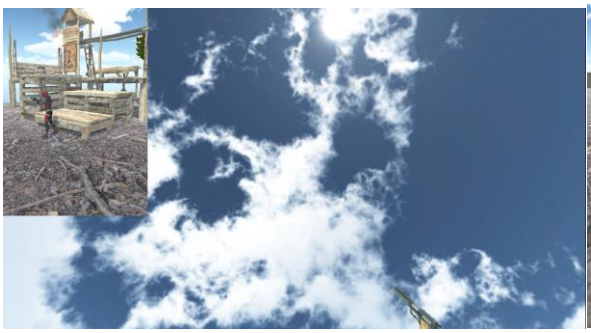
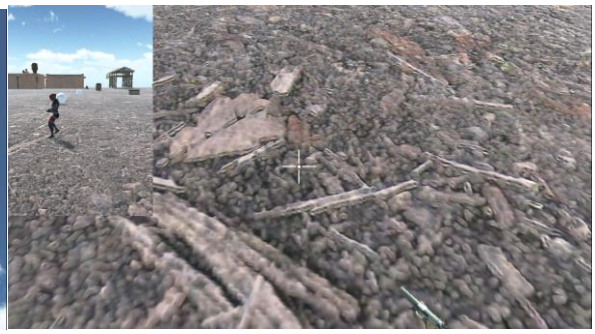
Green Salar
salar.rezayani@gmail.com

First person shooter game

1) Basic camera control for movement and look.

a. mouse look – able to look around

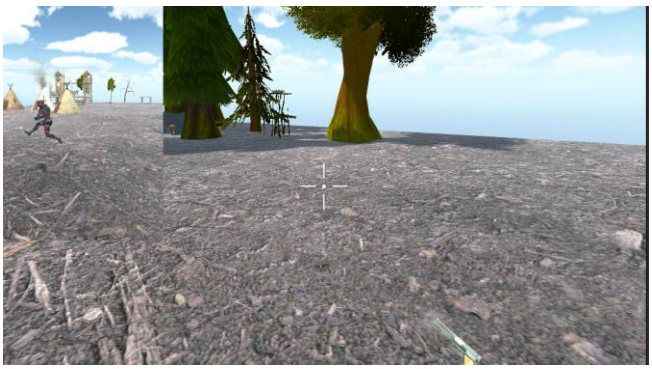




```
void Start()
{
    Cursor.lockState = CursorLockMode.Locked;
}
void Update()
{
  float mouseX = Input.GetAxis("Mouse X") * mouseSensitivity * Time.deltaTime;
  float mouseY = Input.GetAxis("Mouse Y") * mouseSensitivity * Time.deltaTime;
  playerBody.Rotate(Vector3.up*mouseX);

  xRotation -= mouseY;
  xRotation = Mathf.Clamp(xRotation,-90,90);
  transform.localRotation=Quaternion.Euler(xRotation + headRotationX,0f,0f);

  float headRotationX = headTracker.position.y*10;
```

b. WASD keyboard move – able to move in XZ plane using keypad

c. Spacebar jump (with gravity) –



Green Salar
salar.rezayani@gmail.com

```
void Update()
{

    if(Input.GetKey("up")||Input.GetKey("w")){
      forward = true;
      backward = false;
    }
    if(Input.GetKey("s")||Input.GetKey("down")){
      forward = false;
      backward = true;
    }

    isGrounded = Physics.CheckSphere(groundCheck.position, groundDistance, groundMask);
    if(isGrounded){
      airTime = 0;
    } else{
      airTime = airTime + Time.deltaTime;
    }
    if(airTime > 1){
      isGrounded = true;
    }

    if(isGrounded && velocity.y<0){
      velocity.y = -2;
    }
    float x = Input.GetAxis("Horizontal");
    float z = Input.GetAxis("Vertical"); // vertical in out cordination is forward!
    Vector3 move = transform.right * x + transform.forward * z ;// trasform correspond to the transform of the player object componnent
    controller.Move(move/10);
    anim.SetBool("isGrounded",isGrounded);
    anim.SetFloat("speed",move.magnitude);
    anim.SetBool("forward",forward);
    anim.SetBool("backward",backward);

    velocity.y += gravity * 1.8f * Time.deltaTime;
    controller.Move(velocity*Time.deltaTime);
    if(Input.GetButtonDown("Jump") && isGrounded){
      anim.SetBool("isGrounded",false);
      velocity.y = Mathf.Sqrt(jumpHeight * -2 *gravity);

    }
```

