234329 - Project in VR

Rubik's Cube Trainer Project

Final Report

By: Alexander Gurevich, Denys Svyshchov
Advisors: Boaz Sterenfeld, Yaron Honen

Spring 2018

Content

1. Introduction

Rubik’s Cube Trainer is VR application the main purpose of which is step by step learning of solving Rubik’s Cube and opportunity to solve cube on time and compete with friends who is the best Cube solver. The game uses VR gloves allowing the player realistic interaction and better understanding of solving process.

2. System & Technologies

Application uses VR headset HTC Vive[1] and Manus VR gloves[2]. It was developed on Unity game engine.

[1] https://www.vive.com/us/product/vive-virtual-reality-system/
[2] https://manus-vr.com/

3. Application Overview

The game include 2 modes: Studying and Freestyle. In Studying mode player follow the step by step instructions on the board which appears as karaoke. In parallel on cube there are flashing arrows which suggest right moves.

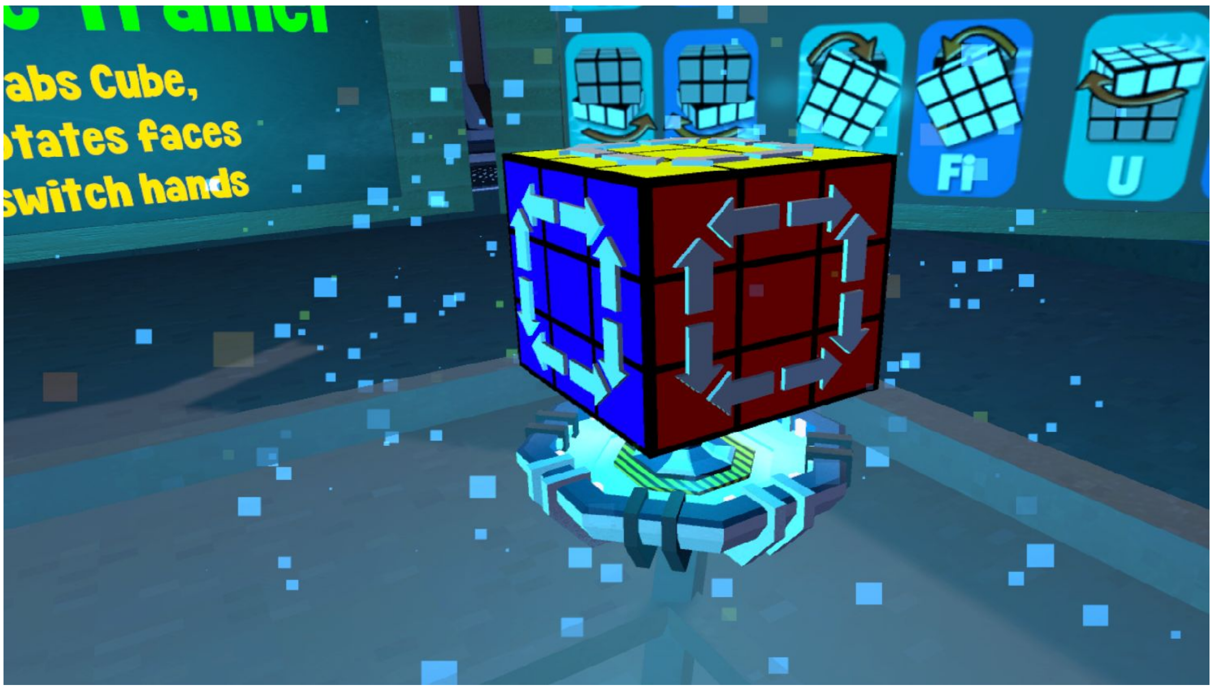
Second mode is Freestyle. The player must first enter his name and after that starts solving Rubik's Cube on time. After successful solving players result is adding to scoreboard.

3.1 Main Scene

Main scene represents place with table on which are located cube, two stands: cube holder and cube solver, button to return cube to initial position, lever for changing hand that grab the cube. There are boards opposite the table: scoreboard, main board with descriptions and instructions, and board with move definitions. Also there is menu beside the table.

3.1.2 Cube holder

Cube holder is stand which has area so if cube gets into it, cube will hover above stand.

