Gesture-Controlled LED Coffee Table with Tetris

Project Proposal

Josh Pack | Esther Kim | Ryan Dwyer

Project #16

September 19, 2012

ECE 445 Fall 2012

TA: Lydia Majure

Contents

1. Introduction

Those classic, oversized arcade games like Pac-Man and Space Invaders just don't fit in with the atmosphere of a relaxing, stylish living room. We want to solve this problem by putting a modern twist on a retro video game. Our idea is to design and build a coffee table with an LED matrix display surface on which we can implement the game Tetris. We will use microcontrollers to control the LEDs. Some projects along these lines have already been completed, but we want to make the game much more interactive by adding motion sensors, which would allow control of the game through hand gestures. We also want to make the appearance of the table more refined, adding light-diffusing films above the LEDs to give a warm glow. The surface will be sturdy enough to be used as a common coffee table, as well.

Objectives

Goal: Our project goal is to have a fully functional coffee table with an LED matrix display surface that can play Tetris using gesture recognition technology.

Functions: The intended functions are to have a coffee table that users can play Tetris using hand gestures above the table to drop, move, and rotate the pieces. The Tetris game will have the basic functions of removing a row when it fills, randomizing the next piece, and ending the game when a piece has reached the top row.

Benefits: Benefits of the LED Tetris Coffee Table include:

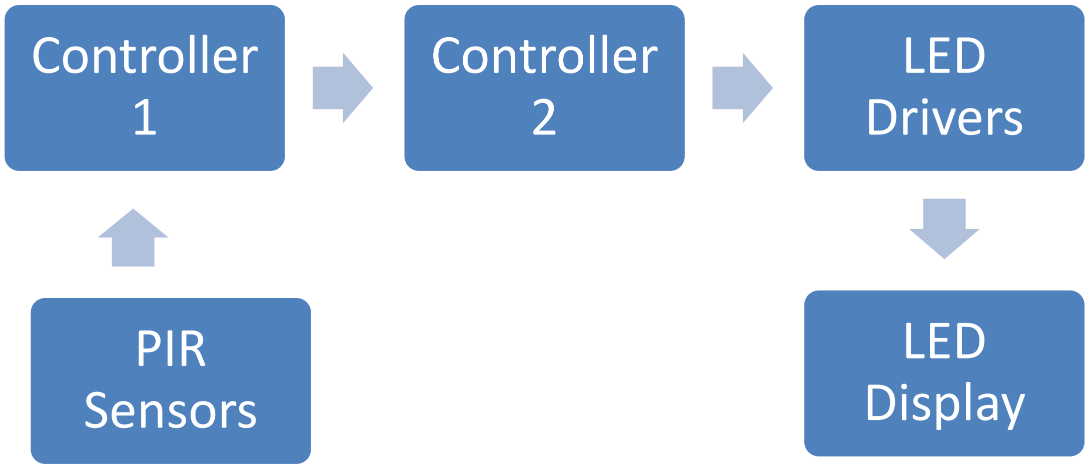
- Entertainment to users with the Tetris game
- Aesthetically pleasing in the living room
- Supportive coffee table

Features: Features of the LED Tetris Coffee Table include:

- Rows are removed when it fills
- The next piece is randomized
- Game ends when a piece reaches the top row
- Drop, move, and rotate are controlled with hand sensors

2. Design

Block Diagram



Block Descriptions

Controller 1:

(1) Arduino Uno board
Controller 1 will be an Arduino Uno used to run the main program, which will take input from the PIR sensors to determine the user input and send control signals to Controller 2. This controller will keep track of the locations of the game pieces and the progress of the player. Controller 1 will notify controller 2 when a row is cleared or when the player has lost.

The Arduino Uno board has 14 digital I/O pins, 6 analog inputs, a 16 MHz crystal oscillator, and USB connection. The board can be powered through USB, an AC-DC adapter, or a battery. Operating voltage is 5V; recommended input voltage is 7-12V.

