

How to survive in a barren wasteland in a supernatural setting: the game

How to survive in a barren wasteland in a supernatural setting: the game

COSC 729

Group Members: Andrew Benya and Aishwarya Reehl

Faculty: Dr. Sharad Sharma

Bowie State University

How to survive in a barren wasteland in a supernatural setting: the game

Overview

The goal of our project is to create a zombie game where the player has to escape the zombie attack and try to reach the helicopter to win.

Health bar is used to track if the player has been attacked by the zombie, if so then the health bar reduces.

In case the player gets stuck he can press T to get evacuated.

This paper contains the details about how the environment is defined and the technologies used in creating this game. As the game designed involves a lot of hypothetical situations, virtual reality is the best option to design it. This game is extremely fun to play and is definitely a stress buster. The purpose of this game is to enjoy the game and have fun.

Environment

The environment resembles a haunted place. The environment is mostly wilderness and it includes mountains, trees, sea and dust storms to make it look more like a haunted place. The abandoned houses have been added to create a feel in the game. Terrain was “painted” onto the plane with a preview package. This package allows us to use painting tools to elevate and depress the ground, apply textures with discretion, and paint any prefab tree available in the scene.

Vision



Figure 1 Environment showing the dry lake



Figure 2 Terrain with water, haunted house and zombies

How to survive in a barren wasteland in a supernatural setting: the game



Figure 3 Terrain with billboard, house and zombies



Figure 4 Scary trees in the terrain

For sky we added the night sky with stars to create the perfect environment. A scary music is added throughout the game and helicopter hovering sounds for evacuation.

Inputs

Player can use keyboard to control the navigation. The player can use A to turn left and D to turn right or the Left Arrow and Right Arrow keys. The player can use W to go up and S to go down or the Up Arrow and Down Arrow keys.

Sensors

Several proximity sensors have been used to detect if the player is going near the zombie. There is time sensor for evacuation and touch sensor for radio.

How to survive in a barren wasteland in a supernatural setting: the game



Figure 5 Zombie proximity sensor

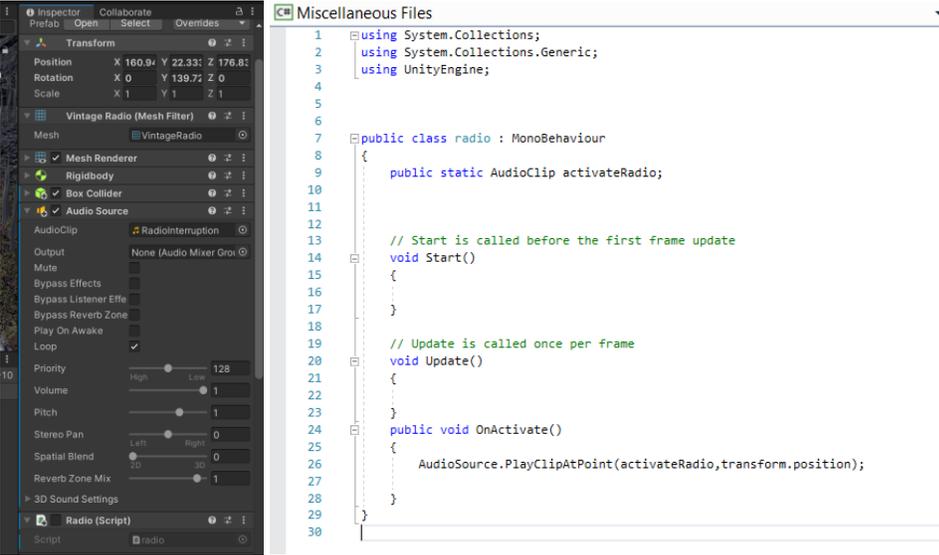


Figure 6 Radio sensor and script

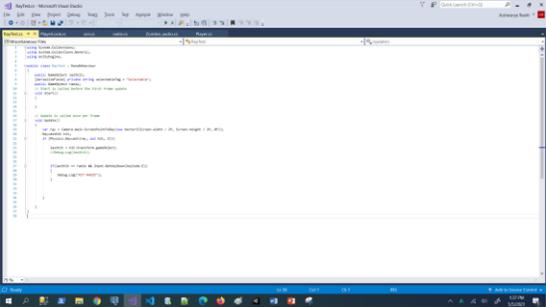


Figure 7 Hit radio script

How to survive in a barren wasteland in a supernatural setting: the game

Animation and scripting

The zombies are animated using custom and predefined scripts. Script to make the zombie look at the player as shown in Figure 8.

```
using UnityEngine;
using UnityEngine.AI;

public class LookAtPlayer : MonoBehaviour
{
    private NavMeshAgent agent;
    private Transform player;

    void Start()
    {
        agent = GetComponent<NavMeshAgent>();
        player = GameObject.FindGameObjectWithTag("Player").transform;
    }

    void Update()
    {
        agent.SetDestination(player.position);
    }
}
```