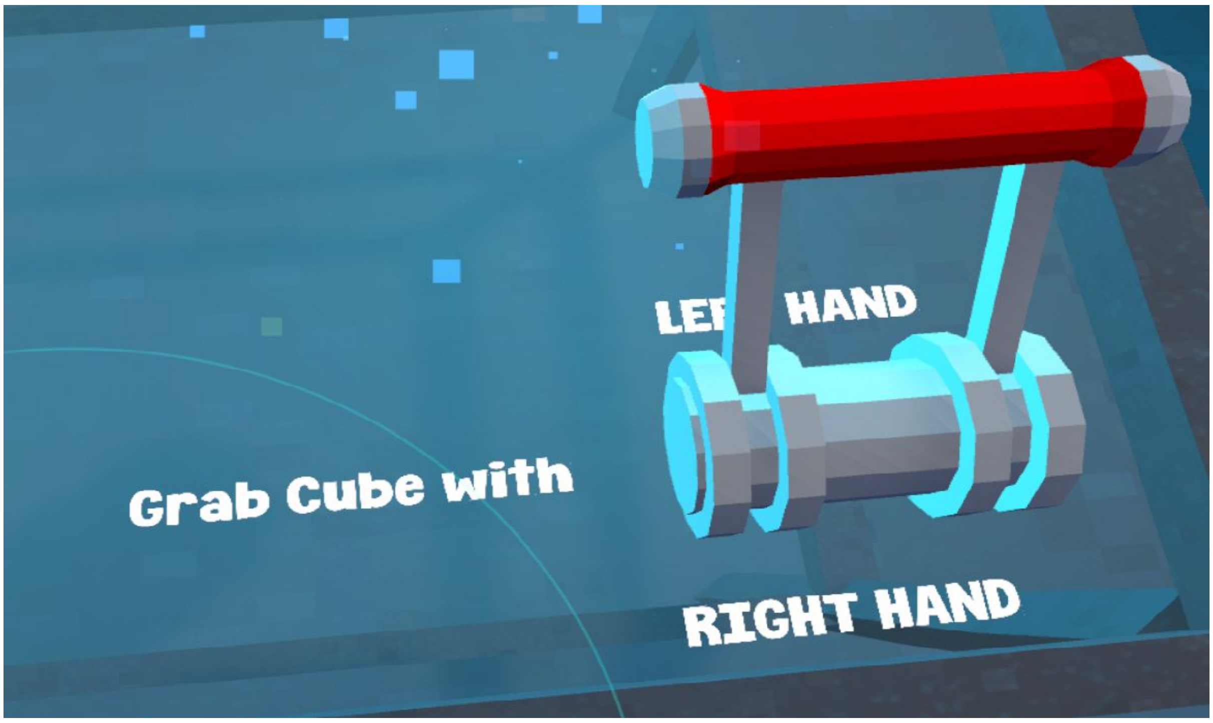


3.1.2 Cube Solver

Same object as cube holder, but additionally cube starts solving.

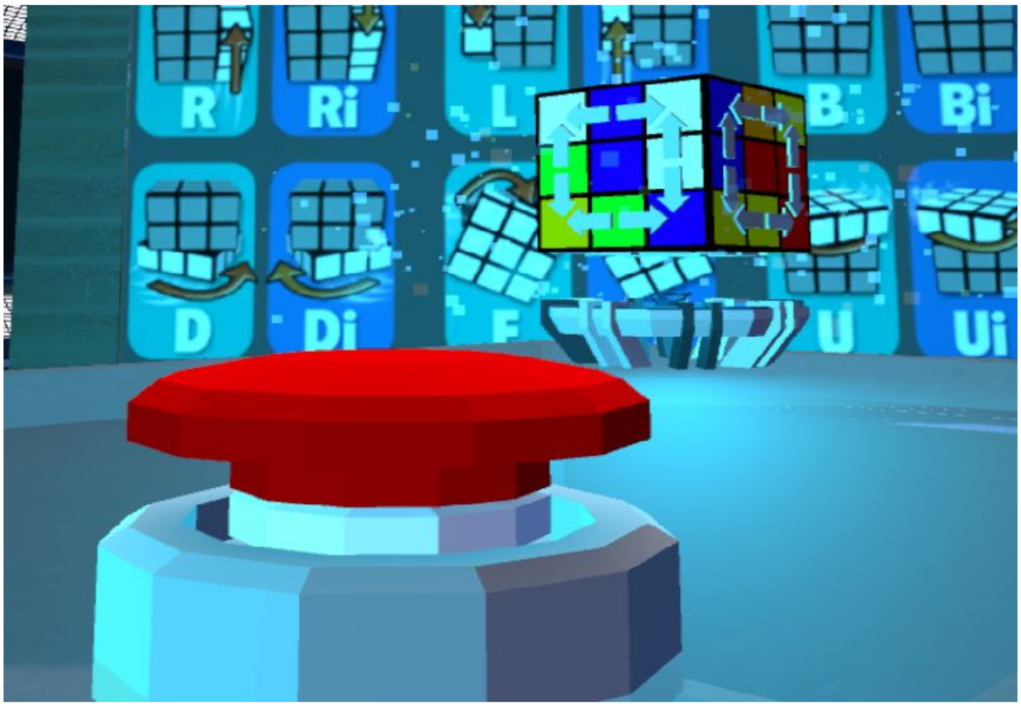
3.1.3 Lever

Lever which depending on the position change hand that grab the cube.



