**GOLDBERG MACHINE**

TEAM 지곡동 골드버그

김진수, Dept. of Computer Science and Engineering, POSTECH, 20202728

문민재, Dept. of Computer Science and Engineering, POSTECH, 20202405

이화윤, Dept. of Computer Science and Engineering, POSTECH, 20150221

1. **Project Description**

이번 프로젝트에서는 Unity를 이용하여 3D 골드버그 머신을 구현한다. 다양한 물체를 이용하여 골드버그 머신을 구현함으로써 다양한 material들과 prefab들을 활용하고, object를 제작하며 물체의 충돌과 physics-based animation에 대해 이해할 수 있다.

골드버그 머신은 아주 간단한 일을 하기 위해 지나치게 복잡하고 간접적인 방법을 하는 머신을 의미한다. 관련 없는 물체들의 각각의 동작이 그 다음 동작을 실행시키는 방식으로 구성된다. 따라서 골드버그 머신을 설계하기 위해서는 사용 될 물체의 특성과 물리 법칙에 대한 이해가 필요하다. 본 프로젝트에서는 다양한 물성을 가진 물체들을 활용하여 여러 기능들로 구성된 골드버그 머신을 구현하고자 한다.

골드버그 머신에서는 주로 물체 간의 충돌을 통해 한 장치의 동작에서 다음 장치의 동작으로 넘어간다. 충돌을 감지하는 방식에는 AABB, OBB 크게 두 가지가 존재한다. AABB란 Axis Aligned Bounding Box의 약자로 Bounding Box가 축에 나란하게 만들어진다. 따라서 구현이 쉽고 많은 연산이 필요하지 않다. 하지만 실제 물체와 Bounding Box의 차이가 클 수 있다는 단점이 있다. OBB는 Oriented Bounding Box의 약자로 Bonding Box가 축에 나란하지 않을 수 있다. 따라서 충돌을 확인하는 것이 더 복잡하다. 하지만 AABB보다 물체와 Bounding Box의 차이가 크지 않다는 장점이 있다. 이번 프로젝트에서 골드버그 머신을 구현해봄으로써 다양한 Collision Detection 방식에 대해 이해할 수 있다.

Computer Graphics로 사실적인 골드버그 머신을 구현하기 위해서는 물체가 현실에서 가지는 물리적 특성을 프로그램을 통해 재현해야한다. Unity에서는 RigidBody라는 component로 해당 object에 물리적 특성(ex. mass, friction, drag 등)을 부여할 수 있다. Rigidbody를 활용하여 여러 물성은 가진 object들을 생성할 수 있으며, 해당 물체가 가진 고유의 특성을 활용하여 다양한 물성의 물체들로 구성된 골드버그 머신을 구현할 수 있다.

1. **Programming Environment**

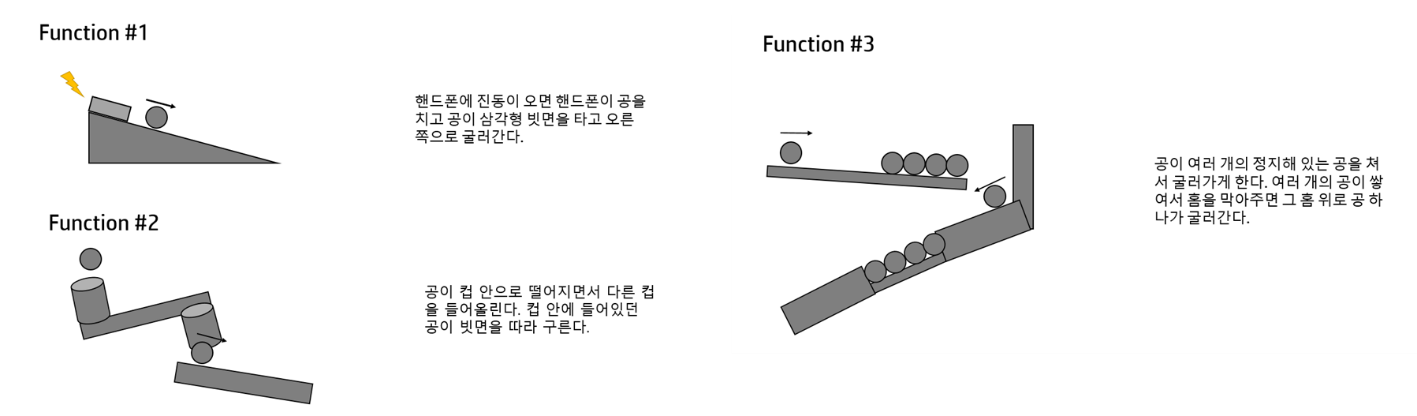
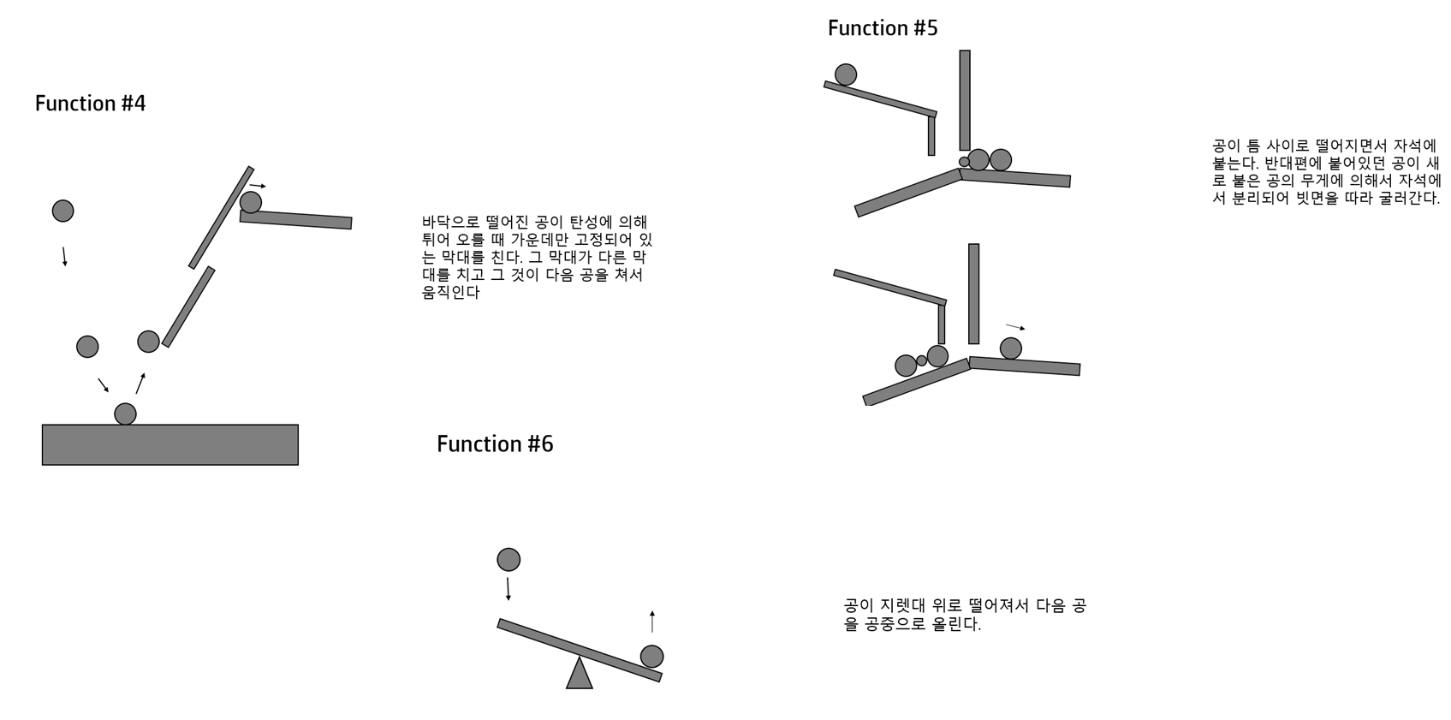
Unity 2019.3.15f1

Visual Studio 2019

Target Platform: Windows x86\_64

Api Compatibility Level: .NET Standard 2.0

1. **Design and Implementation**
   1. **Part 1**
      1. **Design**

****

Function1: 핸드폰에 진동이 오면 핸드폰이 공을 치고 공이 삼각형 빗면을 타고 오른쪽으로 굴러간다.

