Motion Control Gaming with Kinect

Jonathan Fong
jonathanfong3.14159@gmail.com

Jacob Gorman
jacobgormandevelopment@gmail.com

Zachary Panzarino
zachary@panzarino.com

Steven Topper
sttopper@ctemc.org

Sara Weill
saracweill@gmail.com

New Jersey Governor's School of Engineering and Technology

22 July 2016

Abstract

The Kinect motion-sensing device, created by Microsoft, allows users to play games by interacting with both the device and the interface without any kind of handheld controller. The motion control technology that the Kinect implements has the potential to impact vast aspects of everyday life, such as virtual robotic control, rehabilitation, and exercise.

In this study, multiple Kinect games were created using the Microsoft Kinect device and the programming language Python. These games were created with an aspect of exercise in each to suggest how these games can be implemented in exercise and rehabilitation regimes, creating an encouraging environment that athletes and patients can use in the comfort of their own homes.

1. Introduction

The Kinect is part of a line of motion-sensing devices created by Microsoft for the Xbox video game console. Its motion control sensors allow users to play games and interact with the device without using any kind of handheld controller. The natural user interface (NUI) allows players to control games intuitively with both voice and motion control [1]. This study implemented the programming language Python and several game modules that interact with the Kinect: Python 2.7, Pygame, PyKinect, Kinect for Windows SDK, and the Kinect Developer Toolkit. Python is a relatively new language as well as an extremely versatile one, allowing developers to take full advantage of recent technological advances.

Motion control technology has the potential to impact vast aspects of daily life, including medicine, physical therapy, and exercise. Recent studies have shown that using repetitive activities on a virtual interface over a long period of time is helpful for both physical and mental recovery [2]. With the help of affordable Kinect motion-sensing devices and personalized exercise games, patients and other users can achieve rehabilitation and exercise anywhere, including at home. The games created in this study focus on the possibility of physical exercise in motion-control gaming through activities such as push-ups, running in place, and sprinting for short distances.

2. Background

2.1 Kinect Systems Overview

The Kinect device has two infrared (IR) sensors, a color sensor, a multi-microphone array, and a tilt motor (Appendix Figure 1). The RGB color sensor camera makes capturing color image possible, storing the data in 1280x960 resolution. One of the infrared sensors emits an IR light (see below) and the other reads the IR beams reflected back at the sensor. These two sensors make capturing depth information possible [3]. The multi-array microphone contains four microphones for capturing sound. The four microphones can identify the source of sounds, identify the direction of the audio wave, and filter out background noise [4]. The tilt motor can automatically adjust the sensor based on user. If the user is short, the Kinect sensor will angle down, and if the user is tall, it will angle itself up [4].

2.2 Technology Behind the Kinect

The Kinect needs to know where the user's body is relative to the machine. Finding body position is a two part process. First, it must create a depth map, and then it must infer the user's body position. Once the Kinect carries out these two steps, it can interpret the gestures made by a user, carry out commands, and give feedback based on the user's motions.

2.2.1 Creating a Depth Map

The Kinect has two 3D Depth Sensors (Appendix, Figure 2) and an RGB Camera. The Kinect creates a depth map by analyzing a speckled pattern of near-IR laser light. (Figure 3). This technique of analyzing a known pattern is called structured light [5]. Structured light works by projecting a known pattern onto an object so as to infer depth based on the deformation of that pattern. The Kinect uses a near-IR laser light with a speckled pattern to do this. (Figure 3) The speckled pattern projected by the Kinect is comprised of three different sized dots that are created optimally for three different depth ranges (Appendix, Figure 2). The Kinect can sense pattern deformation as well as changes in the size and shape of the individual dots [6]. The Kinect's depth and height measurements are extremely accurate: within one centimeter and three millimeters, respectively [7]. There are, however, some downsides to this system: the Kinect will only work inside because sunlight would interfere with the near-infrared speckled dot pattern. Also, two Kinects cannot be used in near vicinity to each other, because if the speckled light projections become combined, the Kinects will misinterpret the resulting deformations of the patterns [5].

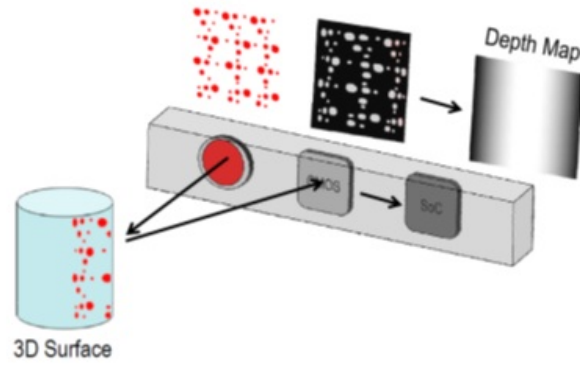

Figure 3: The Kinect uses a system of structured light to infer depth [6].

2.2.2 Inferring Body Position

After the depth map is created, the Kinect performs human subject foreground extraction (i.e. background removal) to extract the human subject [8]. This extracted human subject is matched to the Kinect's database of over 100,000 skeletons of past body positions to estimate the user's current pose. Every time the Kinect detects movement, it uses a system of a "randomized decision forest" to determine how the user's body is oriented [5]. (Appendix, Figure 4) The randomized decision forest is very similar to a highly complex version of 20 questions that the system uses to decide the user's body position. After that, the skeleton joint positions are estimated and refined [8]. Once the Kinect has determined the locations of the user's joints, it can recognize any motions or gestures performed by the user. Finally, the Kinect processes these motions and sends the triggered feedback to its users via the visual screen [8]. These steps take only nanoseconds for the Kinect to carry out.