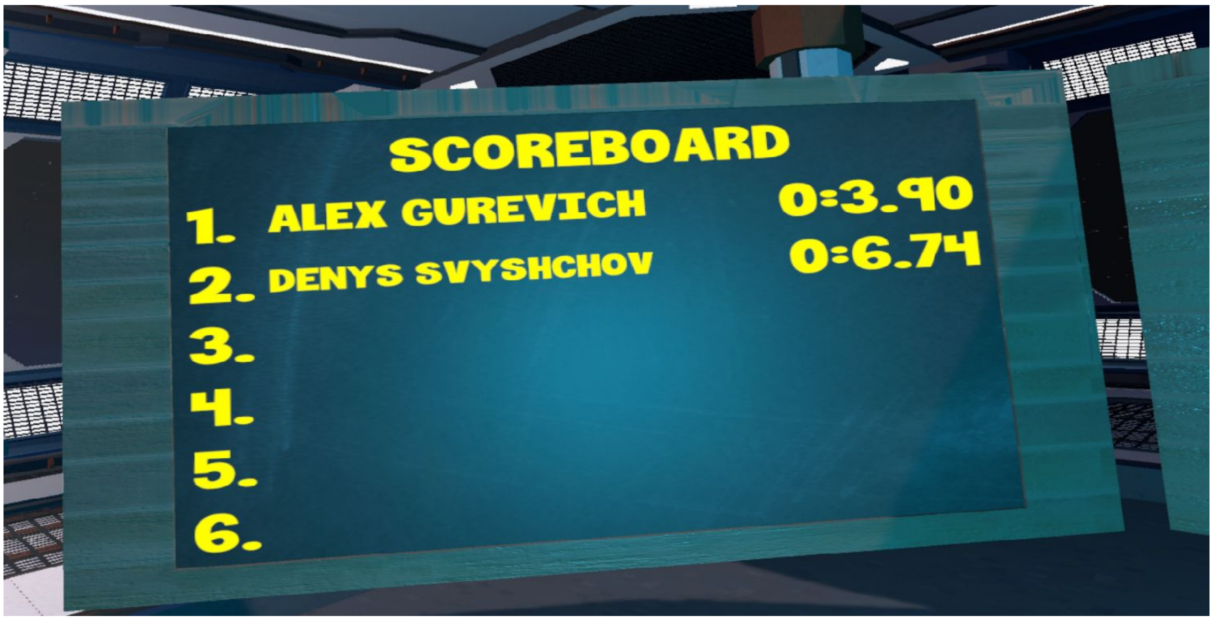
3.1.4 Return button

Button that returns cube to cube holder.



3.1.5 Scoreboard

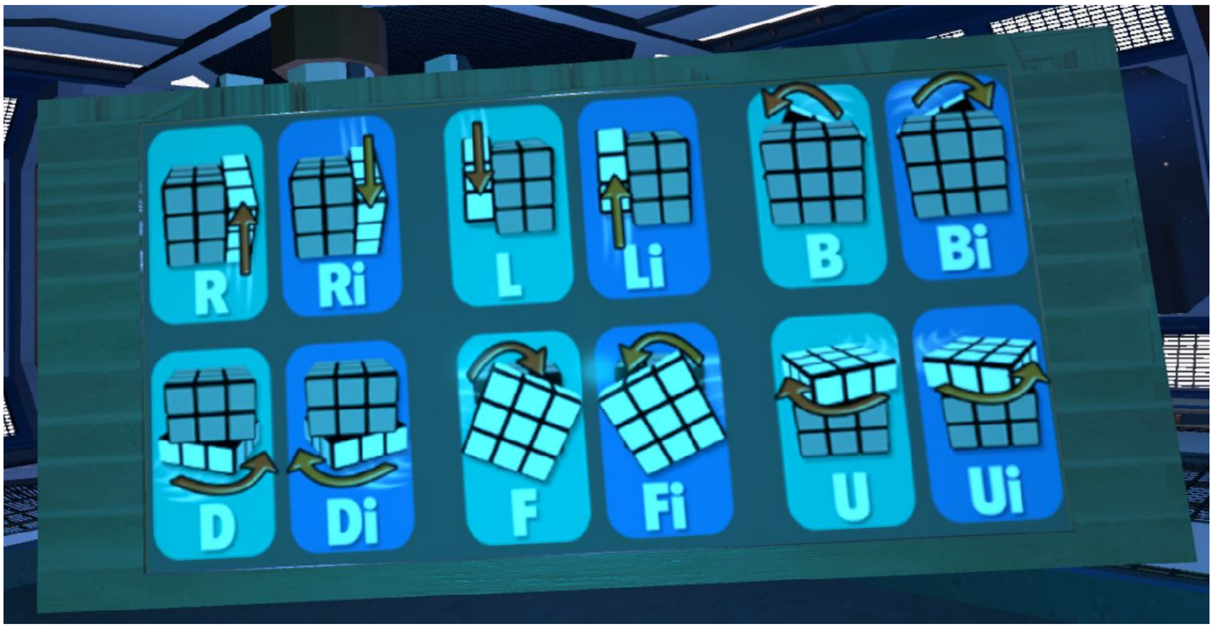
Board on which appears time of solving that players get in Freestyle mode


