in the outer ring than when located in the inner ring. In the medium and long target durations, distractor location had no effect on target accuracy.

There was no interaction between distractor duration and distractor location, nor a three-way interaction between target duration, distractor duration and distractor location.

**Distractor accuracy**

We similarly used a 3 (target duration: long, medium, short) \(\times\) 3 (distractor duration: long, medium, short) \(\times\) 2 (distractor location: inner ring, outer ring) repeated-measures ANOVA to analyze the results on distractor accuracy. The ANOVA revealed main effects and two-way interactions.

The ANOVA revealed a main effect of target duration on distractor accuracy \((F(2, 46) = 83.6, p < .001, \eta^2_p = .291)\). The longer the target duration, the higher the distractor accuracy. Post-hoc pairwise comparison \(t\)-tests of target duration on distractor accuracy revealed significant differences between all durations \((p < .001)\). There were a total of 144 trials per target duration with mean accuracy: long 43.3\% \((m = 62.3, sd = 4.2\%\)), medium 35.8\% \((m = 51.6, sd = 6.5\%\)), short 25.2\% \((m = 36.3, sd = 6.3\%\)).

The ANOVA revealed a main effect of distractor duration on distractor accuracy \((F(1.46, 33.5) = 266.1, p < .001^*, \eta^2_p = .729)\). The longer the distractor duration, the higher the distractor accuracy. Post-hoc pairwise comparison \(t\)-tests of distractor duration on distractor accuracy revealed significant differences between all durations \((p < .001)\). There were a total of 144 trials per distractor duration with mean accuracy: long 61.4\% \((m = 88.5, sd = 10.6\%\)), medium 24.4\% \((m = 35.2, sd = 4.5\%\)), short 18.5\% \((m = 26.6, sd = 4.1\%\)).

The ANOVA revealed a main effect of distractor location on distractor accuracy \((F(1, 23) = 15.49, p < .001, \eta^2_p = .012)\). When distractors were located in the inner ring, distractor accuracy was better than when distractors...
were in the outer ring. There were a total of 216 trials per distractor location (inner/outer) with mean accuracy: inner 36.8% ($m = 79.5, sd = 5.6$%), outer 32.7% ($m = 70.7, sd = 3.9$%)

There was a two-way interaction of target duration and distractor duration on distractor accuracy ($F(2.56,58.91) = 53.55, p < .001^*, \eta_g^2 = 0.364$), indicating that distractor accuracy was not affected by target duration when distractor duration was short.

There was a two-way interaction of target duration and distractor location on distractor accuracy ($F(2,46) = 5.59, p < .01, \eta_g^2 = 0.010$), with distractor location only having an effect on distractor accuracy when target duration was medium. In this case, distractor correctness was higher when located in the inner ring than when in the outer ring.

There was a two-way interaction of distractor duration and distractor location on distractor accuracy ($F(2,46) = 24.57, p < .001, \eta_g^2 = 0.072$), indicating that distractor accuracy was better when the distractor duration was long and the distractor location was in the inner ring than in the outer ring, and that distractor accuracy was not affected by distractor location when distractor duration was medium or short.

There was no three-way interaction between target duration, distractor duration and distractor location.

**Matched Target and Distractor Durations**

Trials had a mix of durations for target and the distractor. In a real application of the technique the durations would be the same for targets and distractors because one user’s distractor would be another user’s target (and vice-versa). Restricting the data to only include trials where target and distractor durations were the same, the means for target accuracy were: long 95.8% ($m = 46.0, sd = 4.4$%), medium 94.2% ($m = 45.2, sd = 4.2$%), short 71.0% ($m = 34.1, sd = 18.3$%). The means for distractor accuracy were: long 83.2% ($m = 40.0, sd = 8.8$%), medium 22.5% ($m = 10.8, sd = 7.5$%), short 18.1% ($m = 8.71, sd = 5.8$%).
Letters

Combining the 5 letter choices with the 3 levels of target duration, 3 levels of distractor duration, and 16 distractor locations would require 720 trials in the experiment to cover each condition once, which is more than the 432 trials that were completed, so including target letter as a factor in the main ANOVA was not done because there would be many combinations of letters and conditions that were not tested. Instead, we tested the target letter in isolation. A one-way repeated measures ANOVA revealed a significant effect of letter on target accuracy ($F(2.97, 68.38) = 15.59, p < .001^*, \eta^2_g = 0.20$). Post-hoc pairwise comparison t-tests showed A was significantly easier than C, D, and E ($p < .001$), but not easier than B ($p = .095$), and that E was significantly harder than A ($p < .001$), B ($p < .01$), and D ($p < .001$), but not C ($p = .57$). The means for target accuracy for each letter were: A 92.8% ($sd = 5.8\%$), B 87.7% ($sd = 10.0\%$), C 83.7% ($sd = 9.9\%$), D 87.4% ($sd = 7.6\%$), E 79.4% ($sd = 10.7\%$).
Questionnaire
The data from the questionnaires that were given to participants after each block of the experiment were analyzed using Friedman tests. All questions were Likert-formatted on a 5-point scale. We report effect size as $r$ and use Cohen’s criteria that a value of .1 is a small effect size, .3 is medium, and .5 is large [26].

How difficult was the target? (BQ1) A Friedman test revealed an effect of target duration on self-assessed difficulty in interpreting the target letter ($\chi^2(2) = 37.76, p < .001$). Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between short and medium ($p < .001, r = .60$), short and long ($p < .001, r = .62$), and long and medium ($p < .01, r = .46$). In this question, ratings ranged from 1, very difficult, to 5, very easy. Participants had the most difficulty during the short duration, and least in the long duration, with medians: long 4, medium 4, short 2.

How difficult was the distractor? (BQ2) A Friedman test revealed an effect of target duration on self-assessed difficulty in interpreting the distractor’s letter ($\chi^2(2) = 23.26, p < .001$). Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between short and long ($p < .001, r = .58$), and long and medium ($p < .01, r = .44$), but not between short and medium ($p = .105$). In this question, ratings ranged from 1, very difficult, to 5, very easy. Participants had the most difficulty during the short duration, and least in the long duration, with medians: long 2, medium 1, short 1.

How distracting was the distractor? (BQ3) A Friedman test revealed no significant difference in self-assessments of how distracting the distractor was across different target durations ($\chi^2(2) = 1.41, p = .494$). In this question, ratings ranged from 1, no effect, to 5, very distracting. Participants found the distractor had little impact on their ability to interpret their own avatar’s letters, with medians: long 2, medium 2, short 2.

How confident were you? (BQ4) A Friedman test revealed an effect of target duration on self-assessed confidence in interpreting the target letter
Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between short and medium ($p < .001, r = .52$), short and long ($p < .001, r = .62$), and long and medium ($p < .001, r = .52$). In this question, ratings ranged from 1, very doubtful, to 5, very confident. Participants were most confident during the long duration, and least in the short duration, with medians: long 4, medium 3, short 2.

How comfortable did you feel? (BQ5) A Friedman test revealed an effect of target duration on self-assessed level of comfort ($\chi^2(2) = 24.4, p < .001$). Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between short and medium ($p < .01, r = .47$), short and long ($p < .001, r = .57$), and long and medium ($p < .05, r = .43$). In this question, ratings ranged from 1, very uncomfortable, to 5, very comfortable. Participants were most comfortable during the long duration, and least in the short duration, with medians: long 4, medium 3, short 2.

Strategy. (BQ6) We asked participants to write down any strategy they employed in the experimental blocks. In the long duration block, 15 participants mentioned primarily focusing on their avatar, and 5 participants described using a wider range of focus. For example, P07 said, “Not to focus on my avatar but spread the sight focus as wide as possible.”

In the medium duration block, 15 participants continued on with the strategy that they used in the long block. One participant, P09 switched from taking a wider view to taking a more focused look at the target, saying “Only focusing on the avatar, no way to get the distractor, attempting only causes me to miss everything!” Four participants began interpreting the letter feedback at the segment level, a strategy that was crucial for the short block. P04 described the strategy as, “I wasn’t reading the letters anymore; just looking for which segments disappear or even less, which part of the 8 disappears.”

In the short duration block, the strategy of interpreting the flickering segments was widely adopted, with 18 participants describing it explicitly. For example, P22 said, “Could only watch for the flicker and subtract the negative space from the 8 to determine the expected letter. Did not watch for distractor at all.” Five participants did not mention using this strategy, and
instead tried to see the letter the best they could. P19 said, “I closed my eyes after the flash and hope for the flash to appear.”

The mean target accuracy on the short block was 70.5%. If we separate the accuracy scores based on those who used the missing segment strategy from those who did not, the mean accuracy goes up to 76.5% for those that did use the strategy, and down to 42.6% for those that did not.

9.3.9 Discussion

Our findings indicate that displaying simple letter feedback to users by briefly displaying the letter on an avatar is a viable method of presentation.

\( H1 \) was partially supported. Participants had little difficulty getting the target’s value accurately during the long and medium durations, with mean accuracy ratings 95.6% and 92.5% respectively, while in the short duration accuracy dropped to 70.5%. The accuracy of the medium duration fell slightly short of the hypothesized 95% mark. It was common for users to mention accidentally hitting the wrong button or entering the target and distractor letters in reverse order, which may explain the lower accuracy. On trials where both target and distractor duration were the same, long accuracy increased to 95.8% and medium accuracy to 94.2%, closer to our expected threshold and a more realistic estimate of performance in actual usage when target and distractor duration would always be identical because of their symmetry.

Several participants surprised us with their target accuracy on the short duration, with seven participants scoring above 85%, four of whom were over 90%. Those that had already adopted the strategy of interpreting the flashing segments to determine the letter prior to the start of the short block tended to score higher, while some participants only figured out the strategy part way through. Some did not figure out the strategy, with four participants scoring under 50%, one as low as 30.6%. This variance explains the relatively large standard deviation seen in target accuracy in the short duration block.

\( H2 \) was supported. Distractor accuracy matched our expected thresh-
olds. However, we suspected the means for the distractors may have been artificially lowered due to the final block of the experiment when the target duration was shortest. In this block, participants had to focus entirely on the avatar, almost completely giving up on looking for the distractor at all. If we exclude this block from the calculation of means across distractor accuracy by distractor duration, we get: long 74.3% (+12.8%), medium 25.7% (+1.3%), short 18.6% (+0.1%). Furthermore, if we compare the means for when the distractor and target durations were both long, the distractor accuracy increases to 83.2% (+21.7%). Given this, we advise against making use of the long duration for targets and distractors if it is important to minimize the ability to interpret other avatars’ feedback.

For the medium target duration, the means were target accuracy 92.5%, distractor accuracy 35.8%. Reducing the data set to only consider trials where the target and distractor duration were both medium, the means were target accuracy 94.1%, distractor accuracy 22.5%. This duration thus provides a highly accurate target presentation, while keeping distractor accuracy close to random chance (20%).

\(H3\) was supported. Although over time, all users could be taught the strategy needed to handle the short duration, users found it the least comfortable of all the durations. Given this result, we advise against the short duration as a viable speed for application use.

\(H4\) was supported. Results from the post-block questionnaires indicated that participants did not find the distractor made it more difficult to interpret their own avatar, regardless of target duration.

\(H5\) was supported. While results indicated distractor accuracy increased when the distractor was located closer to the target, we did not expect this to be at a cost of target accuracy. Our findings indicate that when the distractor had longer duration or was closer to the target, target accuracy decreased. We suspect this is due to an increase in distraction. Participants may think they have a good chance of getting the distractor on one of those trials, and so they perhaps attempt to interpret it prior to solidifying their comprehension of the target’s letter. Notably, the effect sizes (.007 for distractor duration and .012 for location) were below the small threshold put forth by Cohen
(.02) [26], so it may be that these effects can be ignored.

We found that there was an effect on target accuracy of letter, with a medium effect size (.2). After the experiment had concluded, several participants commented on how some letters were harder than others, with A being the easiest, which is consistent with our results. Many mentioned difficulty in accidentally swapping the meaning of b and d, and especially in discerning the difference between C and E in the short target duration block. With the other letters, it was possible to easily determine which was shown because of the locations where segments were missing, but the overlap between C and E was too similar to confidently distinguish. This is likely a drawback of the 7-segment display that we used. We had thought that using a 7-segment display would be familiar to participants, but instead it brought issues with inconsistencies in capitalization (b and d were lower case while A, C, and E were upper case), and we suspect the unnatural formations of the letters increased the time required to interpret what was shown. The same camouflage effect provided by the fully-lit figure-8 could be accomplished using more segments to provide more natural looking characters. It may even work by simply stacking all five characters overlaid on one another and simply fading out all but the active letter when feedback is presented.

9.4 Experiment 2: Abrupt Onset Display

After completing Experiment 1, we wanted to verify that using the 7-segment display in the way that we had done, with no-onset techniques (a full figure-8, followed by the letter, followed by the full figure-8), was the reason we could provide such a strong difference in target and distractor accuracy while keeping distraction levels low. To test this, we ran the experiment again with the variation being that the letter displays would use abrupt onset by beginning blank, displaying the letter, then returning to being blank.

9.4.1 Participants

We recruited 12 participants (7 male) between 20 and 29 years of age, all of whom were compensated $20 for their efforts. This experiment used the
same recruitment methods and inclusion criteria as Experiment 1. None of these participants were participants in the first experiment.

9.4.2 Apparatus, Procedure, and Design
This experiment used the same apparatus, procedure, and design as Experiment 1.

9.4.3 Task
The task in this experiment was very similar to that of Experiment 1, with the only change being in how the letters were displayed on avatars. Instead of having a full 7-segment display, flashing the letter, then showing the full 7-segment display, as in Experiment 1, we had the 7-segment display be blank before and after showing the letter, as depicted in Figure 9.7. In the 5x5 grid, no 7-segment displays were visible except those of the target and distractor avatars for the brief moment they simultaneously displayed their letters.

![Figure 9.7: The sequence the target avatar moves through in each trial in Experiment 2: fade in, reveal letter, hide letter, accuracy feedback, fade out.](image)
9.4.4 Results

For all post-hoc comparisons, the Bonferroni method was used to adjust $p$-values. We again report effect size using generalized eta-squared ($\eta^2_g$) [14], where .02 is a small effect, .13 is medium, and .26 is large [26].

**Target accuracy**

We again used a 3 (target duration: long, medium, short) $\times$ 3 (distractor duration: long, medium, short) $\times$ 2 (distractor location: inner ring, outer ring) repeated-measures ANOVA to analyze the results on target accuracy. The ANOVA revealed no interactions amongst target duration, distractor duration, and distractor location on target accuracy.

The ANOVA revealed a main effect of target duration on target accuracy ($F(2, 22) = 30.04, p < .001^*, \eta^2_g = .273$). Target accuracy was lower than otherwise when the target duration was short, but under all durations target accuracy was still over 90%. Post-hoc pairwise comparison $t$-tests of target duration on target accuracy revealed significant differences between short and medium, and short and long ($p < .001$), but not between medium and long ($p = 1.0$). There were a total of 144 trials per target duration with mean accuracy: long 97.7% ($m = 140.8, sd = 1.3\%$), medium 98.0% ($m = 141.2, sd = 1.2\%$), short 93.1% ($m = 101.5, sd = 2.5\%$).

The ANOVA revealed no main effect of distractor duration on target accuracy ($F(2, 22) = 1.13, p = .340$). There were a total of 144 trials per distractor duration with mean accuracy: long 95.7% ($m = 137.8, sd = 1.9\%$), medium 96.4% ($m = 138.8, sd = 1.3\%$), short 96.8% ($m = 139.3, sd = 2.0\%$).

The ANOVA revealed no main effect of distractor location on target accuracy ($F(1, 11) = 4.30, p = .062$). There were a total of 216 trials per distractor location (inner/outer) with mean accuracy: inner 95.99% ($m = 207.2, sd = 1.3\%$), outer 96.6% ($m = 208.8, sd = 1.2\%$).