2.3 Current Kinect Applications for Exercise and Physical Therapy

The Kinect is capable of a myriad of practical uses besides simple gaming. Microsoft Research, along with Seoul National University, is sponsoring Stroke Recovery with Kinect, a project that will provide a low-cost and at-home motor ability rehabilitation solution for victims of both past and recent strokes. Simple games, similar to those created in this study, can be used by patients to rehabilitate motor skills in a fun and encouraging way. Patients will have the advantage of recovering at home in a comfortable setting with their families, instead of in an unfamiliar hospital or rehabilitation center [9]. Researchers have evidence to suggest that the game-like atmosphere that the Kinect program provides is encouraging and inviting for the patients which often leads to faster recoveries [9].

Reflexion Health created a digital medicine software system called Vera that uses Kinect technology to help patients with musculoskeletal rehabilitation. In November 2015, the US FDA approved this system, and the CDC has provided a $1 million grant to use this system to aid in fall prevention for seniors [10]. Patients see an avatar on the screen that coaches them and encourages them to practice the exercises at home. The program will track their progress and report the data back to the user's physical therapist who can monitor their activity in real time [10]. A similar program, created by the startup company Jintronix, has also been approved by the FDA.

2.4 Other Upcoming Kinect Applications

Kinect has many real-world applications beyond simple gaming. Researchers in China have developed a program called Kinect Sign Language Translator, which translates the user's sign language into spoken and written language in real time. This Kinect program can translate all 300 different versions of sign language practiced around the world, creating an extremely useful bridge between those who use sign language and those who do not [9].

Motion Control technology might also have extremely beneficial medical applications in the future. Today, when surgeons need to find information about their patient or for the medical procedures in the middle of a surgery, today they often must interact with non-sterile surfaces, which can be detrimental to the surgery [1]. GestSure is a program created with the intent of allowing surgeons to find critical information simply by making certain gestures in front of a Kinect Sensor [9]. Simple skeletal recognition, such as that demonstrated in this project, will be implemented in this program.

Finally, NASA has paired the Kinect with an Oculus Rift virtual reality headset to control a robotic arm [9]. In this situation, the robot would use the Kinect's skeletal tracking technology to copy the motions that a human user is making. If further testing proves successful, NASA hopes to install this system on the International Space System. With this technology, robots could act with human versatility in places too dangerous or costly for humans to go.

3. Experimental Approach

3.1 Setup

The games and interface were designed on Windows computers and can be run on computers that have access to Python, Pygame, PyKinect, and the Kinect SDK. The games were designed for and tested using Kinect for Windows. During testing, the Kinect was mounted on a table several meters away from the player.

To develop the games, Python and a few game modules that interact with the Kinect were used. The games were created using Python 2.7, Pygame, PyKinect, Kinect for Windows SDK, and the Kinect Developer Toolkit.

Python is a relatively new programming language, meaning that it is able to take full advantage of recent computing advances [11]. Python was developed to be an interpreted, object-oriented programming language that relies on indentation for syntax. The language includes features such as modules, exceptions, dynamic typing, and classes which makes it a very high-level language. In addition, Python is portable across all major hardware and software platforms, making it very versatile [12].

Pygame is a module that allows developers to create graphics when writing software in the Python programming language. Specifically, it is designed for game creation and therefore includes ways to render motion of graphics around a screen. Pygame is built on multiple back ends, which allow it to render on computers that have different graphics systems. The core functions are written in C and Assembly, which allows it to run 20 to 100 times faster than strictly Python, which allows these games to run faster with this library than they would be able to with other libraries. Despite being extremely powerful, Pygame has a simple syntax that allows complex games to be created with ease [13].

Although Pygame allows users to create graphics in their applications, it does not track Kinect events on its own. In order to allow developers to track Kinect within their games, Microsoft has developed a library called PyKinect. This library allows users to track all events of the Kinect, including both visual and audio stimulation. Within the visual tracking sub-module, the package provides tools to track skeletons and depth maps and allows users to easily display these camera feeds. The PyKinect package is specifically designed to work well with Pygame so that users are able to integrate the Kinect into their own custom games [14].

3.2 Game Design for Physical Rehabilitation

The three games were designed to complement each other in a way that gives the player the ability to work out any and all muscle groups. Red Light, Green Light allows the user to get a cardio workout as well as decide whether he would like to exercise his arms, his legs, or his full body. Push-up Pong complements the intense cardio of Red Light, Green Light with intense core and upper body workouts, as well as an option to rest the upper body in the sit-up variant of the game. Breakout also provides a cardiovascular workout by having the user run back and forth.

3.2.1 Designing Red Light, Green Light

Red Light, Green Light follows similar rules to those followed by kindergarteners during recess. The classical rules of the game are relatively simple; during a "green light," players run towards another player who is the "traffic light." During this time, that player's back is turned. When the "traffic light" turns around, the "light" turns red, and anyone caught moving is sent back to the start. A player wins when he or she reaches the "traffic light." In this version, however, players are encouraged to move as much and as quickly as possible while the score box is green, and to halt all movement when it turns red. Additionally, the score is time and score oriented, as opposed to goal oriented.

