Title: Linear motion of objects (multi-dimension) variable speed

EXPERIMENT NO: 2 **Name:** SHAKYA JAY JITENDRA

Date:11-8-2025 **Reg. No:** 24BCG10123

Aim: To write a C# program in Unity that performs linear motion of objects (multidimension) variable speed using Unity’s Game engine.

Description/Concept:

1. Linear Motion Linear motion (or rectilinear motion) is movement along a single straight line. In a 3D environment like Unity, this "line" can be oriented in any direction in space (e.g., horizontally, vertically, or at any diagonal angle).

2. Vectors (Vector3) In Unity, positions and directions in 3D space are represented using a data structure called Vector3. A Vector3 holds three floating-point numbers: x, y, and z, corresponding to the three spatial axes.

Position: transform.position is a Vector3 that stores the object's exact location in the world.

Direction: A Vector3 can also represent a direction. For example, (1, 0, 0) points along the positive X-axis. To ensure direction only represents orientation and not magnitude, we use a normalized vector (a vector with a length of 1).

3. Speed and Velocity

Speed: A scalar (a single number, like 5.0f) that represents how fast an object is moving.

Velocity: A vector that represents both the speed and the direction of motion. It's calculated by multiplying the direction vector by the speed: $Velocity = Direction \times Speed$.

4. Frame Rate Independence (Time.deltaTime) Computers run games at different frame rates (Frames Per Second or FPS). If we simply move an object by a fixed amount each frame, it will move faster on computers with higher FPS. To solve this, we use Time.deltaTime. This variable holds the time (in seconds) it took to complete the last frame.

By multiplying our movement by Time.deltaTime, we ensure the object moves a consistent distance over one second, regardless of the frame rate.

Program/Coding:

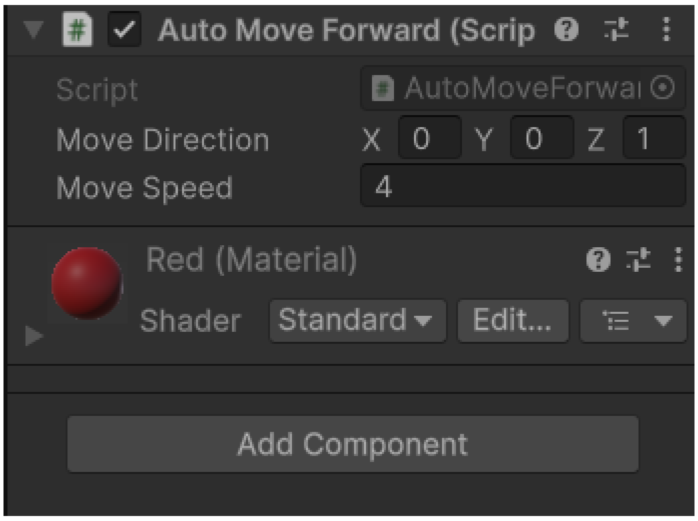
For AUTO CUBE:

```
using UnityEngine;
public class AutoMoveForward : MonoBehaviour

{

    public Vector3 moveDirection = Vector3.forward; // Default direction

    public float moveSpeed = 2f; // Adjustable speed


    void Update()

    {

        transform.Translate(moveDirection.normalized * moveSpeed *
     Time.deltaTime, Space.World);

    }

}
```

For NORMAL CUBE:

```
using UnityEngine;

using UnityEngine.InputSystem;


[RequireComponent(typeof(Rigidbody))]

public class PlayerMovement : MonoBehaviour

{

    public float moveSpeed = 5f; // Movement speed


    private Rigidbody rb;

    private Vector2 input;


    void Start()

    {

        rb = GetComponent<Rigidbody>();

        rb.freezeRotation = true; // Prevent physics rotation

    }
    void Update()
    {
        // Read WASD input directly (simple method)
        input = Vector2.zero;
        if (Keyboard.current.wKey.isPressed) input.y += 1;
        if (Keyboard.current.sKey.isPressed) input.y -= 1;
        if (Keyboard.current.aKey.isPressed) input.x -= 1;
        if (Keyboard.current.dKey.isPressed) input.x += 1;
        input = input.normalized;
    }

    void FixedUpdate()
    {
        // World-space movement
        Vector3 moveDir = new Vector3(input.x, 0f, input.y);
        Vector3 velocity = moveDir * moveSpeed;
        velocity.y = rb.linearVelocity.y; // Preserve vertical motion

        rb.linearVelocity = velocity;
    }
}
```