Function2: 공이 컵 안으로 떨어지면서 다른 컵을 들어올린다. 컵 안에 들어있던 공이 빗면을 따라 구른다.

Function3: 공이 여러 개의 정지해 있는 공을 쳐서 굴러가게 한다. 여러 개의 공이 쌓여서 홈을 막아주면 그 홈 위로 공 하나가 굴러간다.

Function4: 바닥으로 떨어진 공이 탄성에 의해 튀어 오를 때 가운데만 고정되어 있는 막대를 친다. 그 막대가 다른 막대를 치고 그 것이 다음 공을 쳐서 움직인다

Function5: 공이 틈 사이로 떨어지면서 자석에 붙는다. 반대편에 붙어있던 공이 새로 붙은 공의 무게에 의해서 자석에서 분리되어 빗면을 따라 굴러간다.

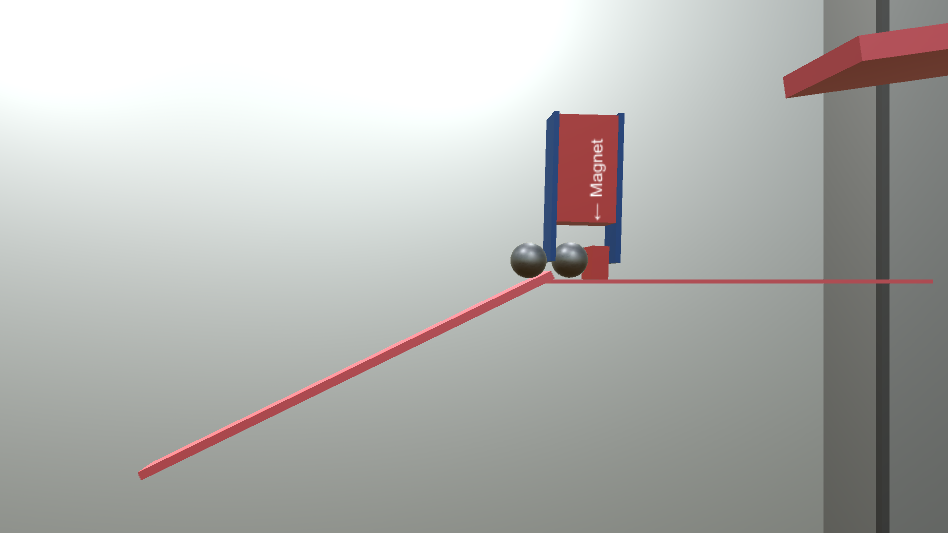
Function6: 공이 지렛대 위로 떨어져서 다음 공을 공중으로 올린다.

* + 1. **Implementation (Magnetic Force)**

이 스크립트의 아이디어는 https://luv-n-interest.tistory.com/252 에서 착안했다.

자석과 쇠공이 서로 잡아당기는 힘을 이용하여 골드버그 머신의 일부를 구현하였다. 자석의 자력이 작용하는 구간을 구형 collider로 설정하여 자석을 parent로 가지게 했다. 이 구역에 들어오면 자석에는 쇠 공을 향해 자석과 쇠 공의 거리에 반비례하는 힘을 주고 같은 크기의 힘을 쇠 공에는 자석을 향해 주게 하여 서로를 끌어당긴다. 자력이 작용하는 공간을 벗어나면 서로를 향한 인력이 사라진다.

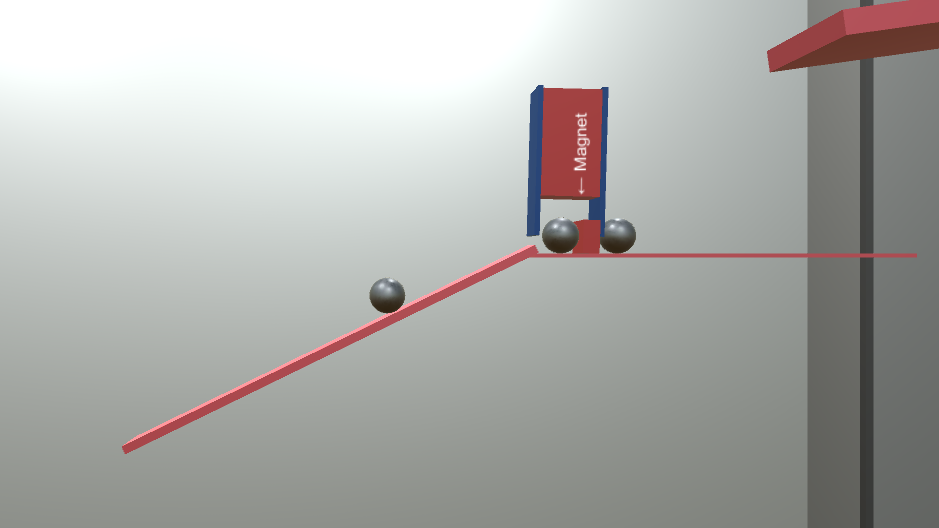
골드버그 머신에서 자력을 이용하여 구현한 부분은 다음과 같다. 벽을 사이에 두고 자력으로 붙어있는 빗면에 쇠 공1이 놓여있다. 이 쇠 공1은 자석과의 인력으로 빗면으로 굴러 떨어지지 않고 붙어있다. 반대편에서 굴러온 쇠 공2가 자석과 서로 잡아 당기게 되어 쇠 공2 쪽으로 자석이 이동하게 된다. 벽 때문에 쇠 공1은 자석에 끌려가지 못하게 되어 자력이 작용할 수 있는 공간을 벗어나게 된다. 자석으로부터 영향을 받지 않게 된 쇠 공1이 빗면을 따라 구르게 된다.



[그림 1] 자력으로 인해 굴러가지 않고 있는 빗면에 놓인 쇠 공1

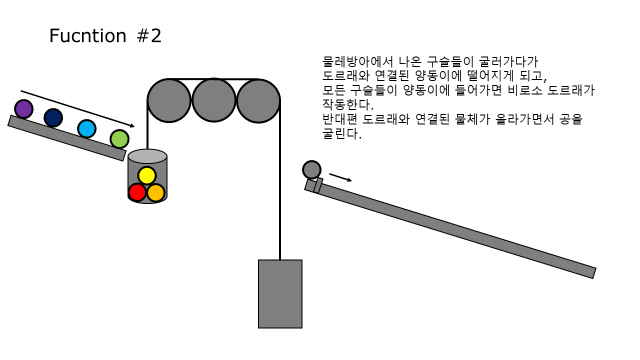
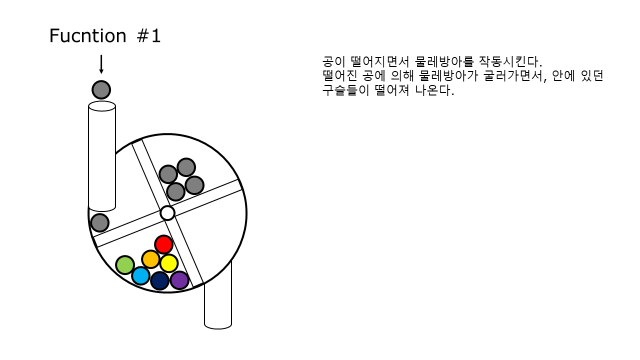


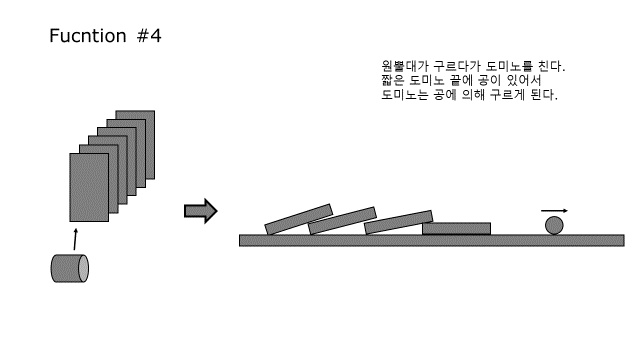
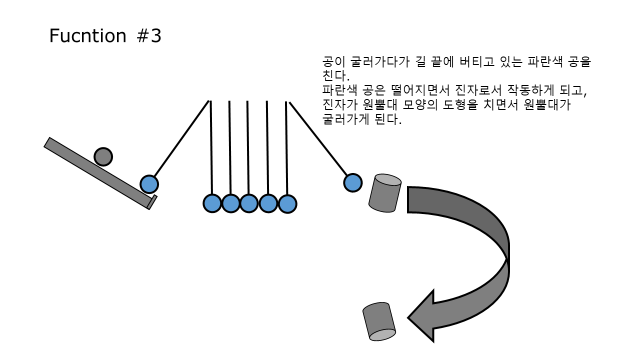
[그림 2] 쇠 공2에 의해 오른쪽으로 끌려가는 자석