Score is calculated by taking the distance between body part locations at 10-millisecond intervals, effectively measuring movement speed. Body part position was taken using JointID. Many body parts have a specific location tracked by the Kinect on a scale from (0, 0) to (1, 1). These values were taken every 10 milliseconds and compared to the positions values from the previous loop. When the light is green players gain points for how fast they move their limbs and when it is red they have points deducted. The color of the light is a global variable that is used by many methods including the score keeping methods, the music playing methods, and of course, the rendering methods. The value of the color variable was used to determine whether score should be increased or decreased. At the end of the game, the user is shown not only the game score, but also the net score (the sum of points added and points deducted). This feature informs players of how much they exercised during the course of the game.

Initially, Red Light, Green Light used the only the left and right hands as reference points for tracking the user's motions. Later, the left and right feet were added as well as the head. In order to keep Red Light, Green Light versatile and player-friendly, major design changes were put in place. In the current version of the game, players get to choose whether they would like to exercise their upper, lower, or full body. This change allows players fatigued from other exercises to still enjoy the game. This flexible gameplay along with the lack of a specific challenge to conquer, leads to a strong gameplay contrast when compared to the other games designed over the course of the project. Red Light, Green Light was designed to be casual, and as such, the users define many of their challenges. Users get to choose which muscles they prefer to have tracked and which time limit they prefer to implement. Additionally, the lack of specific goals or targets leads to a greater availability of the game to users. Users do not have to be physically fit enough to reach certain goals or beat a computer AI. Instead, one's only competition is oneself, and to beat oneself, one must become more physically fit.

In addition to offering single-player gameplay, Red Light, Green Light offers a cooperative mode. In this mode, score is calculated using the same method as is used in single-player mode. The players not only have double the opportunity to earn points, but also have double the opportunity to lose them. The idea for this game mode actually originated from a bug that caused all movement to be tracked as zero. The bug was caused by faulty looping in the score keeping function. The loop calculated the change in position of each skeleton and then returned the score change. The loop, however, returned the value after tracking only the first skeleton, so half of the time a blank skeleton was tracked instead of the user. In the process of fixing this bug, it was realized that with a few modifications, the method could track the score change for all skeletons in the frame. These modifications were implemented in what is now the multiplayer game mode.

One of the earliest issues with the game was its length. In older versions of the game, the game was terminated when the music ended. While the light was green, Scott Joplin's "Maple Leaf Rag" would play, and during red lights a ticking noise would play. "Maple Leaf Rag" is already a three-minute piece, leading to six minute games. To reduce this speed and maintain the versatile gameplay, a timer option was created. Users can now select how long the game will last. Another benefit of this feature is that the ability to restart the piece each time the light turned green became feasible. Now, players have a distinctive audio cue at which point they resume moving. During play testing this was found to be helpful and more enjoyable than resuming the piece at the point at which it was last paused.

In early stages of game development, a skeleton display was added for debugging purposes (Figure 5). The skeleton display is an outline of the user's body. Despite being added for the purpose of debugging, many of the test group enjoyed the presence of their skeleton. Based on this feedback, the skeleton was incorporated into the final design. At the beginning of the game, in addition to choosing which body parts will be exercised and for how long the game will be played, the user is also prompted to choose whether or not a skeleton will be drawn. The skeleton is drawn in the same color as the light to aid in the user's ability to react quickly.

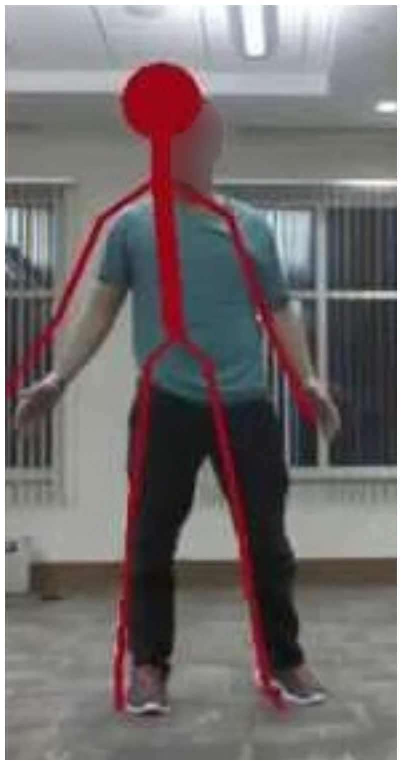
Figure 5: A skeleton is displayed during a red light.

3.2.2 Designing Push-up Pong

Push-up Pong is designed after one of the first computer games ever developed: Pong. The original Pong was designed as a game in which two paddles hit a ball back and forth. When one player fails to return the ball, the other player gains a point [15]. Push-up Pong was designed to be a modern version of the age-old classic.

The first development stage of Push-up Pong was creating a game of normal pong that implements Python. Designing the game of Pong without involving the Kinect allowed for easy testing and graphics tuning. This stage consisted of creating a core library to handle graphics generation and rendering, which was extended to allow creation of each type of object in the game. Using this library allows the program to draw all parts of the game and easily move them around the screen. The controller of the game creates the objects, calculates movement (i.e. bouncing) and renders those movements to the screen using the core library. Once all movement was calculated and the graphics were finalized, the stage was set to incorporate the Kinect as the controller.

