02122 Project course, Spring 2014
DTU Compute
Technical University of Denmark

Katabasis
Group 6.1

Carsten Nielsen
Cebrail Erdogan
Philip Berman
Jonathan Becktor

May 19, 2014

Abstract

Using the Arduino hardware with Gameduino 2, well create an advanced game. The game will have elements as AI and Map generation. Coding in arduinos environment and make it work in the limited hardware is quite a challenge and fun.

Contents

Chapter 1

Introduction

The main purpose of the project is to make an advanced game using the Arduino hardware. Arduino is a programmable piece of hardware. Combining it with the Gameduino 2 makes it possible for us to create a rich game. The Gameduino extends the Arduino with a touch screen, extra space and processing power.

Chapter 2

Problem Analysis

Define domain specific concepts to be used in the rest of the report.
Explain the problems considered, features to be implemented, etc.
Possibly merge with the next section.

Chapter 3

Requirements specification

Be more precise about what the system should and should not satisfy. Be sure to use the vocabulary introduced in the problem analysis.

3.0.1 Timeplan

Our timeplan follows an agile development system called waterfall. The waterfall system is a sequential design process. It is designed to get through the project phases and have a product as soon as possible. The phases in our project can be seen in the figure below.

Project management When the waterfall ends and we still have time we will go back to the start and check for new requirements and the whole waterfall process starts again. The report is both written seamlessly and separately in the end of the project, therefore it does not has to be illustrated, as it will create confusion.

Tasks management When providing the tasks we use the timeboxing method. This method increases the productivity significantly. Timeboxing works by breaking a big task into smaller tasks with better manageable time frames. It kinda works like a stopwatch.

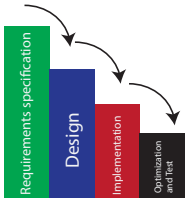

Figure 3.1: An overview of the waterfall timeplan.

Chapter 4

Overall design

Describe the overall structure of the system, the different components of the system and interfaces between these.

Chapter 5

Implementation and detailed design

Present implementation details, program structure etc.

Further, interesting algorithms and data structures should be presented here.

5.0.2 Game loop

Essentially the game is one long loop, which ends whenever the game does. Each iteration executes the gamelogic per frame, such as moving the player and monsters around. Gameloops vary from simple while(true) loops to dynamic separate gameloops handling separate calculations (such as separating rendering from gameplay).For this we only needed a simple game loop, updating the world in each iteration. To ensure a consistent gameplay experience, we needed to know how much time passes betweenframe updates. Without this, the game would run faster or slower on different processors or inconsistently in different ingame events, throwing off the players sense of timing. For example, the game would be equally playable with either 30 or 60 frames per second. If the difference isnt incorporated in the gamelogic though,the game would run twice as fast with 60 frames per second! Our implementation of a frames per second counter, is simply calculating how many frames was calculated the last frame, and then assuming the next frame has the same amount. This is a pretty simple solution, which could easily be expanded upon if needed. The obvious criticism of this method, is that the frame calculations between each second are not accounted for, and that the framerate is essentially a second behind any framerate changes.

5.0.3 Movement

The movement of our hero will mainly be right and left. He will have also have the ability to jump. Beside the basics movements, our hero will also have fight moves. The fight moves contains moves like sword swinging and archery.

Input

The Gameduino 2 comes with a touch screen and accelerometer. These modules gives us an opportunity to interact with the game in many ways. We could use the accelerometer or the screen to move the hero. We dont have much experience with the screen and its capabilities. Therefore another option as backup is preferable.

Another option will be using an external controller. The Wii nunchuck is popular and is very suitable for this game. Using the buttons we game can be controlled as seen in the figure below.

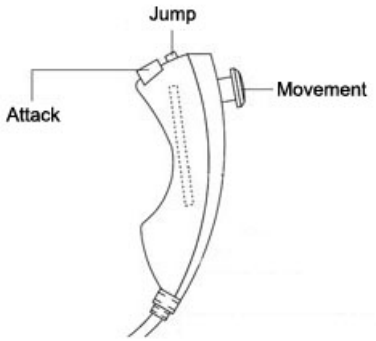

Figure 5.1: Button specifications

Chapter 6

Hardware

Here are the specifications of the Arduinos we have:

bleh bleh

6.0.4 input

The Wii nunchuk is one of the ways to control the game, aside from the touchscreen. The nunchuk has to be connected to the Arduino by hardware. Even though they use same slots in the board, it is possible to use both the Gameduino 2 and Wii nunchuk at the same time because they use different hardware interfaces. The Gameduino uses an ISP interface, while the Wii nunchuk uses an I2C interface. The Gameduino requires 5 volt to work, which forces us may shorten the lifetime of the nunchuk - as it normally operates at 3.3 volt, but it should not be a big concern.

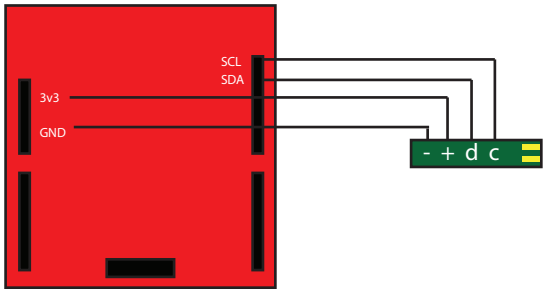

Figure 6.1: Hardware connection of an nunchuk

6.0.5 I2C

The I2C is the interface, which is used by the Wii nunchuk adapter. This bus interface allows easy communication between components and only requires two bus lines. These lines are both bidirectional. These bus lines are called SCL (Serial Clock Line) and SDA (Serial Data Line).

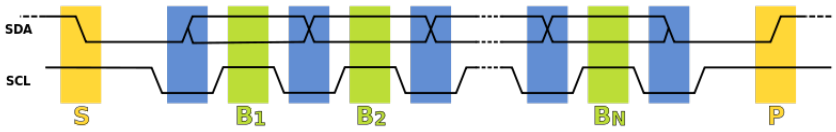

Figure 6.2: Hardware connection of an nunchuk

1. Data transfer is initiated with a START bit (S) signaled by SDA being pulled low while SCL stays high.
2. SDA sets the 1st data bit level while keeping SCL low (during blue bar time.) and the data is sampled (received) when SCL rises (green).
3. When the transfer is complete, a STOP bit (P) is sent by releasing the data line to allow it to be pulled high while SCL is kept high continuously.
4. In order to avoid false marker detection, the level on SDA is changed on the SCL falling edge and is sampled and captured on the rising edge of SCL.

Chapter 7

Testing and performance analysis

Present test methodology as well as results in this section. In addition, if performance analysis of the system is interesting, present it here as well.

A few screenshots of the program can be included here as well.

Chapter 8

Discussion

Discuss the problems, challenges and solutions encountered. What did you learn? What gave you troubles? Interesting future extensions and improvements of the system can be discussed here as well.

Chapter 9

Conclusion

Summarize the main results. This section should make sense even if the reader has only read the introduction.

References

List your references here.

Chapter 10

Appendices

E.g., examples, screenshots, test cases, tables, if any.

Include source code in the electronic version of the report