**Distractor accuracy**

We again used a 3 (target duration: long, medium, short) $\times$ 3 (distractor duration: long, medium, short) $\times$ 2 (distractor location: inner ring, outer ring)
repeated-measures ANOVA to analyze the results on distractor accuracy. The ANOVA revealed a main effect of target duration on distractor accuracy ($F(2, 22) = 3.80, p < .05, \eta^2_g = .021$). Post-hoc pairwise comparison $t$-tests of distractor duration on distractor accuracy revealed no significant differences between durations, but suggested that medium was trending to be better than long ($p = .11$). There were a total of 144 trials per target duration with mean accuracy: long 71.5% ($m = 102.9, sd = 7.8\%$), medium 75.2% ($m = 108.2, sd = 5.9\%$), short 74.1% ($m = 106.8, sd = 8.0\%$).

The ANOVA revealed a main effect of distractor duration on distractor accuracy ($F(2, 22) = 188.74, p < .001, \eta^2_g = .780$). The longer the distractor duration, the higher the distractor accuracy. Post-hoc pairwise comparison $t$-tests of distractor duration on distractor accuracy revealed significant differences between all durations ($p < .001$). There were a total of 144 trials per distractor duration with mean accuracy: long 95.8% ($m = 137.9, sd = 2.1\%$), medium 78.1% ($m = 112.5, sd = 8.9\%$), short 46.9% ($m = 67.5, sd = 11.5\%$).
The ANOVA revealed a main effect of distractor location on distractor accuracy (\(F(1, 11) = 173.52, p < .001, \eta^2_g = .373\)). When distractors are located in the inner ring, distractor accuracy is better than when distractors are in the outer ring. There were a total of 216 trials per distractor location (inner/outer) with mean accuracy: inner 81.9% (\(m = 176.8, sd = 6.5\%\)), outer 65.3% (\(m = 141.1, sd = 7.6\%\)).

There was an interaction of target duration, distractor duration and distractor location on distractor accuracy (\(F(4, 44) = 3.53, p < .05, \eta^2_g = 0.023\)), indicating that distractor accuracy was only affected by target duration when both distractor duration was medium and distractor location was in the outer ring.

There was an interaction of target duration and distractor duration on distractor accuracy (\(F(4, 44) = 2.79, p < .05, \eta^2_g = 0.029\)), but post-hoc tests revealed no significant differences among pairs.

There was an interaction of distractor duration and distractor location on distractor accuracy (\(F(2, 46) = 24.57, p < .001, \eta^2_g = 0.072\)), indicating that distractor accuracy was unaffected by distractor location when the distractor duration was long. When the duration was medium or slow, distractor accuracy was higher when the location was in the inner ring than in the outer ring.

There was no interaction of target duration and distractor location on distractor accuracy (\(F(2, 22) = 2.76, p = .085\)).

**Letters**

As in Experiment 1, we tested the effect of target letter in isolation due to a lack of sufficient trials across all conditions. A one-way repeated measures ANOVA showed a significant main effect of letter on target accuracy (\(F(4, 44) = 3.65, p < .05, \eta^2_g = 0.23\)). Post-hoc pairwise comparison \(t\)-tests of letter on target accuracy revealed no significant differences between letters, but suggested D was trending to be worse than A (\(p = .16\)), B (\(p = .23\)), and C (\(p = .35\)). The means for each letter were: A 97.9% (\(sd = 1.6\%\)), B 97.0% (\(sd = 2.2\%\)), C 97.6% (\(sd = 2.1\%\)), D 93.7% (\(sd = 4.5\%\)), E 95.3%
Questionnaire

The following results use the effect size measurement \( r \) where a value of .1 is small, .3 is medium, and .5 is large [26]. These results are analyzed from the questionnaires that were given to participants after each block of the experiment. All questions were Likert-formatted on a 5-point scale.

**How difficult was the target?** (BQ1) A Friedman test revealed an effect of target duration on self-assessed difficulty in interpreting the target letter \( (\chi^2(2) = 18.93, p < .001) \). Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between short and medium \( (p < .01, r = .63) \), and short and long \( (p < .01, r = .63) \), but not between medium and long \( (p = .33) \). In this question, ratings ranged from 1, very difficult, to 5, very easy. Participants had the most difficulty during the short duration, and least in the long duration, with medians: long 4.5, medium 4, short 2.

**How difficult was the distractor?** (BQ2) A Friedman test revealed an effect of target duration on self-assessed difficulty in interpreting the distractor’s letter \( (\chi^2(2) = 8.71, p < .05) \). Post-hoc pairwise Wilcoxon Signed-Rank tests showed no significant differences between pairs of durations. In this question, ratings ranged from 1, very difficult, to 5, very easy. The median ratings were: long 2, medium 2, short 1.5.

**How distracting was the distractor?** (BQ3) A Friedman test revealed a significant difference in self-assessments of how distracting the distractor was across different target durations \( (\chi^2(2) = 9.63, p < .01) \). Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between short and long \( (p < .05, r = .55) \), but not between short and medium \( (p = .80) \) or medium and long \( (p = .33) \). In this question, ratings ranged from 1, no effect, to 5, very distracting. Participants found the distractor had moderate impact on their ability to interpret their own avatar’s letters, with medians: long 2, medium 3, short 3.

**How confident were you?** (BQ4) A Friedman test revealed an effect
of target duration on self-assessed confidence in interpreting the target letter ($\chi^2(2) = 10.89, p < .01$). Post-hoc pairwise Wilcoxon Signed-Rank tests showed that there was a significant difference between short and long ($p < .05, r = .59$), but not between short and medium ($p = .19$), or medium and long ($p = .36$). In this question, ratings ranged from 1, very doubtful, to 5, very confident. Participants were most confident during the long duration, and least in the short duration, with medians: long 4, medium 4, short 2.

**How comfortable did you feel?** (BQ5) A Friedman test revealed an effect of target duration on self-assessed level of comfort ($\chi^2(2) = 7.33, p < .05$). Post-hoc pairwise Wilcoxon Signed-Rank tests revealed no significant differences across target durations. In this question, ratings ranged from 1, very uncomfortable, to 5, very comfortable. The median ratings were: long 4, medium 4, short 3.5.

**Strategy.** (BQ6) We asked participants to write down any strategy they employed in the experimental blocks. In the long duration block, 7 participants mentioned primarily focusing on their avatar. For example P12 said, "Be alert - look at my own avatar first." Four participants described using a wider range of focus. For example, P09 said, "I tried to use a softer focus on the center of the screen to both initially see my avatar’s letter and then hopefully jump to the distractor in time to see the letter. That didn’t work most of the time, I think."

In the medium duration block, 8 participants continued on with the strategy that they used in the long block. One participant, P06 switched from taking a wider view to taking a more focused look at the target, saying "This time the avatar’s timer was faster, so it was more important to focus on that first."

In the short duration block, 7 participants again explicitly mentioned continuing with their previously used strategies, which typically involved focusing on the target primarily. For example, P01 said, "Same strategy as before. I focused on my own avatar a little more since it was at a faster speed." There were no common strategies outside of generally concentrating hard on the target.
9.4.5 Discussion

In this experiment, we saw that while there was a significant difference in target accuracy across target duration, all accuracy levels were above 90%, even in the short duration. Participant strategies indicated they were able to see the letters directly in the short duration, which helps explain the high percentage. This is further supported by comparing the means of distractor accuracy by distractor duration across all blocks (long 95.8%, medium 78.1%, short 46.9%) with the same means excluding the short target duration block (long 96.0%, medium 76.6%, short 47.3%). Here we see the means are roughly equal, implying participants had enough time and cognitive resources to focus on and interpret both the target letter and the distractor letter even in the short duration block, in contrast to Experiment 1.

While the questionnaire results suggest that participants found the distractor somewhat distracting (median rating 3 out of 5, higher than 2 in Experiment 1), the actual results show that distractor duration and location had no significant effect on target accuracy. Perhaps it was the case that distractors felt like they were distracting, but reading the letters in the presentation that was given was easy enough to not effect the target results. This explanation is further supported by the result that distractor location did not have a significant effect on distractor accuracy when distractor duration was long. In this case, there was enough time to see both target and distractor, but when distractor duration was short or medium and the distractor was in the inner ring, accuracy increased.

While there was a significant difference found in the self-assessed comfort level participants expressed between experiment blocks, post-hoc tests did not reveal any additional insights into the difference. The median ratings were all relatively close (long 4, medium 4, short 3.5), with all coming above a neutral rating of 3. This suggests that despite the very brief duration seen in the short block, participants did not feel overly taxed by the task.
9.5 General Discussion

In our first experiment, we found a balance between target accuracy, distractor accuracy, and participant comfort in the medium duration block (80ms). Our second experiment confirmed that the no-onset presentation style was the primary reason we were able to approach the desired accuracy levels we observed in Experiment 1: 92.5% for targets, and 24.4% for distractors, roughly the same as random chance (20%). While participants were more comfortable with the abrupt onset presentation used in Experiment 2, the increase in comfort was modest and was overshadowed by the large increase in distractor accuracy.

Furthermore, participants reported a higher degree of distraction incurred by the abrupt onset presentation style. We wanted to limit this for two reasons: we feared it would decrease participants’ ability to interpret the target letter, and it would make it easier to notice another avatar’s feedback without attending to it a priori. The first reason turned out to be a concern: we saw higher distraction ratings and higher target accuracy in Experiment 2 than Experiment 1. The latter reason does seem to be a concern: we saw a large increase in distractor accuracy in the second experiment.

The experiments uncovered issues with using a 7-segment display to present letters, with multiple participants expressing trouble interpreting the letters, especially at high speed. Other sources of errors participants noted were accidentally entering the letters in the wrong order (i.e., distractor before target) or entering a B when they meant D due to the similarity in their lowercase shapes. Issues with 7-segment displays are well documented [72], so perhaps it would be best that future work avoid their usage.

Both experiments were limited by the timing precision afforded by Firefox, within which they were run. While we aimed for roughly 16ms, 80ms, and 400ms of display time, we logged the visible durations with mean values: 23.1ms (sd = 5.2ms), 85.9ms (sd = 5.6ms), and 404.7ms (sd = 3.6ms). Given that we were using a 60Hz monitor, the granularity provided should still have granted us 1 frame in the short duration, 5 frames in the medium duration, and 25 frames in the long duration, despite the variances in actual
9.6 Conclusion and Future Work

We examined a novel technique for displaying feedback in such a way that only those attending to it a priori can reliably interpret the results. To do this, we made use of a no-onset presentation style. We ran two experiments to verify our hypotheses and discovered that the no-onset presentation had significant advantages over abrupt onset presentation when it comes to interpreting a second randomly selected avatar’s feedback, while preserving a high accuracy rating for interpreting your own statically located avatar’s feedback.

Our approach required a trade-off between reliability and privacy. While longer visible durations resulted in higher accuracy levels, they also made it easier to interpret the feedback of another user’s avatar. Shorter visible durations, on the other hand, resulted in higher error rates but reduced the possibility of guessing another user’s feedback to slightly worse than random chance. At a medium length duration (80ms) of feedback reveal time, we found a balance of high reliability and fairly high security with over 90% target accuracy and below 25% distractor accuracy.

We believe that we have demonstrated the efficacy of leveraging perceptual limitations to provide semi-private feedback to users of a shared display. While we used a 7-segment display to present the letters, there were issues with interpretation, so future work might explore more generalized camouflaging of the feedback with no-onset presentations. We would like to explore this further with different experimental protocols, for example, testing only random interpretation of letters, or investigating more than two avatars at once. Other directions for future work include exploring equiluminant texture variations to reveal feedback as opposed to using no-onset presentations.

9.7 Acknowledgments

This research was supported by the GRAND Network of Centres of Excellence under the SHRDSP project and by the Natural Science and Engineering...
Research Council of Canada under the Discovery Grant program. Facilities used for the research in the Institute for Computing, Information and Cognitive Systems at the University of British Columbia were provided by funding from the Canada Foundation for Innovation. We thank Ben Janzen and Francisco Escalona for their assistance in running Experiment 1, and Ron Rensink for his insightful discussions about visual perception.
Chapter 10

Conclusions and Future Work

Despite the rapid expansion of technology in our everyday life, the usage of technology in the classroom has seen very modest changes in recent years. To help facilitate the adoption and creation of rich technological interactive systems in classrooms, we developed an architecture for Classroom Synchronous Participation Systems (CSPS). These systems are designed to allow students to engage in interactive activities while in lecture, getting real-time feedback, which is often a result of their individual actions. The architecture we propose can be implemented on a local level, running on a single user’s computer, or can be scaled to be supported by institutional servers accessible across multiple courses in a university.

We demonstrated the efficacy of our architecture by implementing a system that followed it, called Rhombus Classroom Synchronous Participation System. This system makes use of a suite of servers and web technologies to allow students to use clickers as generic 5-button controllers to applications that can be run in a classroom environment. The Clicker Server was developed to function as a multi-platform standalone interface to stream $i>clicker$ clicks across a network socket, negating the need for any software interested in working with clickers as input from having to interface directly with the hardware. The ID Server stands between the Clicker Server and the Web Server to translate device IDs into more usable names for use in applications. The Web Server receives the clicks from the aforementioned servers, routes
them to the main Controller activated in the Web Framework, and coordinates communication between the main Controller and the various Viewers an application may be using. The Web Framework provides the infrastructure to create stateful applications that can easily associate incoming clicks with individuals, simplifying the creation of interactive classroom activities. The web technologies allowed us to offer displays of applications to any computer connected to the Web Server over the network, without requiring any additional installation on their part.

We developed a novel method of simultaneously and anonymously registering a large group of clickers in the system with the Sequence Aliaser application. By providing each person a sequence of buttons to press, they were able to associate their clicker to a predefined alias that matched the given sequence. This sped up the task of registering clickers in the system, and transformed it from rote administrative work into a fun, engaging activity for participants. This technology should be explored further to learn its limitations, both with regards to number of participants it can support and the types of perceptual cues it requires to assure users they have correctly associated their clickers.

Using Rhombus, we implemented a suite of game-theoretic exercises on it (e.g., Prisoner’s Dilemma, Ultimatum Game, etc.) that were then used multiple times in a third-year Cognitive Systems course during lecture. In the first term of use, Rhombus was very positively received by both the students and the instructor. The students understood the controls and results, enjoyed playing the games, and felt engaged while playing. The instructor was pleased to finally have the ability to give students firsthand experience playing classic games, as opposed to the games he was using in previous offerings that were limited to what he could support with the technology he had available to him, and not ideal for educational purposes.

Encouraged by the results of the first term of use, the instructor took on using Rhombus on his own in the following term. While the majority of students still enjoyed playing the games and felt engaged, the margins were much smaller. The instructor was still very pleased with the system and looked forward to continuing to use it in the future, with hopes to expand
his curriculum of games. He provided several reasons for the change in student attitude this term compared to the previous, including a change of co-instructor, a lack of student incentive, and an initial awkwardness with using the system.

We developed a novel display technique for providing semi-private feedback to users of Rhombus. Our technique gives a balance of high accuracy in interpreting your own feedback, while preserving near-random accuracy in interpreting the feedback of another when that feedback is displayed at the same time as your own.

The results from the classroom evaluation of Rhombus were encouraging, but also revealed areas for future work. There should be usability testing on the instructor interface to the system to ensure less technologically savvy users can easily use the system. This may include improving the web-based instructor controller, as well as the data output, and general system activation controls.

The focus of Rhombus is on providing a system to support more interactivity in the classroom. As such, it is not solely tied to using clickers as input, although they were our initial choice. The architecture supports a wide variety of inputs beyond clickers, such as mobile phones, laptops, tweets and text messages, all of which could be explored to learn how the students behaviour changes depending on their mode of input. Furthermore, the architecture of the system allows for every participant to have their own personal display, which should be explored as it opens a wider variety of interactive applications and feedback mechanisms.

Rhombus has only been used with game-theoretic exercises in a classroom, but we have demonstrated some of its flexibility by conducting two experiments in it (Chapter 9). Future work could explore using the system in other domains, such as community discussions and negotiations for city planning, or algorithmic education (e.g., learning sorting or data structures like linked lists). We foresee the system being useful whenever information displayed on screen is informed by the inputs of a group of people.
Bibliography


135


[56] A. Michotte. The perception of causality. 1963. → pages 21


→ pages [104]
Appendix A

Experiment Resources

This appendix includes supplemental resources that were used in running the experiments described in Chapter 9. The same resources were used for both experiments.

