Title: Vector Operations and Their Application in Unity Game Engine

EXPERIMENT NO: 1 **Name:** SHAKYA JAY JITENDRA
Date:20-7-2025 **Reg. No:** 24BCG10123

Aim:

To perform basic vector operations such as addition, subtraction, dot product, and cross product using Unity and understand their application in a game environment.

Description/Concept:

Vectors are fundamental in game development for representing direction, position, and force. Unity's Vector3 class allows developers to easily perform vector operations that control game mechanics like movement, collision, lighting, and AI behavior.

This experiment demonstrates how to implement and visualize:

1. **Vector Addition** – Combining directions/forces.
2. **Vector Subtraction** – Finding direction between two points.
3. **Dot Product** – Measuring angle/alignment.
4. **Cross Product** – Finding perpendicular vectors or surface normals.

Program/Coding:

csharp

```
using UnityEngine;


public class VectorOperations : MonoBehaviour

{

    public Transform cube1;

    public Transform cube2;

    public Transform resultCube;


    private Vector3 originalPosition1;

    private Vector3 originalPosition2;
```

```
void Start()
{
    if (cube1 != null) originalPosition1 = cube1.position;
    if (cube2 != null) originalPosition2 = cube2.position;
}

void Update()
{
    if (cube1 == null || cube2 == null || resultCube == null) return;

    Vector3 v1 = cube1.position;
    Vector3 v2 = cube2.position;

    // A - Vector Addition
    if (Input.GetKeyDown(KeyCode.A))
    {
        Vector3 addition = v1 + v2;
        Debug.Log("Addition: " + addition);
        resultCube.position = addition;
    }

    // S - Vector Subtraction
    if (Input.GetKeyDown(KeyCode.S))
    {
        Vector3 subtraction = v1 - v2;
        Debug.Log("Subtraction: " + subtraction);
        resultCube.position = subtraction;
    }

    // D - Dot Product
```

```
        if (Input.GetKeyDown(KeyCode.D))
        {
            float dot = Vector3.Dot(v1.normalized, v2.normalized);
            Debug.Log("Dot Product: " + dot);
        }

        // C - Cross Product
        if (Input.GetKeyDown(KeyCode.C))
        {
            Vector3 cross = Vector3.Cross(v1.normalized, v2.normalized);
            Debug.Log("Cross Product: " + cross);
            resultCube.position = cross * 5f; // scaling for visibility
        }

        // R - Reset
        if (Input.GetKeyDown(KeyCode.R))
        {
            cube1.position = originalPosition1;
            cube2.position = originalPosition2;
            resultCube.position = Vector3.zero;
            Debug.Log("Reset to original positions");
        }
    }
}
```

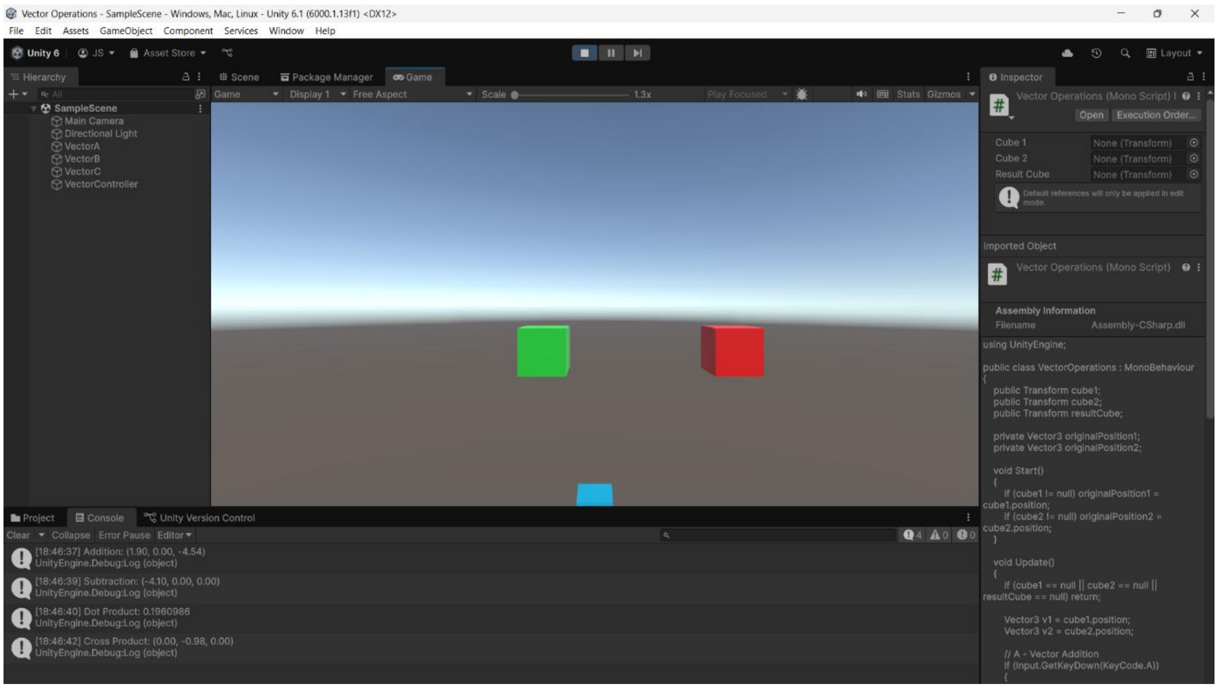
Analysis/Summary/Conclusion:

- **Addition** shows the combined effect of two vectors (useful for motion).
- **Subtraction** gives the direction from one object to another.
- **Dot Product** helps determine how aligned two directions are (used in lighting and AI).
- **Cross Product** returns a perpendicular vector (used in normals and rotation

axes).

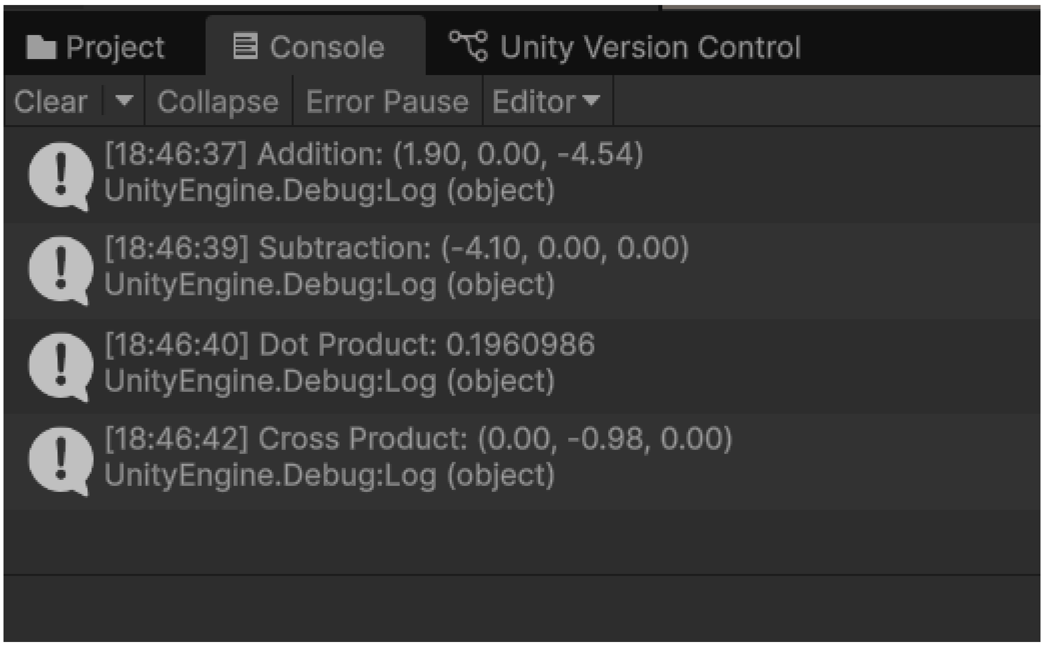
These operations are core to physics, movement, camera control, lighting, and AI behavior in Unity.

Output:



```
using UnityEngine;

public class VectorOperations : MonoBehaviour
{
    public Transform cube1;
    public Transform cube2;
    public Transform resultCube;

    private Vector3 originalPosition1;
    private Vector3 originalPosition2;

    void Start()
    {
        if (cube1 != null) originalPosition1 =
cube1.position;
        if (cube2 != null) originalPosition2 =
cube2.position;
    }

    void Update()
    {
        if (cube1 == null || cube2 == null ||
resultCube == null) return;

        Vector3 v1 = cube1.position;
        Vector3 v2 = cube2.position;

        // A - Vector Addition
        if (Input.GetKeyDown(KeyCode.A))
        {
```

```
[18:46:37] Addition: (1.90, 0.00, -4.54)
UnityEngine.Debug:Log (object)
[18:46:39] Subtraction: (-4.10, 0.00, 0.00)
UnityEngine.Debug:Log (object)
[18:46:40] Dot Product: 0.1960986
UnityEngine.Debug:Log (object)
[18:46:42] Cross Product: (0.00, -0.98, 0.00)
UnityEngine.Debug:Log (object)
```



```
[18:46:37] Addition: (1.90, 0.00, -4.54)
UnityEngine.Debug:Log (object)
[18:46:39] Subtraction: (-4.10, 0.00, 0.00)
UnityEngine.Debug:Log (object)
[18:46:40] Dot Product: 0.1960986
UnityEngine.Debug:Log (object)
[18:46:42] Cross Product: (0.00, -0.98, 0.00)
UnityEngine.Debug:Log (object)
```

Result:

Script was created for Vector operations and were successfully implemented and tested in Unity Game Engine. Their application in game mechanics was understood through coding and console outputs.