Controller 2:

(1) Arduino Uno board
Controller 2 will be an Arduino Uno used solely to interface with the drivers to render the display based on the received control signals from Controller 1. Having a second controller dedicated to creating the display will allow us to ensure a smooth image with minimal flickering while the first controller deals with executing the Tetris program.

(See controller 1 description for specs)

LED Drivers:

(7-13) TLC5940
The LED drivers we will use are the TLC5940, which use PWM to control the LEDs to produce the appropriate color from the 7 options in our Tetris design. They also supply the proper driving voltage for the LEDs without having to connect each LED to a power supply through a resistor. Controller 2 will send control signals to the appropriate LED drivers to create the desired display on the LEDs.

The TLC5940 features 16 channels with dot correction and grayscale PWM control. DC operating voltage is 3V-5.5V. Power dissipation rating is 2W-4W. We expect to use between 7 and 13 of these to be able to properly control all 200 RGB LEDs.

LED Display:

(200) RGB LEDs
These RGB LEDs will be arranged in a 10x20 matrix such that each RGB LED represents one pixel. These LEDs will have diffused lenses so that any colors which combine the red, green, or blue will look pure rather than looking like multiple, separate colors. They will be controlled by the TLC5940 LED drivers, which use PWM for each of the red, green, and blue components to create the desired color. Our Tetris implementation will consist of seven different colors for game pieces.

We are planning to use 5mm 4-pin RGB diffused, common anode LEDs. These have an operating voltage of approximately 2.1V-3.4V.

PIR Sensors:

(3) Passive-Infrared (PIR) sensors
These three sensors will be used to implement gesture control of our Tetris game.

The sensors have an operating voltage of 3.3V to 5V and a sensing range of up to 20 feet.

3. Block Level Requirements and Verification

Requirements

1. Controller 1 must be able to interpret each of the five input controls from the PIR sensors.

2. Controller 1 must keep track of the position of the game pieces so that the pieces will stack properly and completed rows will be removed properly.

3. Controller 1 must send control signals to controller 2 with instructions for the next cycle of the display.

4. Controller 2 must be able to translate the control signals from controller 1 in order to properly direct the LED drivers.

5. The LED drivers must provide each LED in the display with the proper operating voltage, which ranges from 2.1V to 3.4V

6. The LED drivers must use proper PWM on the red, green, and blue portions of each RGB LED such that each LED can light with each of the seven necessary colors (cyan, blue, orange, yellow, green, purple, red in accordance with The Tetris Company's color scheme).

7. Each LED must light with each of the seven necessary colors (cyan, blue, orange, yellow, green, purple, red in accordance with The Tetris Company's color scheme).

8. The sensor block must be able to distinguish five control motions for the falling Tetris pieces: drop, move left, move right, rotate clockwise, and rotate counterclockwise). These control motions must be detectable at a reasonable height from the surface of the table, which will be no more than 2 ft.

Verification

1. For initial testing, we will have controller 1 output five separate signals to respective LEDs so that only one LED will light at a time for each of the possible control motions.

2. Before implementing our full Tetris game, we will send individual pieces to the display without any user control to make sure they stack on top of each other. A second test program will drop pieces in a complete row to make sure controller 1 detects a complete row and removes this row.

3. We will use the test program mentioned in verification (2) to ensure that the display updates at a smooth rate and that the control motions are actualized on the display in a responsive manner with no more than 20ms delay between control motion and piece movement.

4. We will use the test program mentioned in verification (2) to ensure that the pieces enter the display from the top and accumulate at the bottom as expected.

5. We will test each output pin of the LED drivers with a voltmeter to ensure that the output voltage is in the operating range of 2.1V to 3.4V

6. We will connect one RGB LED to the output of one of the LED drivers with the controllers sending the proper signals to the driver to create the desired colors. We will repeat for each of the seven colors to make sure our duty cycles are visibly correct for each color.