SCOREBOARD

1. ALEX GUREVICH 0:3.90
2. DENYS SVYSHCHOV 0:6.74
3.
4.
5.
6.

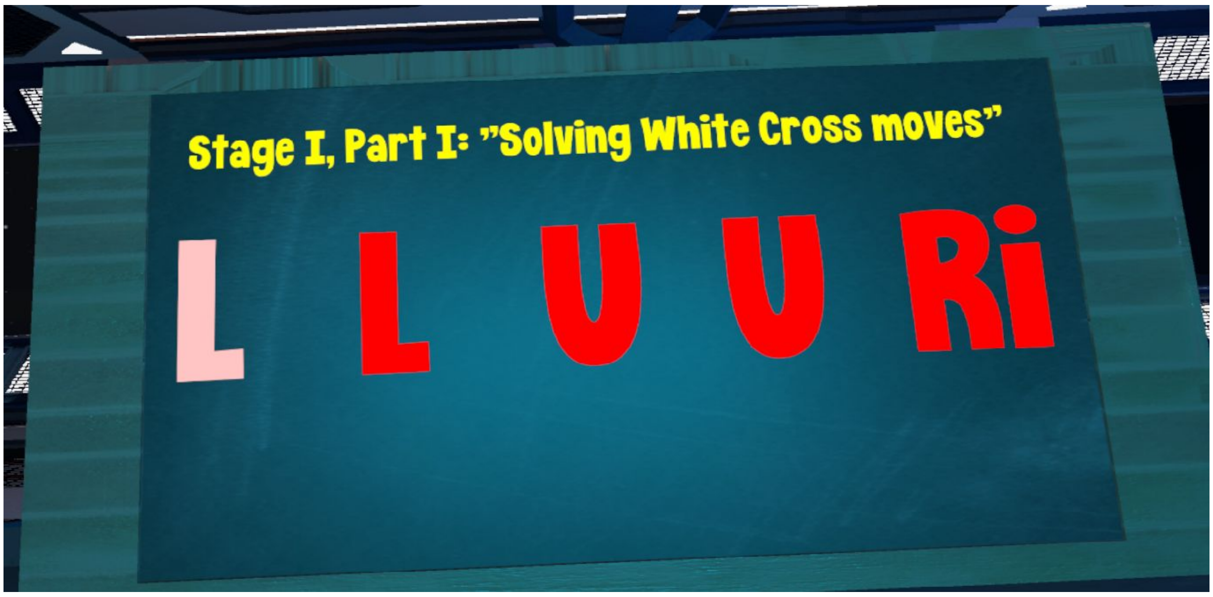
3.1.6 Signboard

Board with move definitions.

