ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Σχεδίαση Παιχνιδιού Εικονικής Πραγματικότητας Χρησιμοποιώντας Τεχνολογίες Oculus Rift και Leap Motion

Σταυρουλάκης Αλέξιος

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ

Αναπ. Καθ. Μανιά Αικατερίνη (Επιβλέπουσα)

Αναπ. Καθ. Δεληγιαννάκης Αντώνιος

Αναπ. Καθ. Ουγγρίνης Κωνσταντίνος-Αλκέτας

Χανιά, Φεβρουάριος 2018

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την επιβλέπουσα καθηγήτρια κα. Κατερίνα Μανιά για την καθοδήγηση και εποπτεία κατά την εκπόνηση της διπλωματικής μου εργασίας. Μου παρείχε και μου μετέδωσε το ενδιαφέρον για το ψηφιακό κόσμο δίνοντάς μου παράλληλα τον έναυσμα και την δυνατότητα να ασχοληθώ σε μεγάλο βαθμό με τον τομέα της εικονικής πραγματικότητας.

Περιεχόμενα

Περίληψη 9
Κεφάλαιο 1ο Εισαγωγή 11
1.1 Εισαγωγή 11
1.2 Αντικείμενο της Πτυχιακής Εργασίας 11
1.3 Δομή της Εργασίας 13
Κεφάλαιο 2ο Θεωρητική Βάση 15
2.1 Ιστορική αναδρομή 15
2.2 Κατηγορίες Βιντεοπαιχνιδιών 17
2.2.1 Δράσης 17
2.3 Εικονική, Επαυξημένη, Μικτή Πραγματικότητα και Επαυξημένη Εικονικότητα 17
2.3.1 Εικονική Πραγματικότητα 17
2.3.2 Επαυξημένη Πραγματικότητα 18
2.3.3 Μικτή Πραγματικότητα 19
2.3.4 Επαυξημένη Εικονικότητα 19
2.4 Εμβύθιση (Immersion) 20
2.5 Ρυθμός μετάδοσης εικόνων (Frame Rate) 22
2.6 Τα κύρια μέρη του Virtual Reality Headset 23
2.7 Head Tracking, Positional Tracking, Eye Tracking, Hand Tracking 25
2.7.1 Head Tracking 25
2.7.2 Positional Tracking 26
2.7.3 Eye Tracking 27
2.7.4 Hand Tracking 27
Κεφάλαιο 3ο Γραφικά Υπολογιστών και Μηχανές Παιχνιδιών 29
3.1 Γραφικά Υπολογιστών και Μηχανές Παιχνιδιών 29
3.2 Τρισδιάστατα Γραφικά Υπολογιστών (3D computer graphics) 29
3.2.1 Τρισδιάστατη Μοντελοποίηση (3D modeling) 29
3.2.2 Animation-3D Animation 30
3.2.3 Rendering-3D Rendering 30
3.3 Μηχανές Παιχνιδιών 32
3.4 Δημοφιλείς Μηχανές Παιχνιδιών 34
3.4.1 Cry Engine 34
3.4.2 Amazon Lumberyard 34
3.4.3 Unreal Engine 35
3.4.4 Unity3D 35
3.5 Επιλογή της Κατάλληλης Μηχανής 35
Κεφάλαιο 4ο Τεχνολογική βάση 37
4.1 Εισαγωγή 37
4.2 Βασικές Έννοιες της Unity 37

4.3 Φωτισμός (Lighting) 39
4.4 Τεχνικές Φωτισμού 41
4.4.1 Επιλογή της Κατάλληλης Μεθόδου 43
4.5 Textures, Shader, και Materials 31
4.6 Φυσική Αντικειμένων, Στερεά Σώματα, Σύγκρουση (Physics, Rigidbody, Colliders) 43
4.7 Scripting ή Script Language (Γλώσσα Σεναρίων) 44
4.8 Κινούμενα Σχέδια (Animation) 46
4.9 Τεχνητή Νοημοσύνη 22
4.10 Πλοήγηση και Εύρεση Καλύτερου Μονοπατιού (Navigation and Path Finding) 47
4.11 Διεπαφή Χρήστη (User interface) 47
4.12 Ήχος (audio) και Κάμερα (camera) 47
4.13 Πλέγμα Πολυγώνων 49
4.14 Particle System, Prefab και Assets 49
4.15 Leap Motion 50
4.16 Oculus Rift 55
4.16.1 Εισαγωγή 55
4.16.2 Binocular Vision 56
4.16.3 Field of View & Scale 56
4.16.4 Rendering Techniques 57
4.16.5 Παρακολούθηση (Tracking) 58
4.17 Ασθένεια Εικονικής Πραγματικότητας (Virtual Reality Sickness) 59
Κεφάλαιο 5ο Σχεδιασμός Παιχνιδιού 62
5.1 Εισαγωγή 62
5.2 Σενάριο και Σύντομη Περιγραφή του Παιχνιδιού 62
5.3 Terrain 63
5.3.1 Texture (υφές) 67
5.3.2 Βλάστηση, Άνεμος, Δέντρα 68
5.3.3 Στατικά Μοντέλα, Σύστημα Σωματιδίων 70
5.4 Χειρισμός Παιχνιδιού 73
5.4.1 Είδη Κινήσεων 73
5.4.2 Επιθέσεις 77
5.5 Κίνηση Χαρακτήρων στο Χώρο 79
5.5.1 Αρκούδα 79
5.5.2 Λύκος 81
5.5.3 Σκελετός 82
5.5.4 Αρχηγός Δαίμονας 82
5.5.5 Ελάφι 84
5.6 Αποστολές 85
5.7 Γραφική Διεπαφή Χρήστη 86
5.8 Μενού Εφαρμογής 86

5.8.1 Αρχικό Μενού 86
5.8.2 Μενού Παύσης 87
5.8.3 Μενού Θανάτου 88
5.8.4 HUD και Γραφικές Διεπαφές 89
5.9 Ήχοι (Audio) 90
Κεφάλαιο 6ο Υλοποίηση Παιχνιδιού 93
6.1 Εισαγωγή 93
6.2 Χειρισμός Παίχτη 93
6.2.1 Ray Casting και Μοντέλο Παίχτη Κατάλληλο για Κίνηση στο Χώρο 93
6.2.2 Ενσωμάτωση Oculus Rift και Mounted Leap Motion 95
6.2.3 Χειρισμός Παίχτη 95
6.2.4 Κινήσεις Χεριών 96
6.2.4 Τεχνικά Στοιχεία Μπάλας Φωτιάς 100
6.3 Συστήματα Σωματιδίων και Χειρονομίες 101
6.4 Τεχνητή Νοημοσύνη και Χαρακτήρες 103
6.5 Χαρακτήρες – Αυτόνομοι Πράκτορες και Animation 104
6.5.1 Αρκούδα 104
6.5.2 Λύκος 107
6.5.3 Ελάφι 108
6.5.4 Σκελετός 109
6.5.5 Τελικός Αρχηγός, Δαίμονας 110
6.6 Διαχείριση Ενεργειών Μέσω Script 112
6.7 Γραφική Διεπαφή Χρήστη 113
6.8 Αλλαγή Σκηνής και Υπομενού 114
6.8.1 Αλλαγή Σκηνής 114
6.8.2 Εμφάνιση Υπομενού 114
6.9 Μενού Παύσης 115
Κεφάλαιο 7ο Αποτελέσματα, Εκτίμηση και Μελλοντικές Επεκτάσεις 118
7.1 Εισαγωγή 118
7.2 Αποτελέσματα-Εκτίμηση 118
Στόχος όπως ήδη αναφέραμε ήταν η δημιουργία ενός παιχνιδιού εικονικής πραγματικότητας συνδυάζοντας το Oculus Rift και το Leap Motion ώστε ο χρήστης να εμβυθιστεί με επιτυχία στο εικονικό περιβάλλον. 118
7.3 Μελλοντικές Επεκτάσεις και Βελτιώσεις 119
Κεφάλαιο 8ο Επίλογος 120
Βιβλιογραφία 121

Πίνακας Εικόνων

Εικόνα 1: Παιχνίδια Δράσης Αριστερά: Bloodborne, Δεξιά: Doom (2016) 17
Εικόνα 2: Virtual Reality 18
Εικόνα 3: Επαυξημένη πραγματικότητα (Pokemon 2016) 19
Εικόνα 4: Μικτή πραγματικότητα 19
Εικόνα 5: Σύγκριση της επαυξημένης πραγματικότητας και της επαυξημένης εικονικότητας 20
Εικόνα 6: The Reality-Virtuality Continuum 20
Εικόνα 7: Immersion (εμβύθιση) 21
Εικόνα 8: Η χρονική καθυστέρηση μεταξύ των πράξεων του χρήστη και της αλλαγής στο εικονικό περιβάλλον ονομάζεται Latency 22
Εικόνα 9: Head Mount Display 23
Εικόνα 10: Field of View 25
Εικόνα 11: Head Tracking 26
Εικόνα 12: Positional Tracking 27
Εικόνα 13: Eye Tracking. Κέντρο Ίριδας (κόκκινο), Ανάκλαση Κερατοειδούς (πράσινο) και Διάνυσμα Εξόδου (μπλε) 27
Εικόνα 14: Hand Tracking 28
Εικόνα 15:Material, Texture, Shader 32
Εικόνα 16: Ο Σκοπός της Game Engine 33
Εικόνα 17: CryEngine 34
Εικόνα 18:Amazon Lumberyard 34
Εικόνα 19: Unreal Engine 35
Εικόνα 20: Unity3D 35
Εικόνα 21: Συστήμα Συντεταγμένων 37
Εικόνα 22: Parenting 39
Εικόνα 23: Basic Lighting Diagram 39
Εικόνα 24: Directional Light 40
Εικόνα 25: Point Light 40
Εικόνα 26: Η ίδια σκηνή: χωρίς lighting (αριστερά), μόνο με ‘direct light’ (κέντρο) & με έμεσο 'global illumination' (δεξία). Παρατηρήστε πως μεταφέρονται τα χρώματα καθώς το φως ‘αναπηδά' πάνω στις επιφάνειες, δίνοντας πολύ περισσότερο ρεαλισμό 41
Εικόνα 27: Real Time Lighting 42
Εικόνα 28: Διαφορά Compiler-Interpreter 45
Εικόνα 29: Audio Source-Listener Diagram 48
Εικόνα 30: Camera 49
Εικόνα 31: Particle System 50
Εικόνα 32: Πεδίο Hand Tracking του Leap Motion 51
Εικόνα 33:Δεδομένα αισθητήρα σε μορφή στερεοσκοπικής εικόνας γκρι φάσματος 51
Εικόνα 34: Η άποψη του Motion Leap για τα χέρια σας 52
Εικόνα 35: Το σωστό σύστημα συντεταγμένων Leap Motion 53
Εικόνα 36: Το “χέρι” PalmNormal και η κατεύθυνση του διανύσματος (Direction vectors) ορίζουν τον προσανατολισμό του χεριού 53
Εικόνα 37: Τα σύμβολα "TipPosition" και "Direction" του δάκτυλου παρέχουν τη θέση ενός άκρου δακτύλου και τη γενική κατεύθυνση στην οποία δείχνει ένα δάχτυλο. 54
Εικόνα 38: Ανατομία χεριού 55
Εικόνα 39: Oculus Rift Consumer Version 1 56
Εικόνα 40: 6DoF 58
Εικόνα 41: Oculus Rift CV1 Features 59
Εικόνα 42: Virtual Reality Sickness 61
Εικόνα 43: Terrain Περιοχή 1: Χωριό(σημείο εκκινησης), Περιοχή 2: Δάσος Αρκούδων, Περιοχή 3: Δάσος Λύκων 64
Εικόνα 44: Village 64