A.1 Pre-experiment Questionnaire

This questionnaire was given to participants prior to taking part in the experiment.
Private Feedback on a Shared Display
Pre-Experiment Questionnaire

1. How old are you?
   ☐ ☐ years

2. What is your gender? (tick one)
   ☐ Male
   ☐ Female
   ☐ Other

3. How much time do you spend per week using a computer? (tick one)
   ☐ Less than 1 hour
   ☐ 1 to 3 hours
   ☐ 4 to 8 hours
   ☐ More than 8 hours

4. How many times have you used an i-clicker? (tick one)
   ☐ Never
   ☐ Once
   ☐ 2 to 5 times
   ☐ More than 5 times

5. Have you previously used or seen Rhombus Participation System? (tick one)
   ☐ Yes
   ☐ No

6. Do you normally wear glasses or contact lenses? (tick one)
   ☐ Yes
      If yes, what is your prescription?
      ☐ ☐
      ☐ I don’t know
   ☐ No
A.2 Post-block Questionnaire

This questionnaire was given to participants after each of the three blocks of the experiment.
1. How difficult was it to interpret your avatar’s letter? (circle one number)
   
   1 - Very Difficult  2 - Difficult  3 - Neutral  4 - Easy  5 - Very Easy

2. How difficult was it to interpret the distractor’s letter? (circle one number)

   1 - Very Difficult  2 - Difficult  3 - Neutral  4 - Easy  5 - Very Easy

3. On a scale from 1 to 5, how distracting was the distractor? With 1 meaning it had no effect and 5 meaning it made it very difficult to interpret your avatar’s letter. (circle one number)

   (No effect)  1  2  3  4  5 (Very Distracting)

4. How confident are you that you could reliably interpret your feedback at this speed?

   1 - Very Doubtful  2 - Doubtful  3 - Neutral  4 - Confident  5 - Very Confident

5. How comfortable did you feel while completing this task?

   1 - Very Uncomfortable  2 - Uncomfortable  3 - Neutral  4 - Comfortable  5 - Very Comfortable

6. Did you employ any particular strategy in completing the task? Please explain.

   ____________________________________________
   ____________________________________________
   ____________________________________________
   ____________________________________________
   ____________________________________________

7. Please write any other comments you have regarding your experience with this task:

   ____________________________________________
   ____________________________________________
   ____________________________________________
   ____________________________________________
   ____________________________________________
A.3 Post-experiment Questionnaire

This questionnaire was given to participants after they had completed all the blocks of the experiment.
Private Feedback on a Shared Display
Post-Experiment Questionnaire

1. Overall, how much did you like using this display technique? (circle one number)
   1 - Really Dislike  2 - Dislike  3 - Neutral  4 - Like  5 - Really Like

2. Please write any other comments you have regarding your overall experience with the display mechanism:

______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________

Version 1.0 2014-May-25
A.4 Consent Form

This consent form was provided to participants prior to running the experiment. All participants that completed the experiment signed the form.
Private Feedback on a Shared Display
UBC Department of Computer Science
ICICS/CS Building
201-2366 Main Mall
Vancouver, B.C., V6T 1Z4

Consent Form

Principal Investigator
Kellogg S. Booth, Professor, Department of Computer Science, (604) 822-8193

Co-Investigator
Peter Beshai, M.Sc. Student, Department of Computer Science, (604) 339-4003

Project Purpose and Procedures
The purpose of this study is to evaluate a novel method of providing private individual feedback on a shared display. You will be asked to interpret letters on a display and press the corresponding button on an i-clicker remote.

Confidentiality
Your identity will remain anonymous and will be kept confidential. A computer will record performance as you perform the tasks, but no identifying information (such as your name) will be stored with this data, nor will it be associated with the data after it has been analyzed.

The results will be made public through publications; however, no identifying information will be included in any published disclosure of the research.

No audio recordings or photographs will be made of your participation.

Risks/Remuneration/Compensation
There are no anticipated risks to you participating in this research. You are free to take a break or withdraw from the study.

You will receive an honorarium of $20 for your participation. You will be eligible for the honorarium even if you withdraw from the study.
Contact Information about the Project
If you have any questions or require further information about the project you may contact Peter Beshai (pbeshai@cs.ubc.ca or 604-339-4003) or Dr. Kellogg Booth (kbooth@cs.ubc.ca or (604) 822-8193).

Contact for Concerns About the Rights of Research Subjects
If you have any concerns or complaints about your rights as a research participant and/or your experiences while participating in this study, contact the Research Participant Complaint Line in the UBC Office of Research Services at 604-822-8598 or if long distance e-mail RSIL@ors.ubc.ca or call toll free 1-877-822-8598

Consent
We intend for your participation in this project to be pleasant and stress-free. Your participation is entirely voluntary and you may refuse to participate or withdraw from the study at any time without consequence.

Your signature below indicates that you have received a copy of this consent form for your own records. Your signature indicates that you consent to participate in this study.

I, (print name) ___________________________ agree to participate in the project as outlined above. My participation in this project is voluntary and I understand that I may withdraw at any time.

____________________________________________________________________

Subject Signature     Date   _______________________________

Printed Name of Subject
Appendix B

Classroom Evaluation

Resources

This appendix contains supplemental resources that were used when running the classroom evaluation of Rhombus CSPS.

B.1 Student Clicker Questionnaire

These questions were administered via the Question app in Rhombus, as shown in Figure[B.1]. Students responded by pressing buttons on their clickers. A 20 second timer was displayed on screen to allow those who did not wish to answer questions the option of abstaining, as opposed to waiting for the number of responses to reach the total number of participants available.

1. It was easy to find myself on the screen.
   A: Strongly Agree
   B: Agree
   C: Neutral
   D: Disagree
   E: Strongly Disagree

2. I understood the rules of the game.
   A: Strongly Agree
Figure B.1: The Question app in Rhombus. The question is presented in large text at the top with the answers mapping to clicker buttons displayed on the left side. As students click in their responses, their names appear on the right side of the page directly below a count of the responses. Pressing a clicker button after one’s name is already visible causes the name to flash, indicating that the new button press has been received.

B: Agree
C: Neutral
D: Disagree
E: Strongly Disagree

3. I understood the controls of the game.
   A: Strongly Agree
   B: Agree
   C: Neutral
   D: Disagree
   E: Strongly Disagree

4. I understood the results of the game.
   A: Strongly Agree
   B: Agree
5. I liked playing the game.
   A: Strongly Agree
   B: Agree
   C: Neutral
   D: Disagree
   E: Strongly Disagree

6. I felt engaged during the game.
   A: Strongly Agree
   B: Agree
   C: Neutral
   D: Disagree
   E: Strongly Disagree

7. I would like to use this system in other classes.
   A: Strongly Agree
   B: Agree
   C: Neutral
   D: Disagree
   E: Strongly Disagree

8. It was satisfying to use this system to play the game.
   A: Strongly Agree
   B: Agree
   C: Neutral
   D: Disagree
   E: Strongly Disagree

9. How was the pace of the game?
   A: Very Fast
   B: Fast

C: Neutral
D: Disagree
E: Strongly Disagree
C: Good
D: Slow
E: Very Slow

10. **How would you rate the system compared to typical iClicker usage?**
    A: Much Better
    B: Better
    C: Equivalent
    D: Worse
    E: Much Worse

11. **How helpful are the in-class multiple choice questions with regards to learning?**
    A: Very helpful
    B: Helpful
    C: Not helpful or harmful
    D: Harmful
    E: Very Harmful

12. **How helpful was playing the games with this system with regards to learning?**
    A: Very helpful
    B: Helpful
    C: Not helpful or harmful
    D: Harmful
    E: Very Harmful

13. **It is worth taking class time to do multiple choice questions.**
    A: Strongly Agree
    B: Agree
    C: Neutral
    D: Disagree
    E: Strongly Disagree
14. **It was worth taking class time to play games with this system.**
   A: Strongly Agree  
   B: Agree  
   C: Neutral  
   D: Disagree  
   E: Strongly Disagree

**B.2 Student Short Answer Questionnaire**

The short answer questionnaire was administered to students on the final session of term 1 and term 2. The questionnaire was printed and students filled in responses by hand.
Post-Study Questionnaire

Pseudonym: ____________

Thanks for participating in a research study using the Rhombus Clicker System (RCS), an interactive classroom response system that allows inputs from iClickers to be used with real-time visual feedback. Please take a few minutes and answer the following questions.

[ ] I consent to have my data used in the study and have signed a consent form.

Which input device(s) would you prefer to use with an interactive classroom response system?

clicker, mobile phone, tablet, laptop, other: ______ (please circle)

Please explain your choice(s) below.

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

Describe any issues or problems you had with RCS.

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

Describe what you liked most about RCS.

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

Describe any suggestions you have for new features or improvements to the system.

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

In COGS 300, we used RCS to play game theory games. Do you have any suggestions for other applications of the system?

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

Other comments

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

____________________________________________________________________________

Thanks for participating!

Researcher: Peter Beshai pbeshai@cs.ubc.ca
B.3 Consent Form

This consent form was presented to students prior to any use of the system in term 1. In term 2, this form was given to the students before having them complete the digital and written questionnaires. Students who did not sign a consent form had their data pruned prior to analysis.
THE UNIVERSITY OF BRITISH COLUMBIA
Department of Computer Science
2366 Main Mall
Vancouver, B.C., V6T 1Z4

Consent Form

Principal Investigator: Kellogg S. Booth, Professor, Department of Computer Science, University of British Columbia, ksbooth@cs.ubc.ca 604-822-8193

Co-Investigator: Peter Beshai, pbeshai@cs.ubc.ca 604-339-4003

Recruitment: Participation in this project is open to all students enrolled in COGS 300.

Purpose: The overall purpose of this research is to evaluate the effectiveness of a novel system extending the use of typical classroom response systems (e.g., iClickers) with regards to student engagement, enjoyment, and comprehension.

What you will be asked to do: After you have read this document, I/we will respond to any questions or concerns that you may have. Once you have signed this consent form, you will be asked to:

- interact with a digital system (e.g., the iClicker system with custom software)
- complete a questionnaire about the experience

This should take about 90 minutes and be completed over 6 lectures. The interaction with the iClickers is part of the regular classroom activity but the questionnaire and analysis of the data are not. Unless you consent, you will not be included in the questionnaire or the data analysis.

Risks: There are no anticipated risks to you by participating in this research. Your responses and performance will have no impact on your grade in COGS 300.

Compensation: There is no compensation for participating in this study.

Confidentiality: The clicker system is pseudonymous. You will register your clicker with the pseudonym randomly selected and provided to you, which will be the only link between you and the data that you provide. This link will be erased at the completion of the study. The data you provide will be kept in a secure database and will only be accessible to the research team. You may choose to stop clicking/responding to questions at any time. If you decline to give your consent, we will remove any of your responses prior to analyzing the clicker data for experimental purposes.
The results of the research will be made public through publications; however, no identifying information will be included in any published disclosure of the research.

No audio recordings or photographs will be made of your participation.

**Contact for information about the project:** If you have any questions or require further information about the project, you may contact Peter Beshai (pbeschai@cs.ubc.ca or 604-339-6003) or Dr. Kellogg Booth (ksbooth@cs.ubc.ca or 604-822-8193).

**Contact for information about the rights of research subjects:** If you have any concerns about your treatment or rights as a research subject, you may contact the Research Subject Information Line in the UBC Office of Research Services at 604-822-8598 or if long distance, email ORSIL@ors.ubc.ca.

**Pseudonym:** Please enter the pseudonym (also known as an alias, handle, or nickname) that has been provided to you.

Your pseudonym: ______________

**Consent:** We intend for your participation in this project to be pleasant and stress-free. Your participation is entirely voluntary and you may refuse to participate or withdraw from the study at any time without consequence.

Your signature below indicates that you have received a copy of this consent form for your own records. Your signature indicates that you consent to participate in this study.

I, (print name) ___________________________ agree to participate in the project as outlined above. My participation in this project is voluntary and I understand that I may withdraw at any time.

_______________________________
Subject Signature

_______________________________
Date

_______________________________
Printed Name of Subject
B.4 Sequence Aliaser Mapping

The sequence aliaser requires a mapping from sequences to aliases in order to work. We used sequences of length 4 to map to 64 possible aliases, as shown in Table B.1. In term 1, we provided slips of paper that indicated their assigned alias and the associated sequence to students, which they then used to register with the system.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Alias</th>
<th>Sequence</th>
<th>Alias</th>
<th>Sequence</th>
<th>Alias</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAAD</td>
<td>leo</td>
<td>BBCA</td>
<td>adele</td>
<td>CCDC</td>
<td>yeezy</td>
</tr>
<tr>
<td>AABC</td>
<td>martha</td>
<td>BBDB</td>
<td>gosling</td>
<td>CDAD</td>
<td>jobs</td>
</tr>
<tr>
<td>AACA</td>
<td>jordan</td>
<td>BCAB</td>
<td>jackson</td>
<td>CDBC</td>
<td>fey</td>
</tr>
<tr>
<td>AADB</td>
<td>zooey</td>
<td>BCBA</td>
<td>keanu</td>
<td>CDCA</td>
<td>owen</td>
</tr>
<tr>
<td>ABAA</td>
<td>angie</td>
<td>BCCC</td>
<td>potter</td>
<td>CDDB</td>
<td>whoopi</td>
</tr>
<tr>
<td>ABBB</td>
<td>perry</td>
<td>BCDD</td>
<td>cruise</td>
<td>DAAC</td>
<td>portman</td>
</tr>
<tr>
<td>ABCD</td>
<td>stiller</td>
<td>BDAC</td>
<td>arnie</td>
<td>DABD</td>
<td>julia</td>
</tr>
<tr>
<td>ABDC</td>
<td>pink</td>
<td>BDBD</td>
<td>diaz</td>
<td>DACB</td>
<td>alba</td>
</tr>
<tr>
<td>ACAC</td>
<td>halle</td>
<td>BDCB</td>
<td>murray</td>
<td>DADA</td>
<td>liz</td>
</tr>
<tr>
<td>ACBD</td>
<td>lopez</td>
<td>BDDA</td>
<td>cruz</td>
<td>DBAB</td>
<td>maddy</td>
</tr>
<tr>
<td>ACCB</td>
<td>marilyn</td>
<td>CAAB</td>
<td>bee</td>
<td>DBBA</td>
<td>vaughn</td>
</tr>
<tr>
<td>ACDA</td>
<td>spears</td>
<td>CABA</td>
<td>leia</td>
<td>DBCC</td>
<td>oprah</td>
</tr>
<tr>
<td>ADAB</td>
<td>aniston</td>
<td>CACC</td>
<td>hova</td>
<td>DBDD</td>
<td>gaga</td>
</tr>
<tr>
<td>ADBA</td>
<td>spock</td>
<td>CADD</td>
<td>scarjo</td>
<td>DCAD</td>
<td>ellen</td>
</tr>
<tr>
<td>ADCCC</td>
<td>freeman</td>
<td>CBAC</td>
<td>audrey</td>
<td>DCBC</td>
<td>marley</td>
</tr>
<tr>
<td>ADDD</td>
<td>pitt</td>
<td>CBBB</td>
<td>elvis</td>
<td>DCCA</td>
<td>ford</td>
</tr>
<tr>
<td>BAAA</td>
<td>will</td>
<td>CBCB</td>
<td>deniro</td>
<td>DCDB</td>
<td>bruce</td>
</tr>
<tr>
<td>BABB</td>
<td>lucy</td>
<td>CBDA</td>
<td>stark</td>
<td>DDAA</td>
<td>carrey</td>
</tr>
<tr>
<td>BACD</td>
<td>rihanna</td>
<td>CCAA</td>
<td>holmes</td>
<td>DDBB</td>
<td>bond</td>
</tr>
<tr>
<td>BADC</td>
<td>cere</td>
<td>CCBB</td>
<td>timber</td>
<td>DDCD</td>
<td>samuel</td>
</tr>
<tr>
<td>BBAD</td>
<td>swift</td>
<td>CCCD</td>
<td>gates</td>
<td>DDDC</td>
<td>mila</td>
</tr>
<tr>
<td>BBBC</td>
<td>depp</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table B.1: The sequence-alias mapping used in Sequence Aliaser to allow participants to register their clickers to celebrity aliases using 4 letter sequences.
Appendix C

