In Class Practice #2

(Collision Detection and Tunneling)

Name, Surname:	İnan Kazancı
Group #:	2
Group Member Names:	Alperen Garip, İnan Kazancı, Abdurrahman Arslan, Asya Su Şen

STEP 1: System Summary

Component	Your Information
Operating System	Ubuntu 24.04.4 LTS
CPU	Intel® Core™ i7-7700HQ × 8
RAM	16.0 GiB
Unity Version	2022.3.62f3
Fixed Update Rate	50fpx (per 0.02 second)

STEP 2: Download and Run the Project

- Download the Unity project from the link provided by the instructor
- Open the scene: Scenes/TunnelingDemo
- Press Play and observe the tunneling effect in action

What to look for: The ball may pass through barriers without collision being detected (appears on the other side without bouncing or slowing down).

STEP 3a: Identify Parameters Affecting Tunneling

Task: Discuss as a group and identify 4-6 parameters that affect the Tunneling Phenomenon.

Requirements for each parameter:

- Must be adjustable in Unity Inspector OR modifiable in C# scripts
- Must be related to physics simulation
- Specify WHERE the parameter is located (e.g., "Rigidbody component → Mass")

Parameter List

Parameter	Location	Expected Effect
#1: Barrier Thickness	Barrier GameObject → Transform → Scale (X-axis)	Thinner barriers = more tunneling
#2:direction	Roller GameObject → Ball Booster → Direction → X	smaller magnitude direction = less tunelling
#3:scale	Roller GameObject → Transform → Scale	larger scale => les tunelling
#4:mass	Roller GameObject → Rigidbody → Mass	larger mass = > les tunelling
#5:drag	Roller GameObject → Rigidbody → Drag	larger drag => les tunelling
#6: (Optional)		

STEP 3b: Exploratory Testing

Instructions:

- Each group member should experiment with different parameter values on their own machine
- Try extreme values (very high/low) to see clear effects
- Note which parameters seem to have the strongest impact on tunneling
- Share your findings within your group

Note your observations here:

Everything related to speed seems to affect tunelling ie if it has more force, less mass, less drag the ball will move faster and we'll experience tunelling

STEP 4: Design Collision Detection Experiments

Important: For all experiments, Collision Detection Mode must be set to Discrete (found in the Rigidbody component of the ball).

Hypothesis Section

Before conducting experiments, discuss and record your predictions:

Which 2 parameters do you predict will have the strongest effect on tunneling?

direction in ballbooster script, and the thickness of the

barrier

Why? (Based on your understanding of physics engines)

if the ball is moving so fast that it moves beyond the barrier in one frame tunelling happens. If the ball is slower or if the barrier is thicker it will be more probable to detect the collision

Experimental Design

Choose 2 parameters from your Step 3a list to test systematically.

Parameter	Details	Test Values
Direction	Name: Direction Location: BallBooster script of the roller	Low Value: -15 High Value: -100
Thickness of the barrier	Name: Thickness of the barrier Location: x value of the scale of the brick wall	Low Value:0.02 High Value: 2

Control Parameters

List all other parameters and their FIXED values for all experiments:

Parameter	Fixed Value
Collision Detection Mode	Discrete
mass of the roller	1
drag of the roller	0
scaleof the roller	1

Experimental Procedure (Filling in the data collection table)

- Set Parameter 1 to its first test value
- Set Parameter 2 to its first test value

- Press Play and observe the ball
- Record whether tunneling occurred (Yes/No)
- Press the Reset button
- Repeat steps 3-5 for a total of 6 trials (1 run = 3 trials)
    - Calculate the tunneling frequency (e.g., "3/6" means tunneling observed in 3 out of 6 trials)
- Move to the next parameter combination
- Test all 4 combinations (2 values × 2 values)

Data Collection Table

Exp #	Direction	Thickness	Tunneling (X/6)	Comments
1	Low	Low	4/6	as per expected
2	Low	High	0/6	as per expected
3	High	Low	6/6	as per expected
4	High	High	4/6	as per expected

Total Trials Conducted: 24 (4 combinations × 6 trials each)
Total Runs Conducted: 8 (1 run = 3 trials)

STEP 5: Analysis and Discussion

Instructions: Collaboratively discuss and answer the following questions as a group. All group members should submit the same text.

1. Did the results match your initial predictions? Why or why not?

Yes making the wall thicker made the tunelling problem appear less and making the ball faster made the tunelling problem appear more

2. Which parameter had a stronger effect on tunneling? How do you know?

From this experiment since low high contrast cases were both 4/6 we can't determine

3. Describe any patterns you observed in the data:
   - At what threshold values did tunneling become consistent

(approaching 6/6)?

- At what values did tunneling become rare (approaching 0/6)?

- Were there any surprising or unexpected results?

When the wall was thin and roller speed was high we got 6/6
When the wall was thick and roller speed was low we got 0/6
These were what we expected

STEP 6: BONUS (15% Extra Credit)

Instructions: This step should be carried out individually. However, you are free to discuss and ask questions among group members.

Task:

Modify the BallBooster.cs script so that when a ball hits the first barrier, a visual action occurs.

Required Visual Action:
Change the size of the ball (scale it up or down)

Optional Additional Actions:
- Change the ball's color
- Add a particle effect
- Change the ball's material
- Add rotation
- Play a sound

Submission:

Paste your complete modified BallBooster.cs script below:

// Paste your modified script here

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BallBooster : MonoBehaviour
{
```

```
    [SerializeField]
    private Vector3 direction = new Vector3(-5, 0, 0);
    private Transform _ball;
    private Vector3 startPos;
    private Renderer _renderer;

    private Rigidbody rb;

    // Start is called before the first frame update
    void Start()
    {
        rb = GetComponent<Rigidbody>();
        _renderer = GetComponent<Renderer>();
        _ball = GetComponent<Transform>().transform;
        startPos = transform.position;
    }

    void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.name == "Brick Wall")
        {
            _renderer.material.color = Color.red;
            _ball.localScale = new Vector3(2, 2, 2);

        }
    }
    void FixedUpdate()
    {
        rb.AddForce(direction);
    }
}
```

Grading Rubric

Criteria	Points	Description
System Summary	5%	Complete and accurate system information
Parameter Identification	20%	4-6 relevant parameters with correct locations and expected effects
Experimental Design	20%	Proper value selection, controls identified, procedure followed
Data Collection	20%	Complete table, accurate counting, 24 total trials

Analysis & Discussion	25%	Insightful answers, identifies patterns, relates to physics theory
Presentation	10%	Clear formatting, proper grammar, organized responses
BONUS: Code Implementation	+5%	Working collision detection with visual feedback
TOTAL	100%	(+5% possible bonus)

Submission Requirements

What to Submit:

- This completed document (PDF or Word format)
  - All sections filled out
  - Data table complete with 4 experiments
  - Analysis section completed (same for all group members)
  - System summary filled in

- Modified Script (for bonus)
  - Paste the complete BallBooster.cs code in the space provided above

Good luck with your experiments!