Figure 8 Script to make zombie look at the player

Zombie fall animation at the start of the game:

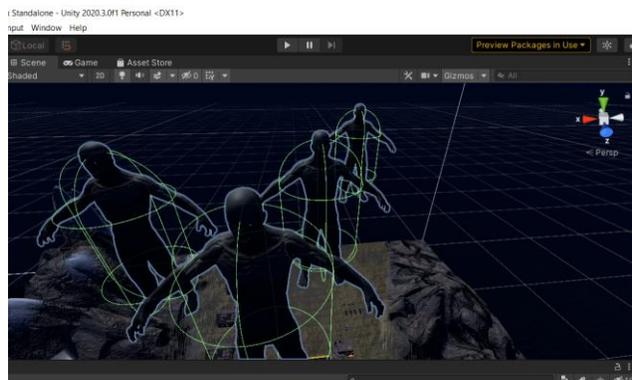


Figure 9 Zombies falling at the start of the game

Helicopter navigation using animation:

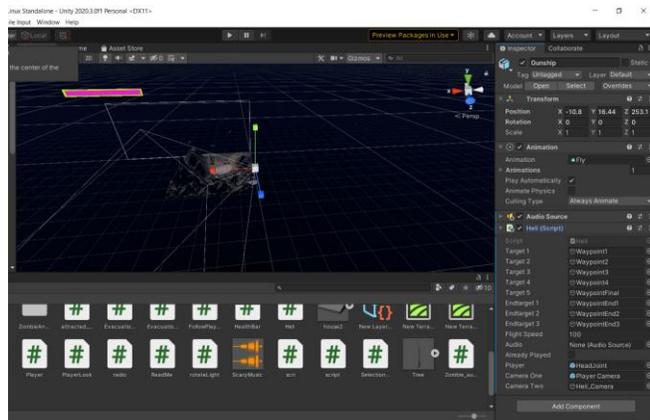


Figure 10 Helicopter navigation animation

Zombie walking animation is as shown in Figure 11:

How to survive in a barren wasteland in a supernatural setting: the game

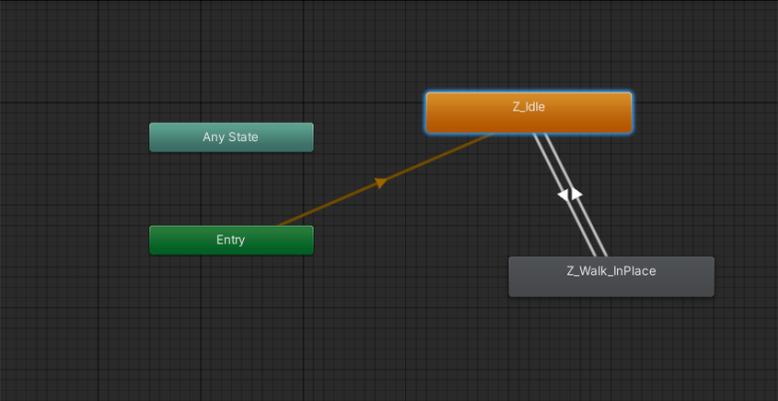


Figure 11 Zombie animation to walk

Zombie various other animations are as shown in Figure 12:

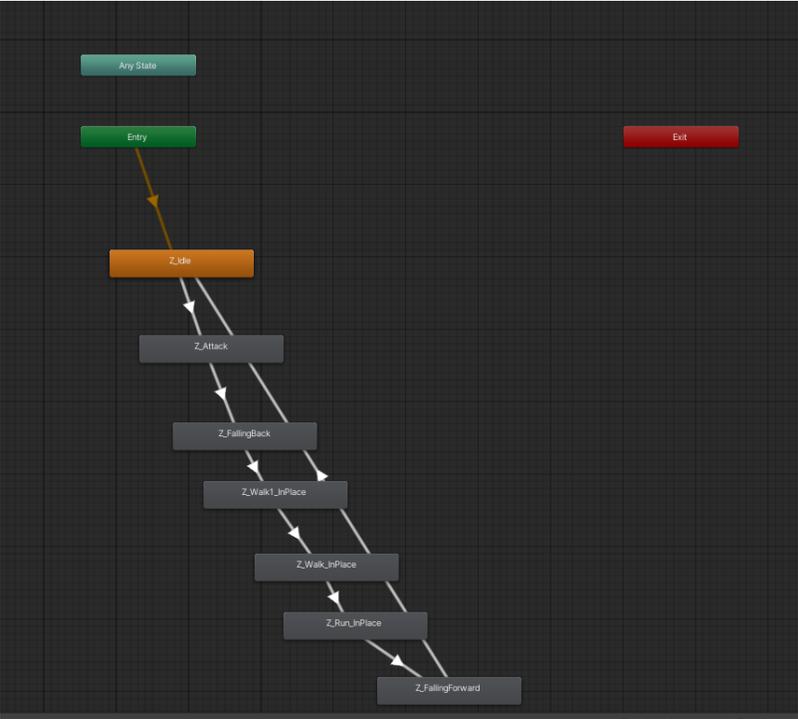


Figure 12 Zombie various animation

Health bar script:

How to survive in a barren wasteland in a supernatural setting: the game

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5
6 public class Player : MonoBehaviour
7 {
8     public int maxHealth = 1000;
9     public int currentHealth;
10
11     public HealthBar healthBar;
12     // Start is called before the first frame update
13     void Start()
14     {
15         currentHealth = maxHealth;
16         healthBar.SetMaxHealth(maxHealth);
17     }
18
19     // Update is called once per frame
20     public void Update()
21     {
22         //TakeDamage(20);
23         if(currentHealth <= 0)
24         {
25             SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
26         }
27     }
28
29     public void TakeDamage(int damage)
30     {
31         currentHealth -= damage;
32
33         healthBar.SetHealth(currentHealth);
34     }
35 }
```

Figure 13 Health bar manipulation script

Rotate light script to follow the player:

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class rotateLight : MonoBehaviour
6 {
7     public float rotateSpeed = 2f;
8     Transform transform;
9     // Start is called before the first frame update
10    void Start()
11    {
12        transform = GetComponent<Transform>();
13    }
14
15    // Update is called once per frame
16    void Update()
17    {
18        transform.Rotate(Vector3.down * Time.deltaTime * 100);
19    }
20 }
21
```

Figure 14 Rotate light script

Zombie audio script that is triggered with proximity:

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class Zombie_audio : MonoBehaviour
6 {
7     Transform playerTransform;
8     AudioSource audioSource;
9     public GameObject player;
10    //bool sound_play;
11    //bool toggle;
12    // Start is called before the first frame update
13    void Start()
14    {
15        audioSource = GetComponent<AudioSource>();
16        //sound_play = false;
17    }
18
19    // Update is called once per frame
20    void Update()
21    {
22        playerTransform = player.transform;
23        var distance = Vector3.Distance(playerTransform.position, transform.position);
24        if(distance <= 10.0f)
25            audioSource.Play();
26        else
27        {
28            audioSource.Stop();
29        }
30    }
31 }
--
```

Figure 15 Zombie audio script

Script to control the zombie's movement thc6:

How to survive in a barren wasteland in a supernatural setting: the game

```
using UnityEngine;
using System.Collections;

public class MZCtrl : MonoBehaviour {

    private Animator anim;
    private CharacterController controller;
    private int battle_state = 0;
    public float r_speed = 0.0f;
    public float runSpeed = 1.0f;
    public float turnSpeed = 10.0f;
    public float gravity = 20.0f;
    private Vector3 moveDirection = Vector3.zero;
    private float w_sp = 0.0f;
    private float r_sp = 0.0f;

    // Use this for initialization
    void Start () {
        anim = GetComponent<Animator>();
        controller = GetComponent<CharacterController> ();
        w_sp = speed; //read walk speed
        r_sp = runSpeed; //read run speed
        battle_state = 0;
        runSpeed = 1;
    }