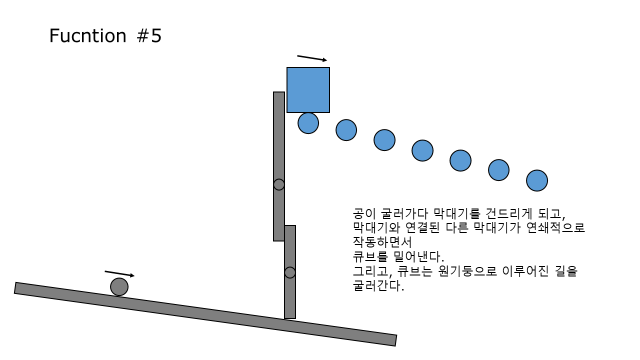


[그림 3] 자력이 작용하는 공간을 벗어나 빗면을 따라 구르는 쇠 공 2

* 1. **Part 2**
     1. **Design**







Function 1: 공이 떨어지면서 물레방아를 작동시킨다. 떨어진 공에 의해 물레방아가 굴러가면서, 안에 있던 구슬들이 떨어져 나온다.

Function 2: 물레방아에서 나온 구슬들이 굴러가다가 도르래와 연결된 양동이에 떨어지게 되고,

모든 구슬들이 양동이에 들어가면 비로소 도르래가 작동한다. 반대편 도르래와 연결된 물체가 올라가면서 공을 굴린다.

Function 3: 공이 굴러가다가 길 끝에 버티고 있는 파란색 공을 친다. 파란색 공은 떨어지면서 진자로서 작동하게 되고, 진자가 원뿔대 모양의 도형을 치면서 원뿔대가 굴러가게 된다.

Function 4: 원뿔대가 구르다가 도미노를 친다. 짧은 도미노 끝에 공이 있어서 도미노는 공에 의해 구르게 된다.

Function 5: 공이 굴러가다 막대기를 건드리게 되고, 막대기와 연결된 다른 막대기가 연쇄적으로 작동하면서 큐브를 밀어낸다. 그리고, 큐브는 원기둥으로 이루어진 길을 굴러간다

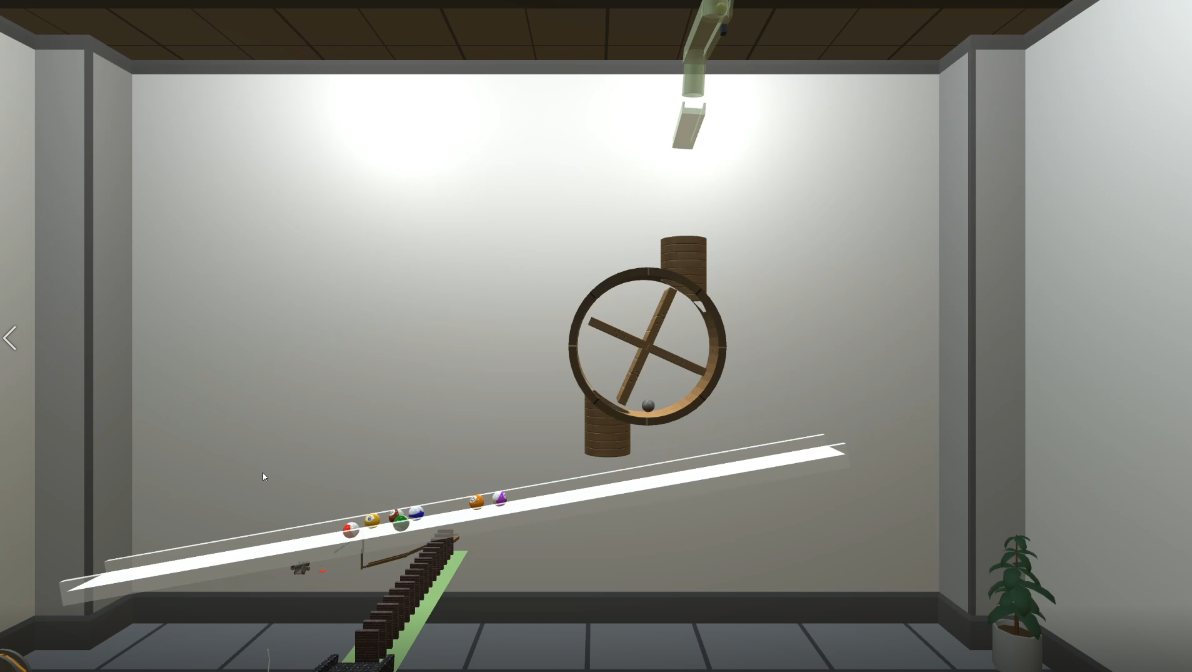
Function 6: 원기둥으로 이루어진 길을 굴러가던 큐브가 길이 끝남과 동시에 땅에 떨어져

바닥에 있던 버튼을 누르게 되고, 공 발사 장치를 작동시켜 공을 발사시켜 날려보낸다.

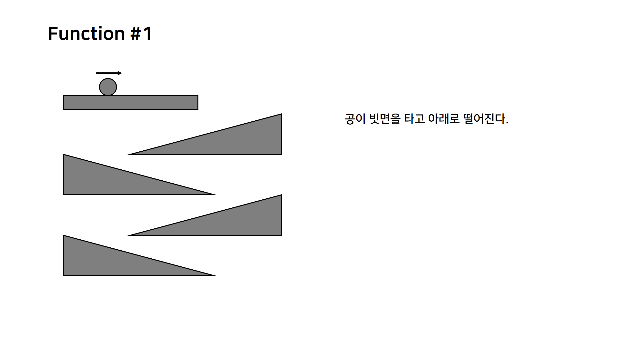
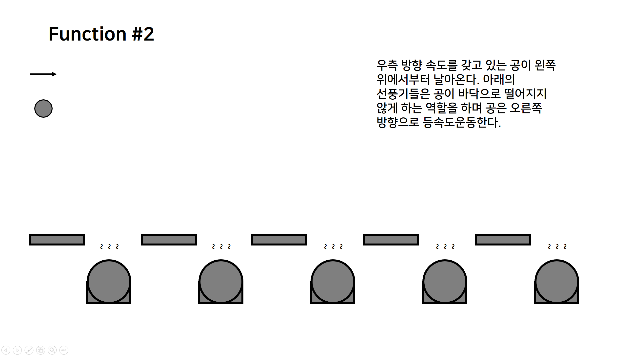
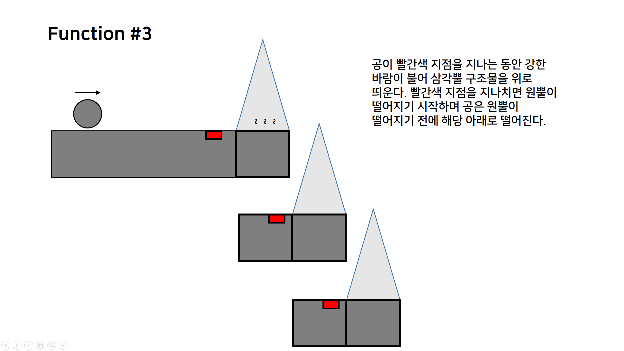
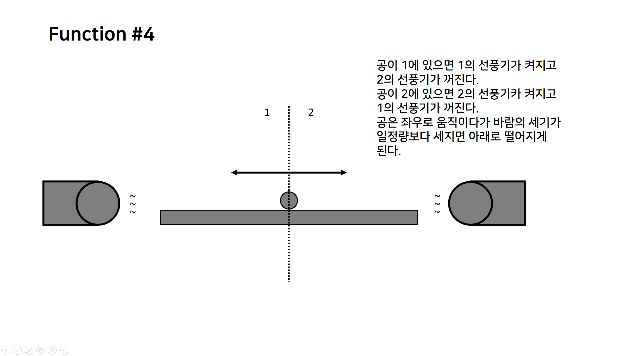
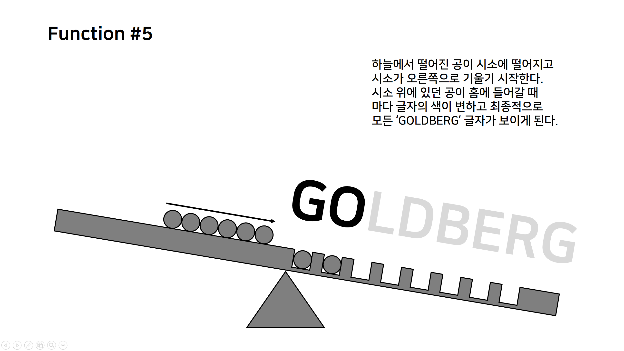
* + 1. **Implementation**

Function 2 의 양동이에 공이 가득 차면 반대편에 달려 있던 오브젝트가 올라오면서 공을 건드려 다음 function으로 넘어가게 하는 대신, 양동이에 공이 가득 차면 케이블이 끊어지면서 반대편에 매달린 오브젝트가 떨어져 다음 기능으로 넘어가게끔 구현되는 등 구현의 한계로 인해 디자인했던 바와 일부 다르게 구현되었다. 도르래에 사용되는 케이블의 구현은 Unity의 Hinge joint 기능을 사용하여 구현되었다.