Many methods had to be created to handle all of the input that comes from the Kinect before the game could be controlled with it. By adding Kinect events to Pygame, the program was able to detect when the Kinect had detected user movement and could react accordingly. The most essential part of the data provided for this game was the coordinates of body joints, specifically the hand and head. At first, the game was created to move to the position of the right hand, and the opposing player was controlled by the left hand, as see in Figure 6. After this stage, the left hand was replaced by an Artificial Intelligence (AI) which allowed the user to play alone against an opponent. The AI was specifically designed to either hit or miss the ball at certain positions in order to mimic the actions of a real player (Figure 7). Originally, the user's paddle jumped abruptly to the *y* coordinate of his or her hand as it moved up and down, which allowed for an effective but laggy experience. Therefore, a system was created to move the paddle from its old position to its new position over five frames (about as long as the Kinect takes to update), which allowed for a much smoother transition.

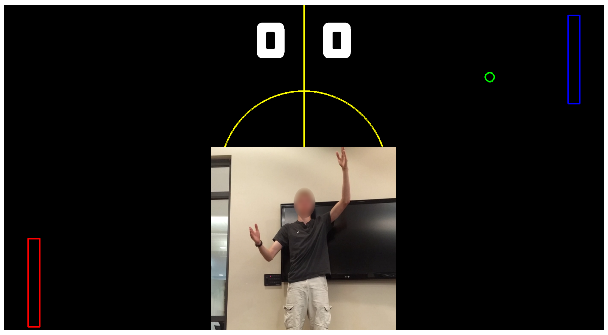

Figure 6: The user controls both paddles with two hands.

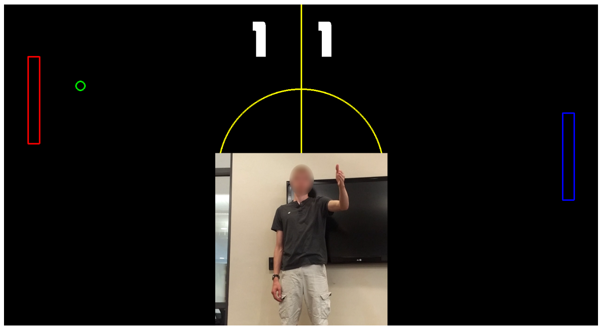

Figure 7: The AI (left) returns the ball to the user.

After the initial implementation of hand tracking, the game was changed to track the user's head in order to induce more physical activity. This change was designed to track physical movement in a vertical direction in order to control the paddle. Push-ups are a great physical exercise for working various muscles in the upper body and therefore were chosen to be the main controlling action for the game (Figure 7). The exercise works well for most users because there are different types of push-ups that fit most body sizes and strengths. The action of a push-up involves the user moving from an elevated position to a lower position just above the ground, thus creating the vertical movement for the Kinect to track [16]. Originally, the game tracked the user's movement from top to bottom of the Kinect viewport. However, this would not be effective for push-ups because users would not be able to reach the top of the screen with their paddle. Therefore, the game scales movement from the bottom of the Kinect's view to match the size of the display screen, effectively allowing the user to reach the top of the screen while still remaining in push-up position. The game is able to effectively track a user's progress in a push-up and update the screen accordingly, providing an incentive for users to increase personal fitness by performing push-ups to control the game.

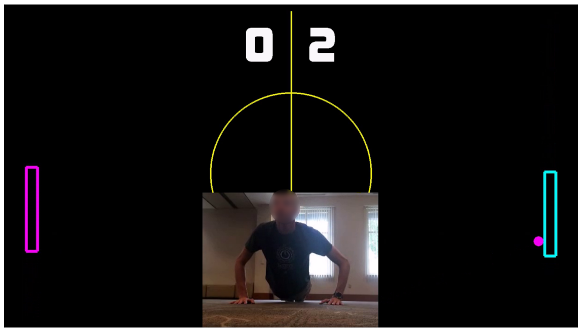

Figure 8: The user controls the right paddle by performing a push-up.

3.2.3 Designing Breakout

Breakout is similar to Push-up Pong in that it is based on a classic video game: the original arcade game of the same name. Also known as Brick Breaker, Breakout was originally a clone of Pong designed for a single player. Breakout's gameplay involves players controlling a single paddle at the bottom of the screen, which is used bounce a ball around the level and into blocks arranged at the middle and top of the screen. When the ball collides with a block, the block disappears and the ball bounces off in another direction. The player's goal is to make all of the blocks disappear [17]. This Breakout served as an inspiration for the Kinect version.

The first step in developing the new Breakout was to create a working version of the original Breakout using Python. Without the complications of the Kinect and PyKinect, it was much easier to develop a game with working graphics and a simple physics engine (Figure 9). The graphics and engine allow objects to be easily drawn on a graphics window and make moving the ball and paddle easy. The engine controlled the motion of the ball and calculated the angles and vectors when it bounced off a wall, paddle, or block, while the graphics window drew the ball onto the screen and controlled the game's rendering speed.