Εικόνα 45: Δάσος αρκούδων 65
Εικόνα 46: Δάσος λύκων 65
Εικόνα 47: Cemetery 1 66
Εικόνα 48: Cemetery 2 66
Εικόνα 49: Cemetery 3 66
Εικόνα 50: Βωμός 67
Εικόνα 51: Υφές Εφαρμογής 68
Εικόνα 52: Επικάλυψη Τερέν με Υφές 68
Εικόνα 53: Δέντρα 69
Εικόνα 54: Γρασίδι 69
Εικόνα 55: Skybox 70
Εικόνα 56: Μπάλα Φωτιάς 72
Εικόνα 57: Μπάλα Πάγου 72
Εικόνα 58: Πράσινη Μπάλα 73
Εικόνα 59: Μαγική Μπάλα 73
Εικόνα 60: Πρόσθια κίνηση προς την κατεύθυνση του οπτικού πεδίου του χρήστη 74
Εικόνα 61: Όπισθεν Κίνηση 74
Εικόνα 62: Δεξιά Κίνηση 75
Εικόνα 63: Αριστερή Κίνηση 75
Εικόνα 64: Διαγώνια Κίνηση με φορά Έμπροσθεν και Δεξιά 75
Εικόνα 65: Διαγώνια Κίνηση με φορά Έμπροσθεν και Αριστερά 76
Εικόνα 66: Όπισθεν Δεξιά Κίνηση 76
Εικόνα 67: Όπισθεν Αριστερή Κίνηση 76
Εικόνα 68: Επίθεση Μπάλας Φωτιάς 77
Εικόνα 69: Επίθεση Μπάλας Πάγου 78
Εικόνα 70: Επίθεση Πράσινης Μπάλας φάση 1 78
Εικόνα 71: Επίθεση Πράσινης Μπάλας φάση 2 79
Εικόνα 72: Αρκούδα 79
Εικόνα 73: Πάγος 80
Εικόνα 74: Λύκος 81
Εικόνα 75: Σκελετός 82
Εικόνα 76: Τελικός Αρχηγός Δαίμονας με Ασπίδα 83
Εικόνα 77: Τελικός Αρχηγός Δαίμονας χωρίς Ασπίδα 83
Εικόνα 78: Ελάφι 84
Εικόνα 79: Αρχικό Μενού 87
Εικόνα 80: Μενού Παύσης 88
Εικόνα 81: Μενού Θανάτου 89
Εικόνα 82: HUD, Η πράσινη μπάρα απεικονίζει την ενέργεια του παίχτη και η μπλε το mana. 89
Εικόνα 83: PlayerCapsule Floating ..... **Σφάλμα! Δεν έχει οριστεί σελιδοδείκτης.**
Εικόνα 84: Κώδικας Floating του PlayerCapsule 94
Εικόνα 85: Συνθήκη Εντοπισμού Κλειστής Αριστερής Παλάμης 97
Εικόνα 86: Όρια Δεξιάς Στροφής Καρπού 97
Εικόνα 87: Όρια Αριστερής Στροφής Καρπού 98
Εικόνα 88: Έλεγχοι για Εκτόξευση Πρασινης Μπάλας ή Φωτιάς 99
Εικόνα 89: Κώδικας Επίθεσης Αριστερού Χεριού 99
Εικόνα 90: Κώδικας Προεξέχοντα Αντίχειρα και Δείκτη Δεξιού και Αριστερού Χεριού 100
Εικόνα 91: Κώδικας Ελέγχου Χρονικών Περιθωρίων Μπάλας Επίθεσης 101
Εικόνα 92: Δήλωση Συστήματος Σωματιδίου 102
Εικόνα 93: Ενεργοποίηση Συστήματος Σωματιδίου 102
Εικόνα 94: Απενεργοποίηση Συστήματος Σωματιδίου 102
Εικόνα 95: Κώδικας Ενεργοποίησης Συστήματος Σωματιδίου σε Σχέση με το Χρόνο 103
Εικόνα 96: Εύρεση του Καλύτερου Μονοπατιού 104
Εικόνα 97: Κώδικας Αναπαραγωγής Ήχου 107

Εικόνα 98: Κώδικας Πεδίου Καλέσματος Λύκου 108
Εικόνα 99: Κώδικας Συμπεριφοράς Ελαφιού 109
Εικόνα 100: Κώδικας Ενημέρωσης Animation Controller σύμφωνα με συνθήκη if 111
Εικόνα 101: Εντολές Ενημέρωσης των Slider 114

Περίληψη

Οι εξελίξεις στην τεχνολογία πάντοτε υπήρξαν ένας καθοριστικός παράγοντας που σηματοδοτούσε σημαντικές αλλαγές στην πορεία της ανθρωπότητας με πολλαπλές επιδράσεις σε μία σειρά τομέων της ζωής των ανθρώπων όπως η υγεία η ψυχαγωγία κ.ά. Ιδιαίτερο και σημαντικό κομμάτι αυτής της τεχνολογικής ανάπτυξης είναι η εικονική πραγματικότητα. Αφού, η εικονική πραγματικότητα είναι μία από τις πιο ακμάζουσες τεχνολογίες και ιδιαίτερα στον τομέα της ψυχαγωγίας, έτσι και εμείς εμπνευστήκαμε τη δημιουργία ενός παιχνιδιού γραφικών τριών διαστάσεων εικονικής πραγματικότητας βασισμένο στις πρωτοποριακές τεχνολογίες Oculus Rift και Leap Motion, καθώς πιστεύουμε ότι αποτελεί επιταγή των καιρών μας η ενασχόληση και ανάπτυξη τέτοιου είδους εφαρμογών.

Η εφαρμογή ανήκει στην κατηγορία δράσης πρώτου προσώπου (First Person Shooter) και δημιουργήθηκε κάνοντας χρήση της μηχανής γραφικών Unity3D πάνω στην οποία βασίστηκε όλη η μηχανική του παιχνιδιού, από τη σύνδεση των τεχνολογιών, τους νόμους της φυσικής, τη σύνθεση των γραφικών και ήχων μέχρι την κατασκευή μοντέλων-χαρακτήρων και της συμπεριφοράς τους στον ψηφιακό κόσμο. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η C#, η οποία αποτελεί γλώσσα σεναρίων (scripting language) και είναι κατάλληλη για εφαρμογές που σχετίζονται με γραφικά, αφού εκτελεί τα διάφορα κομμάτια κώδικα σύμφωνα με τον ρυθμό των frames.

Μέσα σε αυτήν την εφαρμογή, ο χρήστης βρίσκεται σε όρθια θέση φορώντας το Oculus Rift στο κεφάλι και αντιλαμβάνεται το εικονικό περιβάλλον μέσω των δύο φακών που βρίσκονται στη συσκευή. Εμπεριέχει σύστημα ανίχνευσης τοποθεσίας καθορίζοντας την ακριβή θέση του χρήστη στο εικονικό περιβάλλον και μέσω του γυροσκοπίου (σύστημα ενσωματωμένο στη συσκευή), ανιχνεύεται η οποιαδήποτε περιστροφική κίνηση της κεφαλής αποτυπώνοντάς την στον ψηφιακό κόσμο. Επιπλέον, χρησιμοποιείται η συσκευή Leap Motion, λόγω αναγκαιότητας της ακριβής ανίχνευσης χεριών και δακτύλων που επιτυγχάνεται μέσω αισθητήρων και κάμερας της συσκευής. Αυτό έχει σαν αποτέλεσμα, η πραγματική κίνηση του χεριού και των δακτύλων να αντικατοπτρίζεται με ακρίβεια στο εικονικό περιβάλλον. Αξίζει να αναφερθεί εδώ ότι δε γίνεται χρήση των κλασσικών συσκευών εισόδου και εξόδου, όπως ποντίκι, πληκτρολόγιο, ηχεία και οθόνη.

Η δράση διαδραματίζεται σε ένα πολύπλοκο, ορθογώνιο, γεωμετρικό τερέν με πλούσια γεωμορφολογία εδάφους, ποικιλόμορφα τοπία και διάφορες υφές, τοποθετημένες με καλλιτεχνικό τρόπο πάνω σε αυτό, για τη δημιουργία ενός ρεαλιστικού, ευχάριστου και ελκυστικού προς το χρήστη παιχνιδιού εικονικής πραγματικότητας. Πιο συγκεκριμένα, ο χρήστης εμβυθίζεται στο εικονικό περιβάλλον, καλούμενος να διεκπεραιώσει μία σειρά τεσσάρων αποστολών μέσα στις οποίες παρεμβάλλονται εχθροί που παρεμποδίζουν την ολοκλήρωσή τους. Οι εχθροί είναι χαρακτήρες-μοντέλα προγραμματισμένοι με διαφορετική συμπεριφορά ανάλογα το είδος τους. Είναι εμπλουτισμένοι με κινούμενα σχέδια (animation) ώστε η κίνησή τους να είναι ρεαλιστική και λειτουργούν ως αυτόνομοι πράκτορες,

βασισμένοι στις αρχές της τεχνητής νοημοσύνης. Επίσης, σχεδιάστηκε ένα εύχρηστο User Interface, δίνοντας τη δυνατότητα στο χρήστη να επιλέγει μόνος του το μενού που θέλει να ενεργοποιήσει κάθε φορά με τη χρήση των δακτύλων του μέσω του Leap Motion.