Function 3 에서는 진자 여러 개가 연속되어 붙어있는 뉴턴의 요람 대신, 진자 한 개만을 사용하여 구현되었고, 마지막에 굴러가는 도형이 원뿔대에서 원기둥으로 바뀌는 등 디자인했던 바와 일부 다르게 구현되었다. 진자에 사용되는 케이블의 구현은 Unity의 Hinge joint 기능을 사용하여 구현되었다.



* 1. **Part 3**
     1. **Design**

Function 1: 공이 빗면을 타고 아래로 떨어진다.

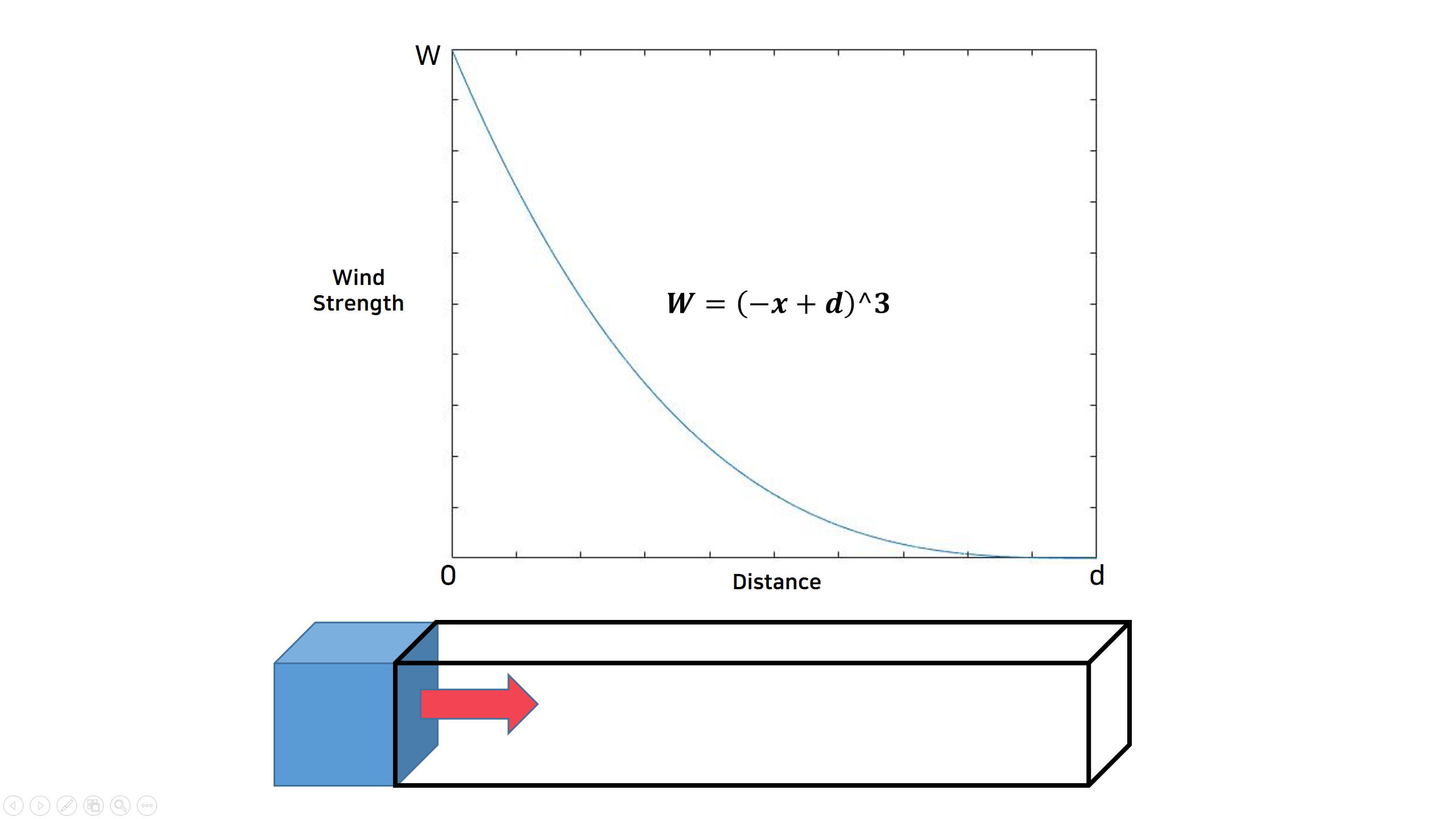
Function 2: 우측 방향 속도를 갖고 있는 공이 왼쪽 위에서부터 날아온다. 아래의 선풍기들은 공이 바닥으로 떨어지지 않게 하는 역할을 하며 공은 오른쪽 방향으로 등속도운동한다.

Function 3: 공이 빨간색 지점을 지나는 동안 강한 바람이 불어 삼각뿔 구조물을 위로 띄운다. 빨간색 지점을 지나치면 원뿔이 떨어지기 시작하며 공은 원뿔이 떨어지기 전에 해당 아래로 떨어진다.

Function 4: 공이 1에 있으면 1의 선풍기가 켜지고 2의 선풍기가 꺼진다. 공이 2에 있으면 2의 선풍기카 켜지고 1의 선풍기가 꺼진다. 공은 좌우로 움직이다가 바람의 세기가 일정량보다 세지면 아래로 떨어지게 된다.

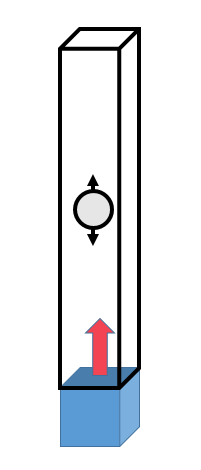
Function 5: 하늘에서 떨어진 공이 시소에 떨어지고 시소가 오른쪽으로 기울기 시작한다. 시소 위에 있던 공이 홈에 들어갈 때 마다 글자의 색이 변하고 최종적으로 모든 ‘GOLDBERG’ 글자가 보이게 된다.

* + 1. **Implementation (Wind & Air resistance)**



Part 3의 function 2, 4, 5는 바람을 통해 공을 움직인다. 바람의 세기는 바람이 발생하는 지점으로부터 멀어질수록 약해진다. 이를 구현하기 위하여 object에 가해지는 힘을 다음 식과 같이 매핑하였다.

위 식에서 W는 바람의 세기, x는 바람이 적용되는 영역 안에서 바람이 발생하는 지점으로부터 object까지의 거리, d는 바람이 적용되는 영역의 길이를 의미한다. 위 식과 같이 객체에 힘을 적용하였을 때, 실제 바람으로부터 적용되는 힘과 유사한 결과를 얻을 수 있었다.



바람이 구현되어 있을 때 공기 저항이 구현되지 않는다면 다음과 같이 바람이 x축 방향으로 적용되는 상황에서 object는 영원히 진동한다. 이를 해결하기 위해서는 공기저항을 적용 해야하며, 공기 저항이 적용된다면 위의 공은 진동하다가 결국 중력과 바람의 힘이 평형인 점에서 정지해 있게 된다. 공기저항을 구현하기 drag equation을 근사 하였으며 drag equation은 다음과 같다.

위 식에 따라 공기저항은 object의 속도의 제곱에 비례하도록 구현하였다.