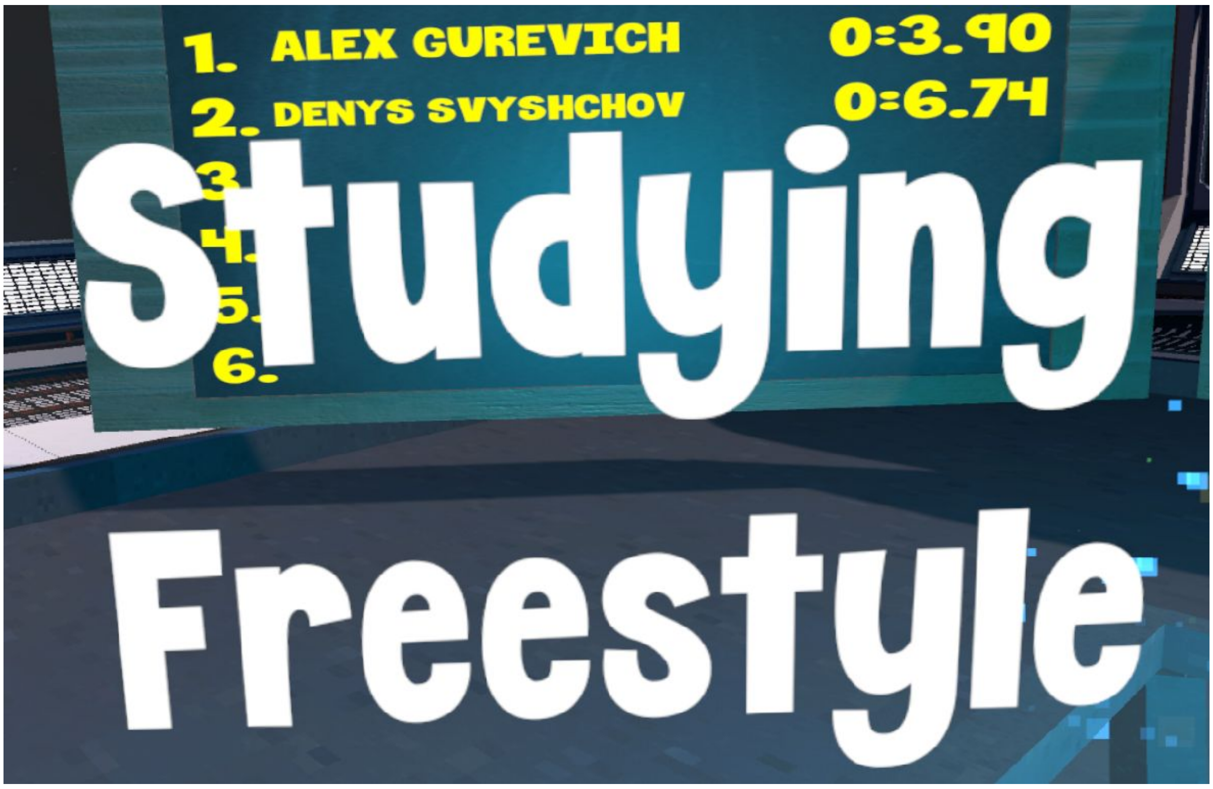


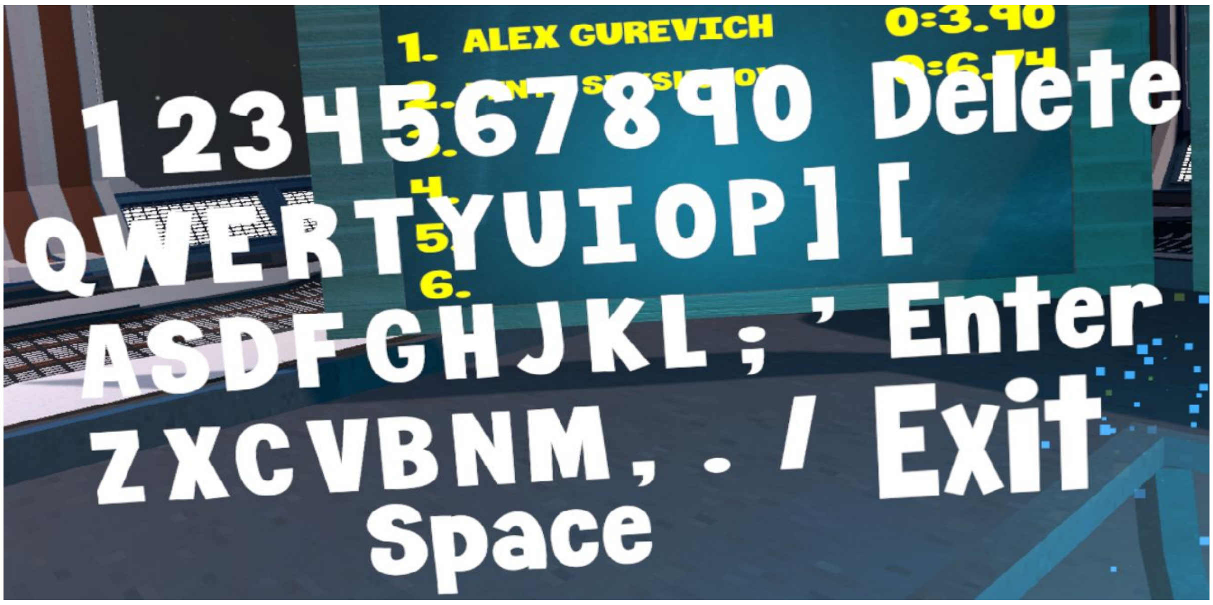
3.1.7 Mainboard

Board on which on start greetings and basi c rules are shown and in Studying mode appears step by step instructions and definition of current level of solving.