Rhombus Game Details

This appendix covers screenshots and explanations of the games that were played in the classroom evaluation of Rhombus. The games are covered in the following order:

- Prisoner’s Dilemma
- Iterated Prisoner's Dilemma
- Ultimatum Game
- Stag Hunt
- Coin Matching
- Coordination
Figure C.1: The standard check-in screen that was used in all games played with Rhombus. When users press a button on their clicker, their avatar appears on screen with a green overlay and big check mark fading in. Users are inserted in lexicographic order.
C.1 Prisoner’s Dilemma

These screenshots are of the Prisoner’s Dilemma game. In each round of play, users are randomly partnered, meaning that results from previous rounds may not predict what happens in future rounds because users play with different partners. A graph is displayed during play phases to remind users of how the general population played in previous rounds.
Figure C.2: The Prisoner’s Dilemma play screen in round 1. Users press C to cooperate and D to defect. The scores that will be assigned to each user depending on the outcome of their match-up are indicated in the Pay-off Matrix. Avatars of users that have pressed a button on their clicker are darkened and display the word “Played”. Subsequent clicker button presses cause the word “Played” to flash to provide feedback to the user.
Figure C.3: The Prisoner’s Dilemma results screen after the first round. Percentage breakdown of how the class played is given by the bar above the avatars, with blue representing the number of cooperators and orange representing the number of defectors. User action is represented by the hue of the avatar. User score is shown numerically and encoded in the lightness of the avatar. The two letters on the avatar represent the user’s action followed by their partner’s action. The total score for each individual user is accumulated at the bottom of each avatar. A histogram shows the average scores of cooperators (0.7) and defectors (2.3), as well as the overall average (1.5). Hovering over the bars in the histogram with the mouse produces the tooltip shown.
Figure C.4: The Prisoner’s Dilemma play screen in round 2. The score each user received in the previous round is displayed at the bottom of their avatar. The histogram from the previous results screen persists below the avatars.
Figure C.5: The Prisoner’s Dilemma results screen in round 2. Instead of a histogram representing the average scores, a line chart is used to demonstrate trends over the completed rounds.
Figure C.6: The Prisoner’s Dilemma play screen in round 5. The line chart persists from the previous results screen. Hovering over the chart with a mouse produces a tooltip that shows summary information.
Figure C.7: The Prisoner’s Dilemma total results screen. After completing all of the rounds of the game, cumulative results are shown. The avatars are coloured with varying lightness to encode their different scores, with the lighter colours representing higher scores. The highest scoring user is outlined in yellow.
C.2 Iterated Prisoner’s Dilemma

This version of the Prisoner’s Dilemma is very similar to the Prisoner’s Dilemma described in Section C.1, with the main difference being that users maintain the same partner for five consecutive rounds, as opposed to being randomly assigned a different partner each round. The class is split up into two teams of equal size and the partnerships are formed across the teams. The other difference was that this game was run in three phases where students alternating between playing as humans and playing with scripts “as computers”. Phase 1 was team 1 human vs. team 2 human, phase 2 was team 1 human vs. team 2 computer, and phase 3 was team 1 computer vs. team 2 human.
Figure C.8: The Iterated Prisoner’s Dilemma play screen in round 1. Users press C to cooperate and D to defect. The scores that will be assigned to each user depending on the outcome of their match-up are indicated in the Pay-off Matrix. Avatars of users that have pressed a button on their clicker are darkened and display the word “Played”. Subsequent clicker button presses cause the word “Played” to flash to provide feedback to the user. The subtitle for both teams was configured to be ‘Human’ to indicate to users they were in the human vs. human phase of the game.
Figure C.9: The Iterated Prisoner’s Dilemma round results screen. Total scores for each team are displayed in the team headings (14 for team 1 and 20 for team 2). Percentage breakdown of how the team played are given by the bars at the top of the team groups, with blue representing the number of cooperators and orange representing the number of defectors. User action is represented by the hue of the avatar. User score is shown numerically and encoded in the lightness of the avatar. The two letters on the avatar represent the user’s action followed by their partner’s action. The total score for each individual user is accumulated at the bottom of each avatar.
Figure C.10: The Iterated Prisoner’s Dilemma play screen in round 2. The totals for team score are indicated in the team headings (14 for team 1 and 20 for team 2) and the score each user received in the previous round is displayed at the bottom of their avatar.
Figure C.11: The Iterated Prisoner’s Dilemma phase 1 results screen. After completing all rounds in a phase, cumulative results are shown. The avatars are coloured with varying lightness to encode their different scores, with the lighter colours representing higher scores. The highest scoring user is outlined in yellow.
Figure C.12: The Iterated Prisoner’s Dilemma cumulative phase results screen. After phase 2 and phase 3, users see a cumulative total of scores over all the phases so far completed. This figure shows the results after all phases are complete.
C.3 Ultimatum Game

The following screenshots are of the Ultimatum Game. In this game, the users take on two roles in each round: as giver and as receiver. They first play the role of giver, where they must decide how much they wish to offer their partner. After everyone has completed this step, all users play the role of receiver, where they decide whether or not they wish to accept the offer a giver made them. The partnering in this game is asymmetric; the users form a directed cycle where their forward partner is the person they are making an offer to (they are in the role of giver), while their backward partner is the person they are receiving an offer from (they are in the role of receiver).

This game used three phases, one where everyone played as humans in both roles, one where the givers were played with scripts “as computers”, and one where the receivers were played with scripts “as computers”.
Figure C.13: The Ultimatum Game play screen, where the users are acting as givers. Users press A to offer 5 (they keep 5) or press B to offer 1 (they keep 9). Avatars of users that have pressed a button on their clicker are darkened and display the word “Played”. Subsequent clicker button presses cause the word “Played” to flash to provide feedback to the user.
Figure C.14: The Ultimatum Game play screen, where the users are acting as receivers. The offer made to users is displayed on the user’s avatar. Users press A to accept or B to reject it. The percentage breakdown of how many users offered 5 and how many offered 1 is indicated by the bar above the avatars, with the blue colour representing those that offered 5, and orange those that offered 1.
Figure C.15: The Ultimatum Game results screen, showing how users fared when they played as givers. A green square indicates that the offer they made was accepted, while a red square indicates the offer was rejected. When an offer was accepted, the score received by the giver is indicated by the number on the avatar. When an offer was rejected, the demand they made is shown with an arrow to 0 (e.g., if a user offered 1 and was rejected, they would see “9 → 0”). The percentage of acceptances and rejections is indicated by the bar above the avatars.
Figure C.16: The Ultimatum Game results screen, showing how users fared when they played as receivers. A green square indicates they accepted the offer they were given, while a red square indicates they rejected it. When an offer was accepted, the score they received is indicated by the number on the avatar. When an offer was rejected, the amount they were offered is shown with an arrow to 0 (e.g., if a user was offered 1 and rejected it, they would see “1 → 0”). The percentage of acceptances and rejections is indicated by the bar above the avatars.
Figure C.17: The Ultimatum Game combined results screen. Here the number on the avatar represents the sum of the scores they received playing as a giver and as a receiver. The avatars are coloured with varying lightness to encode their different scores, with the lighter colours representing higher scores. The total score for each individual user is accumulated at the bottom of each avatar.
Figure C.18: The Ultimatum Game phase 1 results screen. After completing all rounds in a phase, cumulative results are shown. The avatars are coloured with varying lightness to encode their different scores, with the lighter colours representing higher scores. The highest scoring user is outlined in yellow.
Figure C.19: The Ultimatum Game cumulative phase results screen. After phase 2 and phase 3, users see a cumulative total of scores over all the phases so far completed. This figure shows the results after all phases are complete.
C.4 Stag Hunt

The following screenshots are of the Stag Hunt game. This game is similar in mechanics to the Prisoner’s Dilemma (Section C.1), but the pay-off matrix is modified, which produces different incentives.

As with the Prisoner’s Dilemma, the game features three phases: human vs. human, human vs. “computer”, “computer” vs. human.
Figure C.20: The Stag Hunt play screen in round 1. Users press A to hunt a stag and B to hunt a hare. The scores that will be assigned to each user depending on the outcome of their match-up are indicated in the Pay-off Matrix. Avatars of users that have pressed a button on their clicker are darkened and display the word “Played”. Subsequent clicker button presses cause the word “Played” to flash to provide feedback to the user. The subtitle for both teams was configured to be ‘Human’ to indicate to users they were in the human vs. human phase of the game.
Figure C.21: The Stag Hunt results screen. Total scores for each team are displayed in the team headings (28 for team 1 and 28 for team 2). Percentage breakdown of how the team played are given by the bars at the top of the team groups, with green representing the number of stag hunters and purple representing the number of hare hunters. User action is represented by the hue of the avatar. User score is shown numerically and encoded in the lightness of the avatar. The two letters on the avatar represent the user’s action followed by their partner’s action. The total score for each individual user is accumulated at the bottom of each avatar.
Figure C.22: The Stag Hunt phase 1 results screen. After completing all rounds in a phase, cumulative results are shown. The avatars are coloured with varying lightness to encode their different scores, with the lighter colours representing higher scores. The highest scoring user is outlined in yellow.
Figure C.23: The Stag Hunt cumulative phase results screen. After phase 2 and phase 3, users see a cumulative total of scores over all the phases so far completed. This figure shows the results after all phases are complete.
C.5 Coin Matching

The following screenshots are of the Coin Matching game. In this game, users are split into two equal teams: half as matchers, and half as mismatches. The matchers aim to enter the same choice as their anonymous (e.g., both select heads or both select tails), random partner from the mismatches team, while the mismatches aim to enter a different choice than their partner on the matchers team (e.g., one selects heads while the other selects tails).

There were three phases to this game, similar to the Prisoner’s Dilemma (Section C.1): human vs. human, human vs. computer, computer vs. human.
Figure C.24: The Coin Matching play screen in round 1. Users press A to choose heads and B to choose tails. Avatars of users that have pressed a button on their clicker are darkened and display the word “Played”. Subsequent clicker button presses cause the word “Played” to flash to provide feedback to the user. The subtitle for both teams was configured to be ‘Human’ to indicate to users they were in the human vs. human phase of the game.
Figure C.25: The Coin Matching round results screen. Total scores for each team are displayed in the team headings (4 for matchers and 8 for mismatches). User score is shown numerically and encoded in the lightness of the avatar. The two letters on the avatar represent the user’s action followed by their partner’s action. The total score for each individual user is accumulated at the bottom of each avatar.
Figure C.26: The Coin Matching play screen in round 2. The totals for team score are indicated in the team headings (4 for matchers and 8 for mismatches) and the score each user received in the previous round is displayed at the bottom of their avatar.
Figure C.27: The Coin Matching phase 1 results. After completing all rounds in a phase, cumulative results are shown. The avatars are coloured with varying lightness to encode their different scores, with the lighter colours representing higher scores. The highest scoring user is outlined in yellow.
Figure C.28: The Coin Matching cumulative phase results. After phase 2 and phase 3, users see a cumulative total of scores over all the phases so far completed. This figure shows the results after all phases are complete.
C.6 Coordination

The following screenshots are of the Coordination game. This game is similar in mechanics to the Coin Matching game (Section C.5), but the scoring system is different. If both partners select the same choice, they both get 0 points, while if they select different choices, they both get 1 point. The phases of the game function the same as in Coin Matching: human vs. human, human vs. “computer”, “computer” vs. human.
Figure C.29: The Coordination game play screen in round 1. Users press A to choose “A” and B to choose “B”. Avatars of users that have pressed a button on their clicker are darkened and display the word “Played”. Subsequent clicker button presses cause the word “Played” to flash to provide feedback to the user. The subtitle for both teams was configured to be ‘Human’ to indicate to users they were in the human vs. human phase of the game.
**Figure C.30:** The Coordination game round results screen. Total scores for each team are displayed in the team headings (4 for matchers and 8 for mismatchers). User score is shown numerically and encoded in the lightness of the avatar. The two letters on the avatar represent the user’s action followed by their partner’s action. The total score for each individual user is accumulated at the bottom of each avatar.
Figure C.31: The Coordination game phase 1 results. After completing all rounds in a phase, cumulative results are shown. The avatars are coloured with varying lightness to encode their different scores, with the lighter colours representing higher scores. The highest scoring user is outlined in yellow.
Figure C.32: The Coordination game cumulative phase results. After phase 2 and phase 3, users see a cumulative total of scores over all the phases so far completed. This figure shows the results after all phases are complete.
Appendix D

Term 1 Interview Transcript

This appendix presents the full interview transcript from the interview held with the instructor that used Rhombus during the first term. The interviewer was the researcher and is denoted by R: in the script, while the instructor’s dialog is denoted by P:.

R: Why did you decide to run games in the classroom in the first place?

P: One of the things we did coming into this class, COGS 300, so this is Introduction to Cognitive Systems, was to upgrade the game theory content. The program the course had used games informally on occasion, especially iterated Prisoner’s Dilemma in a kind of very informal way. And there was some pressure, especially from the Computer Science department, that the game theoretic aspect of the course be made a little tighter. Our [inaudible] had experienced using games both in Computer Science classes and in Philosophy classes and the experience was that unless people actually played the games, the theory didn’t really, was very difficult to teach, was harder to teach, unless people... So we began using games, in class games, a parallel stream that reinforced this was that we had decided in the class to use clickers for quiz taking and so it was clear that for, this apparatus would also be useful to support gaming, although the built in software was quite deficient for this. There was no special support for gaming, but it let us collect moves from class and play N-person games, so we did that.

R: So immediately you started using clickers?
P: I think probably we used clickers, I taught this course this is the seventh time, I think the first term we didn’t have clickers, right away the second term. Another stream, just to mentioned it, I had a student who was interested in games who became the TA the second term and he was willing to do the extra work of let’s figure out how to actually use clickers. So it was his class presentation that convinced me that the clickers would really support gaming more than I, I had been using them quite loosely, just collecting move for this, move for that. So Eric Tulin(?) who was an undergraduate, became the TA and that was influential. To have a student actually work it through, so then I think the next year, we both terms, used the clickers to play games. And altogether, the feedback we got were the games were the good part. The third strand the stream in COGS, the stream of thinking is the original Turing idea that we should think through a computation in terms of what a humanly emulated computer could do. Could we write code that a human could understand? And so that streamed into it. We began saying you should play this game both as a human and as some code you wrote out that you could then give to another person that they then could execute. So we’re using games as a model for simple computer program creation. This is a mixed crowd, maybe only a third of them have any real programming experience. So it gave people who were at the lower end of that range a chance to think through a problem in terms of a series of instructions that someone else can understand.

R: So then the reason that the games are pedagogically viable is that people weren’t really grasping the games without playing them themselves?

P: Well I mean, [inaudible] your own background, but he was not in computer science, but one can teach you game theory as an axiomatic mathematical theory, in which case it has almost no impact on most students I would say. Or you can teach it philosophically as puzzles, or as in out of.. um, so, isn’t it amazing that the theory predicts this, but people want to do this? Which doesn’t have much impact either because people haven’t bought enough into the axiomatic theory to feel that there is a difference. OR you could, you can actually get people to play the games, which often people who do the theory say “well, oh that’s just stupid” I mean that’s certainly
a tradition I came from but when you actually do it, it’s the theory that’s [inaudible], but actually doing it and watching it, the people that follow the theory actually do better. Or the theory is predicting opportunities that... and so, I think that’s where you get the, a big plus, is that, and it also now you’ve thrown out in the class, you’ve made it their problem to try and figure out. You have people, you get the various possible solutions being voiced by different people. Then you go, wait! no I think you should always cooperate even if you get worse. Well then maybe these scores aren’t actually your utilities. Oh, well I see. So you work back through it that way and I think that is probably the impact it has had. I think teaching it just as theory you get a very small number who really get it.

Another thing, the games we use have also been used a lot in psychological research and so as a kind of follow on to understand those experiments, it helps to have been a subject by having played the games, otherwise you’re, once again it’s like “they did something”, which doesn’t know, “there was something done in the experiment”, but y’know...