* 1. **Camera Movement**

카메라의 위치나 움직이는 방식에 따라 공의 움직임을 보다 생동감 있게 느끼거나, 기능의 구현을 보다 자세하게 확인할 수 있다. 여러 시점에 따라 해당 골드버그 머신을 분석할 수 있기 위해 동적인 카메라와 정적인 카메라 두 가지를 구현하였다.

동적인 카메라는 키보드 입력을 통해 카메라의 x축, y축, z축 좌표를 변경할 수 있으며 y축 + 방향과 – 방향으로 카메라를 회전할 수 있다. 해당 카메라를 사용한다면 기능의 진행방향을 따라 카메라를 따라 움직일 수 있으며, 회전 및 이동을 통해 사용자에게 골드버그 머신을 실제로 눈앞에서 보고 있는 듯한 느낌을 전달할 수 있다.

정적인 카메라는 키보드 입력에 따라 카메라의 transform을 변환할 수 있다. 해당 프로그램에는 총 17개의 카메라의 position과 해당 position의 rotation 값을 갖고 있으며 키보드의 오른쪽, 왼쪽 방향키를 통해 다음 상황이나 이전 상황으로 카메라의 transform을 변환할 수 있다. 해당 카메라는 동적인 카메라에 비해 작동하기 쉬우나, 카메라의 위치가 움직이기 않기 때문에 생동감을 느끼기는 어렵다.

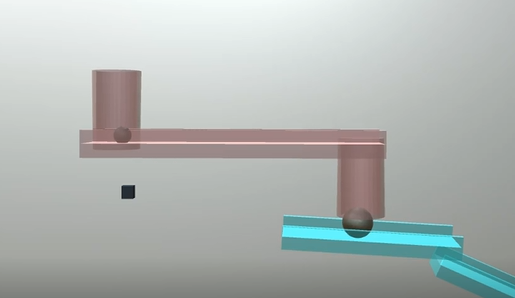
1. **How to Run**

진수

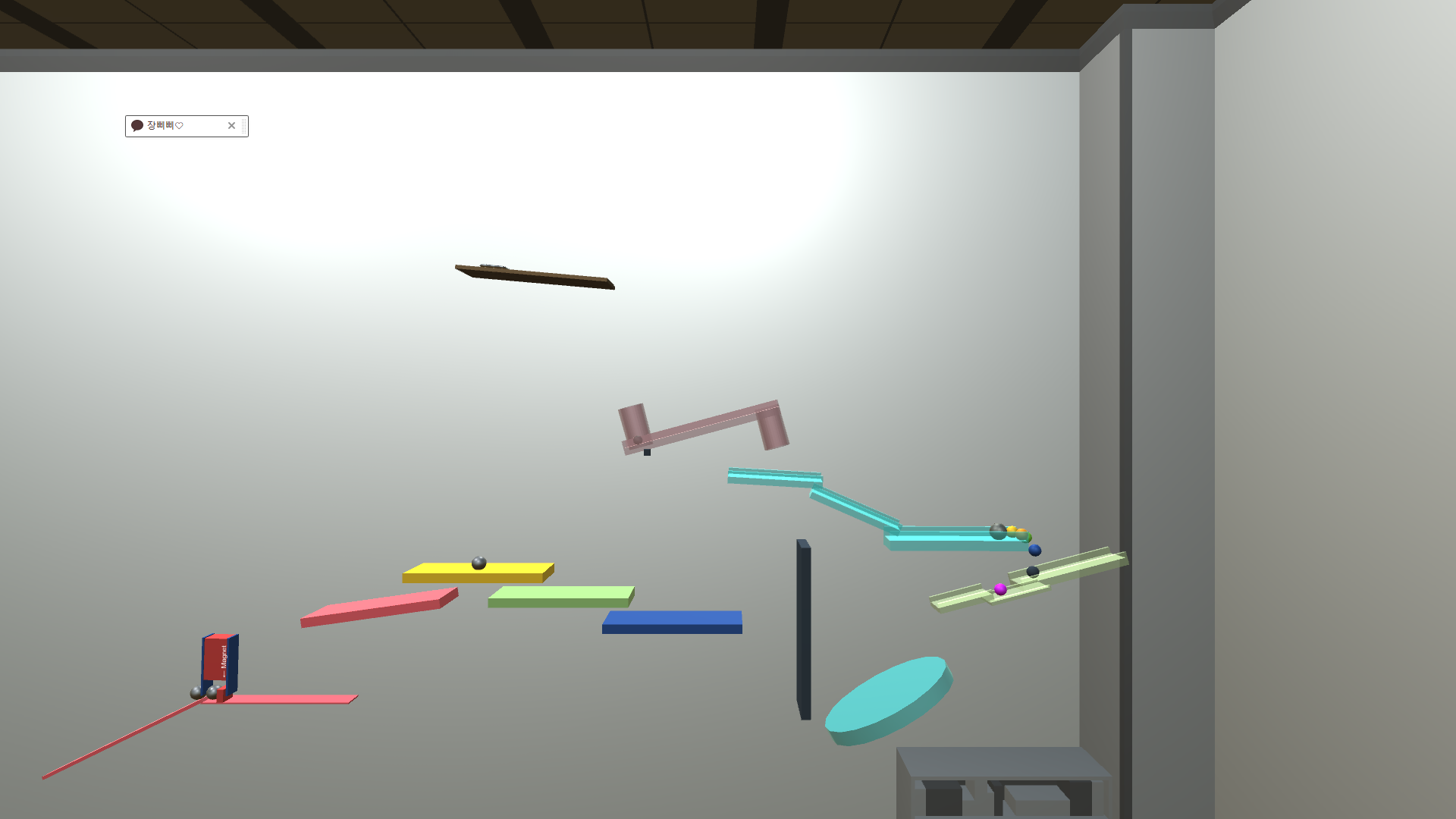
1. **Example**
   1. **Part 1**



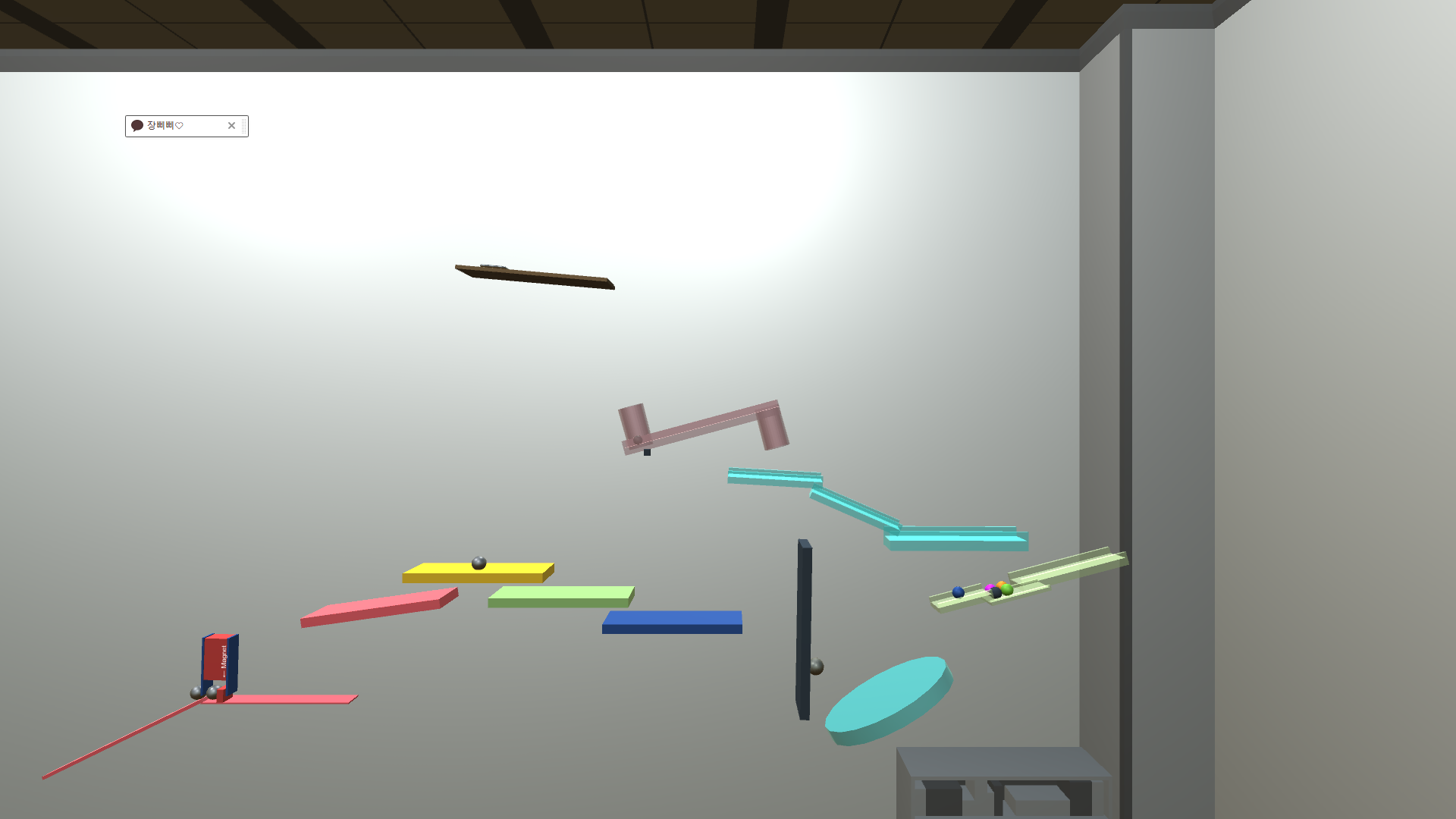
핸드폰에 진동이 울리면 작은 턱에 걸쳐있던 공이 빗면을 따라 움직인다.