Τέλος, ιδιαίτερη έμφαση δόθηκε στην καλλιτεχνική πλευρά της εφαρμογής επιλέγοντας με ευαισθησία τα διάφορα μοντέλα, υφές, ήχους κλπ. Η εφαρμογή αξιολογήθηκε μέσω του ερωτηματολογίου Simulator Sickness Questionaire, από χρήστες και των δύο φύλων, διαφορετικής ηλικίας, καθώς θέλαμε να ελέγξουμε αν η σχεδίαση της εφαρμογής με γνώμονα την αποφυγή του simulation sickness ήταν επιτυχής.

Όλα τα παραπάνω έχουν στόχο την εξερεύνηση της διευρυμένης εικονικής πραγματικότητας και τη δημιουργία εφαρμογής όπου θα εξασφαλίζεται η επιτυχημένη διαδραστικότητα μεταξύ χρήστη και ψηφιακού κόσμου.

Κεφάλαιο 1^{ο} Εισαγωγή

1.1 Εισαγωγή

Η εικονική πραγματικότητα (virtual reality), η προσομοίωση δηλαδή ενός περιβάλλοντος από έναν υπολογιστή, εξελίσσεται σε μια από τις βασικές τάσεις της παγκόσμιας αγοράς των ψηφιακών τεχνολογιών καθώς προϊόντα και εφαρμογές έχουν αρχίσει να αυξάνονται και να διατίθενται προς κατανάλωση στην αγορά και μάλιστα με εντυπωσιακούς ρυθμούς. Καθώς η VR έχει χρήσεις σε πολύ μεγάλο εύρος της ανθρώπινης δραστηριότητας (ψυχαγωγία, τέχνη, εκπαίδευση, ιατρική, στρατό, διάστημα) αναμένεται ότι θα φέρει την επανάσταση και θα αλλάξει ριζικά βασικούς τομείς της ζωής μας. Αυτό ήταν εξάλλου υπόσχεση της τεχνολογίας από τις αρχές της δεκαετίας του πενήντα, θα αρχίσει δε να γίνεται περισσότερο αντιληπτό όταν η εικονική πραγματικότητα θα γίνει προσιτή σε όλους. Ανατρεπτικός παράγοντας μέχρι τώρα για την απόκτηση ενός «δυνατού» Virtual Reality Head Set και ενός αντίστοιχου ισχυρού υπολογιστή από το ευρύ αγοραστικό κοινό, είναι τα οικονομικά μεγέθη που απαιτούνται για την αγορά τους. Ήδη όμως και το εμπόδιο αυτό έχει αρχίσει να φθίνει καθώς μεγάλες εταιρίες του χώρου της πληροφορικής επενδύουν υπέρογκα χρηματικά ποσά στην τεχνολογία της εικονικής πραγματικότητας. Ένας από αυτούς τους επενδυτές είναι και ο ιδρυτής-ιδιοκτήτης της πασίγνωστης ιστοσελίδας κοινωνικής δικτύωσης FaceBook Mark Zuckerberg ο οποίος μάλιστα τον Μάρτιου του 2014 αγόρασε την εταιρία Oculus έναντι του υπέρογκου ποσού των δύο δισεκατομμυρίων δολαρίων.

Δεν προκαλεί επομένως έκπληξη το γεγονός ότι και η βιομηχανία βιντεοπαιχνιδιών είναι ένας από τους μεγαλύτερους υποστηριχτές του χώρου αυτού . Έτσι, με ταχείς ρυθμούς επαναπροσδιορίζει και δημιουργεί τεχνικές που θα έχουν εφαρμογή σε video games τα οποία τρέχουν σε συσκευές VR αξιοποιώντας την εκρηκτική επέκταση των δυνατοτήτων της ψηφιακής τεχνολογίας και των ηλεκτρονικών υπολογιστών.

1.2 Αντικείμενο της Πτυχιακής Εργασίας

Στις μέρες μας όλο και περισσότερο χρησιμοποιείται η τεχνολογία της εικονικής πραγματικότητας και σε πολλές πτυχές της ζωής μας. Με τη χρήση της, έχει γίνει εφικτό να πλησιάσουμε μέρη που δεν μπορούσαμε διαφορετικά, να προσφέρουμε γνώση σε ανθρώπους με κινητικά προβλήματα, να κάνουμε πιο διασκεδαστική την εκπαίδευση για τα παιδιά και όχι μόνο, αλλά και να διασκεδάσουμε. Όσο περισσότερο χρησιμοποιείται η τεχνολογία της εικονικής πραγματικότητας σε τομείς της ζωής μας, τόσο θα ανακαλύπτονται και νέες τεχνικές, πιο βελτιωμένες από τις προηγούμενες με σκοπό την καλύτερη απεικόνιση και αλληλεπίδραση.

Η παρούσα πτυχιακή ασχολείται με την εξερεύνηση της διευρυμένης εικονικής πραγματικότητας στο gaming δημιουργώντας ένα επαναστατικό βιντεοπαιχνίδι δράσης πρώτου προσώπου, κατά το οποίο ο χρήστης χρησιμοποιεί καινοτόμες τεχνολογίες εικονικής πραγματικότητας, όπως το Oculus Rift και το Leap Motion.

Το παιχνίδι σχεδιάστηκε με γραφικά 3 διαστάσεων (3D) για να γίνει πιο ρεαλιστικό και εντυπωσιακό στο χρήστη. Τα 3D είναι γραφικά που στηρίζονται συχνά στο καρτεσιανό σύστημα (x,y,z), αποθηκευμένα στη μνήμη του υπολογιστή με σκοπό την εκτέλεση υπολογισμών και την απόδοση εικόνων 2 διαστάσεων. Η δημιουργία τους χωρίζεται σε 3

φάσεις: την τρισδιάστατη μοντελοποίηση (3D Modeling), τα κινούμενα σχέδια (Animation), και την απόδοση δισδιάστατης εικόνας (3D Rendering).

Αυτά τα γραφικά 3 διαστάσεων συντέθηκαν και αποδόθηκαν μέσω της Unity 3D. Η Unity 3D αποτελεί μία από τις καταλληλότερες επιλογές των προγραμματιστών λόγω συμβατότητάς της με μεγάλη γκάμα πλατφόρμων και προσαρμοστικότητας στα hardware των υπολογιστών αποδίδοντας γραφικά υψηλής ποιότητας. Διαθέτει ισχυρές μηχανές φυσικής και αποτελεί ελεύθερο λογισμικό. Προκειμένου όμως, να επεκταθούν οι λειτουργίες αυτής της πλατφόρμας γίνεται κατά κόρον χρήση των λεγόμενων scripts. Τα scripts είναι πηγαίος κώδικας, ο οποίος μεταφράζεται από κάποιο πρόγραμμα και τρέχει σε πραγματικό χρόνο. Στη συγκεκριμένη εφαρμογή, τα scripts αποδόθηκαν σε C#, γλώσσα προγραμματισμού στην οποία οι κώδικες εκτελούνται σε κάθε καινούργιο frame.

Πέραν όμως των τεχνικών στοιχείων, ο χρήστης προκειμένου να εμβυθιστεί στον εικονικό κόσμο και να επιτευχθεί ποιοτική διαδραστικότητα, χρησιμοποιεί τις συσκευές Oculus Rift και Leap Motion.

Το Oculus Rift είναι συσκευή εικονικής πραγματικότητας που τοποθετείται στο κεφάλι και μέσω των φακών και οθονών που υπάρχουν, ο χρήστης εμβυθίζεται στο εικονικό περιβάλλον. Μέσα σε αυτό το περιβάλλον, υπάρχει μία κάμερα που συνδέεται με το Oculus Rift. Η πληροφορία του οπτικού πεδίου της κάμερας μεταφέρεται στην κάθε οθόνη του Oculus Rift υπό διαφορετική γωνία δίνοντας στο χρήστη την αίσθηση παρουσίας του στον ψηφιακό κόσμο. Ένα από τα σημαντικότερα χαρακτηριστικά της συσκευής είναι το Head Tracking, σύστημα που αποτελείται από επιταχυνσιόμετρο, μαγνητόμετρο και γυροσκόπιο. Μέσω του γυροσκοπίου, ανιχνεύεται οποιαδήποτε περιστροφική κίνηση της κεφαλής στον πραγματικό κόσμο και αυτή η πληροφορία μεταβιβάζεται στην κάμερα του εικονικού κόσμου με αποτέλεσμα να περιστρέφεται και αυτή.

Για την εκτέλεση της εφαρμογής, πέρα από την ανίχνευση οποιασδήποτε κίνησης της κεφαλής, είναι απαραίτητη και η ανίχνευση της κίνησης των χεριών και των δαχτύλων του χρήστη. Αυτό επιτυγχάνεται μέσω της συσκευής Leap Motion. Η συγκεκριμένη αποτελείται από δύο κάμερες και τρία LED, που παράγουν υπέρυθρο φως και συνθέτουν ένα αισθητήρα που καταγράφει τις κινήσεις των χεριών του χρήστη σε αρκετά frames/second. Η ακατέργαστη εικόνα που συντίθεται από τον αισθητήρα, αντλείται από τον υπολογιστή και το λογισμικό του Leap Motion και με προηγμένους μαθηματικούς αλγόριθμους την επεξεργάζεται. Έτσι, οι εικόνες αναλύονται για να δημιουργήσουν μία 3D αναπαράσταση του τι βλέπει η συσκευή και στη συνέχεια να αποδοθεί στον ψηφιακό κόσμο.