    // Update is called once per frame
    void Update () {
        if (Input.GetKey ("1")) // turn to still state
        {
            anim.SetInteger ("battle", 0);
            battle_state = 0;
            runSpeed = 1;
        }
        if (Input.GetKey ("2")) // turn to battle state
        {
            anim.SetInteger ("battle", 1);
            battle_state = 1;
            runSpeed = r_sp;
        }
        if (Input.GetKey ("w"))
        {
            anim.SetInteger ("moving", 1); //walk/run/moving
        }
        else
        {
            anim.SetInteger ("moving", 0);
        }
        if (Input.GetKey ("down")) //walkback
        {
            anim.SetInteger ("moving", 12);
            runSpeed = 1;
        }
        if (Input.GetKeyUp ("down"))
        {
            if (battle_state == 0) runSpeed = 1;
            else if (battle_state >0) runSpeed = r_sp;
        }
        if (Input.GetMouseButtonDown (0)) { // attack1
            anim.SetInteger ("moving", 2);
        }
        if (Input.GetMouseButtonDown (1)) { // attack2
            anim.SetInteger ("moving", 3);
        }
        if (Input.GetMouseButtonDown (2)) { // attack3
            anim.SetInteger ("moving", 4);
        }
        if (Input.GetKeyDown ("space")) { //jump
            anim.SetInteger ("moving", 6);
        }
        if (Input.GetKeyDown ("c")) { //roar/howl
            anim.SetInteger ("moving", 7);
        }
        if (Input.GetKeyDown ("y")) { //powerhit
            anim.SetInteger ("battle", 1);
            battle_state = 1;
            runSpeed = r_sp;
            anim.SetInteger ("moving", 5);
        }
        if (Input.GetKeyDown ("u")) //hit
        {
            battle_state = 1;
            runSpeed = r_sp;
            anim.SetInteger ("battle", 1);
            int n = Random.Range (0, 2);
            if (n == 1)
            {
                anim.SetInteger ("moving", 8);
            }
            else
            {
                anim.SetInteger ("moving", 9);
            }
        }
        if (Input.GetKeyDown ("k")) { //rising
            anim.SetInteger ("battle", 1);
            battle_state = 1;
            runSpeed = r_sp;
            anim.SetInteger ("moving", 15);
        }
        if (Input.GetKeyDown ("i")) anim.SetInteger ("moving", 13); //die/fall
        if (Input.GetKeyDown ("o")) anim.SetInteger ("moving", 14); //die2
    }