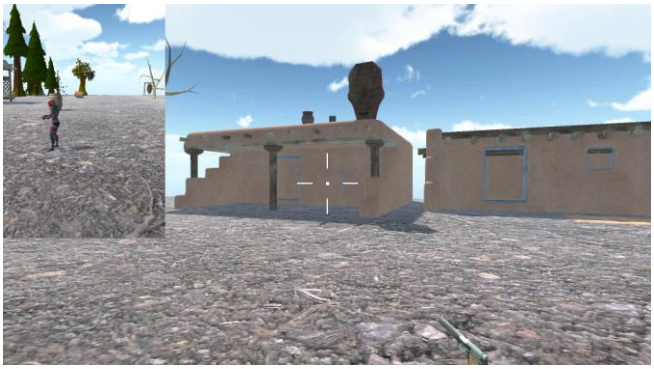
(if air-time exceed 1 second, it means it is on an object)

Crosshair

→ by Using a Canvas and UI elements + Anchors for changing the size of crosshair

a. visible at all time (does not get occluded by objects)

b. cross-shaped

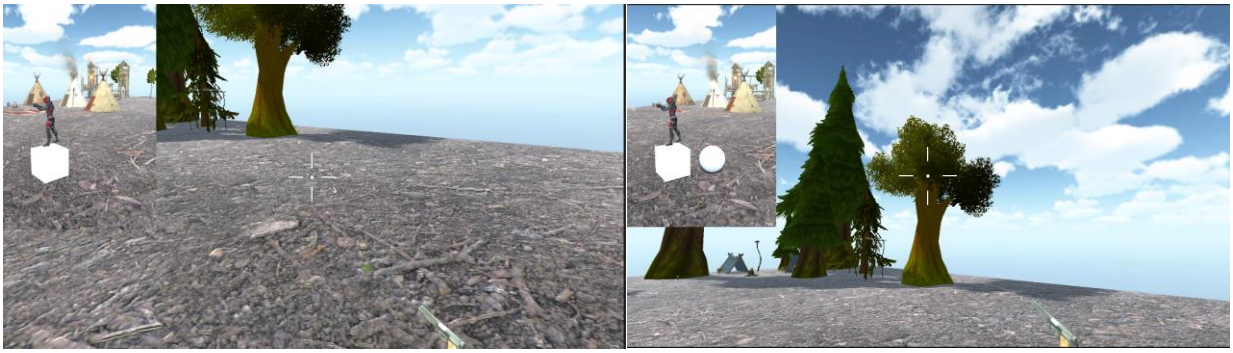


Green Salar
salar.rezayani@gmail.com

c. changes color when over enemy → turn to red (White Objects enemy + a static barrel for shouting)

```
middleCrosshair.GetComponent<RawImage> ().texture = null;
leftCrosshair.GetComponent<RawImage> ().texture   = null;
rightCrosshair.GetComponent<RawImage> ().texture  = null;
topCrosshair.GetComponent<RawImage> ().texture    = null;
bottomCrosshair.GetComponent<RawImage> ().texture = null;

RaycastHit hit;
if(Physics.Raycast(raycastExitPiont.position,raycastExitPiont.forward,out hit,200))
  {
//    Debug.Log(hit.transform.name);
    if(hit.transform.name=="enemy"){
        middleCrosshair.GetComponent<RawImage> ().texture = redTexture;
        leftCrosshair.GetComponent<RawImage> ().texture   = redTexture;
        rightCrosshair.GetComponent<RawImage> ().texture  = redTexture;
        topCrosshair.GetComponent<RawImage> ().texture    = redTexture;
        bottomCrosshair.GetComponent<RawImage> ().texture = redTexture;
    }
    }
            crosshair.GetComponent<Crosshair>().size= (hit.distance*2+55);
```

d. changes of crosshair by distance → by using RaycastHit . distance



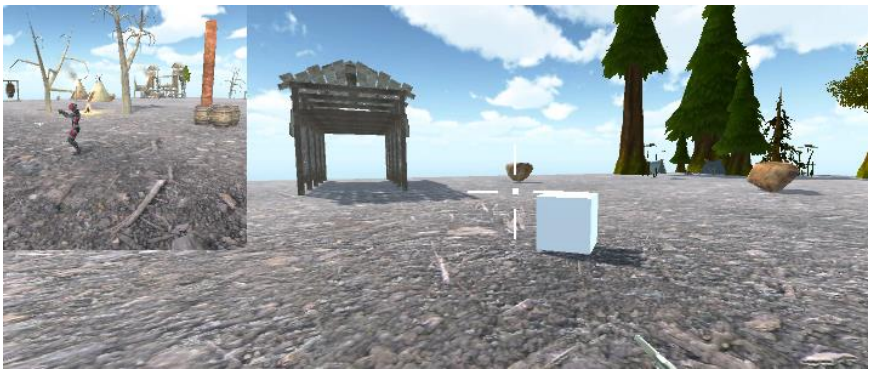
Green Salar
salar.rezayani@gmail.com

```
  private void Start(){

    rect = GetComponent<RectTransform>();

  }

  private void Update(){
      currentSize = Mathf.Lerp(currentSize, size, Time.deltaTime * speed);
    rect.SetSizeWithCurrentAnchors(RectTransform.Axis.Horizontal, currentSize);
     rect.SetSizeWithCurrentAnchors(RectTransform.Axis.Vertical, currentSize);
    }
}
```