Όλα τα παραπάνω λοιπόν συνετέλεσαν στη δημιουργία του παιχνιδιού που ονομάζεται Eternal Conflict. Ο χρήστης βρίσκεται στη θέση του μάγου Aidunghast και αφού πρώτα περάσει τις δοκιμασίες των τεσσάρων αποστολών, έχει ως τελικό στόχο την εξόντωση του δαίμονα Baphomet, ο οποίος προσπαθεί να εξουσιάσει όλη την ανθρωπότητα. Για την επίτευξη των στόχων, ο χρήστης θα πρέπει να εκτελεί συγκεκριμένες κινήσεις του αριστερού χεριού και περιστροφικές της κεφαλής.

Οι 4 αποστολές τις οποίες καλείται να διεκπεραιώσει ο χρήστης, διαφέρουν μεταξύ τους ως προς τον τρόπο ολοκλήρωσης τους. Κάθε αποστολή διαθέτει και έναν διαφορετικό εχθρό. Τα 4 είδη εχθρών είναι: α) η αρκούδα, β) ο λύκος, γ) ο σκελετός και δ) ο τελικός αρχηγός δαίμονας. Όλοι οι εχθροί αποτελούν αυτόνομους πράκτορες, δηλαδή η συμπεριφορά τους έχει διαμορφωθεί σύμφωνα με τις βασικές αρχές της τεχνητής νοημοσύνης. Για την δημιουργία της τεχνητής νοημοσύνης, εκμεταλλευτήκαμε τις δυνατότητες της Unity 3D και πιο συγκεκριμένα τη λειτουργία εύρεσης του καλύτερου μονοπατιού. Δηλαδή, η αυτόματη επιλογή της πιο σύντομης διαδρομής που ένα αντικείμενο θα ακολουθήσει για να πάει σε ένα συγκεκριμένο στόχο. Κάθε εχθρός εκτός από ιδιαίτερη συμπεριφορά έχει και δικά του κινούμενα σχέδια, ώστε να αντιλαμβάνεται ο χρήστης την φυσική κίνηση του κάθε εχθρού, όπως τρέξιμο, περπάτημα, επίθεση κλπ.

Η εφαρμογή έχει προγραμματιστεί ώστε ο ψηφιακός χαρακτήρας του εικονικού περιβάλλοντος να εκτελεί κάποιες ενέργειες ανάλογα με τις χειρονομίες του χρήστη, που έχουν κωδικοποιηθεί μέσω scripts. Τις χειρονομίες που δημιουργήσαμε και προγραμματίσαμε για την εξέλιξη του παιχνιδιού περιέχονται σε 8 ευθύγραμμες κινήσεις, καθώς και άλλες οι οποίες αναφέρονται στο αντίστοιχο κεφάλαιο και σε 3 επιθέσεις, με την πράσινη μπάλα, την μπάλα της φωτιάς και του πάγου. Για παράδειγμα, για να εκτελεστεί μία επίθεση θα πρέπει η παλάμη να αποκτήσει την κατάλληλη στιγμιαία ταχύτητα, έτσι ώστε να δοθεί εντολή να δημιουργηθεί η αντίστοιχη μαγική μπάλα. Η πληροφορία της ταχύτητας της παλάμης αντλείται μέσω εντολών του Leap Motion.

Όλο το παιχνίδι εκτυλίσσεται σε ένα κοινό τερέν, ορθογώνιου σχήματος με πλούσια γεωμορφολογία. Μέσα σε αυτό το τερέν υπάρχουν ταυτόχρονα πέντε νοητές περιοχές. Αυτές είναι: 1) το χωριό, 2) το δάσος αρκούδων, 3) το δάσος λύκων, 4) το κοιμητήριο και 5) η περιοχή του βωμού. Για τη ρεαλισμό του τερέν και κάθε περιοχής του, χρησιμοποιήσαμε μεγάλη ποικιλία υφών, οι οποίες επιλέχθηκαν με μεγάλη ευαισθησία.

Ακόμα, σημαντικό κομμάτι της εφαρμογής είναι τα ηχητικά εφέ, τα οποία με τη σειρά τους αποτελούν αναπόσπαστο κομμάτι της εμβύθισης του χρήστη στο εικονικό περιβάλλον. Αυτά πραγματοποιούνται από τη Unity παράγοντας ήχους από ηχητικές πηγές (audio sources), τοποθετημένες πάνω στα αντικείμενα και εντοπίζονται από έναν ηχητικό ακροατή (audio listener). Υπάρχουν δύο ειδών ηχητικές πηγές, η μία καθορίζει την ένταση του ήχου με βάση την απόσταση της από τον ηχητικό ακροατή και η άλλη είναι ανεξάρτητη με σταθερή ένταση και γίνεται αντιληπτή από τον ηχητικό ακροατή σε οποιοδήποτε σημείο της σκηνής κι αν βρίσκεται αυτός.

Τέλος, προκειμένου ο χρήστης να περιηγηθεί με ευελιξία στην εφαρμογή σχεδιάσαμε ένα εύχρηστο User Interface. Μέσω αυτού, ο χρήστης μπορεί να ξεκινήσει το παιχνίδι, να προχωρήσει σε οποιαδήποτε αποστολή επιθυμεί, να κάνει επανεκκίνηση της αποστολής που βρίσκεται, να επιστρέψει στην αρχική οθόνη αλλά και να κλείσει την εφαρμογή. Για να εκτελεστεί μία από τις παραπάνω ενέργειες, θα πρέπει ο χρήστης με το δείκτη του δεξιού του χεριού να πιέσει το αντίστοιχο κουμπί στο εκάστοτε μενού.

Μέσα από αυτή την εφαρμογή λοιπόν, καταφέραμε να συνδυάσουμε αρμονικά, φυσικά τοπία με μαγικά εφέ και προηγμένη τεχνολογία για τον βέλτιστο τρόπο εμβύθισης και αλληλεπίδρασης του χρήστη με το εικονικό περιβάλλον.

1.3 Δομή της Εργασίας

Κλείνοντας με το πρώτο κεφάλαιο το οποίο ήταν εισαγωγικό, γίνεται μια συνοπτική αναφορά των υπόλοιπων έξι που ακολουθούν.

Στο κεφάλαιο 2, πλην της ιστορικής αναδρομής που αναφέρεται εκτενώς, αναφέρονται και οι κατηγορίες βιντεοπαιχνιδιών, γίνεται αναλυτική περιγραφή της εικονικής πραγματικότητας ως και των υποκλάδων αυτής. Επίσης, αναλύεται η έννοια του Head Mount Display, της μηχανής παιχνιδιών και της τεχνητής νοημοσύνης.

Στο κεφάλαιο 3 διεισδύουμε στις έννοιες των γραφικών υπολογιστών και ειδικότερα στα τρισδιάστατα γραφικά. Αναλύεται η τρισδιάστατη μοντελοποίηση και τα βήματα που απαιτούνται για την απόδοση ενός εικονικού μοντέλου στην οθόνη. Επίσης, περιγράφονται οι τέσσερις περισσότερο διαδεδομένες μηχανές παιχνιδιών και ο λόγος που επιλέχθηκε η πλατφόρμα Unity3D για τη δημιουργία της συγκεκριμένης εφαρμογής.

Στο κεφάλαιο 4 αναφέρονται οι βασικές έννοιες της Unity3D. Ακόμα, περιγράφονται και αναλύονται οι τρόποι και οι τεχνικές φωτισμού. Περιγράφεται η διαδικασία με την οποία η επιφάνεια ενός μοντέλου αποκτά σχήμα, χρώμα και φωτεινότητα (Textures, Shaders, Materials). Γίνεται αναφορά στις μηχανές φυσικής και στον τρόπο με τον οποίο αποδίδονται οι νόμοι της φυσικής στην εφαρμογή όπως και ανάλυση γίνεται στα κινούμενα σχέδια, στην τεχνική εύρεσης του καλύτερου μονοπατιού, στην τεχνητή νοημοσύνη, στο scripting, στην ηχητική επένδυση και στην έννοια της κάμερας μέσα από την οποία ο χρήστης βλέπει τον ψηφιακό κόσμο. Επίσης, γίνεται αναλυτική περιγραφή του Leap Motion και του Oculus Rift και δίνεται ιδιαίτερη έμφαση στις αρνητικές επιπτώσεις της εικονικής πραγματικότητας (simulation sickness). Τέλος, περιγράφονται οι διεπαφές χρήστη και το εικονικό περιβάλλον εφαρμογής.

Στην αρχή του κεφαλαίου 5 γίνεται σύντομη περιγραφή του παιχνιδιού, του σεναρίου και της πλοκής αυτού. Στη συνέχεια παρουσιάζεται το terrain πάνω στο οποίο εκτυλίσσεται όλη η δράση καθώς και όλα εκείνα τα στοιχεία που το συνθέτουν. Αναλύεται ο χειρισμός του παιχνιδιού και οι κινήσεις του χρήστη που αλληλεπιδρούν με το εικονικό περιβάλλον. Ακόμα, περιγράφονται οι εχθροί και η συμπεριφορά τους η οποία βασίζεται πάνω στην τεχνητή νοημοσύνη. Τέλος, παρουσιάζονται τα μενού και οι ήχοι που στόχο έχουν τη σύνθεση ενός εικονικού ρεαλιστικού περιβάλλοντος.

Το κεφάλαιο 6 είναι η προγραμματιστική και τεχνολογική απόδοση του κεφαλαίου 5. Σε αυτό αναπτύσσονται οι κινήσεις του χρήστη σε προγραμματιστική βάση καθώς και οι τρόποι δημιουργίας στοιχειώδους τεχνητής νοημοσύνης σύμφωνα με τις προγραμματιστικές τεχνικές. Ακόμα, αναλύεται η διαχείριση ορισμένων ενεργειών απαραίτητη για τη σύνθεση του παιχνιδιού μέσω script και ο τρόπος δημιουργίας των μενού, υπομενού και διεπαφών χρήστη.