R: And you said for you having them play as computers as humans, that’s so they can get an understanding of the Turing...

P: Yeah. Well there’s two aspects. One of my, maybe my guilt is, the course used to always start with them reading Turing’s Computing Machinery and Intelligence and working through that which is quite a philosophical paper, and we’ve sort of dropped it out because they read it in other courses and it’s gotten to be kind of a set piece, but it, as Jedediah Pearl reminded me when he came and talked, it’s a luminous paper that really holds up this really high standard that we should both understand computers completely and this transparent mathematical way, but also they should do human- we should always be thinking What can the computer do something? What would it take to make the computers do humanly worthwhile things like understanding causation? So I was going to drop this out, but this idea is really important, so how do you get the idea back is you give people a task that’s Turing like. So actually works out in two ways, the one Turing idea is you’ve got to be able to write code that another person would understand. And so you’re already seeing code, and they’re reading Newell and Simon instead on
Symbol Systems and Search, and so, this is an idea back in kind of the 60s early 70s Computer Science idea [inaudible], Code is a symbol system with an interpreter, but the interpreter can’t be that smart. You have to write this code so that just a dumb, just another student... so it gives us a chance to talk about what they know about real cognitive abilities and bounded rationality by realizing that if you’re going to hand your script to somebody else, it’s got to be written in some way that, y’know, you don’t have an established common language for writing game programs, so. And this year, people, y’know, “is it all right to write is as a kind of diagram.” Well, you have to, you have to figure out for this group. Will this group understand that? So the idea of common code and how you get coordination around it, all of that stuff gets kind of bled into the course around that. So that’s one stream.

The other stream is both in Cognitive Systems, but also especially in the Ethics side, which fears largely the way I teach the course, is the question keeps arising: what things can computers do as well as people? And perhaps better? And what do we do with that cross-over? And what’s the weird stuff around the crossing over point? And I don’t do it in terms of the transcendence stuff, the singularity, whatever. I don’t care. I mean, I care about the really practical way, as like, once driverless cars can drive better than people, then we have a real ethics problems because then human driving becomes worse than what we can... There’s a phrase that my mind isn’t jumping at the moment, but in certainly medicine, you’re not allowed to do something once there’s a practice that is better. And so, there’s experimental literature on that, and there’ll be that later on, but this is an opportunity to talk about that, which is, you’re writing these little programs, but you’re also playing these games as a full-fledged human. Which games, if any, will the computers do better than? Is the rigidity and limited repertoire of the computer program sometimes an advantage rather than a disadvantage? And especially you get to the Ultimatum Game and it looks like that might be, so that’s an edge to it that’s especially relevant to the way we’re doing COGS, right? Because you’re not going to actually get inside of the design of a driverless car or drone, but they raise up the question really fiercely,
which is, what if these things are actually better than human soldiers or human drivers? Well you can raise it in, and once again, the game kind of becomes this little toy world that you can raise this question in. Which of these games would you bet the program might do as well.

**R:** For example, you had the students play the ultimatum game. Is there anything in particular you wanted them to learn playing that game?

**P:** Well, the Ultimatum Game, is in a way the richest one, probably only makes sense to play it after you’ve played other games. I mean, backing into the critique of the way it used to be done. It used to be done with “God, there’s this counterintuitive and hard game the Prisoner’s Dilemma and look, you don’t expect people to cooperate, but they do” All of which is wrong in theory. So what you want to do is, that’s not, none of that’s true. What’s true is that single play game is an easy to play game, it’s just, it has a miserable outcome. Right? And the repeating game actually is easy to play, but it has this [inaudible: illian] outcome so there’s no, the great mystery, the Hofstadter kind of mystery about it is just confusing the most basic things that you shouldn’t confuse in game theory, between single shot and repeated games, and so having done all that, so in a way, in the Prisoner’s Dilemma there’s no mystery left. In a way, the only mystery is what we generated, which was a good idea, I mean if you knew in advance when the game ends, then there is an interesting case because it should roll back logically or rationally, but people don’t roll it back, and so there’s that. But, I mean there’s, so you dispel the mystery of Prisoner’s Dilemma, but it opened up the whole folk theorem and social choice and all this interesting stuff, but there *IS* mystery about the Ultimatum Game because threats are rationally very disturbing. Clearly, a core part of human and animal behaviour repertoires. And utterly mysterious from the point of view of game theory which transparently says “no, why would I ever take anything?” “If I’m offered anything greater than zero, good deal.” So there’s this really clear contrast which doesn’t come up, so now the mystery is back. There’s psychological evidence that people do make threats and they accept threats better from ... so that’s why the Ultimatum Game. The Ultimatum Game is important, but I don’t think if you only played it, once again, it’s kind
of, you need a kind of game curriculum to give people enough of. And so this is the case for the approach we’re taking. You have to do it enough so that they’re not confusing, “God this is interesting, we’re putting numbers on things” or “God, a series makes a difference”. All of each of these things is really important and has to be figured out. You don’t want them all piled up into.. and I doubt you could go in in one day, one lecture, and do “here’s all the cool things about game theory.” So it’s a little bit like going in and saying “here’s all the cool things about computers, or programming, or anything” You have to have enough of a track record so you can see why this thing is important. So the Ultimatum Game puts the, the way I see it now, the Ultimatum Game brings these things together. Especially because we then do evolutionary game theory and say “actually, in evolution, threats are selected for” Right so, now we’re beginning, maybe nature does select some times for non-rational agents. David loves [inaudible] and know Krustov(?)’s shoe example and you connect this all up with politics so that ending was really good and it also motivated, evolutionary game theory is kind of a mystery if you don’t know regular game theory. So this time they’re thinking in terms of looking ahead and what .. they’re doing all that stuff and now wait you’ve got this approach that has very simple agents and it gets results, which can be related systematically to the Nash results and that’s really interesting.

R: So then, what you want them to learn by playing the game is kind of that it’s different from the theory, or ?

P: Well, there’s multiple levels of learning. You want people to learn the mathematical discipline, or the case where thinking in terms of inside a model pays off and is interesting, so there’s all of that. Taking symbols seriously and really making sure the model fits and so doing all that stuff. And also that’s enabling you to raise all sorts of questions about “does this little model actually fit?” so on the exam there’ll be a little coordination game, but then it will say here’s [how the case?] these old friends don’t want to eat at the same restaurant, here’s this coordination game. Is that the game they’re playing? And people should become sophisticated enough to say, no that doesn’t because you said they don’t ... And also secondary
preferences, does that capture ... there's all of that. And taking that understanding and putting a tiniest program, which is a strategy, and making that strategy available. And then, I think, more subtly, now they're, I mean, just a background. COGS' old frame was first you build an agent, then you give it an environment and then you add other agents to it, which is kind of an old timey way to build things, since you want an evolutionary approach that would say there is actually always other agents. And the MIT way would say at least there's always an environment, so brains come after all that stuff. So [bradically? -> practically?] for me the social would come before even ... what environment your stuck in is probably going to be a product of what group you're actually, can afford to get..., Anyway, there's going to be a reversal of that ordinary thinking. And so games are super simple models of social environments that can let us pull that rehearsal off. The old way is that way(?) we go in, we just have one sensor, 1 bit sensor for light or dark, or touch or not, and that's great, but in a game you can say, well there could be a 1 bit sensor for success or failure and you're trying a coordination problem. So you have a really simple model of social environments. It also enables people to think about social environment in kind of that simple, COGgy simple minded sort of way, but also by doing that thinking, they're thinking that all agents aren't going to be the same. I mean there's no reason to think that all of us, 40 of us, are going to come up with the same agent, and so variety and how do you deal with it falls out very quickly from that, which is to my way of thinking, a good thing. Other approaches to COGS had all agents being the same because it fell out of a kind of simplicity, I mean, how else would you build a suction(?) sensor, a very simple cognitive agent, this has [sid other] effects

R: So then, after they've played the game, or maybe during, how were you able to establish whether or not they learned what you had wanted them to?

P: Question sort of presupposes I guess what we're supposed to use, which is in the kind of learning framework, we establish goals, ... you could tell David was probably being reviewed this year. Each class puts (side?) up, yknow, what are learning goals? Not to criticize that, but I think this
probably fits into a more exploratory form of learning. So the games aren’t
designed to reveal failures to understand on the fly. They’re not on say the
incremental testing mode that say you would get from some recent online
teaching environment. I think they’re more done in the mode of throwing
the class into a highly interactive situation where the’s lots of chances for
things to actually go wrong. I mean, now hopefully not going wrong with
the plumbing level, but certainly going wrong in the sense that they’re not
living up to people’s expectations. So I would say, good signs that it is going
well are that we have “WHAT? People are choosing ..? Don’t you people
understand how this works?” That’s a good outcome. You can afford to
do that in smaller classes and yknow you’re not dependent on this working
for 200 or online. I mean, I say all that because there’s lots of environments
where it would be a lot harder to use that method, but I mean, I would think
the problem with university students in this kind of a class is, why would
you come to class and not be looking at something else on your screen? Why
would you be engaged? One way you’re engaged is you’ve invested in this
see-no-light process and invested in and suddenly it’s like the expectations
you invested in that didn’t work out in ways you don’t understand and now
you’ve got to decide what resources to use. How do you explain that? Are
they all wrong? Are you wrong? Was the theory misapplied? And I think
that’s where lots of the learning happens. The more that you can do more
rounds, you can do more variations, there’s more chance for that to happen.
I mean ideally if we change this in this way, would this still happen? At
the end we saw that because the evolutionary stuff is bottled with already
written software where you can change the population proportions. “Yknow,
would this still happen if an invader came in, using this strategy. Yknow
lets poke that there, oh that’s not what I expected at all” So that’s what,
it’s designed with that kind of... Now, ideally, and we’re not there obviously.
You should also be able to say to people, well, some things I’d like, “Well, let’s
see: did computers do better?” and Did people learn over time? Because
you get people coming up afterward saying, “I was looking up there and I
think people changed over time. Will you go do the...” and I would run
stats and bring it and yeah you’re right, over time, over those five rounds,
the proportion of cooperation did change in the way that you suggested. So anything that would make that, would push that towards an open, public set of data would also then give you a more reflective version of this.

The one thing we paired this with that you wouldn’t have seen happen, there was running in my side of the class also a blog. So they got graded for two blog contributions. Certainly last year when I taught this alone, the games figured a lot in the blogs. So people would, yknow, “why this game didn’t really, wasn’t really a game about cooperation” or “how could this have happened?” And so, some of the learning there happens if you have another form for people to um yeah so.

R: So it’s kind of like the learning maybe isn’t so evident from an individual’s play in the game, but kind of the group and the realizations of how the group dynamics work?

P: Exactly and this plays back to why you want to be able to link up a class full of people. It’s because it is a social- it’s a game that has a social component, and otherwise it’s a puzzle. And individual exercise. Which is useful, I mean it’s useful for things that are based on the pure agent side. I mean David has some pretty good exercises where people learn to build a Bayesian model of decision and that you don’t need a group for, but if you are trying to build a model of how many threateners will I face in an environment, then playing it out in a group is going to be much more useful.

Yea, I would say, in asking that question, you could see in one direction you might go, which is, it would help if people’s strategies were more programmatic. And certainly I taught this in AI, that’s the way you did, you’d have to submit your strategy as runnable Lisp code and then there’s many more possibilities of re-running it with different populations and for this course I think that wouldn’t work. I toyed with it, but it’s not going to work. There’s not a common language even though they’re supposed to have one. They take the.. you lose being able to run it in different ways, but actually having people basically enter their outcomes by the clicker, it’s a lot more involving than (so grain?).

R: You taught several games. Do they all have the same type of objective? The same kind of learning objective?
**P:** You can see that, it’s now come together, partly through the process we’ve been through this term. Now since playing the games is easier and easier and that’s dropping more into the infrastructure, you can think about what the ideal curriculum using them. I can give you the next term’s so you can compare them. I mean, you’d like, externally you’d like this curriculum this series of games to make sense and one way to make sense is to fit other things we’re doing in the course. And so one change we’ll make is the first game. The other thing they’re doing in the course is writing a cooperative robot to play simple traffic games in the lab. So their robot is supposed to solve a 4-way stop sign or something like that. So ideally the first game should also be a coordination game. And so that’s a change we’ll make from the zero-sum game we played this term. As it turns out the competitive aspect didn’t, it was too random. But a coordination game has more, it can be very rich with very simple, I mean. Just a game where you have to choose A and B and A and B is the coordination pair is hard. I mean everyone can’t choose that A because now half the people have to switch, but which half switches and how do you distribute that switching. All of that is cool stuff and it’s cool from a deeper computer science perspective as well. So now, here’s the case where it’ll be easy to play that as to play the game we played before, so we’ll just switch the game around in the apparatus, but we’ll get a much nicer external match to what they do in the rest of the course as well as hopefully another interesting game to play. Actually testing out, I put it on the final, so this is one way to pretest it is to put it on the final and then see what people come up with on the final. So I think the curriculum has a simple game that really more about getting a program out and comparing the imminent, but with no literature. I mean except maybe there’s a literature that says only people can solve problems like this. The Shelding literature in game theory. And then move to the Prisoner’s Dilemma single shot, then the iterated one and then talk about iterated game theory and then the ultimatum game. So a chance to talk about bargaining as well. So there’s a little mini curriculum based on games getting harder in one way, or getting, bringing out different features of the social environment.
R: So can you describe maybe in more detail how you ran these games in previous terms?

P: The selection was different because what’s easy and hard to do. So we did one game different term to term, but I think an iterated prisoner’s dilemma where they wrote out a program on paper and switched with another person and then each person ran them and wrote out on the paper the score and I think we collected information just on the blackboard and I think I photographed the blackboard with my camera and then entered the stuff in the spreadsheet, and so the feedback loop was from the end of one session Tuesday and then Thursday. This was before clickers. Then we tried it with clickers with trying to enter. With clickers we definitely went immediately to, we used an n-person game. And that was easy and we got the results right away and entered in the spreadsheet right away and at one point we tried to use the clickers to collect data from the iterated game where they’d already written on the sheets saying how many people’s fell within this amount, but it was a little awkward. This worked, but we then the next, we were teaching that term, I was the only one teaching that term which was not supposed to be, supposed to be team taught. Next term we start alternating and then it becomes really a pain, because you’d do something and the other person would teach and then you know you’d sort of “oh here’s the results from that game you played a week ago”.

R: But with the clickers it still took until the next session before you could show the results?

P: The clickers let us, for that one game, for the N-person Prisoner’s Dilemma, let us right away look at some results, but for the other game, maybe we collected stuff by clicker, and then I had to analyze and came back so. And the delay interacted poorly with coordination with the other instructor. So, that led to two things: one we don’t coordinate that way anymore. David and I find it much [reduced?] to have a week, a week, a week and given that week week week the old way wouldn’t be that bad, but it’s really great now because you get to build up to a game and play it or play it and analyze it all within a week. But the papers go missing, the papers aren’t always legible and the instructions are a little bit too complicated and
so the failure rate was quite a bit higher. I mean we’re talking about at least 10%, you’re kind of guessing at what. Maybe it’s fair to say this, we haven’t been consistent about using them for giving grades or participation points or whatever for games, but I think when you use clickers, they’re associated with the seriousness of, the clickers have a seriousness about them because they’re used for quizzes. Now, whether that, it’s very difficult to know the difference that makes empirically, but you have a sense that there’s a little bit of, they know that when they click they’re actually linked into that and that seems to be a good thing in terms of running the games.

R: So you’re saying you had at least one game that you played that didn’t involve clickers.