    if (controller.isGrounded)
    {
        moveDirection = transform.forward * Input.GetAxis ("Vertical") * speed * runSpeed;
        float turn = Input.GetAxis ("Horizontal");
        transform.Rotate (0, turn * turnSpeed * Time.deltaTime, 0);
    }
    moveDirection.y -= gravity * Time.deltaTime;
    controller.Move (moveDirection * Time.deltaTime);
}
```

Figure 16 Script to animate and control the zombie

Script to make zombies look and follow the player as shown in Figure 17:

How to survive in a barren wasteland in a supernatural setting: the game

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class MonoBehaviour : MonoBehaviour
6 {
7
8     public MonoBehaviour player;
9     private CharacterController controller;
10    public Vector3 characterVelocity;
11    private float groundCheckRadius;
12    private float characterSpeed = 3.0f;
13    private float gravityScale = -0.25f;
14
15    public float attackDistance;
16    public float bufferDistance;
17
18    // Start is called before the first frame update
19    void Start()
20    {
21        controller = GetComponent<CharacterController>();
22    }
23
24    // Update is called once per frame
25    void Update()
26    {
27        MonoBehaviour character = controller.GetComponent<CharacterController>();
28        if (groundCheckRadius >> characterVelocity.y < 0)
29        {
30            characterVelocity.y = 0;
31        }
32
33        Vector3 distance = Vector3.Distance(player.transform.position, transform.position);
34        if (distance <= attackDistance)
35        {
36            if (distance >= bufferDistance)
37            {
38                transform.LookAt(new Vector3(player.transform.position.x, player.transform.position.y, 0f, player.transform.position.z));
39                transform.position = Vector3.MoveTowards(transform.position, player.transform.position, characterSpeed * Time.deltaTime);
40                //transform.position += transform.forward * characterSpeed * Time.deltaTime;
41            }
42        }
43    }
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Figure 17 Script to make zombies look and follow

Interactivity

Evacuation can be triggered in two different ways. When the player interacts with the radio in the scene, a helicopter will come to rescue the player. The radio interactivity is caused by detecting 3 things: Object is 2 meters away from the center of the player’s camera, the player has pressed E, & the object in question has the label “SelectableTag”. Having all three of these conditions satisfied initiates evacuation.

This is the major focus of the game. Escape the area by summoning the helicopter with the radio.

As a developer tool, we have left in pressing T to start evacuation.