- "size" comes from mouselook.cs (last code picture)– RaycastHit.distance

3) Single level with gameplay (must be able to shoot enemies in some type of environment)

a. appearance b. enemy behavior → Enemy follows the player.



c. weapon and effects:

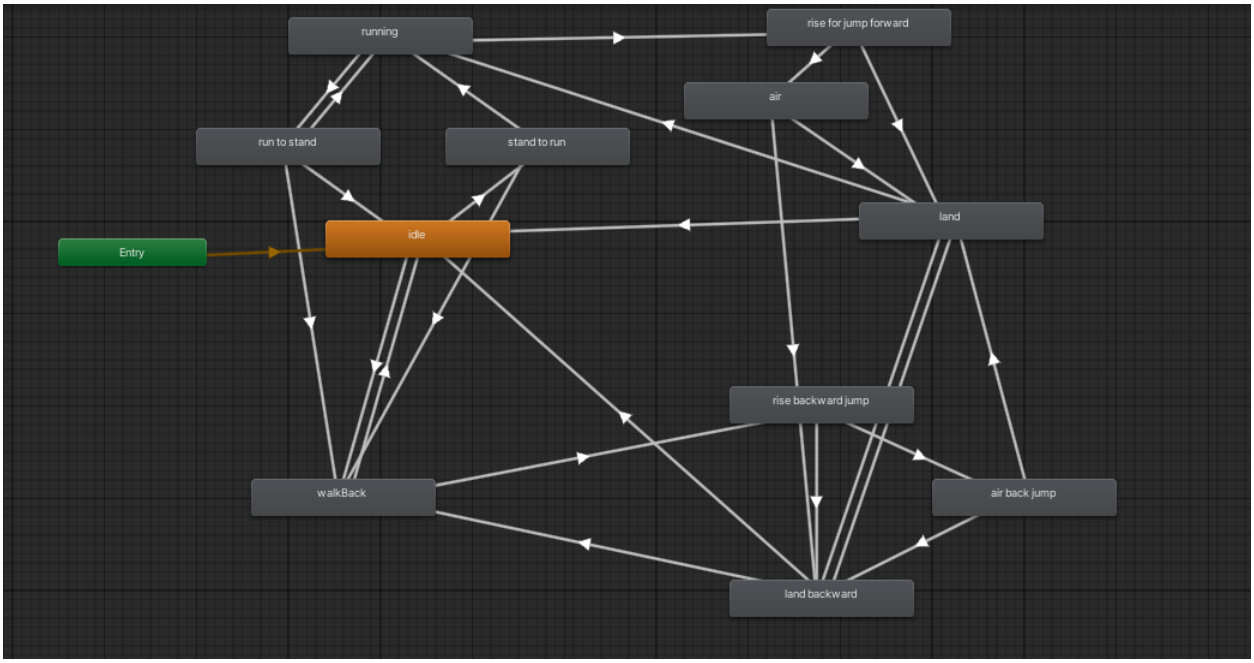
Animations: jump forward, backward, midair, landing, resting to run forward and back,

Sounds: environment music (by myself), steps, jump, land, shooting.

Map: used texture with Hieghtmap

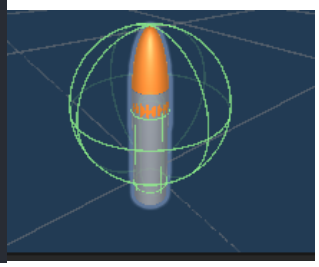
Green Salar
salar.rezayani@gmail.com

- Animation Diagram:



Bullet code:(Bullet prefab instantiates at time of shooting and then will be destroyed after 2 sec)

```
public class Gun : MonoBehaviour
{
      public Transform exitPoint;
      public AudioSource gunShot;
      public GameObject bullet;
      private float timeElapsed = 0;

      void Update()
      {
        if (Input.GetMouseButton(0)){
          GameObject tempBullet;
          if( timeElapsed > 100){
            tempBullet = Instantiate( bullet, exitPoint.position, exitPoint.rotation) as GameObject;
            tempBullet.GetComponent<Rigidbody>().velocity = transform.up*80;
            timeElapsed = 0;
            gunShot.Play();
            Destroy(tempBullet,2.0f);
          }

        }
            timeElapsed += Time.deltaTime * 1000;
      }

}
```



Thank You for Your Time!

--FINISH!--

Green Salar
salar.rezayani@gmail.com