Real world physics VR experience

Gil Beresi and Gil Blinstein

Supervisors: Boaz Sternfeld and Yaron Honen

Submission Date: 01/11/2019

Table of contents:

Introduction

We are undergraduate students, in the computer science faculty in the Technion – Israel's Institute of Technology.

We wanted to explore new environment and VR Technology we were interested on how actions can be done while doing only hand movements.

Our main goal in this project was to simulate real life physics experience.

We developed the project using Unity 2018.4.5f1 environment, scripted with C# in Visual Studio 2019 and used HTC VIVE headset and trackers.

In order to do so, we created a game which contain all the physics elements we wanted to imitate into the virtual world.

The game contains a throwable ball and objects which are moving towards the player, while physics elements are being changed randomly during the game.

Development Environment:

HTC Vive: The HTC Vive is a virtual reality headset developed by HTC and Valve Corporation. The headset uses "room scale" tracking technology, allowing the user to move in 3D space and use motion-tracked handheld controllers to interact with the environment.

VIVE Tracker: The VIVE Tracker creates a wireless and seamless connection between your attached tools and the VIVE system. The VIVE Tracker produces data regarding its movement in real time and use the captured data to faithfully reproduce the movement in virtual reality.

Unity: A cross-platform game engine that can be used to create both three-dimensional and twodimensional games, as well as simulations for desktops and laptops, home consoles, smart televisions, and mobile devices. Unity is scripted with C# in Visual Studio.

Visual Studio: Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Git: Git is a distributed version control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows.

GitHub: GitHub provides hosting for software development version control using Git.

Application Overview

The main menu scene allows the player to choose between two possible options, one of them is the 'Tutorial' and the other is 'Start'.

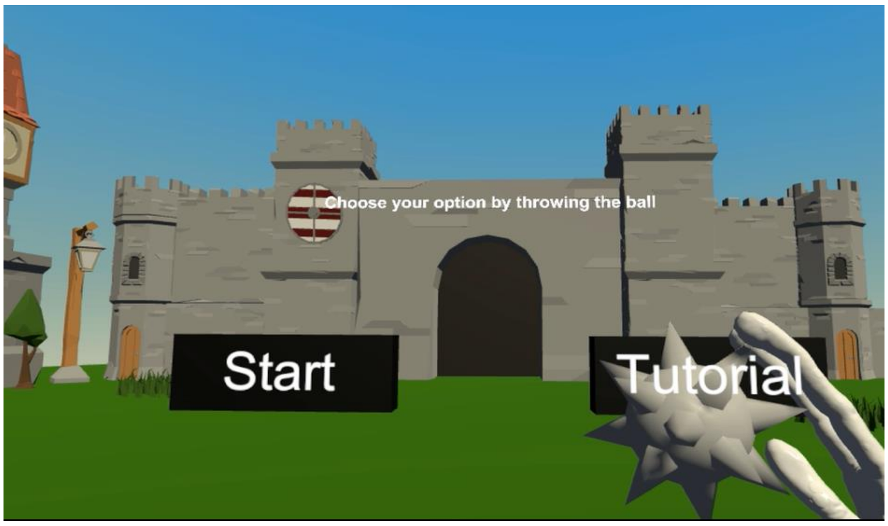
The player can choose the desired option by throwing the ball at the target.

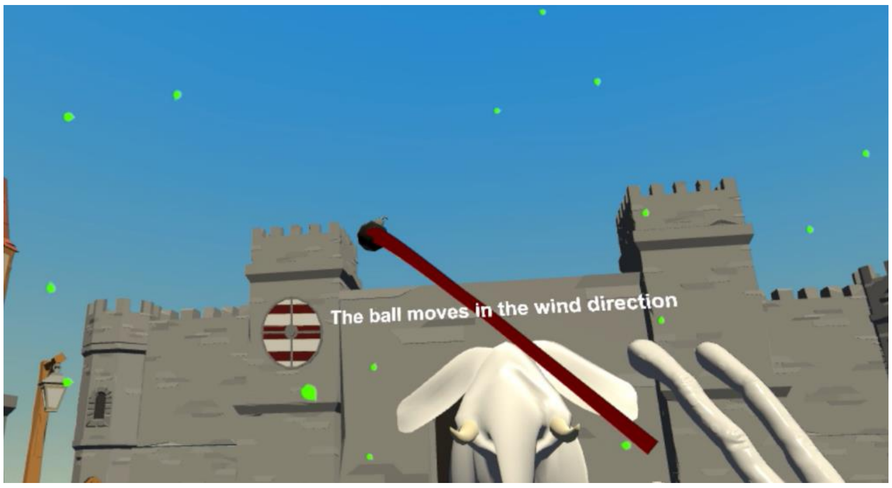
The tutorial consisted of three different levels making the game more interactive:

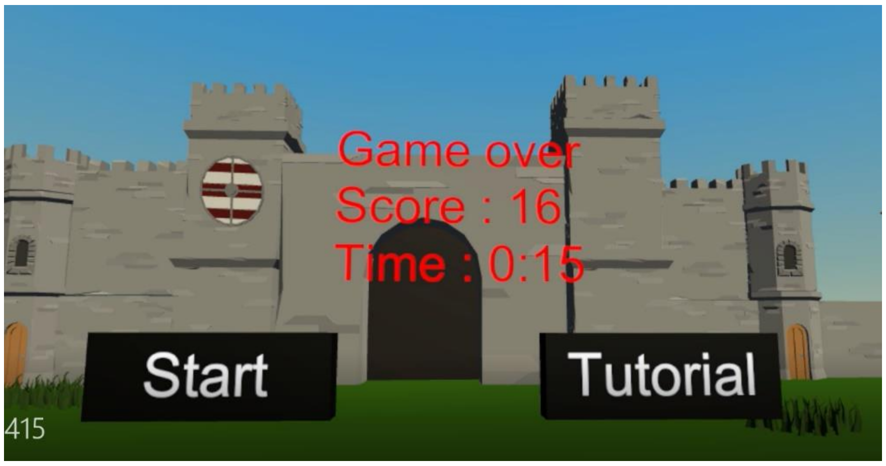
1. The first part of the tutorial helps the player to grasp the basics of how to throw a ball. The tip that is shown for this part is that if you make a faster movement with your hand while throwing a ball you will get a higher velocity.
2. This part helps the player to know that there are different types of balls and objects. Each ball has its own mass, damage power and effect as well as the object has their own different movement speed and health power. The player is given the wooden ball at first, with the lowest power, and he needs to hit the highest health power object the wall. The wooden ball won't destroy the wall, and then the player will get the bomb ball, which destroys every object with one hit, and has an explosion effect on collision with the object.
3. The last part of the tutorial has different physical realism, which is expressed by wind. The wind is shown as leaves flying with the wind direction. Hence the ball direction will be changed as well according to the wind.

If the 'Start' button is chosen, the game will start initializing the game timer and score, it has all the physical elements which are shown in the tutorial, and it changes randomly during the game session.

The game ends when an object hits the player without being destroyed, and then returns to the menu with 'Game Over' announcement and the time and score the player reached.


Choose your option by throwing the ball
Start
Tutorial


The ball moves in the wind direction


Game over
Score : 16
Time : 0:15
Start
Tutorial
415

Development Process

To accomplish our final application, we used Unity environment, which was new to us, therefore we searched for online guides and tutorials, the information was vast online, on the other hand we haven't found sufficient information regarding VIVE Tracker. It took us a while to learn how to use it, and how to integrate it into our project. After getting to know Unity and the VIVE Tracker we started to analyze the "meaning" of a throw. we got to a conclusion that a throw consists of three parts: the starting of a throw, an end of a throw and its direction.

To make it as realistic as possible, we analyzed each part individually:

- The start of a throw: Needs to have an initial velocity in order to divide each movement between a throw and not a throw (not every movement is a throw), Also we took into consideration an initial coordinates number that needs to be taken in order to make an accurate throw.
- The end of a throw: In the real world we move our hand in order to throw a ball and release it at the end, which cause for a decrease in the hand movement velocity. When it happens the current velocity is less than the average velocity and therefore we release the ball from the hand. This part was difficult to grasp and perform due to lack of trigger in our project.

The VIVE Trackers allows us to receive the coordinates of the tracker in virtual space. We started to analyze the trackers features and responsiveness as well as how many coordinates are taken in our definition of a throw. we got to a conclusion it has between 20-50 different coordinates in a throw.

We tried several algorithms to make realistic throw:

- Average direction vector: This algorithm takes all coordinates in the specific throw and calculates the average vector. The result was inaccurate direction.
- Numerical least squares: A numerical method to calculate an approximate direction vector. Our conclusion regarding this method that it is not accurate enough due to lack of coordinates.
- Normal distribution – Gauss bell: After the last two attempts we asked ourselves why they aren't working and what we need to take in more consideration. We got to a conclusion that the coordinates that are close to the median are more accurate, i.e. follows the player intended direction, and the coordinates that in the beginning and at the end are more inaccurate. That’s why we used the normal distribution aka Gauss bell, in order to take in more consideration of the median coordinates. Each coordinate was multiplied by a factor, the coordinates that are close to the median got the highest factor, and the rest got a factor according to their position. The result was a very accurate throw and we have accomplished our main goal.