공이 컵안으로 떨어지면서 다른 컵을 들어올려 다음 공을 출발시킨다.



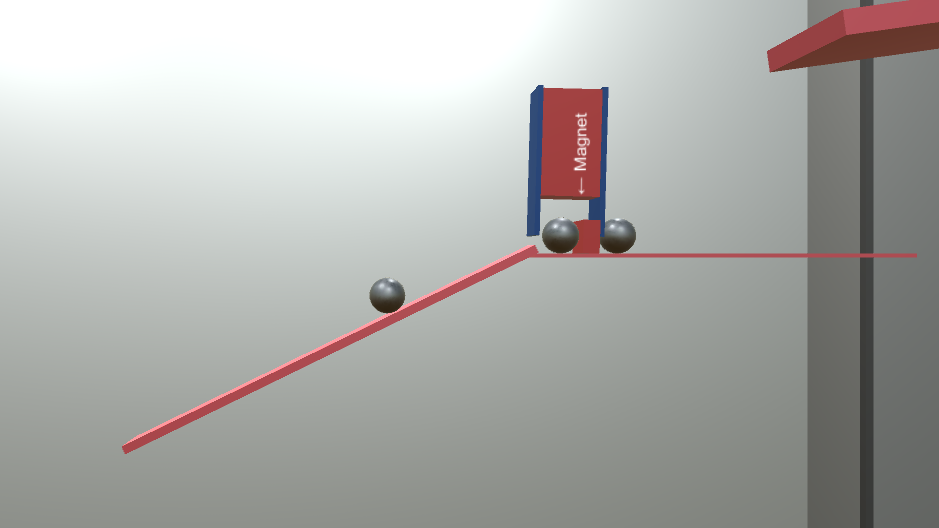
공이 여러 개의 멈춰있던 작은 공들을 친다. 이 공들이 떨어지면서 작은 틈이 있는 다리를 메꾸고 그 다리 위로 큰 공이 지나간다.



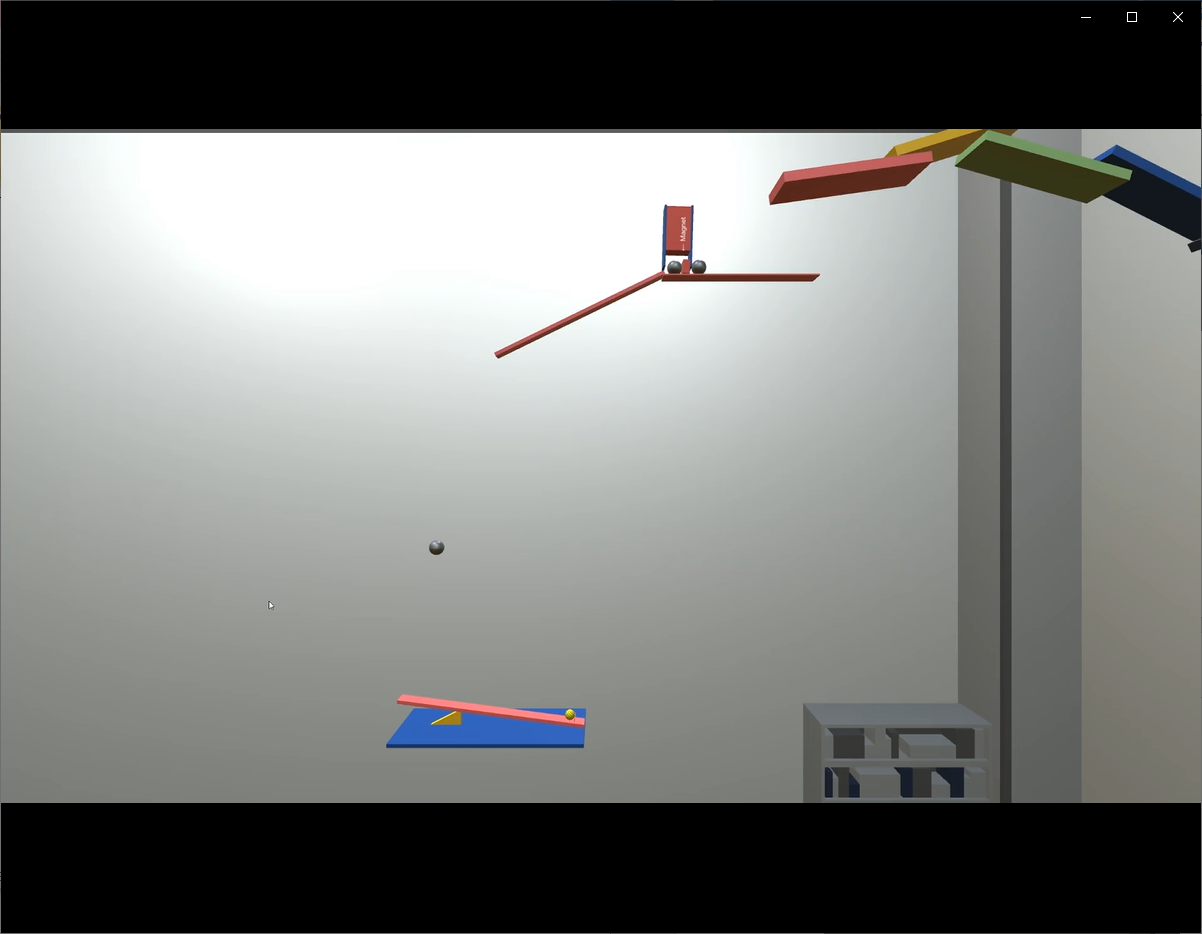
공이 떨어지다가 탄성이 있는 물체를 만나 튕기면서 판자를 치게 된다.