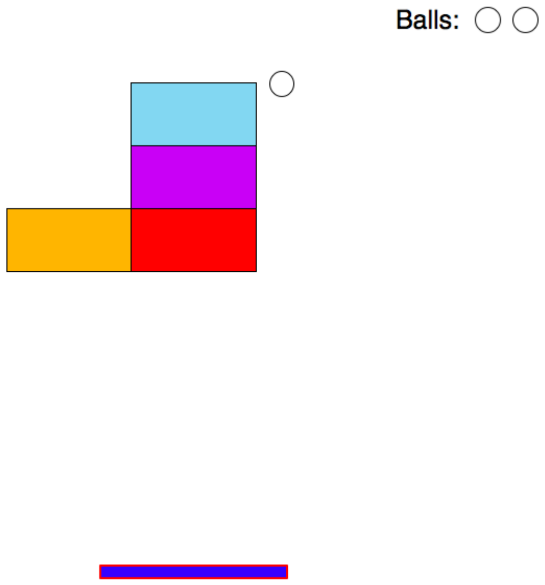

Figure 9: Early versions of Breakout used simple graphics.

In order to transfer the working game over to the Kinect, it was necessary to change the graphics of the game from the originally-used library to Pygame. Pygame was much more suited to the task than the original library because it allowed faster and more accurate motion tracking of the ball and paddle. This change not only increased the rendering speed, but also decreased the amount of lag (Appendix Figure 10). The transition from the original graphics to Pygame involved the creation of objects to represent the ball, paddle, and blocks. These objects contained variables necessary for the creation of each image, such as size, position, and speed. Furthermore, each object was given methods to facilitate the drawing of its graphics; each object, for example, received a few lines of code to render itself into a new position every frame the game was active. Many commands contained by the original engine needed to be changed to commands supported by Pygame, and the entire interface needed to be simplified and object-oriented for ease of editing.

The Kinect tracks the user's head to provide physical activity. The game was designed to move the paddle left or right when the user runs left or right. While the game is running, the Kinect detects the changes in the user's head position and reacts accordingly. Running was used as the game's physical activity because running provides an effective indoor aerobic exercise. The program moves the paddle left and right to match the user's head position and interact with the ball.

Although the original games are similar, the health benefits of the new Breakout are very different from those of Push-up Pong. Push-up Pong develops players' arm muscles and core strength; Breakout gives players a cardiovascular workout as they run back and forth between the two ends of the screen. This type of aerobic workout helps to increase stamina and lower blood pressure and is necessary for a healthy lifestyle [18].

4. Results and Discussion

4.1 Red Light, Green Light

Red Light, Green Light supports both single-player and multiplayer gameplay. Additionally, it provides the user with a choice of game length, type of workout, and skeleton display. The game is an excellent and fun physical challenge because users choose their own difficulty level when they select game duration and type of workout. Red Light, Green Light provides an intense cardio workout with the only limit being the user's willingness to push the limits of his or her body and mind.

Score is calculated by summing the scaled total distance travelled by the tracked joints every 10 milliseconds, effectively measuring the speed at which the player's limbs are moving. If the light is currently red, this distance is subtracted from the player's score; otherwise, it is added. A scale of 100 was used to accommodate for the size of the (x, y) coordinate values output by the Kinect. The x and y values are by default scaled from 0 to 1 leading to extremely small values of distance each time the score was measured. In order to create a scoring system that is both fair and encouraging all scores were multiplied by 100 leading to scores ranging from 50 to 1000 instead of 0 to 10. The large range of 50 to 1000 is a result of differing player skill levels, not inconsistent scoring. By increasing movement speed, players can beat their previous high-scores. Additionally, by increasing reaction times, players can improve their scores.

4.2 Push-up Pong

Push-up Pong offers physically active single-player versus AI gameplay. The game starts by giving players some time to orient themselves and prepare to play the game. Once the game begins, the user is able to control the paddle by performing a push-up. Push-ups performed by the player target the abdominal muscles, coracobrachialis, deltoids, pectoralis major, serratus anterior, and triceps brachii, making it a good exercise to strengthen the entire upper body [16]. The game continues until either the player or the AI scores five points. The player that achieves this first is the winner. The game challenges the user to continue performing push-ups despite becoming tired in order to build muscle and endurance.

The user's paddle position is calculated by taking the user's y coordinate for the head from the Kinect. The value from the Kinect comes as a percentage of the Kinect's viewpoint, so this value is scaled to become a percentage of the bottom third of the viewport. This allows users to control their paddle over the entire screen without having to move out of push-up position. If the ball crosses out of the screen on the left or right, a point is awarded to the respective player controlling the paddle on the other side of the screen. When the ball bounces off of the top or bottom of the screen, its x vector is preserved while the y vector is reversed, causing the ball to bounce off at the same angle that it hit the wall at. Similarly, the x vector is reversed when hitting a paddle in order to create the bouncing effect. Each paddle is divided into five sections that, when hit, change the ball's x and y vectors in a way that correlates to their part of the paddle. The outer parts of the paddle increase the y vector's magnitude more than the x vector's magnitude, and vice versa for the inner parts of the paddle. This creates dynamic gameplay where the ball bounces off at different angles on every hit. In addition, the AI misses the ball more often as the game progresses, which simulates user fatigue and makes it easier for the user to stay competitive as they tire. Through this system, users are able to play a challenging and dynamic game and, at the same time, increase their personal fitness.

4.3 Breakout

Breakout offers a stimulating single-player game using PyKinect. Players are given sufficient time to position themselves and wait for the Kinect to detect them. The players then run left and right to move a paddle laterally across the bottom of the screen. Three seconds after the player is in frame, the ball is released from the center of the paddle. The game continues until the user either successfully breaks all of the blocks on the screen or until three balls fall past the paddle.