Το κεφάλαιο 7 αναφέρεται στην ανάλυση των αποτελεσμάτων του ερωτηματολογίου που δόθηκε σε δεκαπέντε εθελοντές και στην δυνατή μελλοντική επέκταση της διπλωματικής εργασίας. Γίνεται μία σύντομη αναφορά στα χαρακτηριστικά του παιχνιδιού, στα θετικά επιτεύγματα και τις αρνητικές επιπτώσεις, στις δυνατότητες βελτίωσης, αναβάθμισης και επέκτασης της εφαρμογής.

Το κεφάλαιο 8 είναι ο επίλογος του κειμένου της πτυχιακής εργασίας.

Κεφάλαιο 2^{ο} Θεωρητική Βάση

Στο κεφάλαιο αυτό γίνεται ιστορική αναδρομή των βιντεοπαιχνιδιών (video games) και αναπτύσσονται η κατηγορία δράσης πάνω στην όποια βασίζεται και η συγκεκριμένη εφαρμογή. Επίσης, παρουσιάζονται οι πλατφόρμες και τα είδη τους δηλαδή, τα ηλεκτρονικά συστήματα εκτέλεσης των παιχνιδιών αυτών.

2.1 Ιστορική αναδρομή

Για την ιστορική αναδρομή των βιντεοπαιχνιδιών θα πρέπει να ανατρέξουμε στη δεκαετία του ‘50 τότε που ακαδημαϊκοί επιστήμονες υπολογιστών προσεγγίζουν τις βασικές αρχές και τα βήματα εκείνα που στη συνέχεια θα τους οδηγούσαν στο σχεδιασμό και την υλοποίηση των πρώτων video games αλλά και των πρώτων προσομοιώσεων.

Με τον όρο βιντεοπαιχνίδι εννοείται οποιοδήποτε παιχνίδι που πραγματοποιείται με τη χρήση κάποιας ηλεκτρονικής συσκευής. Αυτή μπορεί να είναι ένας ηλεκτρονικός υπολογιστής, μια κονσόλα βιντεοπαιχνιδιών, ένα κινητό τηλέφωνο και άλλα.

Τα video games άρχισαν να γίνονται γνωστά τις δεκαετίες του ‘70 και του ‘80 με την κυκλοφορία κυρίως του arcade, παιχνίδι που διαδόθηκε τάχιστα και έγινε αγαπητό στο ευρύ κοινό. Ήδη με το τέλος της δεκαετίας του ‘80 τα games είχαν γίνει στις περισσότερο βιομηχανικά αναπτυσσόμενες χώρες ένας δημοφιλής τρόπος διασκέδασης και κομμάτι της σύγχρονης ζωής και κουλτούρας. Κατά τη διάρκεια της δεκαετίας του ‘70 εμφανίστηκε επίσης και η πρώτη γενιά από κονσόλες σπιτιών με κυρίαρχο το Pong και τους διάφορους κλώνους του.

Η χρυσή εποχή του arcade θεωρείται η περίοδος από το 1978 έως το 1982 τότε που μηχανές γραφικών με κερματοδέκτες τοποθετούνται στα μεγάλα εμπορικά κέντρα, ενώ οικιακές κονσόλες όπως το Atari και το Intellivision μέσω της προσωπικής τηλεόρασης χαρίζουν την εμπειρία και τη χαρά του παιχνιδιού στο χρήστη. Επιπλέον κατά τη διάρκεια της δεκαετίας του ‘80 παρουσιάστηκαν τα gaming computer και τα πρώιμα διαδικτυακά παιχνίδια ως και οι πρώτες κονσόλες χειρός για να βιώσει όμως στη συνέχεια ο κλάδος την ύφεση λόγω του video games crash.

Από το έτος 1976 έως και το 1982 εμφανίζεται η δεύτερη γενιά κονσόλες σπιτιού, στη συνέχεια από το 1983 έως το 1995 η τρίτη με 8 bit unit ενώ η τέταρτη με 16 bit unit από το 1987 έως το 1999. Η δεκαετία του ‘90 είναι η εποχή της άνθησης αλλά και της παρακμής των Arcade. Στη συνέχεια από το 1993 έως το 2005 θα κυκλοφορήσει η 5η γενιά κονσόλας με 32 έως 64 Bit unit για να τη διαδεχτεί η 6^{η} γενιά που διήρκησε από το 1998 έως το 2013. Κατά τη διάρκεια της χρονικής αυτής περιόδου εμφανίζονται τα games στα κινητά τηλέφωνα ενώ ταυτόχρονα το διαδικτυακό παιχνίδι σε αυτά γνωρίζει μεγάλη εξάπλωση. Από το 2005 έως το 2012 θα κυκλοφορήσει η 7η γενιά με αντιπροσωπευτικότερη κονσόλα το Wii, στην οποία ο χρήστης χειρίζεται το game με πραγματικές κινήσεις, ενώ η 8η γενιά το 2013 με δημοφιλέστερα το Wii U και το 3DS της Nintendo, το X-box της Microsoft και το Play Station 4 της Sony. Παράλληλα στις μέρες μας και ειδικά τη χρονιά του 2017 κυκλοφορούν στην αγορά πληθώρα Head Mount Display προσφέροντας την εμπειρία της εικονικής πραγματικότητας στον κάθε χρήστη.

Η σύγχρονη εικονική πραγματικότητα είναι άρρηκτα συνδεδεμένη με τη τεχνολογία. Τα θεμέλια για τη χρήση της τέθηκαν όμως εκατοντάδες ή και χιλιάδες χρόνια πριν, όταν ο άνθρωπος δημιούργησε μέσω της τέχνης τη δική του πραγματικότητα. Τα νεότερα χρόνια και συγκεκριμένα το 1838 η έρευνα του Charles Wheatstone κατέδειξε ότι ο εγκέφαλος καθώς επεξεργάζεται δύο διαφορετικές δισδιάστατες εικόνες που προσλαμβάνει από κάθε οφθαλμό μπορεί να παράξει ένα ενιαίο αντικείμενο τριών διαστάσεων. Έτσι, αρχικά η επίτευξη της αίσθησης του βάθους έγινε πραγματικότητα από την παρατήρηση δύο συνεχόμενων εικόνων μέσα από το στερεοσκόπιο. Οι σχεδιαστικές αυτές αρχές έχουν εφαρμογή μέχρι και σήμερα και ειδικότερα στη δημιουργία του δημοφιλούς Google Carboard. Το Google Cardboard είναι ένα χαρτόνι το οποίο αναδιπλώνεται και σχηματίζει ένα υποτυπώδες VR Headset, στο οποίο μπορεί να προσαρμοστεί στο κινητό του χρήστη.

Το έτος 1929 ο Edwin Albert Link δημιούργησε το Link Trainer πιθανότατα τον πρώτο ηλεκτρομηχανολογικό εξομοιωτή πτήσης ενώ αργότερα το 1950 ο Morton Heilig το Sensorama μια θεατρική καμπίνα με στοιχεία arcade που θα διέγειρε όλες τις αισθήσεις του χρήστη και όχι μόνο αυτές της ακοής και της όρασης. Έτσι, ο χρήστης στη βόλτα του με τη μηχανή στο Μανχάταν ένοιωθε τον αέρα να του χτυπά το πρόσωπο αλλά και μύριζε τα αρώματα της πόλης. Η επόμενη πατέντα του ίδιου το 1960 ήταν το Telesphere-Mask η πρώτη απόπειρα δημιουργίας οθόνης κεφαλής το Mount Display με παρακολούθηση μη διαδραστικών φιλμ, χωρίς δυνατότητα εντοπισμού κίνησης, γεγονός που επιτεύχθηκε το 1961 με το Head sight που διέθετε οθόνη για κάθε οφθαλμό και μαγνητικό σύστημα εντοπισμού κίνησης συνδεδεμένα με κλειστό κύκλωμα κάμερας.

Το 1968 ο Sutherland με τη βοήθεια του μαθητή του Bob Sproull κατασκευάζουν το πρώτο Virtual Reality / Augmented Reality head mount display , το Sword of Damocles (“Δαμόκλειο Σπαθί”) συνδεδεμένο με υπολογιστή και όχι με κάμερα, με μηχανική ανίχνευσης της κίνησης του κεφαλιού. Η συσκευή ήταν ογκώδης και δύσχρηστη και κρεμόταν από το ταβάνι ενώ ο παίχτης ήταν σταθερά προσδεμένος σ΄αυτήν. Εξάλλου τα γραφικά που διέθετε χαρακτηρίζονται ως πρωτόγονα. Το έτος 1987 ο Jaron Lanier επινόησε για τη συγκεκριμένη περιοχή έρευνας τον όρο “εικονική πραγματικότητα ή virtual reality” αναπτύσσοντας ταυτόχρονα μια σειρά συσκευών εικονικής πραγματικότητας όπως το Data Glove και το Eye Phone (καμία σχέση με το iPhone) προϊόντα που κυκλοφορούν στην αγορά την ίδια χρονιά έναντι όμως μεγάλων χρηματικών ποσών.

Από το 1991 έως και το 1995 σημειώνονται προσπάθειες διάθεσης στο εμπόριο προϊόντων εικονικής πραγματικότητας από διάφορες εταιρίες όπως η Nintendo και η Sega χωρίς όμως να παρατηρείται ιδιαίτερη αγοραστική ανταπόκριση. Παρ’ όλο που η εικονική πραγματικότητα δεν σημειώνει μεγάλη επιτυχία σε επίπεδο πωλήσεων συνεχίζει να εξελίσσεται τεχνολογικά και να φαντάζει ως μια πολλά υποσχόμενη τεχνολογική επίτευξη στο πολύ κοντινό μέλλον, κάτι που επιβεβαιώνει και η κινηματογραφική επιτυχία της ταινίας “Matrix” ταινία που επέβαλε ως επικρατούσα τάση την εικονική πραγματικότητα. Πλέον τον 21ο αιώνα, λόγω της ραγδαίας εξέλιξης των γραφικών σε συνδυασμό με τις εύχρηστες συσκευές και του χαμηλού κόστους δίνεται το έναυσμα της δημιουργίας εφαρμογών και διάθεσης στην αγορά προϊόντων που θα ανταποκρίνονται στον όρο “εικονική πραγματικότητα” [1].

