A Step Forward in Virtual Reality
Team Step

Ryan Daly
Electrical Engineer

Joseph Roberts
Electrical Engineer

Jared Ricci
Electrical Engineer

Steven So
Electrical Engineer
Motivation

▪ Current Virtual Reality has lacked the REALITY aspect

▪ The market is pushing smartphone VR; peripherals can make the experience better

▪ Users do not feel immersed in the environment

▪ VR hasn’t reached its full potential
Introduction

- Step is a new virtual reality environment that will immerse the user with no added hardware

- The user will be able to interact, move, and feel the environment
DEMO
MDR Deliverables Met?

- Precise speed control with elliptical
  - Adequate Kinect motion sensing
  - Data processing and transmitting to android
  - Programmed VR environment using inputs
MDR Deliverables Met?

- Precise speed control with elliptical
- Adequate Kinect motion sensing
  - Data processing and transmitting to android
  - Programmed VR environment using inputs
MDR Deliverables Met?

- Precise speed control with elliptical
- Adequate Kinect motion sensing
- Data processing and transmitting to android
  - Programmed VR environment using inputs
MDR Deliverables Met?

- Precise speed control with elliptical
- Adequate Kinect motion sensing
- Data processing and transmitting to android
- Programmed VR environment using inputs
Our Solution

▪ Create a 360 degree, 3 dimensional virtual environment on the android

▪ Create a system that will detect the user’s movement like walking, running, and arm motions

▪ Create a structure in which the virtual world can be mimicked
Overall Requirements

- User is able to freely move in virtual environment and control movement speed
  - Speed Accuracy within .5 MPH

- Hand and arm motion is translated to in-game action
  - Depth Accuracy, standard deviation within 1 inch

- Control latency less than 200 ms

- User does not have to wear any equipment beyond VR headset

- Maintain framerate at 60 FPS

Overall Requirements

- User is able to freely move in virtual environment and control movement speed
  - Speed Accuracy within .5 MPH

- Hand and arm motion is translated to in-game action
  - Depth Accuracy, standard deviation within 1”

- Control latency less than 200 ms\(^1\)

- User does not have to wear any equipment beyond VR headset

- Maintain framerate at 60 FPS

Design

Elliptical

Raspberry Pi

Router

Android Phone

Kinect

PC
PDR Block Diagram

User Motion
- Elliptical
- Sensing Board
- Raspberry Pi

Hand and Arm Movement
- Kinect Center Sensor
- Kinect Left Sensor
- Kinect Right Sensor

Wireless Network
- PC processor and data transmitter
  - Raspberry Pi Server
    - Data Compiler and interfacing
  - Data Transmitter via Wifi
  - Wifi: 35 ms

PDR Feedback: Interfaces were not well defined

Data Client / VR Game
- Smartphone and VR Headset
- Wifi: 35 ms
- Router
MDR Block Diagram

User Motion
- Elliptical
- Sensing Board
- Raspberry Pi

Hand and Arm Movement
- Kinect Center Sensor

Wireless Network
- Raspberry Pi Server
- Router/Wifi

Speed Data
- 35ms

PC server

Kinect Data
- 35ms

Smartphone/VR Headset
- Data Client
- Parse
- 3D World

35ms
User Motion

Implementation
▪ Raspberry Pi with attached sensor board
▪ Rotational speed measured using magnetic sensor
▪ Data transmitted wirelessly to smartphone

Changes from Initial Design
▪ Multi-magnet sensing
▪ Data sent directly to smartphone, not to PC
User Motion Functionality
User Motion Functionality

- Calculated speed is accurate ±.5 MPH

- Standard Deviation = .152 MPH (n=40)

- Latency is under 200 ms requirement

- Average latency = 70.08 ms (n=100)
Hand tracking - Kinect

**Requirements**
- Depth within 1” standard deviation for arm movement
- Real-time processing and transmitting

**Steps**
- Create server
- Extract Position
- Wait for request
- Write position as string to server
Kinect Joint tracking

- **Skeleton**
  - Each node is a “Joint type” object
  - Using left and right hands
    - Shown as balls in balloon popper