P: Yea typically we had the iterated Prisoner’s Dilemma and it didn’t involve clickers, except maybe once or twice and just for some data collection. And almost every time we played the N-person prisoner’s dilemma which has such a simple game output that the clickers let us both do it and talk about results right away. So the problem there, it meant that, knowing that that worked, we tended to use that game and actually started doing readings around that game. But it’s a really hard game to analyze, so what you’re seeing now is an artifact of a difficulty in classroom procedure. In a stadium we can do this, so I guess we’re going to do a lot of this card flipping. Why are we doing that? Well because that’s the thing we can do with a stadium full of people, well this is the N-person game we can do easily in the class, but that’s not a good reason to choose the game. We should be able, I mean ideally, the instructor should have a palette of games they can choose from and say “No, I’m really interested in this, I want to use this game that has this shape” not just, here the data is... This is an interesting theme. Certainly in the class we played this out, we said “look at this bias” and it’s just like having an MRI and you can’t do certain things and you’re going to get people doing other things in an MRI and there’s going to be your experimental tradition is going to look a certain way because of the bias of that apparatus. So here we go again, this apparatus can give us a bias or reduce the bias. So I think by not being forced to do N-person games, with actually being able to now, I mean, crucially, pair people up individually,
then you can play games, the games you want to play rather than the games you’re playing because that’s ...

R: So just to dig a bit deeper into how it’s actually working, so the N-person would be like you open voting, people click, you close voting, you export the data or something and you analyze it and you run some scripts on it in excel or?

P: Well basically. We’re getting bar charts in class, so we can get a quick take on that, but then we can also come back and say I’ve analyzed this and we can look at a little bit more complex ways. But mainly we could get round 1, okay here’s the mix, round 2, and we’re putting bar charts up and copying the results. But you can’t do that individually because it’s an order of magnitude more. So it’s a really simple game outcome and a really simple epistemology of everyone sees what the proportion was, but even then we were stopped from doing. What the literature has moved to saying that game itself is pretty much, the Nash equilibrium was not to cooperate at all, so unless you put people in smaller subgroups, and then change those groups, it’s unlikely, but now you’re, we can’t do that, so we’re back to yknow.

R: So then, can you describe what improvements the system we used this term were able to let you do?

P: I think the main improvement is that it enables you to play standard games. That is, the games that actually drove game theory as like theoretical development. So in particular, these are two person games people are playing anonymous individuals and often a repeated number of times and so couldn’t do that before except on paper, so maybe once. And now we can do it, each person is playing in two different roles, as a program and as a person, you can stretch the games out, at least we have an order of magnitude maybe actually more than that of compression of how much we can get in in a session. So what we have, sometimes we play three rounds where they play five or ten turns per round, so that’s a lot of game play. So I think that anonymity is clear and clear to everyone and sort of puts the games in another space with separate pseudonyms. So I think that’s all, I mean we’re meeting conditions of the literature. I mean, those games are supposed to be played anonymously between pairs.
So we’re now able to play more games that way. Since we’re linking up with the literature, we’re not doing this weird thing of reading papers of what you don’t want to read because they’re linked to the game that we’re stuck playing and now we’re trying to explain complicated papers. We’re reading classics. We actually read, when we were doing the Prisoner’s Dilemma, we got to read Kahneman’s Nobel Prize winning lecture. That’s an important thing. If you can play the classic game, then you can read the classic paper and look at very simple, but very influential results. And that means the game side is closer to the experimental side when we’re doing experiments. I mean, we actually do Kahneman’s famous experiment you do in class and you do the Trolley problem, the real trolley problem. And it was weird that for games that we weren’t actually doing the standard games. We were playing some weird game that we could jigger into the classroom format. So this is a huge plus. So they know now the standards. If they went from this class to Kevin’s standard game theory class, they wouldn’t “oh wait that’s all different from…” it would be more “no, we know those basic things” so I think that’s kind of the original goal we started with, which was this class should feed the standard view used in Computer Science. I think that’s been a plus.

So the other things we can do then, the data is now in a standard format, it’s already excel friendly and ready. I think at this term each time that I could report out yet further “yes, what we were talking about in class did happen” or “No, even though it looked like it, no computers actually didn’t do better at this game.” So you have a finer level of analysis available because all the data already in a csv, standard form.

R: So how big of an impact did the system have on the games - from administration to pedagogy to engagement? Maybe in comparison to previous terms.

P: I think it doubled the number of games we played. I think we typically play one on paper and one that N-person one, and so this time we played four different games or did we play even five. Oh yes actually we played five because we added the limited Prisoner’s Dilemma and the Stag Hunt game. It let us experiment more and now, since it’s cheap to experiment we could
try more games and modify the curriculum. In the light of that we'll go back to four games next time. So I think that's, it's made it a strand in the course. Something you could do easily. Once people figured it out, it was fast to do. We could do it in a third of a session. Previously to do one on paper it was the whole. It took, it's paper, it's confusing, and oh no wait, so it took a long time and a lot of time was spent on administrative stuff. The administrative stuff was a lot less and the actual pedagogy was more, so that was good. And even though it was in experimental mode, we were probably taking less time and distraction than we were doing when we were doing it the other way.

R: And in terms of engagement from the students, did you think there was, from your perspective, a difference?

P: We'll know more in a week because the number of game, there's game questions on the final and we'll see. You can be surprised - oh wow they were really engaged playing the games, but they really got noth- they didn't obviously learn very much. My impression at this point is yeah, that they became it became more of the common framework of the course. This was almost an ideal case though because both sides of the course were emphasizing similar things. Right I mean, David does a lot of decision theory and making analysis and so back and forth, but I would say it definitely made a difference that way.

R: Do you think that having the real time feedback and showing individual scores was a big improvement?

P: Oh yeah, that’s a huge improvement. Before, there was almost no comparative analysis at the individual level. I took back all the papers and no one ever got to see that. And in the N-person game you’re seeing group results and you’re missing that whole level. We don’t know what people are picking out from there. For all I know, some people are tracking particular others that they... That you know, you’re basically making available huge amount of information for them to do what they like with, but yeah I think that had a definite impact and certainly it added to an engagement factor.

R: And now you're playing more standard games. Is that part of the reason why? You couldn't have played some of the games without that
feature? That is, without being able to show people their feedback-

**P:** You can’t play standard games without people being paired anony-
mously in a group. And all of that takes a way to identify multiple individ-
uals. So that whole individual feedback function is crucial to playing this
game. You can’t say you have a well-informed agent if they don’t, yknow
I chose some things over a few times I don’t know what happened, maybe
yknow.. and often I think frankly when we were playing the other way on
paper, we were probably giving false feedback. It seems like there’s more
cooperation, but we know once you have the more or finer level of analysis,
well wait no no that’s not true, we’re just looking again up there but you
have to take this run in and look over here it’s not what you think.

**R:** What expectations did you have with regards to just getting the real
time feedback of the system?

**P:** I think we had no firm expectations because as we began we didn’t
know how we were going to do that. So this was an open problem. In a
sense you know you have to give people feedback, but it’s difficult to know
how to give a room full of people feedback working with extremely limited
device with no individual level feedback on it. So that cuts out.. So I think
you adapted that problem, solved it in an interesting way. I at first was very
skeptical that these celebrity id things would work. When I first saw the
group sizing, I said is this really going to work? And I think that exceeded
expectations. Expectations were very iffy and I think that worked out really
well. It turned out to be a really good solution to the problem. I think it
turned out, I mean the old way working on paper distracted from, I mean
there were people who just got lost in trying to figure out some one else’s,
the bookkeeping got in the way. And here, instead, the problem of getting
there thing connected was more engaging rather than, that kind of turned
people into the common problem rather than distracted them with their own
little yknow, am I doing this right, being the bookkeeper for another person.
So I think there were lots of good outcomes of that, beyond expectations.

**R:** In terms of your needs, would you say the system met your needs?

**P:** Yea and it was Apple-ish in that way- it met needs I didn’t know I
had, so that’s good. That means that, it certainly solved the problem where
not focusing on a set of problems was easy to do with a very much reduced apparatus. So now we have a much more flexible system. You immediately start thinking of what changes? How can that be changed and what things you’d like to add to it and that’s a huge improvement.

R: Was there anything about the system that was lacking to you?

P: On the system side, it’s a little, the data is a bit difficult to anonymize. I’ve actually tried to do it, only because the pairing information is hard to capture in a... You want to go through and say this clicker gets replaced with some yet other “player 1”, but that clicker is also the opponent in these other places. I don’t know Excel well enough to do. So there are some things that would be nice to have. It would be nice to have an easy way to anonymize data to make it available to students. It’s just a question of it’s in a form that.

R: It’s interesting because I get it anonymized but I change...

P: Yeah, so I was going to say, it’s partly. It’s partly.. The other side is we are using it under research constraints and without the research constraints, if these were simply handles that were being used in class, then the problem would probably solve itself. We would just use it in the raw form that is already being anonymized. But if it’s not being used for grading and it doesn’t ever have to be linked to something else then everything can be simple. And for all I know, people don’t mind using their clicker numbers as anonymous, but yknow they’re not like student numbers, they’re these weird extra number that’s out there. Since I wasn’t sure what I could do with the data anyway at this particular time it wasn’t really stopping me from doing something, but next time I would like to move it to “the data set is available” and we should be talking about that because that’s all a movement to open data and so yes this data is available and anyone, let’s see who’s first to get a blog post finding something in that data we didn’t find yet and that would be cool to do.

R: Were there any parts that were unexpectedly useful in the system?

P: Well, I think the, to give you credit, I think the whole engagement factor that came from the way the pseudonyms were set up. Just turned out to be fun, it wasn’t a chore, turned out to be, turned out those little
half-time exercises, fun that people were having. I think all of that was unexpectedly good. And really I never had this wealth of data. I never had the problem of all this data to analyze. Before it was we had a screenshot of the whiteboard and no idea and it was lost to us what had happened. And now we have problems of a wealth of data and how to deal with that, which is all, fits exactly with the themes of the course. Yknow given a firehose of data, what’re you going to do with it? Data you’re allowed to do this and so all of those questions. I mean the class, as well as David, were all surprised and interested with the research ethics constraints. Oh, how does that work?

**R:** Were the goals you set out to accomplish by using the system reached?

**P:** Yea I think exceeded because we really didn’t expect that we could do this much. Even quite experimentally we got way more done this term than expected. Exceeded.

**R:** What are your thoughts of using multiple displays in the classroom?

**P:** I think I’m sort of being converted to that. I never thought... I mean, I had it as a luxury in earlier versions of the course, hardly used them except to duplicate stuff. I think I don’t even remember a time, maybe once we used a solid object viewer on one side. But now, this made me appreciate how lacking this is and practical concerns come up for next term where I’m not sure whether the person I’m teaching with is actually comfortable doing things like the quizzes. If we had multiple displays, I would simply run the quiz software on my machine and run the quizzes for him. And with a single display this is, I mean, the clicker software is not made to be used by two instructors. They didn’t ever think of that. So literally, David has to send me his session records and then I have to blend them without getting it wrong so it’s fooled into thinking one instructor. And when you’re convincing someone to use clickers for the first time and then to say “oh, by the way,” Well this is open source software, I have a mac but I don’t run OS X, but it’s got to be the instructor running it because yknow it would be really nice if you simply you said “fine, there’s another display, I’ll just run the module in your place. it’s easy to do” So I Think that bottleneck is huge and has helped me appreciate, certainly for any bigger class it would be essential. I mean you can see it in this class. You can see how primitive the
single projector rooms are. I mean I don’t know if running two projectors in that room is easy, but it’s a reminder that probably the base standard that people now should be asking for is yknow two displays.

**R:** How do you think having the multiple displays would affect the games?

**P:** I don’t know I think it would probably open us up as thinking about it from a development standpoint to think what kind of more graphical feedback you could provide. I think we’re limited in thinking what kind of feedback we can provide. And I think it would probably open up our design space. Now maybe more important is, now what would a full feedback game controller, I mean that’s another space. This shouldn’t take all our thinking. What I really like about this project, it made a huge improvement on a very limited, but in-place technology and that’s I think a really cool thing to have done. I mean it’s there, and people want to use it and you can take advantage of that and you build on it and you don’t say “if only we had...” but yeah, given that we’ve got a non feedback providing controller, there’s interesting questions about what else we could show people and how that would work. Well you and... I gather you’ve already worked with Kelly, how could more group on group stuff work.

**R:** Thanks for your time. Do you have any final comments or anything else you wanted to say?

**P:** No, stay with it. No I mean, a thank you it’s been a great project to work on with you. Yeah, I think you took to working with the class really well and that made a difference.

**R:** Thanks so much for the interview.
Appendix E

Term 2 Interview Transcript

This appendix presents the full interview transcript from the interview held with the instructor that used Rhombus during the second term. The interviewer was the researcher and is denoted by R: in the script, while the instructor’s dialog is denoted by P:.

R: What do you think makes playing the games pedagogically valuable? 

P: This is a course context that we’re not stressing proof techniques or we’re not treating game theory very formally and second we’re stressing contrast between rational agents and human agents, and third, there is some discussion of the computational side of rationality. So, actually playing games gives people, well kind of firsthand, it lets them actually write a strategy and execute it. And secondly, it lets use model the .. I mean we’re using the games as models of social interactions, so that lets us actually look at a game in a real social case rather than further modelling the social side through further artefacts. That’s probably the main pedagogical advantage is that it shows the game in a real social context. You’re playing against real responsive, but not identifiable other agents.

R: Would you say you had that opinion before you used the system on your own? 

P: Well I think that when you use it on your own, there’s a bit of plus and minus, it’s a little bit more flexible because you’re not ... well we’ll cut this off here, or I mean, probably a little bit more control over the situation so you
can use more flexibly. On the other hand, you’re operating the system and so there’s further distraction. So I think maybe there’s a bit more interaction around the games. There was no third party; you weren’t there, so there’s a bit of that Westinghouse, negative westinghouse. You’re not there, when you’re there it’s like stop and take. I would think it’s probably a little looser, but I didn’t measure. But you know that’s hard to say because it’s also the second time you do it, some of the modules are now pretty well tested, there were things we expected as instructors, so there were a lot more moves to make, so I think probably that made more of the factor, just that we had more experience with it.

R: But you still had the same pedagogical goals last time that you used it?

P: Yes, yes. Same course, same style of teaching.

R: Was there any evidence in the game or how they played the game that? You said the main goal was kind of this social experience from playing the game actually, and do you feel that they actually achieved that from playing the games? What you wanted them to?

P: Yea I think, seeing the course a better appreciation of the rationality as a way of looking at how agents interact, as a theory of interaction. You’ll see people saying, ‘but yeah, but if humans were playing, would we expect...’ Yknow if humans were interacting with a robot, a car, would we expect some of these, some other problems come up? Yes, I could see the engineer looking at just the rationality of the accident, with, and so I think that contrast between human agents and machine agents is enhanced by playing out the games, I think that’s true.

R: I was wondering if you could describe the preparation you did for using the system before the class.

P: Before the term, we set up, we made sure there were enough pseudonyms for the [inaudible], we made up the, we did the term wise stuff. In the beginning of the term, we got a list of clickers and exchanged,... I don’t remember the details, I think you actually probably matched up those clickers numbers I sent you to set up the... so I entered with a prepared group of, a prepared system. Um, basically I think I, we ended up with more games in the system
than I, we had kind of a surplus, we had to fork some things, change some things, so for each of the times I used it I went through and made sure I remembered which game we were actually going to use and what its details were and reviewed it for myself. I think a couple of times I actually put it up to make sure I understood it. So I did a bit of a rehearsal, maybe 10 minutes.

The only time I had backtrack because I screwed things up was that I forgot that it made a difference whether you use the system first or the clicker system. And so I ended up with whichever way I did it wrong and something not responding. I think I must have done the games first and then the clickers didn’t respond and I just didn’t think of the expedient of unplugging it and plugging it in again. And this time my teaching partner wasn’t a mac person, and so nobody thought of it on the spot and so we ended up doing a paper quiz.

R: So when you did the rehearsal did you use your own clicker for that?
P: I used the built in, the automatic one.
R: /debug?
P: Yea, yea, the debug screen. So I just ran the debug screen.
R: And so, was this preparation from what you did last term?
P: Last term, I think, we often met before class to make sure we had the right game. It was different, Last term it was more, you were the chef and I would make the order and put together the dish and we’re in a cafeteria and I had to make sure is this. Oh right and this is called and... And so what would happen is I had re-written the, I had perhaps changed some detail of the game in the lecture and I want to make sure we have that very still. Because there was a couple of versions of the game and I wanted to make sure there was a version that matched that.