The ball's speed is calculated by storing x and y vectors, which are initialized at the beginning of the game. If the ball hits the side walls, the x vector is reversed; the y vector is reversed if the ball collides with the ceiling. If the ball hits the paddle, the angle at which the ball bounces off is calculated by using trigonometric functions to set the x and y vectors. Whether the ball hits the side walls, ceiling, or paddle, its speed remains the same to provide a realistic bounce.

4.4 Future Plans

If the project were to continue, the next step would be to develop an interface to unify the games. Although all three games are fully functional, the project is still continually changing. In addition to the three aforementioned games, an interface would be created to link the games together. Part of the experience of using Kinect is absolute motion control. As such, an interface that allows players to choose what game they would like to play and which settings they would like to play with would be the next logical step. In Red Light, Green Light, for instance, players currently have to type how long they would like to play, what part of the body they would like to work out, and whether they would like their skeleton displayed into the command-line. A functional interface would keep the atmosphere of a motion-controlled game unbroken while giving the players the option to choose how they would like to proceed with their gameplay session. Designing the interface itself would not be the only challenge, as all three games that constitute the final project would also need to be edited. All three games need to be modified in such a way that they could be opened by the interface. Additionally, the interface would need to pass parameters to the games such as the aforementioned settings in Red Light, Green Light.

5. Conclusion

Over the past decade the distance between users and their computers has dramatically decreased. Users no longer need a mouse to control their computers; they can touch the screen itself, or, in the case of this project, take a step back from the screen. The authors of this paper were tasked with exploring the implications of motion control gaming using the Microsoft Kinect along with several software packages associated with it. The team researched the Kinect's potential in various fields including physical therapy, surgery, and sign language translation in order to better understand different directions in which the project could be taken.

The games designed were directed toward one of the practical applications discovered in the team's research. The games are intended to increase players' physical activity as well as their general physical fitness. Tying their work to physical therapy and rehabilitation helped the group to design coherent games that complement each other. Based on the initial research about the future of physical rehabilitation and exercise with Kinect, the team was able to design games that implemented some of the ideas that these future projects would apply, such as skeletal tracking and motion recognition. In that respect, the team fulfilled one of its principal goals. Red Light, Green Light and Push-up Pong each provide a unique workout that could have real life applications in both physical therapy and physical conditioning. The team hopes to continue its work with a completely motion controlled interface to link its games together. The completion of this goal would mark the actualization of the team's goals as the final product would be completely motion controlled and feature a versatile array of games which complement each other both in terms of style and exercise.

6. Acknowledgements

The authors of this paper would like to thank Lawrence Maceren for the guidance he provided throughout the process of designing and testing the games. The authors would also like to extend their thanks to Program Director Jean Patrick Antoine, the NJ Governor's School Board of Overseers, Dean Ilene Rose, and all of the other dedicated Residential Teaching Assistants for providing them with this incredible opportunity. Finally, without the generous contribution of the NJGSET sponsors, Rutgers University, Lockheed Martin, South Jersey Industries, and Printrbot, this program and project would not have been possible.

7. References

[1] M. Rouse. (2011). *Kinect* [Online]. Available: http://searchhealthit.techtarget.com/definition/Kinect

[2] L. Yao, H. Xu, and A. Li, "Kinect-based Rehabilitation Exercises System: Therapist Involved Approach," CSE Department, NYU, NY, Rep. 24, 2611, 2014.

[3] Microsoft. (2015). *Kinect for Windows Sensor Components and Specifications* [Online]. Available: https://msdn.microsoft.com/en-us/library/jj131033.aspx

[4] J. Tanz. (2011). *Kinect Hackers are Changing the Future of Robotics* [Online]. http://www.wired.com/2011/06/mf_kinect/all/1

[5] J. MacCormick. (2011). *How Does the Kinect Work?* [Online]. Available: http://users.dickinson.edu/~jmac/selected-talks/kinect.pdf

[6] (2011). *How the Kinect Works* [Online]. Available: http://www.depthbiomechanics.co.uk/?p=100

[7] T. Carmody. (2010). *How Motion Detection Works in Xbox Kinect* [Online]. Available: http://www.wired.com/2010/11/tonights-release-xbox-kinect-how-does-it-work/

[8] R. Lun and W. Zhao, "A Survey of Applications and Human Motion Recognition with Microsoft Kinect," Dept. of ECE CSUOhio, Cleveland, OH, Rep. 29, 1555008-1, May 2015.

[9] A. Jamaluddin. *10 Creative and Innovative Uses of Microsoft Kinect* [Online]. Available: http://www.hongkiat.com/blog/innovative-uses-kinect/

[10] K. Yeung. (2015, Nov. 18) *This Microsoft Kinect-Powered Physical Therapy App Now has the FDA's Approval* [Online]. Available: http://venturebeat.com/2015/11/18/this-microsoft-kinect-powered-physical-therapy-app-is-fda-approved/

[11] B. Klein. (2016). *History of Python* [Online]. Available: http://www.python-course.eu/python3_history_and_philosophy.php

[12] M. Lutz. (2011, Dec. 9). *The Python Programming Language* [Online]. Available: http://groups.engin.umd.umich.edu/CIS/course.des/cis400/python/python.html

[13] *About Pygame* [Online]. Available: http://www.pygame.org/wiki/about

[14] *PyKinect - Write Games Using Python* [Online]. Available: https://github.com/Microsoft/PTVS/wiki/PyKinect

