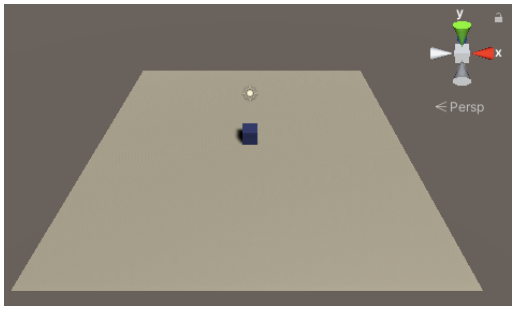
Unity
Lab 2

Steering Behaviour (Arrive)

The Player uses the Arrive script to move to the target position coordinates, which is set before the game starts.



Procedure

1. Start a new 3D project
2. In Hierarchy, rename the SampleScene as **Arrive**

Ground Plane object

1. Add a 3D Plane object to the scene, and rename it 'Ground'
2. Set the following parameters for the Cube's Transform component:
    a. Position: X=0, Y=−0.5, Z=0
    b. Rotation: X=0, Y=0, Z=0
    c. Scale: X=2, Y=1, Z=2
3. Set the following parameters for the Main Camera's Transform component:
    a. Position: X=0, Y=20, Z=−12
    b. Rotation: X=60, Y=0, Z=0
    c. Scale: X=1, Y=1, Z=1

Player object

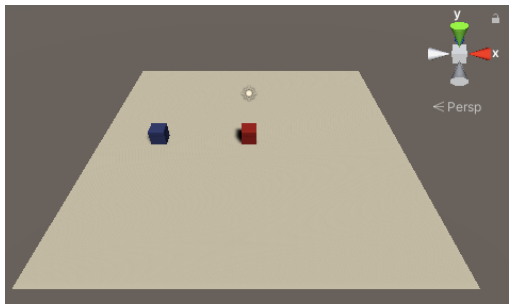
1. Add a 3D Cube object to the scene, and rename it 'Player'
2. Set the following parameters for the Player's Transform component:
    a. Position: X=0, Y=0, Z=0
    b. Rotation: X=0, Y=0, Z=0
    c. Scale: X=1, Y=1, Z=1
3. Create a new Material and rename it as 'PlayerMat'
    a. Select the Blue from the Color palette
    b. Drag the 'PlayerMat' to the Player's cube in the Scene
4. In the Inspector, select Add Component
    a. Select 'New script'
    b. Enter the script name 'Arrive'
    c. In Visual Studio, enter the source code Arrive.cs
5. In the Inspector, enter the following values in the Arrive (Script) component:
    a. Speed: 3
    b. Position: X=4, Y=0, Z=−3
6. Run the game by clicking on the Play button

Source Code (Arrive.cs)

```
// Arrive.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Arrive : MonoBehaviour
{
    public float speed = 3.0f;
    public float stopRadius = 0.2f;
    public float slowRadius = 5.0f;
    public Vector3 targetPosition = Vector3.zero;

    void Start()
    {

    }

    void Update()
    {
        float step = speed * Time.deltaTime;
        float distance = Vector3.Distance(transform.position, targetPosition);
        targetPosition.y = 0.0f;

        if (distance < stopRadius)
        {
            transform.position = Vector3.MoveTowards(transform.position, targetPosition, 0.0f);
        }
        else if (distance < slowRadius)
        {
            transform.position = Vector3.MoveTowards(transform.position, targetPosition,
                step * (distance - stopRadius) / (slowRadius - stopRadius));
        }
        else
        {
            transform.position = Vector3.MoveTowards(transform.position, targetPosition, step);
        }
    }
}
```

Steering Behaviour (Seek)

The Player (Blue) will seek out the static location of the Target (Red). The target uses the Arrive script, so its target position coordinates may be set before the game starts. The Player is completely automated, since it is programmed to chase after the Transform object (Target).