R: So then let’s talk about the experience of using the system in the classroom. What was the procedure that you followed to get the games running? What was a typical, if you were going to play games, what did you do?
P: So, typically there was some preparation. Typically they were given a sheet or some instructions the time before and as before. So that influenced
the design of the course a bit. There was another factor saying let’s do the
course as weekly modules, because we’re two instructors because I wanted
to be able to say on Tuesday, Thursday there is this game coming up rather
than two weeks. So that typically they were either given a sheet or reminded
of the last frame or frames of the lecture that present the games and their
options and remind them that it was coming up. They would have to either
think out or write out a strategy for the game. So there was that much
preparation. And we had talked about experimental game theory in the
course, so I’d explain the reason for this is I want to make sure you really
understand the game and actually let’s, maybe a couple of times, we actually
played out the game with regular clicker mode, just so they’d actually. yknow
this is the ultimatum game, it has this form, somebody does this, somebody
else does this. And, so they had done that preparation and been given often
a handout (maybe 2 out of 4 times). And then that the day, I guess typically
we started with the game. Trying to think, could look at the slides, but I
think typically because of the awareness that it wasn’t, it’s not totally easy
to drop in and out of the.. well the reason is that if you dropped in and out
of it, you’d end up with two clicker files, as the clicker software is background
and stuff, but it’s unforgiving, especially with two instructor’s using it, it’s
not designed for that. So and anyway we were sometimes [inaudible] leaving
files, and I think I didn’t try it, but I didn’t want to end up with two files
the same day because I didn’t know what Connect would do with it. And
this isn’t, this is, it’s kind of background to.. So um, I always wanted to put
the games either in front of or after the, so I think typically we started with
them. And the only bad thing about that in terms of prep was that you
have this trickle in and it’s not easy to have people. It’s not the section of
the class you want people drifting in. So typically I would load something in
front of it, some administrative thing just to give myself a bit of a buffer. I
think I found them that once they’d done it, once they got it pretty quickly,
so then we’d through the sign-in thing and blah blah and put the game up.
I think the first time it was probably a little shaky because I was using the
keyboard. and I think you had said remember the controller will do this.
Once I was using the controller to step through stages, then it was easy and
I was back in control of the pacing.

**R:** So you said when they did the, did you use the Grid program to start them off, or did you just go to the program you wanted them to play and just had them check in and then start playing?

**P:** I think I immediately had, yea I didn’t even think of the Grid program.

**R:** OK, and so then when you said on the previous day you sometimes have them kind of play the game. Was that with the system or was that with just the regular clickers?

**P:** No, without the system. One of the firm rules of experimental economics, you have to make sure that people actually understand what the payoffs are and how the game is played, and so I would typically end with that game they were going to play and maybe playing it just simply if what happens if here’s your choices? You can choose C or D and let’s just try this. I would see what people will think the thing to do is and talk out. People saying, “ohh I get it. No matter what I do I get screwed” So I had that preliminary discussion as a way to motivate people to understand the payoffs. I should say the other thing that was, there was something different this term, which was last term when they played the games at least, they at least were under the impression that the games would count. This term we never, never, I mean there was some vague discussion that maybe there would be some bragging points or something, so there was a difference. And that’s a difference and it does reflect on experimental game theory methodology, because the methodology says it really should count. Otherwise, it’s kind of under motivated. There seemed to be a lot of intrinsic motivation, but probably on the methodological side that’s something we would yknow should... I mean I think because right there I. If I had this from you on some other basis, not that you were in the middle of experimental work with it, I probably would have figured out a way to use it for grades, but I didn’t want to mess around with the BREB. So there’s a kind of conservatism that comes with it being a research project. I think that once it’s another kind of use you’d want to give more attention to, maybe make that a discussion point in the class, which is what’s a fair way to motivate people to...

**R:** I’m just wondering is the reason that you used the regular clicker
software because it was easy to just do right then and would take too much
time to set up the game? Last term we did a prep game the day before

P: We did. I think it was simply a matter of not doing a change-over.
And also, they knew how to use this software, so that wasn’t the question.
The question was, really the prior one, aside from all of the extra stuff, do
you understand what makes this game different from the other games we
played.

R: And the basic clicker technology was enough to get them that

P: Yea, it was enough. You could have done it, you could have asked
people to raise their hands, and so there wasn’t any special reason to match
them up and have a random assignment or anything.

R: So how many times in the term did you end up using the system?

P: We used it four times. If you want the handouts I can dig them up
and send them to you.

R: At the start of the term I gave you some tools so that you could assign
the aliases to the students, I think there was an excel document, there was
a script. I was wondering, it just left wide open the question of how you are
going to tell the students what their aliases are, and I’m wondering how you
ended up doing that?

P: So we did it informally. We said play around with, which fit with
what we were talking about, we were talking about interactive robotics at
that point, so just figure out a way to use feedback in this situation and I
think everyone figured out what they were. But it’s a small class, 25 or 28
at most normally, so I didn’t have a question, I mean I think the question
would, .. we could get away with it being a small class and people figured it
out, but if we were thinking, there’s talk of using it next year in 200 which is
quite a bit bigger class, like 120 or something, I think you probably couldn’t
do that, we’d have to figure out a way. I think it’s, with a small prototype
you can get away with, we had that discussion, but I didn’t, I don’t think I
sent out anything or announced it in any, and there didn’t seem to be anyone
saying I’m forgetting who my alias is.

R: And so, did you use the grid program for them to play around.

P: Yea I think it was at that point, at the beginning, we had the intro-
duction to it, we used the Grid program.

R: So just by them moving around,... P: Yeah, people figured it out. We had a discussion about that and it was interesting that you could do that and people were trying to figure out what things were responding to what, but it was a particularly tolerant group given the subject matter. I don’t think it, now that we’ve talked about it, I don’t think that if we did this next year it would work with 120 people. There’d be chaos.

I think the only user constraint due to the way the user is designed is the original plan was to have some people write scripts and have someone else interpret them and we figured out that was very hard to do. You could give your script and your clicker to someone, but they wouldn’t know your alias. You could give your alias to someone, but ... So that was, so we realized that on the fly. So we said, okay interpret your own, but that was too bad because in being able to write a strategy that somebody else could interpret is a good skill and a check that you’ve actually written it rather than just filling in “do whatever is smart to do” will work for yourself, but not for your buddy, but it also fits in a tradition of the Turing machine and it’s a script involved. That was I think one constraint, it would be nice if there was a way to swap them without ... but the design works for everything else, but there is where it pinches.

R: Hmm, okay. I was wondering how confident you felt using the system on your own?

P: I think we didn’t get into any place where I thought oh, I mean maybe less than fully confident that work was saved. I think it always was, I think the only time I exited too fast and didn’t get the summary, just got the stage result, but enough that I could reconstruct them. So there maybe if there had been some feedback that said “Stage 2 saved, Stage 3 saved, Final Result saved”. I think that was the only lack of confidence.

R: Ok yeah that’s a good point.

P: Because it warns you not to leave, but at some point you can leave. I thought oh my god I have to leave this running all the time. I think I did leave the script always just sitting in the terminal window, and I just backed up and ran it every time, but that was just a simple thing to do. And this
is maybe the best case, I wasn’t trading off with someone else in the class, I mean we weren’t, I had the session each time and could arrange the lecture around it.

R: That’s a good point. I mean it saves after every round, but it’s important I guess to give you the visible feedback that it’s saved.

P: Yea, well it would be good I think, I think that would be a good. Always helpful to. I know I think of this because we’re in the middle of doing 4 people times 2 countries worth of taxes with very different interfaces on the two and it’s “has it saved this? has my daughter’s file?” And that really is a big source of.

R: How was the experience in the classroom from your perspective different this term from the last term, given that you were the one running the show?

P: I think maybe a bit more interaction because it’s less intimidating from their point of view before there was like three senior people in the front and them, but it also intermixed with a different person who had much less interest in game theory. Before there was a bit more of a general kibitzing, “Oh wait, is that really this?” because David a lot more interested in the game theory, and so that may be overshadowed. It was a slightly different environment, but it worked quite relaxed. It wasn’t an intrusion. I think the clickers are familiar and the interface is fun, so between those two things.

R: Can you describe the student engagement level this term compared to the previous term?

P: Yea that’s probably totally overshadowed by the difference in the course. I think, I mean just why it’s overshadowed is because I think David is a much more jump off and have an objection and raise a problem in every class and other people pick up from that and think “oh that’s okay to do” and this term was much less of that. And so I think it carries over in all aspects of the course. So I think the games played out well, and we got some really interesting results, but there was maybe less of that interrupting and debating and that.

R: Just the characteristic of the class was kind of different.

P: Yea, I think it really did probably overshadowed, it was kind of a big
difference in style between the two terms.

R: That’s a good point. Okay, so then almost finished, would you say that the system kind of met your needs for the term?

P: Oh yeah, it exceeded, exceeded my needs. Now we’re even able to pull off results from the and then put them up on the website and having done that first term it was quick and easy second term. It fully met and indeed as I said, I’m going to end up talking about it in Norway because there’s some interest between classroom practice and game theory. So I’m going to give the talk and feature the software there.

R: What issues arose during the term with the system?

P: The only two were my being confused about how to get the clicker system restarted. I deviated from my whatever maybe before I always did it last and anyway I turned it around and didn’t realize. The difference was this, last term you ran it on your system and sent me files. This term I was running it on my own system and just forgot because I’m running it on my own system, the clicker interface needs to be reset. The second was not knowing once whether it did save. I think because we were set up to run three stages and decided after two that we had enough results, so there at that point I stopped and then didn’t get the grand total.

R: So then it sounds like that would be one of the aspects of the system that was maybe confusing, not clear. Where there any aspects?

P: Well the data files, they’re interpretable, it just takes a bit of.. That’s not a confusing part, it’s just a raw interface.

R: Can you elaborate a bit?

P: I would confuse myself forgetting that the pairs, the way the pairs lined up, that was really getting data on each of the pairs, but that duplicated other data. Dumping data like that into a spreadsheet and making sense of it is always... I think that’s not really a problem with the interface, it’s a prototype and that kind of gives you the data. It doesn’t feed into an analytics program. You can’t really complain about that. And I think all the data I wanted to get and all the presentations I wanted to get from it, I could do in fairly short order, it wasn’t “oh my god I can’t figure this out”.

R: Yea I kind of just put everything there.
**P:** What it did is it just invited me to make mistakes because now it’s a little wider than the screen and you have to say, it’s this and that and this, because if you pick up every column you’re going to mix two different - the player and the various opponents together. So you have to do these weird skipping over things and of course I might, I did sometimes get the columns that I..

**R:** Do you think it would be preferable to have.. because I think I did the partner score as well as the is that what you’re saying?

**P:** Yea, so you know, what you’ve got are basically non-homogenous rows, which means you can’t just do sums on them because some of those rows, some of those columns.

**R:** So it’s kind of just a pain in the butt to deal with, I see what you mean.

**P:** Yea, so I mean it’s enough. So you could go through and remove all them, but .. so that’s, I mean, a second look at that data would just show me one player’s homogenous would be easier to write excel stuff on.

**R:** It’s tough because there’s a different partner each round for some of the games.

**P:** Yea and it’s important for verification so you can take a look and say there were really different partners. I think it can have too little information and then you’re stuck and you really can’t say for sure that this is doing what it’s supposed to do, or you can have too much and you’ve got to do some sifting and I think for some experimental product it’s certainly better to err on the side of too much. Because it would be really a pain to say maybe you got paired with 10 different people, but maybe more like 8 because maybe, gee I don’t know, I have no way to tell who you were paired with. So I think for things we talked about in the design of it, that it kind of should be a good experimental platform, that means you can answer those questions. I mean it’s much better to have, I’ve complained about the tables I’ve designed in these experimental data because you can do an audit on it, it really does have an audit trail. It’s a pain, but it lets you say find out that you’re counting the actual responses from people. The mistakes it makes are virtuous. The awkwardness is kind of a virtue.
R: Would you say there’s anything missing or lacking in the software? Like you wanted to do something and you weren’t able to?

P: I think we fixed in the design what displays of results you get and you might want more flexibility than that I’m not sure. So we get a graph of results as we go along, but there are somethings I’d say, I’ll go process this and show you this other thing. You’d have to really go through cases and figure out if there are things standard enough that you’d want to be able to show them. So now we’re saying, since it was humans computers then computers humans, so it’s a little harder to tell just looking at what we’ve got whether the humans or the computers did better and did they do better consistently over the five rounds. But you can’t say in general what the other displays you’d like, but my suspicion is that probably one could cook down a few other display options for end of round data.

R: Yea I see what you’re saying. It would be nice if there was kind of a suite of components you could choose from to customize the results screen.

P: Yea, yea, I think it seems like that, down the road a bit, that “Oh, I’d really like to look at this this way” and to have that ...

R: And what would you say, which features or aspects of the system were the most valuable for you?

P: I think that you basically solved the random assignment to pseudonymously other player really worked well. I mean that cracked the problem of how do you actually do this in the classroom in a fun way. It was not N40520 versus... They were personified enough that people had the sense that they were actually playing a definite other person. I think that was really a good feature that worked, and kind of in a fun way, so that people weren’t irritated by. I mean the clicker software is sometimes annoying and it avoided that annoyance.

R: So last term people were maybe a bit excited about being the celebrities, was there a similar reaction this term?

P: Yea, yea I think. It’s also a prop in an interactive performance. It’s not just collecting data, it’s not putting people in a room and collecting data from them it’s actually a part of interactive stage performance, a classroom, so I mean typically yknow we’re waiting on somebody and yknow you say
“Come on Tony or Barb or Bond” and why are some of them celebrities and some of them the actors? Why are some actors and some characters? There were things you could play with a bit without really nagging how come this person for all I know for very good reasons takes always 35 seconds than 10. And so I think that worked out really well. So that design choice was good, it gave you a bit like a muppet thing. You could make fun of that the muppet is blue without anyone being yknow. Do you feel nvidia(?) is being paired up with and that kind of stuff and nobody is, you’re not really, you’re talking at versions of people that aren’t. So I think that was a good thing, and numbers wouldn’t do it. As we’re tightening it up in an engineering way and you’re going that style, I think it would be a mistake because partly what you’re doing is, it’s a prop and it works really nicely as a prop.

R: That’s interesting. So do you think that say we used shapes, would it be totally different?

P: As it turned out, with this experience, I think that would be really different. There might be problems extending this to larger groups. And you could imagine somebody with a principle complaint of “Why did I get a racially aberrant...” or there were no people of my gender. You could imagine stuff like that, but I’m really glad we didn’t imagine those objections and then make it either more engineery or more bauhaus bland. It had a little bit of edge to it, and the thing is that’s good, it’s a performance. It’s good to have a little weird thing that you can focus on because partly what you’re doing is, between rounds and stuff, giving people other things to come on folks. So think of whatever, it’s Craig Ferguson, and it’s weird chicken jokes. I mean, you’ve got a crowd that you’re trying to involve and I think the interface worked really well for that.

R: This term, was there anything that you didn’t expect to be useful but ended up being useful or you didn’t realize how useful it would be?

P: I think we got a sense of that last term. It continued on. It was fairly fun to, it wasn’t, a mixture of it’s not a nerve-wracking thing to use, say animations in PowerPoint. Think of it often in a presentation and “oh my god, I have to show a movie in the middle, will the connection work, or do I need a local copy?” and it didn’t have that feature. It was run locally,
it didn’t need, it turned out that room had a terrible internet connection and it turned out running it locally and quite secure, yknow use obvious connections once you understood the clicker thing and on the other hand it was fun.

R: And you mentioned that you found using the clickers to be more comfortable than clicking and using the keyboard or whatever?

P: Yea, I use a Macbook Air and it’s tiny. Y’know, just remembering what the keymappings are because there’s three pieces of software in play. Typically I’m using PowerPoint, the clicker stuff and your thing and which ones want this? Is it return or space or? Or there’s VLC, or whatever all the other things. The point is that with that in hand, it’s very easy to click through it and you can walk around because then you’re walking around and sometimes you don’t notice this cell is really being ignored by people, what’s going on here? That kind of stuff. That worked well.

R: Would you say overall that the goals you set out to accomplish were reached while using the system?

P: Yea I think that more than reached because it also was suggestive of these new problems and further developments and using it in another course.

R: What are these new problems?