Developer implementation shown below. We have implemented this as shown in Figure 17, 18 and 19

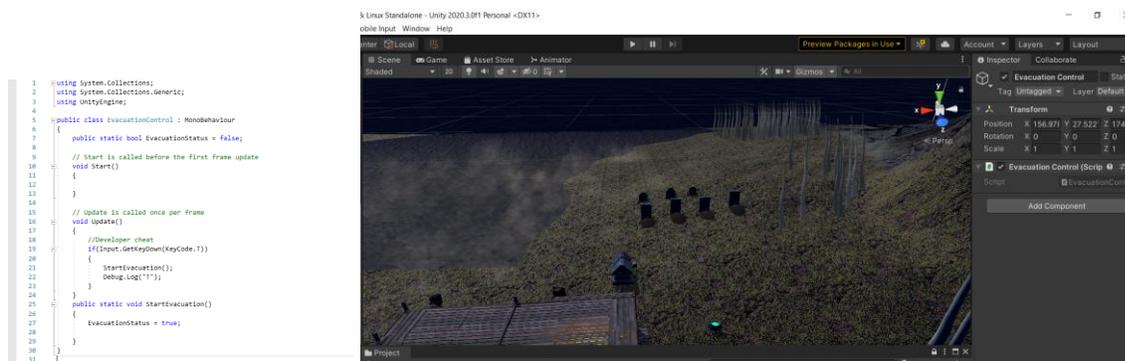


Figure 18 Evacuation script

Helicopter evacuation:

How to survive in a barren wasteland in a supernatural setting: the game

The helicopter speed is a major point of difficulty in this game. Setting the helicopter speed slower dramatically increases the difficulty of the game, since the zombies are all alerted and are all persuing the player. Upon reaching the helicopter, the ending “cut scene” will begin where the camera is fixed to the helicopter and the player can watch from the helicopter as they escape.

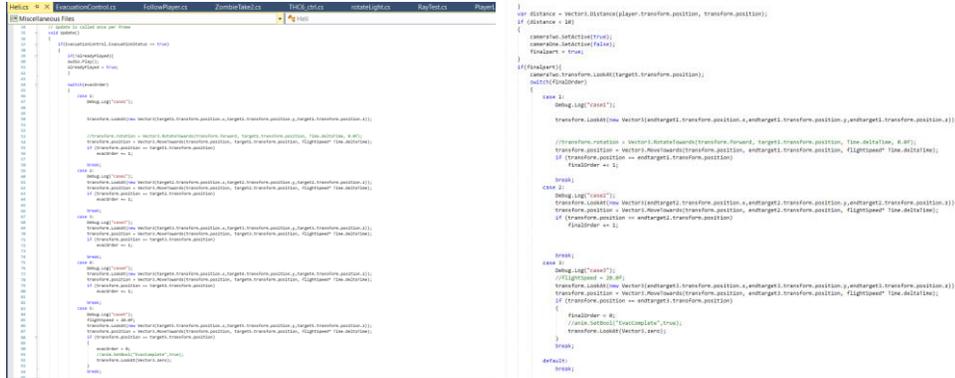


Figure 19 Helicopter animation/navigation

Avatars

Various zombie avatars are used to create this game where zombies would follow the player to create a scary effect like shown in Figure 20



Figure 20 Zombie Avatars

Details of the Virtual Environment

To create the perfect zombie land we added various UI elements. All UI elements used are shown below:

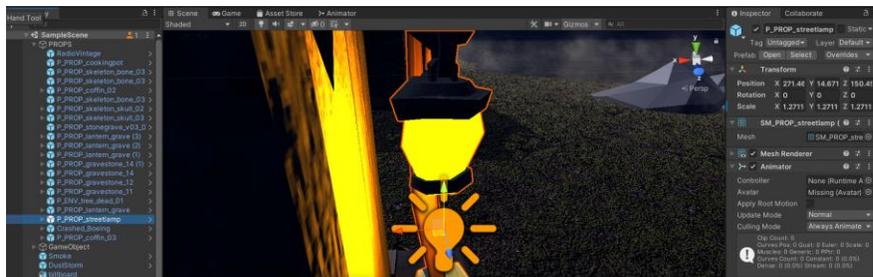


Figure 21 Street lights

How to survive in a barren wasteland in a supernatural setting: the game



Figure 22 Haunted house



Figure 23 Pier House

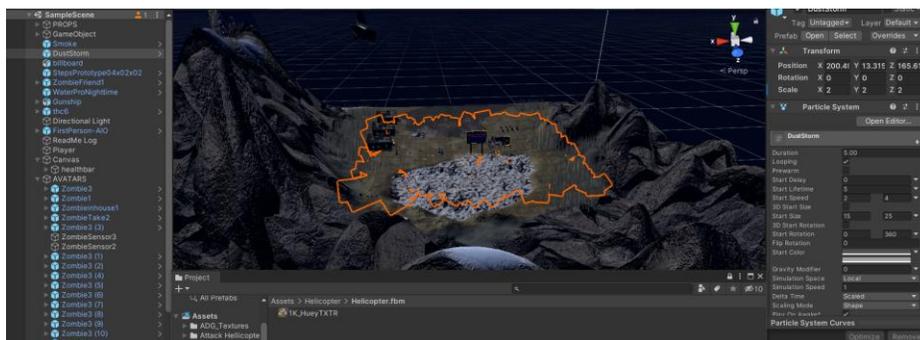


Figure 24 Dust storm to create the fog effect