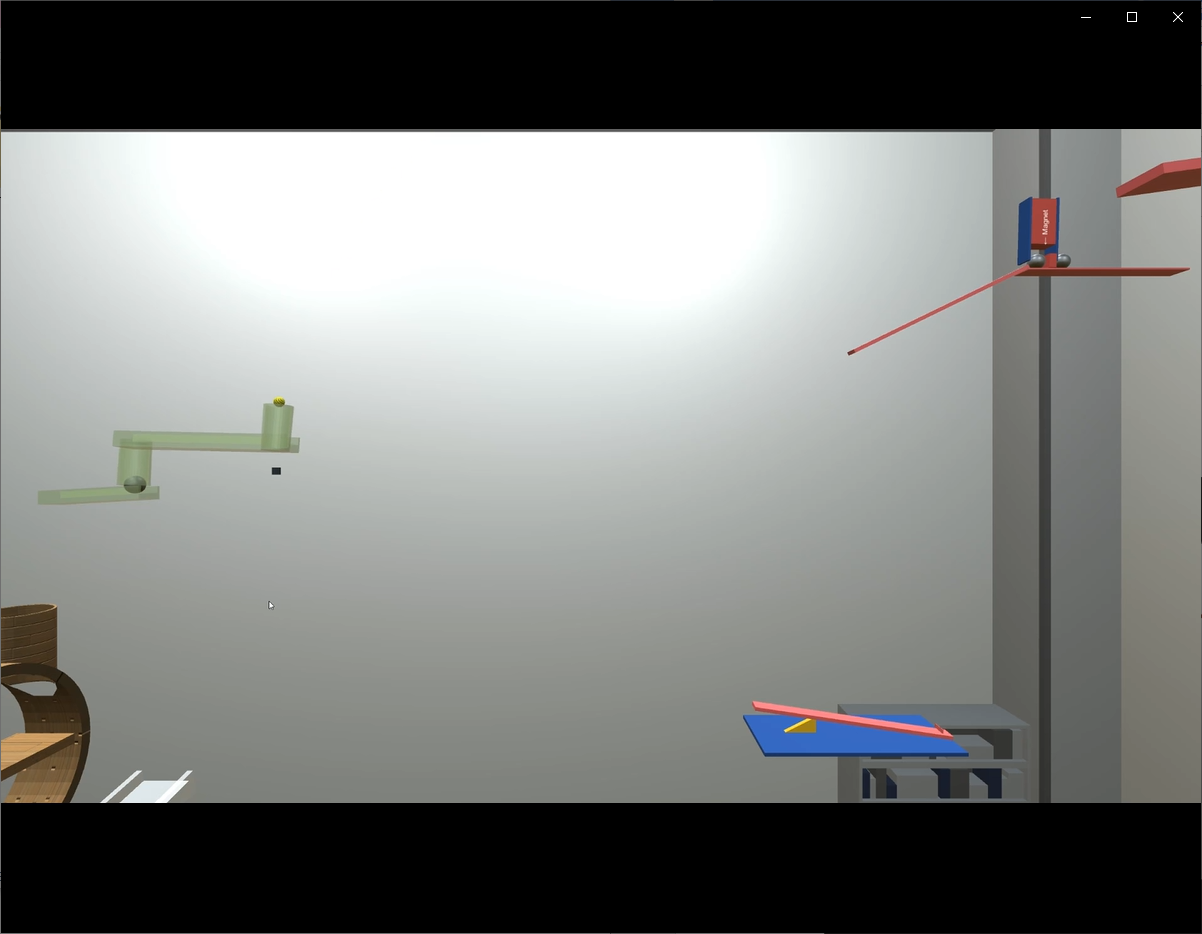
판자끼리 연쇄적으로 충돌이 일어나면서 판자 위에 놓여있던 공이 구른다.



굴러 내려온 공이 자석을 끌어당겨 반대편에 붙어있던 공이 자력의 범위에서 벗어나게 되어 빗면을 따라 구른다.

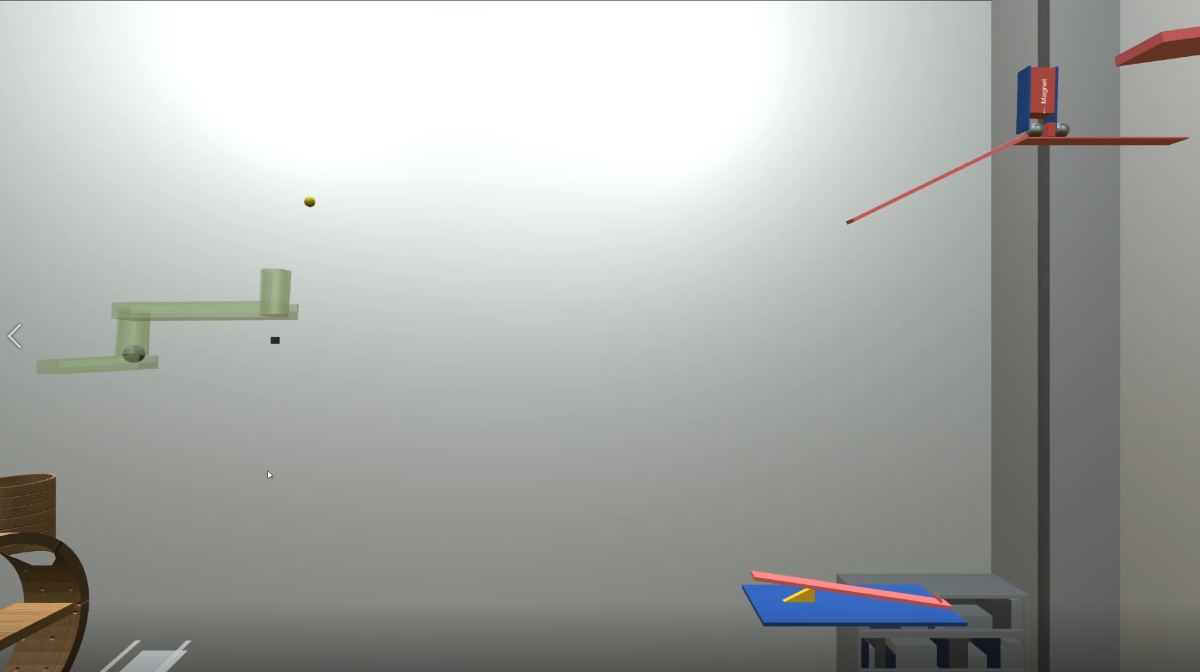


공이 지렛대 위로 떨어져 지렛대 위에 있던 작은 공을 위로 쏘아 올린다.

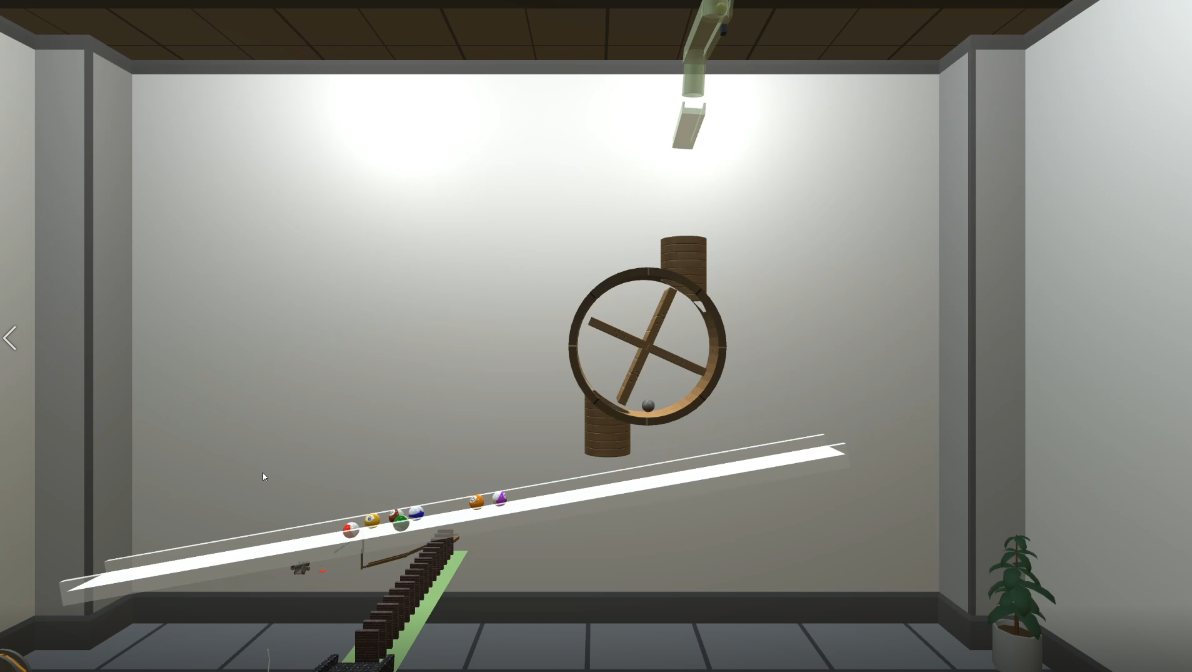


쏘아 올려진 공이 컵에 들어가면서 다른 컵을 들어올려 다음 공을 출발시킨다.