3.1.8 Menu and keyboard



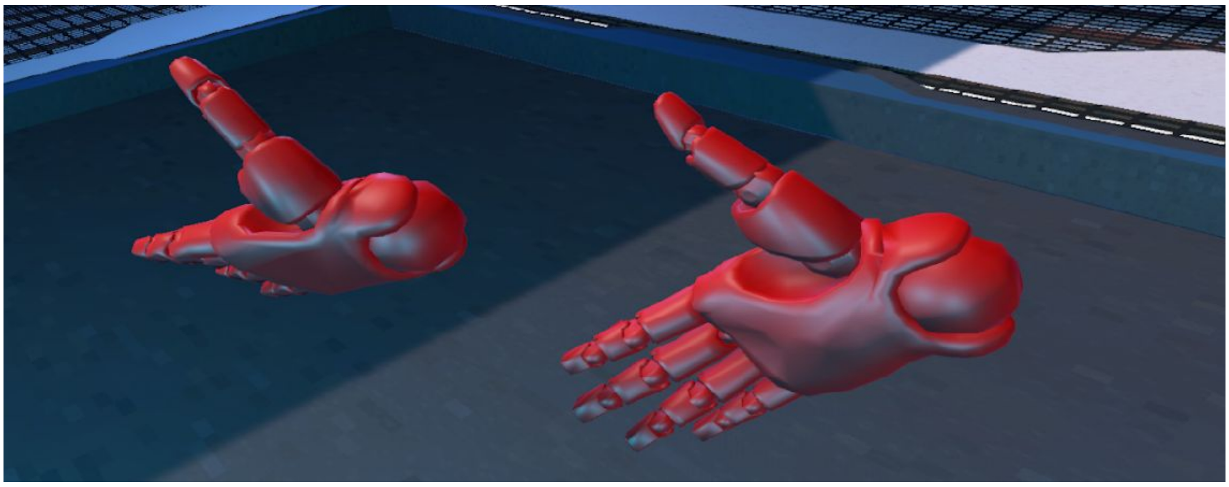


4. Contribution to Manus VR

During our work on the project we faced several issues of ManusVR gloves SDK. But we found a solution for them.

4.1 Finding a solution to save hands settings and load them in the beginning of the game once configured

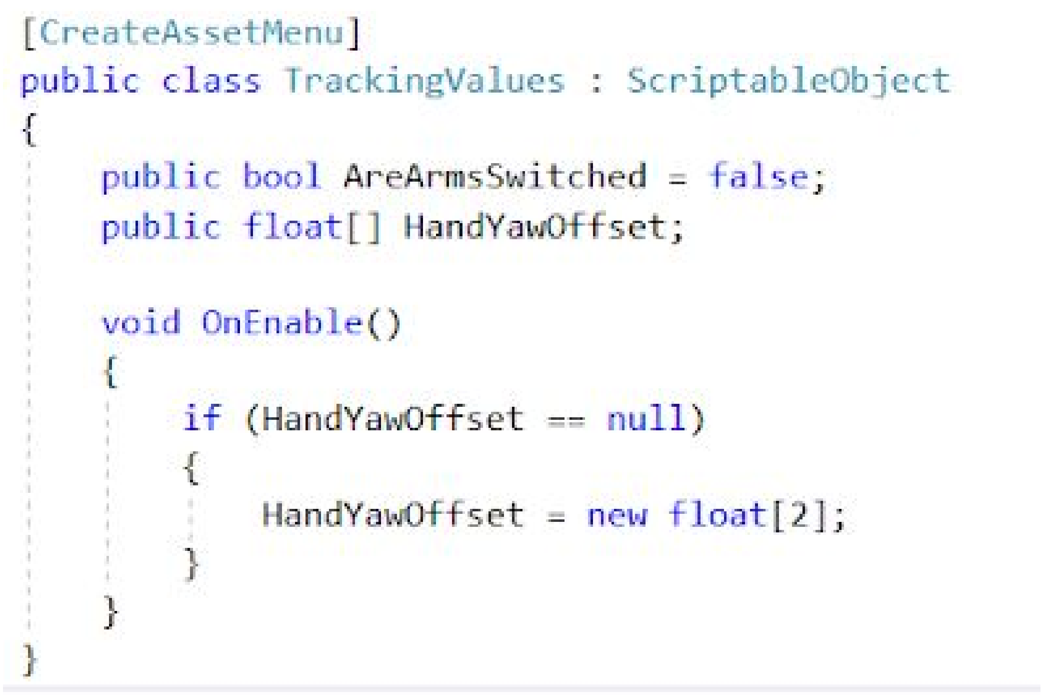
Here is the example of not configured gloves:



At the beginning SDK script does not know which tracker is attached to a specific hand. SDK script could swap trackers using “R” Key and turn hands using “Q”, “W”, “A” and “S” Keys. But the problem was how to save the settings, so you could import them back on the next game initialization.