7. We will create a simple initializing routine on the microcontrollers to cycle all of the LEDs through all seven of the desired colors simultaneously. We will check to make sure all pixels visibly produce the same color.

8. We will connect each of the sensor outputs directly to an LED so that we can observe the proper LEDs lighting when each of the control motion combinations is performed above the sensors. We will measure with a ruler to make sure the maximum distance range of detectable motion for each of the sensors is between 1ft and 2ft.

Tolerance Analysis

The component that will most affect the performance of the project is the synchronization of the LEDs and its response time to gesture controls. This component matters the most because a responsive and synchronous display will influence the functionality of the Tetris game, as well as user interaction and overall enjoyment of the game.

To test the responsiveness of the system, we will start with a new Tetris game. We will set the middle PIR sensor signal to high for the Tetris pieces to drop continuously and as fast as possible. We will measure the time from the piece settling on the surface to a new piece appearing at the top of the display. We will also measure the gravity rate and the drop rate.

The expected time it takes to settle the old piece and have a new piece appear should be the same as the gravity rate. We will aim to have the gravity rate to be 500 ms per row. The tolerance for this time measurement is +/- 250 ms. The drop rate will be twice as fast, or 250 ms per row. When testing the drop rate, we will allow a tolerance of +/-125 ms.

4. Cost and Schedule

Cost Analysis

Parts

Part Name	Amount	Cost per Unit	Price (including shipping est.)
Arduino Uno Microcontroller	2	$23	$65
RGB LED	300	$12.34/100	$45
TLL5940 LED Driver	7	$3.50	$30
PIR Sensor	3	$10	$35
PCB	1	$0	$0
Coffee Table	1	$20	$20
Light Diffusion Grid	1	$10	$15

Labor

Group Member	Rate	Total Number of Hours	2.5 x Total
Josh Pack	$40/hr	150	$15,000
Esther Kim	$40/hr	150	$15,000
Ryan Dwyer	$40/hr	150	$15,000

Parts Total Cost: $ 210
Labor Total Cost: $ 45,000
Grand Total Cost: $ 45,210

Schedule

Week of	Milestone	Overall Task	Josh	Esther	Ryan
09/16	Proposals Due	Brainstorm design Modulate requirements	Get feedback from Lydia on gesture control and initial design Order Bundle 1* of parts		
09/23		Finalize design Gather parts	Order Bundle 2** of parts Write pseudo code for Tetris game		
09/30	Design Review	Feedback on design Tweak design Construct	Design LED matrix on Eagle	Program syncing and rendering on 2nd microcontroller	Program Tetris on 1st microcontroller
10/07		Construct	Submit PCB request Build table with light diffusion grid and sensors	Program syncing and rendering on 2nd microcontroller Write test cases for integration with LED matrix	Program Tetris on 1st microcontroller
10/14		Construct Integrate	Integrate LED matrix with 2nd controller Test LED matrix for synchronization and rendering		Program Tetris on 1st microcontroller
10/21		Integrate	Integrate 1st microcontroller with the 2nd microcontroller and LED matrix Integrate 1st microcontroller with PIR sensors		
10/28		Integrate Test	Integrate 1st microcontroller with the 2nd microcontroller and LED matrix Integrate 1st microcontroller with PIR sensors Start testing sequence		
11/04	Mock-up Demo	Feedback on performance Debug	Debug and test	In charge of presentation PowerPoint	Debug and test
11/11	Mock-up Presentation	Debug	Debug and test		
11/18	Thanksgiving Break				
11/25		Debug Finalize	Debug and test	In charge of documenting test results	Debug and test
12/02	Final Demo	Finalize	Debug and test	In charge of presentation PowerPoint and paper	Debug and test
12/09	Final Presentation Final Paper Due	Finalize	Debug and test	Debug and test	Debug and test

*Bundle 1 includes Arduino microcontrollers, LEDs, LED drivers, PIR sensors

**Bundle 2 includes coffee table, light diffusion grid, additional hardware as needed