P: Led me to think more of how would you really try to crack the motivation problem? What would be fair ways to grade games given that they have this random element? I think that it works so well makes you then want to use it further and you’d like to be able to say to students, yeah you can propose a game and we can test people on it, but that would open up interesting research ethics problems. I think that’s the sign of a good, more than of “argh how can we get it finally to work next term?” it’s rather like “Oh, what new things could we do with it?”

R: You said you’re using it in COGS 200 or thinking about it?

P: There’s some thought of, I’m switching to teaching COGS 200 for various reasons but, one thing about COGS 200 is that the assessment has been a little bit over, I mean too much writing, maybe not enough the most useful kind, and so what I proposed taking the quizzing the system I use there which would have people have clickers. So once people have clickers
then, I would like to also use the game software on them. So the people so far I’ve talked to about it, they’re happy about that, I mean at least one of the people that would replace me in 300 wouldn’t want to do much game theory and people like that there is some game theory done in COGS, so that would be mean... But that will change. Doing it with 120 second year students than 30 third year students. They’ll be more bitchy problems “ohhh how come my name is this? Or I forgot how to do this? I thought this?” I think probably for that the assessment problem is the more important side. You’re going to need to have participation grades or just that you played or not or whatever.

R: Do you have suggestions or directions that this system could move in for that expanding. You mentioned maybe having options for the different results screens, but beyond that do you have other suggestions that could be done to improve on?

P: I think that if anyone less geeky is to use it, I think probably the output of it is going to have to be a little more, a little simplified maybe.

R: Like a report generated or something?

P: Yea, I mean maybe there’s a complete report and there’s just by player because it would be handy if someone could generate. I think that’s probably... The grading thing just opens up other stuff, I mean because if you actually used it with some kind of grading thing you’d also like it finally to interact with the real clicker software, but that’s probably not worth it. I mean it’s not intended to be a grading system, that they’re separate maybe is smart.

No I mean actually I haven’t sat and tried to put a game in it from scratch, so that just says it. So probably if you’re going that direction it would have to be at that level of interface.

R: Yea, certainly that’s not ready yet.

P: But I mean it’s just. You probably know about that it’s just. So yeah.

R: OK thanks. Do you have any other comments before we finish up?

P: No I mean, it’s been really good partnering up. I hope that’s been useful to you. It’s certainly been a good experience. It wasn’t at all a cost
on using it in the classroom, it was much more an enhancement to a class
and as I said I’m going to talk about it in a research setting this summer.
And introduce it most likely to another, bigger class if I can next year.

R: Actually I remembered I have one final follow-up question. Would
you prefer having a TA there operating the system? Or was the pressure not
high enough to be significant to?

P: The benefit of having someone not operate is it’s more flexible, the
pacing, and that performance side works better I think. I guess depends on
the context. Aware as I am of the price of labour, you don’t want a system
where, I mean all the AV systems have now gone to expecting the instructor
simply to use them, right? But in another context you could imagine one of
the students, you could imagine someone, an experimenter wanted someone
to go through a bunch of classes and try a game on them and in that case
you would like some third party, not instructors. For the way we’ve thought
of it as an instructor tool, I think it is better that the instructor gets to do it.
The only thing is it is somewhat demanding on the analysis side. Somebody
has to be comfortable figuring out what the data is saying.

Yea I guess maybe the little slight, probably using the clicker raises ex-
pectations that you have to deal with. Because the clicker could be used
by someone who never looked at the data files, just let the clicker software
crunch them for them. And so it’s operating in class, but it’s also a raw
interface for data.
Appendix F

Rhombus Instructor Manual

This appendix contains the manual provided to the instructor to guide his independent use of Rhombus. Note that at the time the ID Server in Rhombus was referred to as the Clicker Aliaser.
I. Rhombus Participation System

1. Getting Started
   - Requirements
   - Installation
   - Starting the System
   - Stopping the System
   - Configuration
     - Clicker Server
     - Clicker Aliaser
     - Rhombus Participation System

2. Registering Aliases
   - Deleting Aliases

3. Playing Games
   - Controller Interface
   - Viewer Interface
   - Game Results
   - Additional Logs
   - Testing Games

4. References
Rhombus Participation System ♦

**Rhombus Participation System (RPS)** is a web system that allows people to interact with each other through a shared display. It requires a Participation Server to be hooked up to it in order to receive choices from participants. In this package, the Participation Server is the Clicker Server + Clicker Aliaser, which allows participants to make choices via i>clickers.

![System Diagram](image)

The **Clicker Server** is the backend server that listens to the i>clicker base station and outputs clicks over a socket. The **Clicker Aliaser** is an intermediate server that stands between the web server and the clicker server in order to translate clicker IDs into the aliases that show up on screen. The **Web Server** is where RPS resides and it handles all the displays and application logic.

## Getting Started

### Requirements

In order to run RPS, you need to have the following software installed:

- **java** development kit required for the Clicker Aliaser and Clicker Server *(tested with v1.6.0_65)*

- **sqlite3** database required for the Clicker Aliaser and RPS *(tested with v3.7.3)*.

    Most easily installed via Homebrew with the following command:

    ```bash
    brew install sqlite3
    ```

- **node.js** platform required for RPS *(tested with v0.10.12)*

    Download from the [node.js website](https://nodejs.org) and install.
- grunt tool required for RPS *(tested with v0.1.9)*

Install via npm (comes with node.js) with the following command:

```
npm install -g grunt-cli
```

Note: The Clicker Server has only been tested on Mac OS X (10.8+), but should work on any system given the proper HID API library for the clicker driver.

**Installation**

Once all the requirements have been installed, to set up RPS, you simply need to unpackage `clicker_package.tar.gz` to a convenient location. You will be accessing the expanded `clicker_package` directory via the terminal to start and stop the system.

**Starting the System**

Open a terminal at the expanded `clicker_package` directory and run the `start.sh` script found therein.

```
./start.sh
```

This script does the following in order:

- Starts the **Clicker Server** at localhost:4444 by default (See Configuration - Clicker Server)
- Starts the **Clicker Aliaser** at localhost:4445 by default (See Configuration - Clicker Aliaser)
- Starts the **Rhombus Participation System** ("Clicker Web Games" in this package) at [http://localhost:8000](http://localhost:8000) by default (See Configuration - RPS)
- Starts the **Clicker Aliaser web interface** at [http://localhost:8008](http://localhost:8008) by default

**Typical Output Example**
Stopping the System

To shutdown RPS, in the terminal window that is running the `start.sh` script, kill this process.
by pressing `CTRL+C` at the command line.

**Configuration**

All of the servers support a small level of configuration to change basic things such as the ports they run on and databases they use.

**Clicker Server**

To configure the port the Clicker Server runs on as well as the i>clicker base station properties, edit `clicker_package/clicker_server/config.properties`. The following keys are supported:

<table>
<thead>
<tr>
<th>Key</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>4444</td>
<td>The port the Clicker Server runs at. If this changes, update the Clicker Aliaser accordingly.</td>
</tr>
<tr>
<td>instructorId</td>
<td>371BA68A,0D93B52B</td>
<td>The clicker ID of the instructor’s i&gt;clicker. Multiple instructors are supported by using a comma separated string. The instructor’s remote can be used to toggle accepting choices and to move between states. It can be disabled via the Instructor Controller button in the RPS admin.</td>
</tr>
<tr>
<td>channel1</td>
<td>A</td>
<td>The first letter in the frequency channel for the clicker base station.</td>
</tr>
<tr>
<td>channel2</td>
<td>A</td>
<td>The second letter in the frequency channel for the clicker base station.</td>
</tr>
</tbody>
</table>

**Clicker Aliaser**

To configure the port the Clicker Aliaser runs on as well as which Clicker Server it connects to, edit `clicker_package/clicker_aliaser/config.properties`. The following keys are supported:

<table>
<thead>
<tr>
<th>Key</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>4445</td>
<td>The port the Clicker Aliaser runs at. If this changes, update RPS’ configuration as well.</td>
</tr>
<tr>
<td>clickerServerHost</td>
<td>localhost</td>
<td>The host where the Clicker Server is running.</td>
</tr>
<tr>
<td>clickerServerPort</td>
<td>4444</td>
<td>The port the Clicker Server is using.</td>
</tr>
</tbody>
</table>
Rhombus Participation System

To change the port at which RPS connects to its participation server (in this case, the Clicker Aliaser), update `clicker_package/clicker_web_games/fwconfig.json`. Change the `participationServer.port` key accordingly.

To change the port where RPS runs (default 8000), open `clicker_package/clicker_web_games/Gruntfile.js` and change the key `socket-server.dev.options.port`.

**Registering Aliases**

A simple way of managing aliases is to go to the Clicker Aliaser Web Register Page at http://localhost:8008/register (henceforth referred to as the Register Page). Note that the system has to be running for this to be accessible.

To register a single alias, go to the Register Page and under Manual Registration enter the clicker ID in the Participant ID field and the corresponding alias in the Alias field and then click Register. If successful, the new alias will appear at the top of the table on the right side of the page.

The simplest way to register a bulk amount of aliases is to use the companion Excel file (ClickerAliases.xlsm) to generate SQL (output filename from the Excel macro is aliases.sql). With the SQL handy, go to the `clicker_package/clicker_aliaser` directory and run it on the aliaser database as follows:

```
pbeshai: ~/Workspace/research/clicker_package/clicker_aliaser $ sqlite3 aliaser.db < /path/to/aliases.sql
```

You can verify the aliases that are currently registered by going to the Register Page and reviewing the table on the right side.

**Deleting Aliases**

To delete an individual alias, go to the Register Page and click the × beside the alias you’d like to remove.

To delete all registered aliases, click the Delete All Aliases button on the Register Page.

**Playing Games**

To use the RPS system, you need to start the servers then load the Controller interface in your web browser. The URL to access the controller is:
The general procedure is as follows:

1. Start the servers using `clicker_package/start.sh`
2. Go to `http://localhost:8000/m1/controller`
3. Open a new main Viewer
4. (optional) Open a new Instructions viewer if you have an extra display to solely show the instructions
5. Select the game by clicking the button under App Selector
6. (optional) Modify the configuration
7. Enable accepting choices by clicking the “Not Accepting Choices” button. You can also do this on the instructor’s remote by pressing A.
8. Have participants click to play
9. Toggle through states by clicking the Next State and Previous State buttons. You can also do this on the instructor’s remote by pressing C for next state and D for previous. Be sure to reach the final state to get the final output log generated.

**Controller Interface**

The Controller Interface is the main interface the instructor/admin uses while running
applications/games. In this interface you can do the following:

- select the app you would like to run
- review the list of open Viewers
- open new Viewers
- toggle on and off accepting choices
- toggle on and off the instructor’s controller. If disabled, the controller functions as a normal participant’s controller. If enabled, it uses the following mapping:
  - A Toggles on and off accepting choices
  - B Nothing
  - C Goes to the **next state** of the application
  - D Goes to the **previous state** of the application
  - E Nothing
- add or remove a countdown timer
- configure a selected application. When updating a configuration, whatever is entered in the message field will be saved to the logs of the game for future reference. It exists so the instructor/admin can leave a note for why they changed the configuration.
- review which participants are actively playing
- add in latecomers through the queue. Note that not all states support adding latecomers. Play states will add in latecomers automatically when they are loaded, but if somebody joins in the middle of a play state, they will either have to wait until the next play state before they are added, or they can be added in manually via the *Add New Participants* button that shows up on demand.
Controller Interface (annotated, no app selected)
Controller Interface (annotated, Prisoner’s Dilemma (teams) selected)
Controller Interface, Configuration (*Prisoner's Dilemma (teams)* selected)

**Viewer Interface**
Viewer Interface: Prisoner’s Dilemma (teams) Results screen

Viewers can most easily be opened up from the Controller interface by clicking the Open New Main Viewer button. If there is not enough screen space to show the instructions along with the main view, you can open an Instructions only viewer by clicking the Open New Instructions Viewer button. This is handy when you have multiple displays at your disposal. Otherwise, you may have to resort to sharing the instructions over different media (e.g., whiteboards, handouts).
The Viewer Interface will generally have three components: the Status Bar, the Application Pane and the Instructions Pane.

The Status Bar will always be there regardless of the application being played. It displays the status of choices (whether participants are able to make choices or not) as well as a zoom control. The zoom control is slightly different from the built-in web browser zoom in that it only magnifies the Application and Instructions panes in the current window. Built-in browser zoom will magnify the status bar as well as all other browser windows that are open at the same host (e.g., the Controller interface will be affected too).

The Application Pane is where all the application views show up. Typically this involves a visual representation of each participant as well as what choice they've made and what current state they have (e.g., their score). Note that in games where participants are paired and the choices of both players are shown on the results screen with their score (e.g., “AB 5”), the choice on the left (e.g., “A”) belongs to the player whose box it is in and the choice on the right (e.g., “B”) belongs to the partner. The score (e.g., 5) is for the player whose box it is in.

The Instructions Pane shows the instructions for the current state of the application. This pane isn’t always static as you may have different instructions for different states (e.g., playing vs
results, or playing as role A then role B).

**Game Results**

All logs and results from game play can be found in the directory `clicker_package/clicker_web_games/log`. The contents of that directory are described below:

<table>
<thead>
<tr>
<th>Filename</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coin-matching/</td>
<td>Directory containing the results of the Coin Matching Game</td>
</tr>
<tr>
<td>logger.js</td>
<td>Configuration file for RPS logging (should not be modified)</td>
</tr>
<tr>
<td>pd/</td>
<td>Directory containing the results of Prisoner’s Dilemma</td>
</tr>
<tr>
<td>pdm/</td>
<td>Directory containing the results of Prisoner’s Dilemma (multiround)</td>
</tr>
<tr>
<td>pdn/</td>
<td>Directory containing the results of Prisoner’s Dilemma (N-person)</td>
</tr>
<tr>
<td>pdteam/</td>
<td>Directory containing the results of Prisoner’s Dilemma (teams)</td>
</tr>
<tr>
<td>q/</td>
<td>Directory containing the results of the Question app</td>
</tr>
<tr>
<td>stag-hunt/</td>
<td>Directory containing the results of Stag Hunt</td>
</tr>
<tr>
<td>ultimatum/</td>
<td>Directory containing the results of the Ultimatum Game</td>
</tr>
</tbody>
</table>

Note that intermediate logs are saved after each round in the `rounds` subdirectory. For instance, to find an intermediate log for the Ultimatum Game, you’d look in `clicker_package/clicker_web_games/log/ultimatum/rounds`. The final results of the game are stored at the top level (e.g. `clicker_package/clicker_web_games/log/ultimatum`).

**Additional Logs**

The system logs every click and every action in the games. To find more verbose logs, look in these locations:
<table>
<thead>
<tr>
<th>Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>clicker_package/clicker_web_games/log/server.log</code></td>
<td>The verbose output from the Web Server (RPS). Game state logs will be shown here. Also available in JSON form as <code>server.log.json</code>.</td>
</tr>
<tr>
<td><code>clicker_package/clicker_server/log/clicks.log</code></td>
<td>The log of all the clicks that came into the base station.</td>
</tr>
<tr>
<td><code>clicker_package/clicker_server/log/server.log</code></td>
<td>The log of all the actions the Clicker Server has taken.</td>
</tr>
<tr>
<td><code>clicker_package/clicker_aliaser/log/clicks.log</code></td>
<td>The log of all the clicks that came from the Clicker Server to the Clicker Aliaser.</td>
</tr>
<tr>
<td><code>clicker_package/clicker_aliaser/log/server.log</code></td>
<td>The log of all the actions the Clicker Aliaser has taken.</td>
</tr>
<tr>
<td><code>clicker_package/clicker_aliaser/web/log/server.log</code></td>
<td>The verbose output from the Clicker Aliaser web interface. Also available in JSON form as <code>server.log.json</code>.</td>
</tr>
</tbody>
</table>

**Testing Games**

Sometimes you’ll want to have a test run through games without having dozens of clickers available. To do this, you can access the debug interface at:

```
http://localhost:8000/m1/controller/debug
```

This page lets you add in Web Clickers that will simulate regular clickers in games.

**References**

- SQLite [http://www.sqlite.org/](http://www.sqlite.org/)