2.2 Κατηγορίες Βιντεοπαιχνιδιών

Τα βιντεοπαιχνίδια διακρίνονται σε δώδεκα κατηγορίες παιχνιδιών αναλόγως το περιεχόμενο τους. Οι κατηγορίες αυτές είναι παιχνίδια δράσης, Arcade Games, πάλης, αγώνων οδήγησης, παιχνίδια ρόλων, στρατηγικής, προσομοίωσης, γενικής διασκέδασης, μουσικής και ρυθμού, αφήγησης, ικανοτήτων και τύχης και αθλητισμού. Παρακάτω θα αναλυθεί πιο λεπτομερώς η κατηγορία του παιχνιδιού που κατασκευάσαμε, η οποία είναι ένα παιχνίδι δράσης.

2.2.1 Δράσης

Τα παιχνίδια δράσης (Εικόνα 1) είναι ένα είδος βιντεοπαιχνιδιού που δίνουν έμφαση στις φυσικές προκλήσεις συμπεριλαμβανομένου του συντονισμού χειρός ματιού και χρόνου αντίδρασης. Στην εν λόγω κατηγορία εντάσσονται και τα βιντεοπαιχνίδια βολών πρώτου προσώπου (first-person shooter ή FPS). Είναι σειρά βιντεοπαιχνιδιών που βασίζεται σε οπτική πρώτου προσώπου. Σε αυτά ο παίχτης βλέπει τον κόσμο του παιχνιδιού μέσα από τα μάτια του ήρωα. Για να συμβεί αυτό χρησιμοποιούνται μηχανές τρισδιάστατης απεικόνισης, ώστε να παρέχεται η ψευδαίσθηση του πραγματικού χώρου όπως στις εφαρμογές εικονικής πραγματικότητας. Στην κατηγορία αυτών των βιντεοπαιχνιδιών βασίστηκε και η υλοποίηση της παρούσας πτυχιακής εργασίας.

Εικόνα 1: Παιχνίδια Δράσης Αριστερά: Bloodborne, Δεξιά: Doom (2016)

2.3 Εικονική, Επαυξημένη, Μικτή Πραγματικότητα και Επαυξημένη Εικονικότητα

2.3.1 Εικονική Πραγματικότητα

Πολλές και ποικίλες ερμηνείες έχουν δοθεί τις τελευταίες δεκαετίες προκειμένου να αποδοθεί όσο το δυνατόν πιστότερα ο όρος εικονική πραγματικότητα. Ο όρος «εικονική πραγματικότητα» χρησιμοποιήθηκε για πρώτη φορά από τον Jaron Lanier το 1989, τον πατέρα της εικονικής πραγματικότητας που έδωσε τον εξής ορισμό «Ένα αλληλεπιδραστικό, τρισδιάστατο περιβάλλον φτιαγμένο από υπολογιστή στο οποίο μπορεί κάποιος να εμβυθιστεί ». Η εικονική πραγματικότητα χρησιμοποιεί ηλεκτρονικούς υπολογιστές για να δημιουργήσει και να προσομοιώσει υπαρκτά και μη περιβάλλοντα από τα οποία ο χρήστης έχει τη ψευδαίσθηση ότι περιβάλλεται και στα οποία μπορεί να κινηθεί ελεύθερα, αλληλεπιδρώντας παράλληλα με τα αντικείμενα που περιλαμβάνουν όπως θα έκανε και στον πραγματικό κόσμο [2].

Για να είναι επιτυχής η εμβύθιση του χρήστη στο περιβάλλον της εικονικής πραγματικότητας είναι σημαντικό να απομονωθεί ο ίδιος και οι αισθήσεις του από την υπαρκτή πραγματικότητα επικαλύπτοντας τα ερεθίσματα του αληθινού κόσμου με αντίστοιχα εικονικά, φτιαγμένα από το σύστημα της εικονικής πραγματικότητας. Είναι πρωταρχικής σημασίας ένα σύστημα εικονικής πραγματικότητας να παρέχει κατ' αρχήν στερεοσκοπική εικόνα και στερεοσκοπικό ήχο. Εάν το εικονικό περιβάλλον συμπεριφέρεται όπως και το πραγματικό τότε η όλη εμπειρία που θα αποκτήσει ο χρήστης μπορεί να είναι άκρως ρεαλιστική [3].

Παράλληλα πολλά βιβλία και ταινίες επιστημονικής φαντασίας δημιούργησαν χαρακτήρες «παγιδευμένους στην εικονική πραγματικότητα». Η πρώτη ταινία μαζικής αποδοχής του Χόλιγουντ ήταν το «Τρον» του έτους 1982 και οι πιο γνωστές με αναφορά την εικονική πραγματικότητα οι ταινίες των αδερφών Wachowskis, Matrix. Τέλος, δεν πρέπει να αγνοήσουμε ότι η συνεχής ανάπτυξη της τεχνολογίας δίνει τη δυνατότητα για αναβάθμιση της εμβύθισης μέσω της βελτίωσης των συσκευών αλλά και της δυνατότητας απόκτησής τους λόγω χαμηλού κόστους. Ήδη έχουν κυκλοφορήσει στην αγορά τα εμπορικά μοντέλα με πιο διαδεδομένα και δημοφιλή τα Oculus Rift, το Card Board της Google και το Gear VR της Samsung.

Εικόνα 2: Virtual Reality

2.3.2 Επαυξημένη Πραγματικότητα

Πρόκειται για μια τεχνολογία διαφορετική από την εικονική πραγματικότητα αν και έχει ορισμένα κοινά χαρακτηριστικά με αυτήν. Είναι ένα ταχέως αναπτυσσόμενο ερευνητικό πεδίο στο χώρο της εικονικής πραγματικότητας. Επαυξημένη πραγματικότητα είναι η σε πραγματικό χρόνο άμεση ή έμμεση θέαση ενός φυσικού, πραγματικού περιβάλλοντος, του οποίου τα στοιχεία επαυξάνονται από στοιχεία αναπαραγώμενα από συσκευές υπολογιστών, όπως ήχος, βίντεο, γραφικά ή δεδομένα τοποθεσίας. Ο όρος εισήχθη το 1992 από τον Τομ Κάουντελ. Παράδειγμα επαυξημένης πραγματικότητας ήταν η εφαρμογή των HUDs στα μαχητικά αεροσκάφη. Άλλες εφαρμογές της επαυξημένης πραγματικότητας σχεδιάζονται είτε για υπολογιστές με κάποιου είδους Head Mount Display είτε για συσκευές χειρός όπως τα κινητά τηλέφωνα (Εικόνα 3), τάμπλετ ή προσωπικούς ψηφιακούς βοηθούς (PDAs) [4][5].

Εικόνα 3: Επαυξημένη πραγματικότητα (Pokemon 2016)

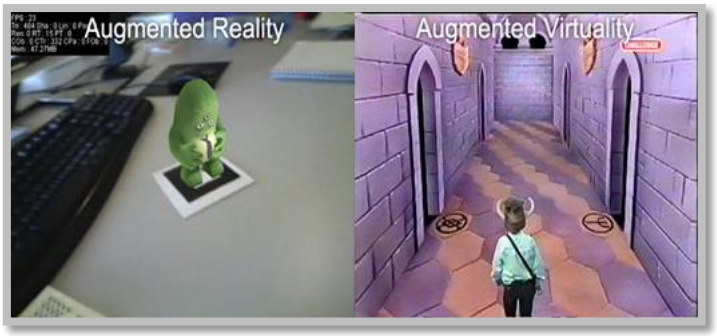
2.3.3 Μικτή Πραγματικότητα

Αναφέρεται και ως υβριδική πραγματικότητα είναι η συγχώνευση πραγματικών και εικονικών κόσμων για την παραγωγή νέων περιβαλλόντων και οπτικοποιήσεων, όπου φυσικά και ψηφιακά αντικείμενα συνυπάρχουν και αλληλεπιδρούν μεταξύ τους σε πραγματικό χρόνο. Η μικτή πραγματικότητα συνδυάζει και συγχωνεύει τόσο την επαυξημένη πραγματικότητα όσο και την επαυξημένη εικονικότητα (Εικόνα 4) [6].

Εικόνα 4: Μικτή πραγματικότητα

2.3.4 Επαυξημένη Εικονικότητα

Επαυξημένη εικονικότητα είναι η συγχώνευση αντικειμένων του πραγματικού κόσμου σε εικονικά ή ψηφιακά περιβάλλοντα. Αναφέρεται δηλαδή, σε εικονικούς χώρους όπου άτομα και φυσικά αντικείμενα ενσωματώνονται δυναμικά επιτυγχάνοντας την αλληλεπίδραση με τον εικονικό κόσμο σε πραγματικό χρόνο. Μπορούμε να πούμε ότι η επαυξημένη εικονικότητα αποτελεί την αντίστροφη διαδικασία της επαυξημένης πραγματικότητας [7].