Figure 25 Billboard with evacuation details

How to survive in a barren wasteland in a supernatural setting: the game

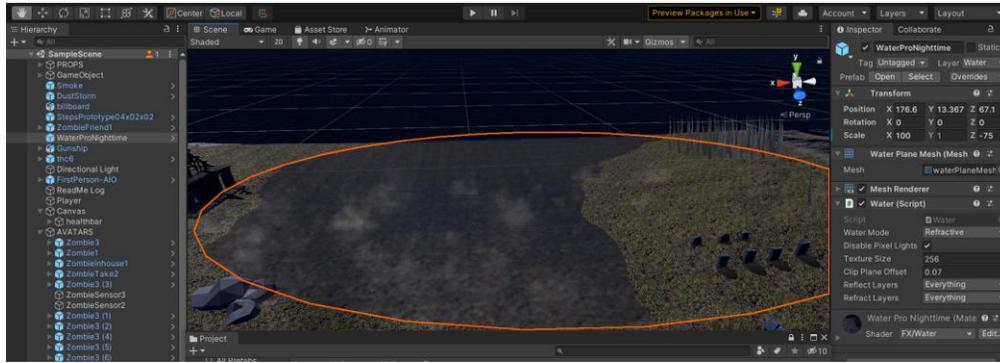


Figure 26 Water effect



Figure 27 Gravestones and crashed Boeing

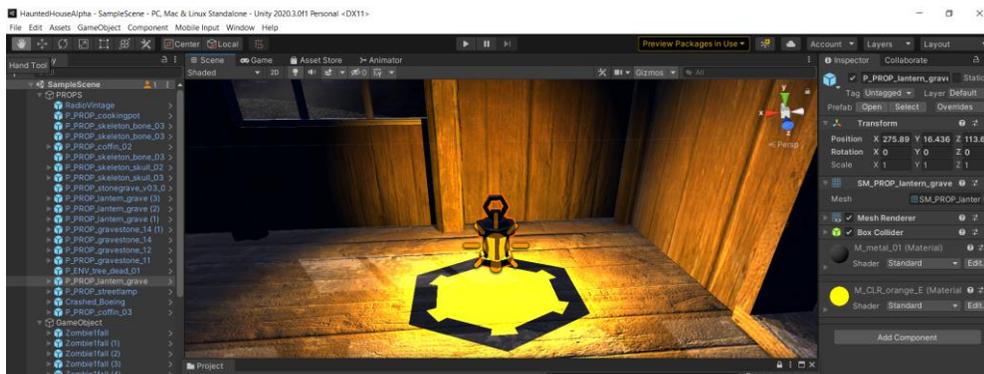


Figure 28 Lantern

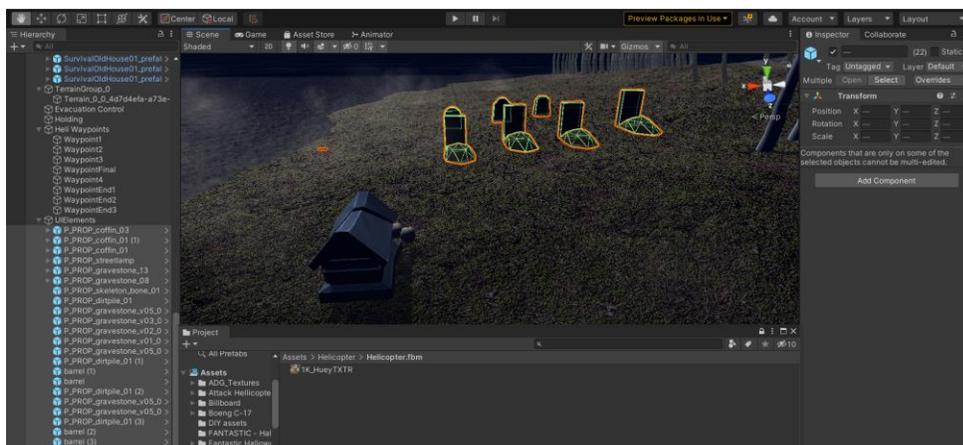


Figure 29 Dirt pile, grave stones and coffins

How to survive in a barren wasteland in a supernatural setting: the game

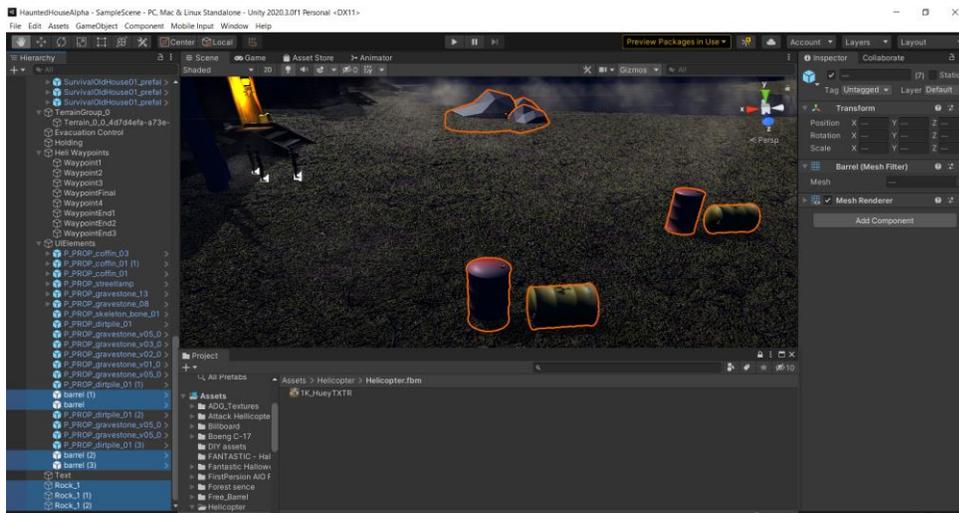


Figure 30 Barrels and rocks

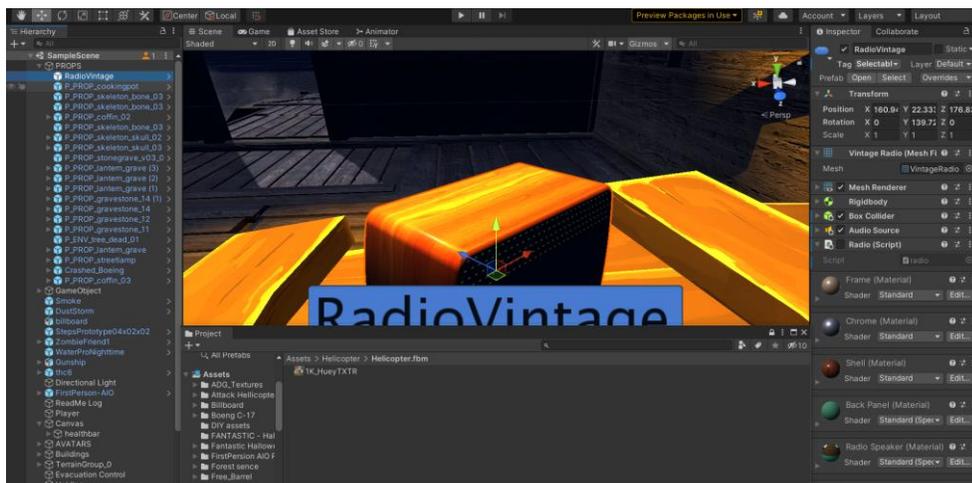


Figure 31 Vintage Radio

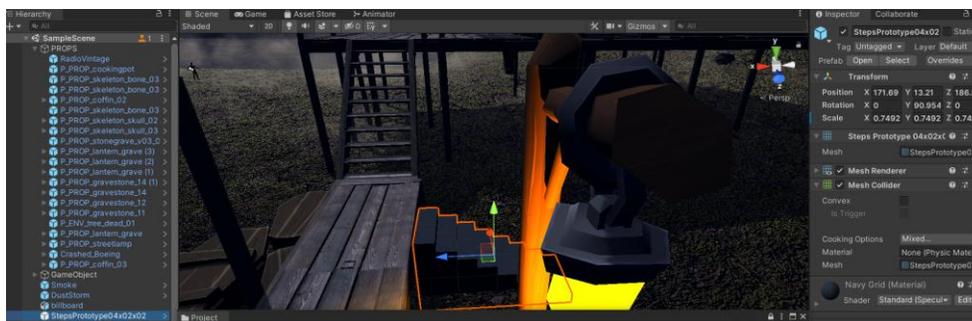


Figure 32 Steps for the pier house

How to survive in a barren wasteland in a supernatural setting: the game

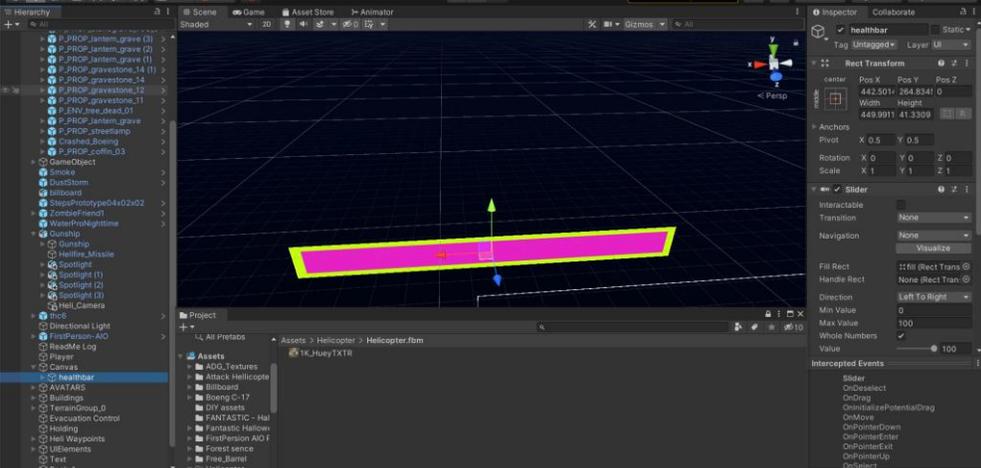


Figure 33 Health Bar

Audio file used:

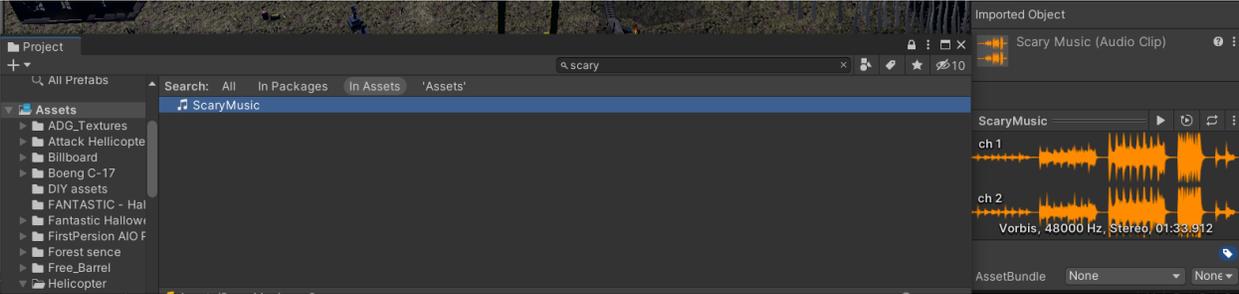


Figure 34 Scary music

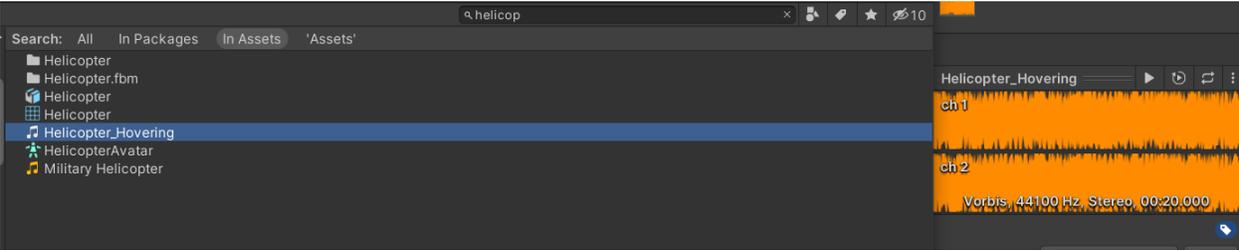


Figure 35 Helicopter music

Further Improvements

We can add how many zombies are tracking the player and also a remaining and elapsed time to create the sense of urgency

Software and Hardware Used

- Windows 10
- 16 bit RAM, 512 GB RAM
- Unity 2020.3.0f1

How to survive in a barren wasteland in a supernatural setting: the game

- C#
- VS 2019

Demo

<https://www.youtube.com/watch?v=-SkJBfldQM4>