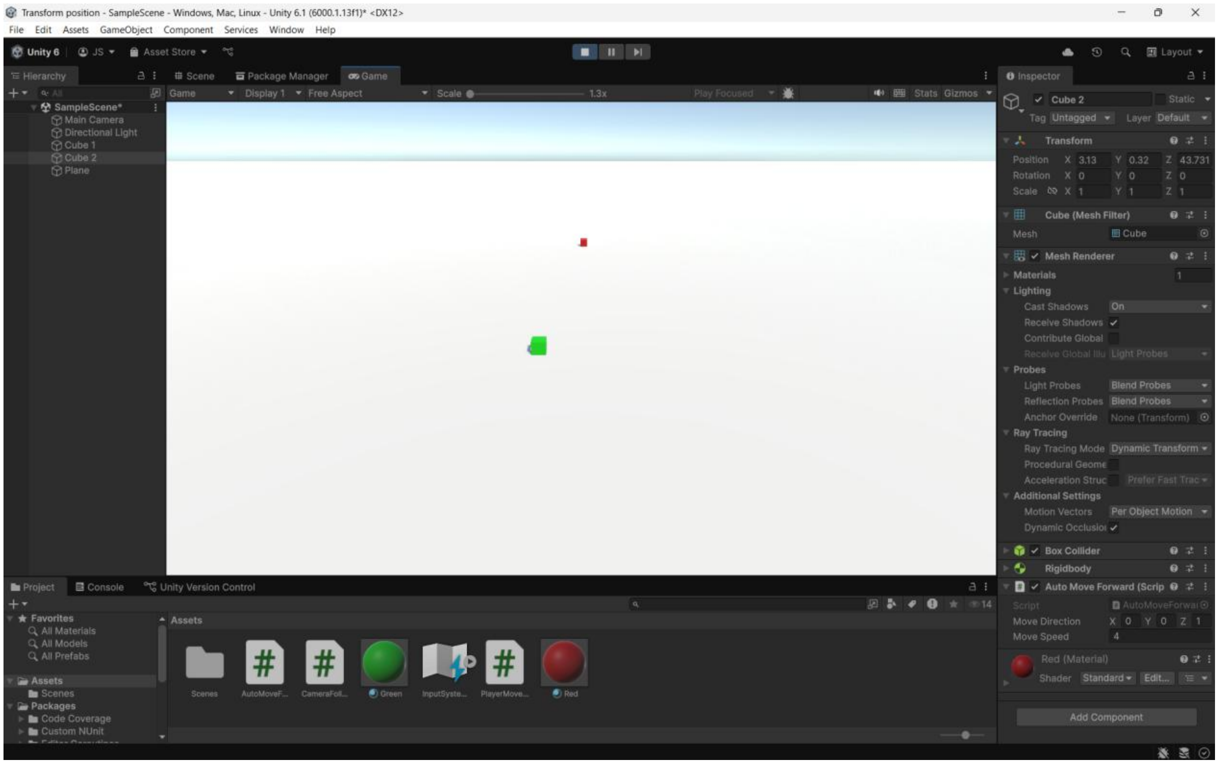
Output: When you press Play, the GameObject will move continuously in the direction defined by moveDirection.The key output is interactive: while the game is running, you can change the speed value in the Inspector. The object will instantly speed up, slow down, stop (if speed is 0), or reverse (if speed is negative), demonstrating variable-speed linear motion.



Auto Move Forward (Script)

Script: AutoMoveForward
Move Direction: X 0 Y 0 Z 1
Move Speed: 4

Red (Material)
Shader: Standard



Transform position - SampleScene - Windows, Mac, Linux - Unity 6.1 (6000.1.13f1)* <DX12>

Hierarchy: SampleScene* — Main Camera, Directional Light, Cube 1, Cube 2, Plane

Inspector: Cube 2 — Tag Untagged, Layer Default

Transform
Position X 3.13 Y 0.32 Z 43.731
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Cube (Mesh Filter) — Mesh: Cube

Mesh Renderer
Materials 1
Lighting: Cast Shadows On; Receive Shadows ✓; Receive Global Illu Light Probes
Probes: Light Probes Blend Probes; Reflection Probes Blend Probes; Anchor Override None (Transform)
Ray Tracing: Ray Tracing Mode Dynamic Transform; Acceleration Struc Prefer Fast Trac
Additional Settings: Motion Vectors Per Object Motion; Dynamic Occlusion ✓

Box Collider
Rigidbody

Auto Move Forward (Script)
Script: AutoMoveForward
Move Direction X 0 Y 0 Z 1
Move Speed 4

Red (Material) — Shader Standard

Conclusion:

This experiment successfully demonstrated programming multi-dimensional linear motion in Unity. By scripting changes to an object's transform.position using a Vector3 for direction, we achieved controlled movement.

The most critical takeaway is the use of Time.delta Time to ensure motion is smooth and independent of hardware performance. Making the speed variable public allowed for real-time adjustments in the Inspector, effectively creating variable speed. This technique is a fundamental building block for numerous game mechanics, from projectiles to moving platforms, proving its importance in game development

Result:

Script was created for Linear motion of objects (multi-dimension) variable speed and were successfully implemented and tested in Unity Game Engine 6.1 Their application in game mechanics was understood through coding and console outputs.