Εικόνα 5: Σύγκριση της επαυξημένης πραγματικότητας και της επαυξημένης εικονικότητας

Εικόνα 6: The Reality-Virtuality Continuum

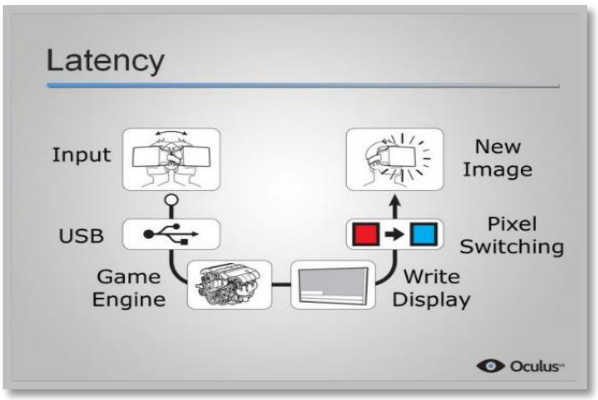
2.4 Εμβύθιση (Immersion)

Οι συσκευές εικονικής πραγματικότητας κάνουν χρήση πολλαπλών τεχνολογιών. Σε συνδυασμό με τις τεχνικές παραγωγής φωτορεαλιστικών γραφικών που βελτιώνονται συνεχώς, το αποτέλεσμα είναι κάτι παραπάνω από πειστικό. Τα VR Headsets στοχεύουν στον αποκλεισμό του χρήστη από το εξωτερικό του περιβάλλον. Η αντικατάσταση των ερεθισμάτων του πραγματικού κόσμου με αυτά του εικονικού περιβάλλοντος είναι ζωτικής σημασίας. Ως γνωστόν οι σημαντικότερες αισθήσεις για την αντίληψη είναι η όραση, η ακοή και η αφή. Η ακοή και η αφή είναι εύκολο να αντικατασταθούν μέσω των ακουστικών και των ειδικών περιφερειακών συσκευών. Η βασική πρόκληση και η μεγαλύτερη δυσκολία έγκειται στην όραση. Ο εγκέφαλος παρέχει την αίσθηση της ιδιοδεκτικότητας. Μέσω της αίσθησης αυτής μπορούμε να αντιληφθούμε πού βρίσκονται τα άκρα μας στο χώρο και να προσανατολιστούμε όταν το κεφάλι κινείται προς κάποια κατεύθυνση. Όταν χρησιμοποιούμε τα εικονικά άκρα, ο εγκέφαλος τείνει να συγχέει το εικονικό περιβάλλον με το πραγματικό. Πράγματι, σε δοκιμές VR οι χρήστες τραβούσαν αντανακλαστικά τα χέρια τους από μια εικονική φωτιά. Ένας ακριβής συνδυασμός χρωμάτων, σκίασης και εφέ φωτισμού είναι απαραίτητος για να δημιουργηθεί ένας ρεαλιστικός κόσμος. Επίσης, ο ήχος πρέπει να έχει κατεύθυνση και βάθος, και να περικλείει τον χρήστη (το λεγόμενο surround) ώστε να κάνει την ακουστική εμπειρία εξίσου πειστική. Καθώς εκ των πραγμάτων τα headset έχουν απλά στερεοφωνικά ακουστικά, θα πρέπει ο ήχος να αλλάζει με την κίνηση του κεφαλιού του χρήστη.

Εικόνα 7: Immersion (εμβύθιση)

Για να είναι αποτελεσματική η εμβύθιση, ο χρήστης θα πρέπει να μπορεί να διερευνήσει το εικονικό περιβάλλον και να είναι σε θέση να αλλάξει την οπτική του γωνία χωρίς δυσκολία. Για να επιτευχθεί κάτι τέτοιο πρέπει τα γραφικά να έχουν αρκετά υψηλή ανάλυση. Επίσης, ο ρυθμός ανανέωσης της εικόνας να είναι τουλάχιστον 30 καρέ ανά δευτερόλεπτο (Frames Per Second). Το όριο των 30 FPS έχει ξεπεραστεί ως πρότυπο πλέον, με τις συσκευές που κυκλοφορούν αυτή τη στιγμή να διαθέτουν πολύ πιο υψηλούς ρυθμούς ανανέωσης. Η χρονική καθυστέρηση είναι το μεγαλύτερο εμπόδιο για την κατασκευή ενός ρεαλιστικού εικονικού περιβάλλοντος. Αν οι εικόνες που φαίνονται μέσα από το HMD (Head Mount Display) δεν αποδοθούν (δεν γίνει rendering) αρκετά γρήγορα, η ψευδαίσθηση της πραγματικότητας χάνεται. Η χρονική καθυστέρηση μεταξύ των πράξεων του χρήστη και της αλλαγής στο εικονικό περιβάλλον ονομάζεται Latency (Εικόνα 8). Ο όρος αυτός μπορεί επίσης να χρησιμοποιηθεί για χρονική καθυστέρηση σε άλλες αισθητήριες εξόδους.

Μελέτες με προσομοιωτές πτήσης δείχνουν ότι οι άνθρωποι μπορούν να ανιχνεύσουν μια καθυστέρηση άνω των 50 χιλιοστών του δευτερολέπτου. Όταν ένας χρήστης ανιχνεύει το latency, αναγνωρίζει την ύπαρξη του τεχνητού περιβάλλοντος και καταστρέφεται η αίσθηση της εμβύθισης. Εάν θέλουμε ο εγκέφαλος να μην είναι σε θέση να ανιχνεύσει οποιαδήποτε βραδύτητα στις εικόνες που προβάλλονται, η καθυστέρηση πρέπει να ελαχιστοποιηθεί κάτω από τα 50 χιλιοστά του δευτερολέπτου. Για παράδειγμα, το Oculus Rift παρακάμπτει αυτό το θέμα με μια τεχνική που οι προγραμματιστές αποκαλούν "έξυπνη παρακολούθηση" (predictive tracking). Αυτή η τεχνολογία "μαντεύει" που μπορεί να κοιτάξει ο χρήστης και παίρνει ένα προβάδισμα κάνοντας rendering το περιβάλλον. Έτσι ελαχιστοποιείται ο χρόνος απόκρισης, σε λιγότερο από 30 χιλιοστά του δευτερολέπτου. Επίσης, για να πετύχουμε μεγαλύτερη εμβύθιση θα μπορούσαμε να αυξήσουμε το μέγεθος της οθόνης. Βέβαια, με την αύξηση του οπτικού πεδίου είναι απαραίτητο να βελτιωθεί και η ανάλυση της οθόνης ώστε να διατηρηθεί καθαρή η εικόνα. Έτσι, αυξάνεται η απαιτούμενη υπολογιστική ισχύς. Ενδέχεται βέβαια, να δημιουργηθούν προβλήματα στην απόκριση του συστήματος και στον ρυθμό ανανέωσης της εικόνας. Τέλος, προκειμένου να επιτευχθεί ο στόχος της αληθινής εμβύθισης, οι προγραμματιστές πρέπει στο μέλλον να καταλήξουν σε μεθόδους εισόδου που είναι πιο φυσικές για τους χρήστες. Όσο ο χρήστης αναγνωρίζει την συσκευή αλληλεπίδρασης, δεν αισθάνεται μέρος του ψηφιακού κόσμου.



Εικόνα 8: Η χρονική καθυστέρηση μεταξύ των πράξεων του χρήστη και της αλλαγής στο εικονικό περιβάλλον ονομάζεται Latency

2.5 Ρυθμός μετάδοσης εικόνων (Frame Rate)

Το Frame Rate (FPS ή frame per second) είναι η συχνότητα με την οποία εμφανίζονται διαδοχικές εικόνες, οι ονομαζόμενες frame σε μια κινούμενη οθόνη. Το Frame Rate έχει μονάδα μέτρησης τα Hertz. Χρονική ευαισθησία (temporal sensitivity) είναι η ελάχιστη χρονική διάρκεια που χρειάζεται ο άνθρωπος για να γίνει αντιληπτό ένα ερέθισμα. Η χρονική ευαισθησία και η ανάλυση της ανθρώπινης όρασης ποικίλει ανάλογα με τον τύπο και τα χαρακτηριστικά των οπτικών ερεθισμάτων και είναι διαφορετική σε κάθε άνθρωπο. Το ανθρώπινο οπτικό σύστημα μπορεί να επεξεργαστεί 10 έως 12 εικόνες ανά δευτερόλεπτο και να τις αντιληφθεί ως μεμονωμένες ενώ περισσότερες από αυτές γίνονται αντιληπτές ως κίνηση. Η δέσμη φωτός που παράγεται από μια ηλεκτρονική συσκευή όπως η οθόνη, θεωρείται σταθερή όταν ο ρυθμός των Frames κυμαίνεται μεταξύ των 50 έως 90 Hertz. Στα γραφικά των ηλεκτρονικών υπολογιστών ακόμα και στην εικονική πραγματικότητα το frame rate είναι εξέχουσας σημασίας καθώς καθορίζει εάν τα διαδοχικά frames θα γίνουν αντιληπτά από τον εγκέφαλο ως μεμονωμένες εικόνες ή εάν θα προσφέρουν την ψευδαίσθηση της κίνησης [8].

Οι συσκευές της εικονικής πραγματικότητας απαιτούν συνήθως πολύ υψηλό frame rate ώστε να δημιουργήσουν την ευχάριστη ψευδαίσθηση της ομαλής κίνησης. Η επίτευξη υψηλού βαθμού frame rate είναι δύσκολο εγχείρημα καθώς απαιτείται hardware με σημαντική επεξεργαστική ισχύ και προσεγμένο προγραμματισμό εφαρμογών .

Ο ρυθμός ανανέωσης (refresh rate), είναι ο αριθμός των frame per second που μπορεί να εμφανίσει μια οθόνη, δηλαδή πόσες φορές το δευτερόλεπτο μια οθόνη ανανεώνει την εικόνα της.

2.6 Τεχνητή Νοημοσύνη

Τεχνητή Νοημοσύνη είναι η νοημοσύνη που εμφανίζεται από τις μηχανές και η ικανότητα τους να ενεργούν ανάλογα με τα ερεθίσματα του περιβάλλοντος σε αντίθεση με τη φυσική νοημοσύνη που εμφανίζεται σε ανθρώπους και άλλα ζώα. Είναι η μελέτη του τρόπου παραγωγής μηχανών οι οποίες κατέχουν μερικές από τις ιδιότητες που έχει ο ανθρώπινος

νους, όπως η ικανότητα κατανόησης της γλώσσας, η αναγνώριση εικόνων, η επίλυση προβλημάτων και η εκμάθηση[36][37]. Στην επιστήμη των υπολογιστών η έρευνα της τεχνητής νοημοσύνης (AI) ορίζεται ως η μελέτη «νοημόνων πρακτόρων» ή «αυτόνομων πρακτόρων»: κάθε συσκευή που αντιλαμβάνεται το περιβάλλον της και λαμβάνει δράσεις που μεγιστοποιούν την πιθανότητα επιτυχίας σε κάποιο στόχο[38]. Στο συγκεκριμένο παιχνίδι, η συμπεριφορά των διαφόρων χαρακτήρων και εχθρών, έχει δημιουργηθεί βάσει των αρχών της τεχνητής νοημοσύνης.

2.8 Τα κύρια μέρη του Virtual Reality Headset

Είναι συσκευή απεικόνισης γυαλιών ή πλήρες κράνος (Εικόνα 9) που τοποθετείται στο κεφάλι του χρήστη. Η συσκευή αυτή μπορεί να διαθέτει μια κοινή οθόνη και για τα δύο μάτια ή δύο ξεχωριστές μία για κάθε οφθαλμό, προσφέροντας στη δεύτερη περίπτωση αποτελεσματικότερα την αίσθηση των τριών διαστάσεων[9].