We found a bug in ManusVR implementation: they used a Dictionary of hands in “TrackingValues.cs” script. According to Unity - Scripting API only simple types could be serialized. So the solution was simply to replace

Dictionary with a float Array.

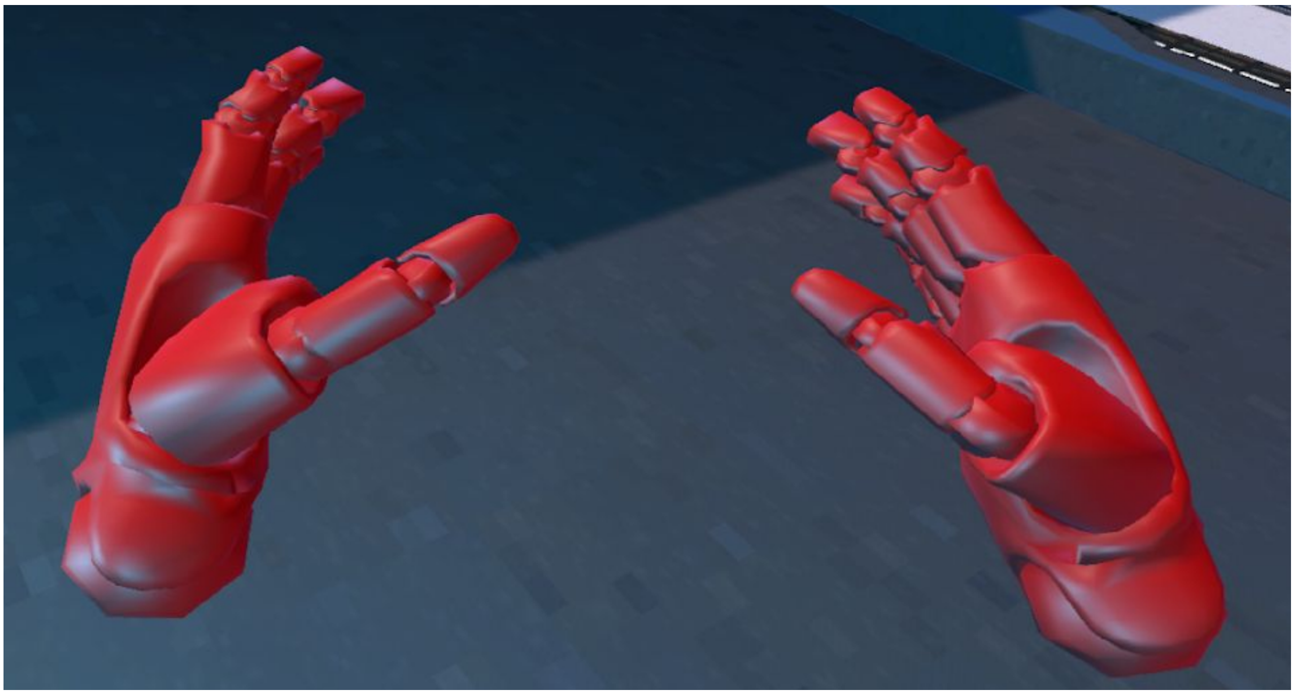


```
[CreateAssetMenu]
public class TrackingValues : ScriptableObject
{
    public bool AreArmsSwitched = false;
    public float[] HandYawOffset;

    void OnEnable()
    {
        if (HandYawOffset == null)
        {
            HandYawOffset = new float[2];
        }
    }
}
```

The only thing left is to fix compilation errors, when accessing HandYawOffset.