* 1. **Part 2**



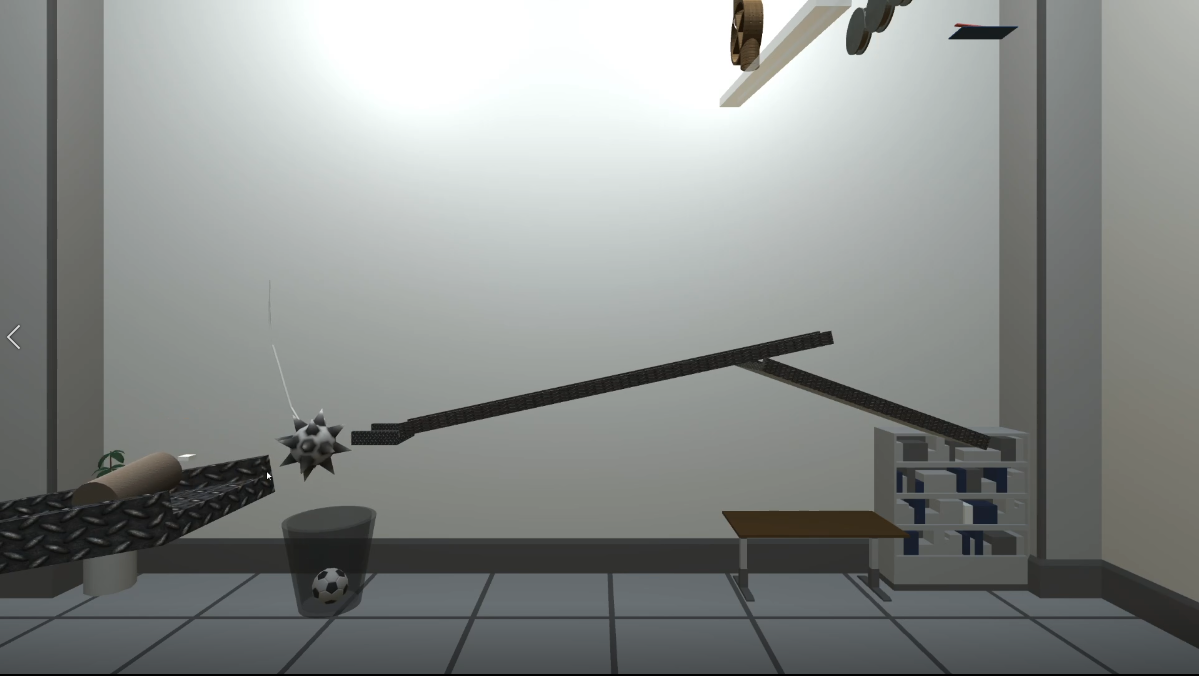
Part 1과 Part 2의 연결부분이다. Part 1에서 Part 2로 좀 더 자연스럽게 연결되게 하기 위해 작은 구조물을 하나 추가하였다. Part 1의 마지막 시소로부터 공이 날아와 컵에 들어가면, 반대쪽에 있던 큰 쇠공이 굴러가면서 Part 2로 이어진다.



Function 1의 작동 모습이다. Part 1과 2의 연결부에서 공이 떨어지며 물레를 작동시키고, 물레 안에 원래 들어있던 7가지 색의 공들이 떨어지면서 굴러간다.



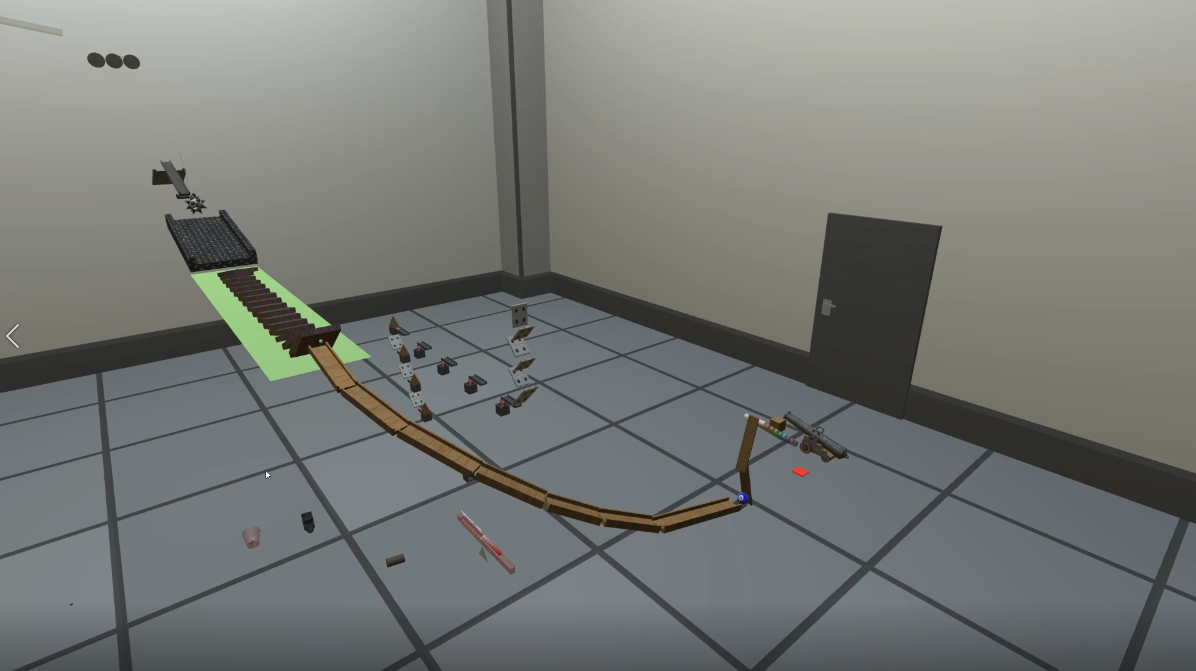
Function 2의 작동 모습이다. Function 1의 공들이 양동이로 모두 떨어지면, 모루와 연결된 케이블이 끊어지면서 양동이와 케이블로 연결된 모루가 떨어진다.



Function 3의 작동 모습이다. 모루가 떨어지면서 판이 기울게 되고, 그로 인해 축구공이 굴러가게 된다. 축구공이 구르다가 철퇴와 부딪히면, 줄에 매달린 철퇴가 떨어지면서 진자 운동을 통해 반대편의 원기둥을 쳐 구르게 한다.



Function 4의 작동 모습이다. 원기둥이 굴러가면서 도미노를 넘어뜨리고, 짧은 도미노 끝에 위치한 파란색 당구공이 마지막 도미노가 넘어짐으로 인해 굴러가게 된다.



Function 5의 작동 모습이다. 파란색 당구공이 굴러가다 나무 판을 건드리면, 나무 판이 회전하면서 나무 큐브를 밀게 되고, 원기둥으로 이루어진 길을 굴러가게 된다.



Function 6의 작동 모습이다. Function 5에 의해 굴러가는 나무 큐브가 떨어지며 버튼에 닿게 되면 버튼이 파란색으로 변하고, 잠시 후 대포에서 포탄이 날아가며 Part 3으로 이어진다.

* 1. **Part 3**

1. **DISCUSSION**
   1. **Part 1**
   2. **Part 2**
   3. **Part 3**
2. **CONCLUSION**
   1. **Part 1**
   2. **Part 2**
   3. **Part 3**
3. **DIRECTION OF IMPROVEMENT**
4. **REFERENCES**
   1. **General**
      1. 다양한 material과 mesh의 공

Ball Pack

https://assetstore.unity.com/packages/3d/props/ball-pack-446

Billiard Balls

<https://assetstore.unity.com/packages/2d/textures-materials/billiard-balls-6353>

Low Polygon Soccer Ball

https://assetstore.unity.com/packages/3d/low-polygon-soccer-ball-84382

* + 1. 삼각 기둥 및 다리 등의 조형물

Prototyping Pack

https://assetstore.unity.com/packages/3d/prototyping-pack-free-94277

* + 1. Materials

Metal, wood, brick

<https://assetstore.unity.com/packages/2d/textures-materials/prototype-materials-pack-65136>

문양을 가진 material

<https://assetstore.unity.com/packages/2d/textures-materials/metals/yughues-free-metal-materials-12949>

* + 1. 배경 장식에 사용된 오피스 테마<https://assetstore.unity.com/packages/3d/environments/snaps-prototype-office-137490>
  1. **Part 1**
     1. Function 1의 스마트폰

https://assetstore.unity.com/packages/3d/props/electronics/free-smartphone-90324

* 1. **Part 2**

1. Function 2의 모루

<https://assetstore.unity.com/packages/3d/props/weapons/cartoon-heavy-weights-2857>

1. Function 6의 대포 및 포탄

<https://assetstore.unity.com/packages/3d/props/cannon-on-a-platform-57534>

* 1. **Part 3**