- **Position**
  - \((x, y, z)\) coordinates
  - Unit: meters
Kinect Depth Results

- Depth was thought to be an issue
  - Collected data and measured depth performance

<table>
<thead>
<tr>
<th>Tape Measured</th>
<th>Calculated Kinect mean (n=100)</th>
<th>Standard Deviation (m)</th>
<th>Standard Deviation (in)</th>
<th>Performance &lt;inch (=0.0254 m)</th>
<th>Performance &lt;4cm</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 m</td>
<td>1.0021 m</td>
<td>0.0214</td>
<td>0.844</td>
<td>80%</td>
<td>93%</td>
</tr>
<tr>
<td>2.25 m</td>
<td>2.2452 m</td>
<td>0.0218</td>
<td>0.859</td>
<td>77%</td>
<td>96%</td>
</tr>
</tbody>
</table>

- Experiment setup
  - Measured points from Kinect with tape measure
  - Compare with what Kinect returned
Kinect Processing and Data Transfer

- Internal frame rate [2]
  - 30 fps

- \((x,y,z)\) coordinates extracted
  - Coordinates converted to string
  - String written to server stream when requested

Kinect Used in the Game
Wireless Network

What it needs?

▪ Should allow the data from inputs to communicate with android
▪ Minimal latency in order to have accurate movements in the game
Wireless Network Blocks

Servers
- The raspberry pi
- PC reading Kinect data

Client
- Android

Router
- Wirelessly connects the Servers to the Client
Does this part work?

Very clearly the Kinect and the Elliptical can communicate with the android phone through the router since the phone can see

The latency

Used slow motion camera and matlab to calculate

End-to-End Latency: ~163ms
Latency: end-to-end
Smartphone Application

Requirements

- Render a 3D virtual world
- Receive and translate data sent through network
- Framerate = 60 FPS (Limited by VSYNC)

Implementation

- Virtual 3D environment developed with Unity
  - C# scripting
- TCP client requesting data through router
  - Data parsed to be usable
- Ensure TCP servers are not capping the framerate through slow data availability
TCP Server → TCP Client → Virtual World

- Uses Microsoft’s .NET Framework

- Steps:
  - Establish client, open stream
  - On every frame render, request data
  - Parse byte data into variables
  - Use variable coordinates and variable speed
MDR Deliverables

Interactive VR game in which the user can walk along a straight path whilst controlling their speed of travel, and using their arms to pop bubbles.

- Demonstrates:
  - a. precise speed control with elliptical
  - b. adequate kinect motion sensing
  - c. data processing and transmitting to android
  - d. a programmed VR environment using inputs

PDR Feedback:
User has no incentive → Game score ✓
Individual Parts

Jared Ricci: Forward Motion → PCB Design
                Fan Controller

Joseph Roberts: Kinect → Structure
                     Safety Harness

Steven So: Wireless Network → Turn sensing

Ryan Daly: VR Application → Refined Game
Gantt Chart

Second Semester Gantt Chart
Team 16: Ryan Daly, Jared Ricci, Joseph Roberts, Steven So

<table>
<thead>
<tr>
<th>Dates</th>
<th>23-Jan</th>
<th>30-Jan</th>
<th>6-Feb</th>
<th>13-Feb</th>
<th>20-Feb</th>
<th>27-Feb</th>
<th>6-Mar</th>
<th>13-Mar</th>
<th>20-Mar</th>
<th>27-Mar</th>
<th>3-Apr</th>
<th>10-Apr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Game Menu Screen (Ryan)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Game Reset (Ryan)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Turning Motion in Game (Ryan)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sound Enhancement (Ryan)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Different Environments (Ryan)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PWM Circuit for Fan Controller (Jared)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Controlled Fan (Jared)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sense backwards Motion (Jared)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Structure to hold kinect, fan, etc (Joe)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Safety Structure and Harness (Joe)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Turning Sensors (Steven)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCB for speed detection and fan (Steven)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CDR Preparation (ALL)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CDR Presentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Refinement of Project (ALL)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CDR Deliverables

1. PCB design / Fan control
2. Refined Game (Menu/Reset)
3. Structure / Safety Harness
4. Turn sensing
Questions?