Procedure

1. In the Scenes folder of Assets, select the **Arrive** scene and duplicate it using CTRL-D
2. Rename the new scene as **Seek**

Player object

1. Add a 3D Cube object to the scene, and rename it 'Player'
2. Set the following parameters for the Cube's Transform component:
    a. Position: X=−6, Y=0, Z=0
    b. Rotation: X=0, Y=0, Z=0
    c. Scale: X=4, Y=4, Z=1
3. In the Inspector, remove the current script and select Add Component
    a. Select 'New script'
    b. Enter the script name 'Seek'
    c. In Visual Studio, enter the source code Seek.cs
4. In the Inspector, enter the following values in the Seek (Script) component:
    a. Target: Target (Transform)
    b. Speed: 1
    c. Future Time: 3

Target object

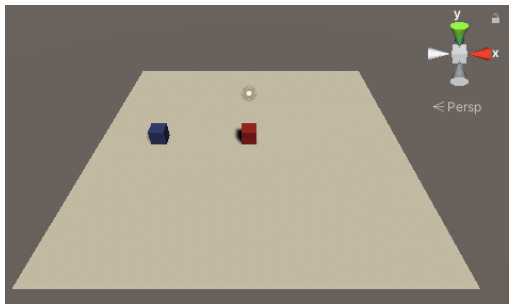
1. Add a 3D Cube object to the scene, and rename it 'Target'
2. Set the following parameters for the Cube's Transform component:
    a. Position: X=0, Y=0, Z=0
    b. Rotation: X=0, Y=0, Z=0
    c. Scale: X=1, Y=1, Z=1
3. Create a new Material and rename it as 'TargetMat'
    a. Select the Red from the Color palette
    b. Drag the 'TargetMat' to the Target's cube in the Scene
4. In the Inspector, select Add Component
    a. Select the script 'Arrive'
5. In the Inspector, enter the following values in the Arrive (Script) component:
    a. Speed: 3
    b. Position: X=6, Y=0, Z=−7
6. Run the game by clicking on the Play button

Source Code (Seek.cs)

```
// Seek.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Seek : MonoBehaviour
{
    public Transform target;

    public float speed = 3.0f;
    private Vector3 targetPosition;

    void Start()
    {

    }

    void Update()
    {
        float step = speed * Time.deltaTime;
        targetPosition = target.position;
        targetPosition.y = 0.0f;
        transform.position = Vector3.MoveTowards(transform.position, targetPosition, step);
    }
}
```

Steering Behaviour (Seek2)

In this game, the Target (Red) will use a modified version of the Arrive script, so its target position can be changed during gameplay with the mouse. The Player behavior remains unchanged.



Procedure

1. In the Scenes folder of Assets, select the **Seek** scene and duplicate it using CTRL-D
2. Rename the new scene as **Seek2**

Player object

1. No changes required

Target object

1. In the Inspector, remove the current script and select Add Component
    a. Select 'New script'
    b. Enter the script name 'Arrive2'
    c. In Visual Studio, enter the source code Arrive2.cs
2. In the Inspector, enter the following values in the Arrive2 (Script) component:
    a. Speed: 3
3. Run the game by clicking on the Play button

Source Code (Arrive2.cs)

```
// Arrive2.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Arrive2 : MonoBehaviour
{
    public float speed = 10f;
    public float stopRadius = 0.2f;
    public float slowRadius = 5.0f;
    private Vector3 targetPosition;
    private Vector3 position = Vector3.zero;

    void Start()
    {

    }


    void Update()
    {
        float step = speed * Time.deltaTime;
        float distance = Vector3.Distance(transform.position, targetPosition);
        RaycastHit raycastHitData;
        Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);

        if (Physics.Raycast(ray, out raycastHitData, 1000f))
        {
            position = raycastHitData.point;
        }

        if (Input.GetMouseButtonDown(0))
        {
            targetPosition = position;
            //Debug.Log(position);
        }

        targetPosition.y = 0.0f;

        if (distance < stopRadius)
        {
            transform.position = Vector3.MoveTowards(transform.position, targetPosition, 0.0f);
        }
        else if (distance < slowRadius)
        {
            transform.position = Vector3.MoveTowards(transform.position, targetPosition,
                step * (distance - stopRadius) / (slowRadius - stopRadius));
        }
        else
        {
            transform.position = Vector3.MoveTowards(transform.position, targetPosition, step);
        }
    }
}
```

END OF LAB EXERCISE