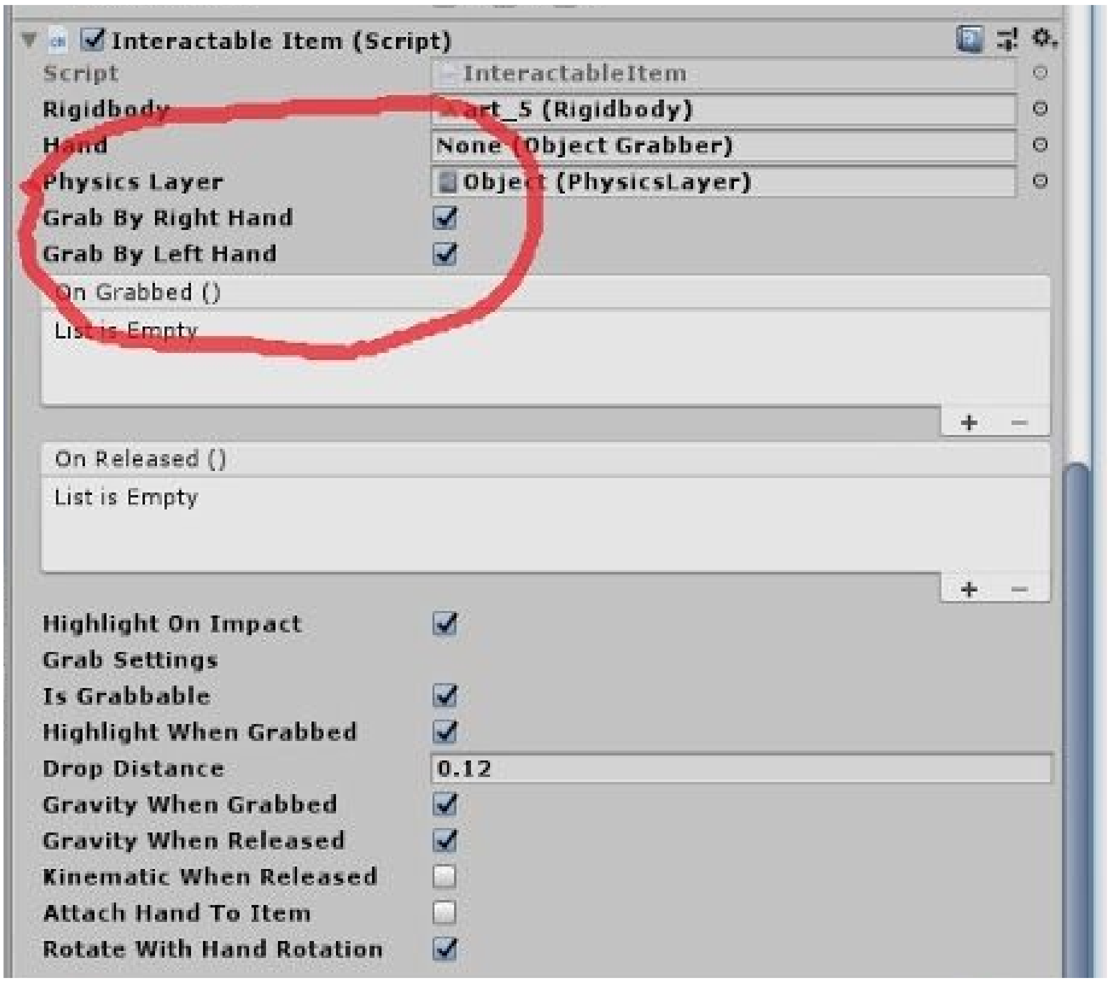
The result - gloves configuration is saved after initial configuration:



4.2 Expanding Manus VR SDK to be able to configure which hand could grab objects

In our project we want user to be able to grab object with one hand while pushing arrows with another. To add this support we improved *class Interactable.*

Now it is easy to configure split hands grab settings:



Interactable Item (Script) ✓	
Script	InteractableItem
Rigidbody	Part_5 (Rigidbody)
Hand	None (Object Grabber)
Physics Layer	Object (PhysicsLayer)
Grab By Right Hand	✓
Grab By Left Hand	✓
On Grabbed ()	List is Empty
On Released ()	List is Empty
Highlight On Impact	✓
Grab Settings	
Is Grabbable	✓
Highlight When Grabbed	✓
Drop Distance	0.12
Gravity When Grabbed	✓
Gravity When Released	✓
Kinematic When Released	☐
Attach Hand To Item	☐
Rotate With Hand Rotation	✓

4.3 How to grab complex object solution

By default ManusVR Interactable Item script adds colliders to all object inside of a parent. Because of this hand attaches to small particle of your object too, causing chaotic movement and disconnection of object particles.

In our project we used one rule to prevent this: If you want to make some item interactable create empty object which will represent interactable item. Add Interactable Item script to this objects. Empty object should have rigidbody and collider. Add simple script that copies transform of empty interactable object to complex object transform.

This way complex object always moves together with empty interactable object, but at the same time simplifies processing and makes it smoother.

5. Bibliography

In out project we used free assets available from Unity Asset Store - https://assetstore.unity.com/

We used a simple cube algorithm and cube model example from GitHub as a reference to create our own Interactable Rubik's Cube Model