[15] *About Pong Game* [Online]. Available: http://www.ponggame.org/

[16] (2016). *What Muscles do Push-Ups Work?* [Online]. Available: http://www.md-health.com/What-Muscles-Do-Push-Ups-Work.html

[17] Procyon. (2008). *Breakout* [Online]. Available: http://classicgaming.gamespy.com/View.php?view=Articles.Detail&id=395

[18] C. Iliades, M.D. (2009, Dec. 9) *Why You Need Aerobic Exercise* [Online]. Available: http://www.everydayhealth .com/fitness/workouts/why-you-need-aerobic-exercise.aspx

[19] N.J. Seo, J.A. Kumar, P. Hur, V. Crocher, B. Motawar, and K. Lakshminarayanan, "Usability Evaluation of Low-Cost Virtual Reality Hand and Arm Rehabilitation Games," Div of Occupational Therapy, Dept. of Health Professions, Charleston, SC, Rep. 53, 321, 2016.

[20] M.J.D. Taylor, D. McCormick, T. Shawis, R. Impson, and M. Griffin, "Activity-Promoting Gaming Systems in Exercise and Rehabilitation," Centre for Sports and Exercise Science, University of Essex, Colchester, Essex, UK, Rep. 48, 1171-1186, 2011.

[21] H. Benko, A.D. Wilson. "Combining Multiple Depth Cameras and Projectors for Interactions On, Above, and Between Surfaces," Microsoft Research, Redmond, WA, 2010.

Appendix

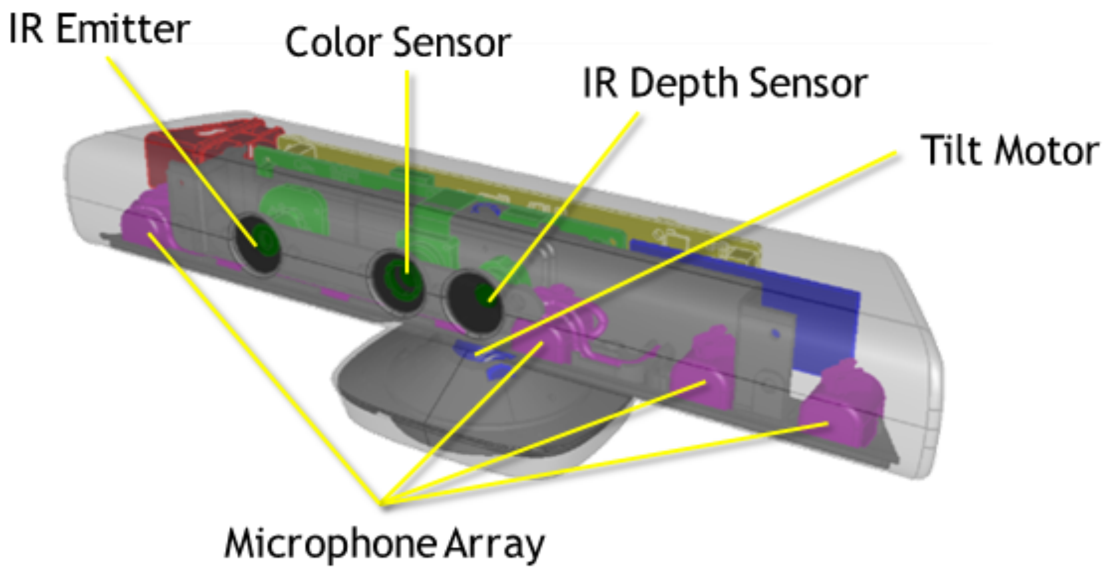

Figure 1: The Kinect system consists of sensors, microphones, and a motor [3].

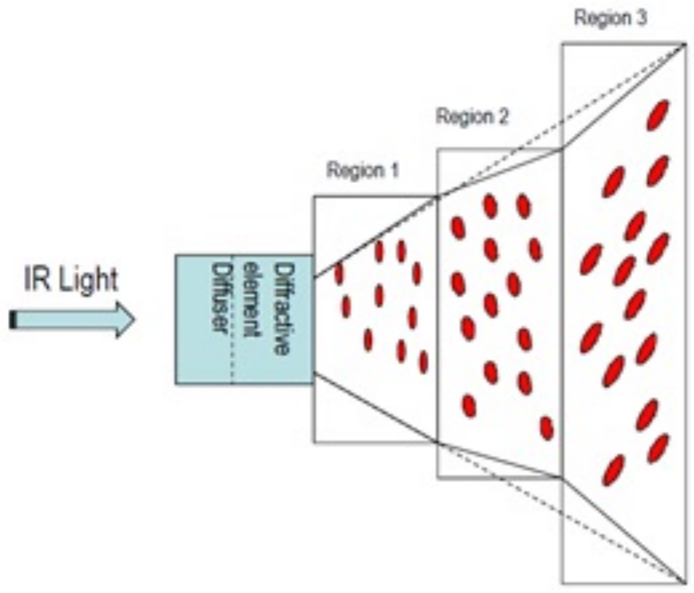

Figure 2: The Kinect uses a system of associated speckle output to infer depth [6].

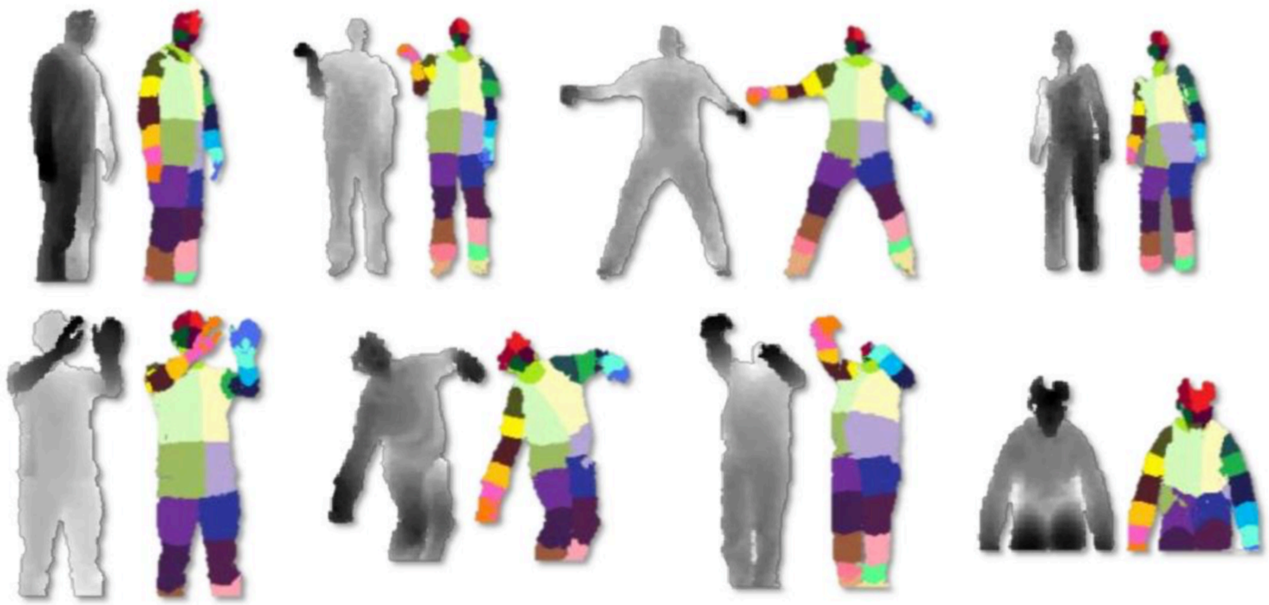

Figure 4: The Kinect compares the created depth map to its database of skeletons [5].

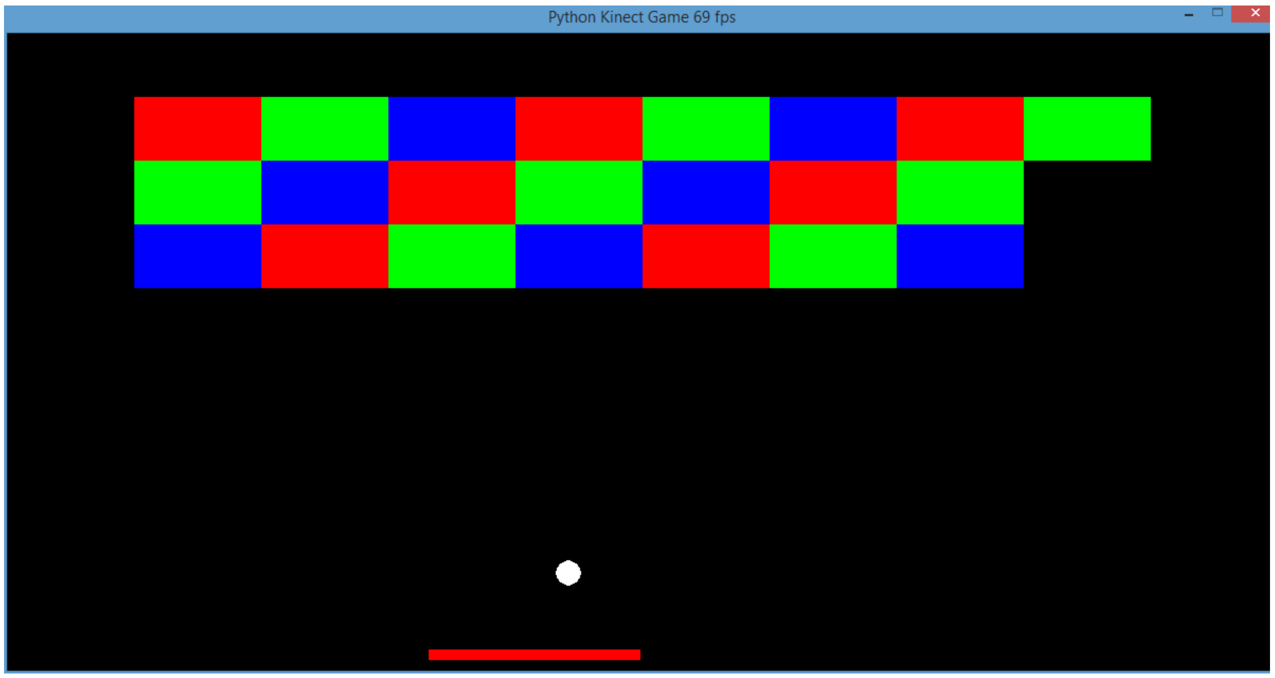

Figure 10: The current version of Breakout uses the Pygame Interface.