Εικόνα 9: Head Mount Display

Η οθόνη είναι το πιο σημαντικό μέρος της συσκευής. Τα περισσότερα headset έχουν δύο οθόνες. Όμως η εικονική πραγματικότητα είναι δυνατή και με μία ενιαία οθόνη, με τους κατάλληλους φακούς. Η κάθε οθόνη ή ο κάθε φακός τροφοδοτεί το αντίστοιχό μάτι ξεχωριστά με εικόνες από διαφορετική οπτική γωνία, όπως λειτουργεί και η όρασή μας. Είναι η βασική αρχή στην οποία στηρίχθηκαν και τα στερεοσκόπια, για τη δημιουργία τρισδιάστατων εικόνων.

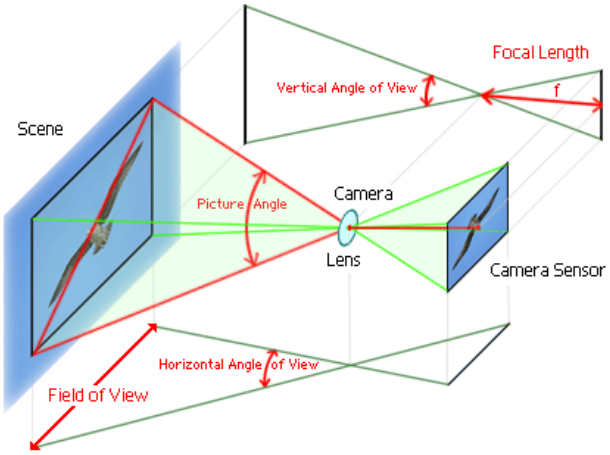
Μπορούμε να καταλάβουμε την ιδέα πίσω από αυτή την υλοποίηση παρατηρώντας οποιοδήποτε αντικείμενο στο φυσικό κόσμο με το ένα μάτι κλειστό και εναλλάξ. Τα δύο μεγαλύτερα ζητήματα για την οθόνη, είναι ο συντελεστής πλήρωσης, ή αλλιώς παράγοντας γεμίσματος (Fill Factor) και η μεταγωγή του. Το γέμισμα της οθόνης είναι ένας συνδυασμός πολλών παραγόντων. Υπάρχουν οθόνες που έχουν πολύ λίγο χώρο μεταξύ των εικονοστοιχείων (pixels) και άλλες που έχουν μια σημαντική ποσότητα μαύρου χώρου. Οι προβολείς DLP είναι ένα παράδειγμα τύπου οθόνης που έχει γενικά πολύ υψηλό συντελεστή πλήρωσης με αμελητέο διάστημα μεταξύ των εικονοστοιχείων. Οι οθόνες LCD, από την άλλη πλευρά, συχνά πάσχουν από το χαμηλό συντελεστή πλήρωσης. Ένα μεγάλο τμήμα της οθόνης καταλαμβάνεται από τον μαύρο χώρο γύρω από τα pixels. Σε πολύ μεγάλες οθόνες αυτό είναι πρόβλημα καθώς τις κάνει να φαίνονται σαν να επικαλύπτεται η εικόνα από ένα λεπτό μαύρο πλέγμα. Συχνά αυτό το φαινόμενο αναφέρεται ως "screen door effect". Ένας

πολύ υψηλότερος συντελεστής πλήρωσης σημαίνει ότι μειώνεται σημαντικά η επίδραση του screen door effect, με αποτέλεσμα την καλύτερη εμφάνιση της εικόνας.

Το δεύτερο σημαντικό θέμα είναι ο χρόνος μεταγωγής. Η διάρκεια δηλαδή του χρόνου που απαιτείται ώστε ένα pixel να μεταβεί από το ένα χρώμα στο άλλο. Η εναλλαγή χρωμάτων γίνεται πολύ γρήγορα, γεγονός θετικό, αφού μειώνεται η καθυστέρηση μεταξύ της κίνησης και της εικόνας. Ωστόσο, το πρόβλημα εντοπίζεται ότι χρειάζεται ένα πολύ μεγάλο χρονικό διάστημα για να ενεργοποιηθούν τα πλήρη χρώματα. Αυτό μερικές φορές οδηγεί σε θάμπωμα της κίνησης όπως αλλάζουν τα frames, ειδικά κατά τη διάρκεια των γρήγορων κινήσεων του κεφαλιού. Αυτό που είναι επίσης σημαντικό για την οθόνη αφορά στην πυκνότητα των pixels, δηλαδή στο συνδυασμό οπτικού πεδίου και ανάλυσης. Για παράδειγμα, το οπτικό μας πεδίο είναι πιο περιορισμένο όταν βλέπουμε μια οθόνη από απόσταση, απ' ότι όταν είναι μπροστά μας. Έτσι αντιστοιχούν περισσότερα pixels σε λιγότερες μοίρες και η καθαρότητα της εικόνας είναι μεγάλη.

Στην περίπτωση του VR Headset ο λόγος pixels ανά μοίρα είναι μικρότερος. Η ανάλυση της οθόνης διανέμεται σε μεγαλύτερο οπτικό πεδίο, πράγμα που μπορεί να παρατηρήσει ο χρήστης από κοντά. Το ανθρώπινο μάτι μπορεί να αναγνωρίσει και να καταλάβει την διαφορά μέχρι τα 60 pixels ανά μοίρα. Τα VR Headsets απέχουν ακόμα πολύ από αυτόν τον αριθμό. Οι κατασκευαστές έχουν επιλέξει τις κατάλληλες διαστάσεις ώστε η πυκνότητα των pixels και ο ρυθμός ανανέωσης της εικόνας να διατηρούνται σε καλό επίπεδο. Η οθόνη από μόνη της δεν είναι ικανή να ξεγελάσει τελείως το μάτι, γι' αυτό χρησιμοποιούνται επιπρόσθετα και φακοί. Οι φακοί δημιουργούν την αίσθηση του βάθους, που σημαίνει πως οι εικόνες καλύπτουν τελείως το οπτικό πεδίο του χρήστη. Σε κάποιες περιπτώσεις, οι φακοί μπορούν να προσαρμοστούν για να ταιριάζουν με την απόσταση των ματιών, που διαφέρει από άτομο σε άτομο

Το πεδίο προβολής (FOV) είναι η παρατηρίσιμη περιοχή που μπορεί να δει κάποιος μέσα από τα μάτια του ή μέσω κάποιας συσκευής και μετριέται. Τυπικά μετριέται σαν γωνία ανεξάρτητα αν αυτή η γωνία είναι οριζόντια, κάθετη ή διαγώνια. Όσον αφορά τα video games, αναφερόμαστε στην παρατηρίσιμη περιοχή που μπορεί να δει κάποιος μέσα στον ψηφιακό κόσμο[49]. Το οπτικό πεδίο του ανθρώπου κυμαίνεται περίπου στις 130-135 μοίρες κάθετα και 200 μοίρες οριζόντια. Οπότε καταλαβαίνουμε ότι η χρήση οθόνης που προσφέρει μεγαλύτερο οπτικό πεδίο εκτός από άσκοπη θα ήταν και μεγάλη σπατάλη πόρων. Τα περισσότερα VR Headsets που κυκλοφορούν προσφέρουν πεδίο οράσεως γύρω στις 100 με 110 μοίρες. Η χρήση των φακών εξισορροπεί το οπτικό αποτέλεσμα.



Εικόνα 10: Field of View

Τέλος, για να εντοπίζουμε με ακρίβεια την κίνηση μέσα στον χώρο χρησιμοποιούνται ενσωματωμένοι αισθητήρες παρακολούθησης κίνησης.

Οι βασικοί αισθητήρες είναι τρεις:

- Γυροσκόπιο, που εξασφαλίζει την κίνηση στους βασικούς άξονες κατεύθυνσης και περιστροφής.
- Μαγνητόμετρο, που μετράει την δύναμη του μαγνητικού πεδίου της γης.
- Επιταχυνσιόμετρο, που μετράει πόσο γρήγορα κινείται η συσκευή στον χώρο.

Ορισμένες συσκευές μάλιστα, βελτιώνουν ακόμα περισσότερο την ακρίβειά τους με τη χρήση υπολογιστικής όρασης. Δηλαδή το VR Headset παρακολουθείται από εξωτερικές κάμερες. Για να αναγνωρίζει ο υπολογιστής την συσκευή, χρησιμοποιείται υπέρυθρο φως ή ανακλαστήρες, που λειτουργούν ως δείκτες. Έτσι, υπολογίζεται η απόλυτη θέση του χρήστη στην εικονική πραγματικότητα. Τέλος, οι συσκευές εισόδου είναι επίσης ένα σημαντικό εξάρτημα για τα συστήματα εικονικής πραγματικότητας. Επί του παρόντος, οι συσκευές αυτές κυμαίνονται από χειριστήρια μέχρι ηλεκτρονικά γάντια μέχρι και αναγνώριση φωνής.

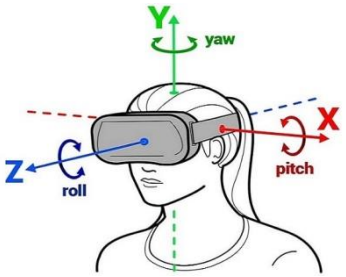
Μέχρι στιγμής, δεν υπάρχει κάποιο στάνταρ σύστημα ελέγχου. Οι επιστήμονες και οι μηχανικοί συνεχώς διερευνούν νέους τρόπους για να κάνουν την είσοδο του χρήστη όσο το δυνατόν φυσικότερη, και επομένως να αυξηθεί η αίσθηση της τηλεπαρουσίας.

2.9 Head Tracking, Positional Tracking, Eye Tracking, Hand Tracking

Πολλά σύγχρονα HMD κυκλοφορούν με χαρακτηριστικά όπως positional tracking, head tracking, hand tracking ακόμα και eye tracking. Παρακάτω θα περιγράψουμε αναλυτικά αυτές τις τεχνολογίες.

2.9.1 Head Tracking

Από τα σημαντικότερα χαρακτηριστικά ενός Head Mount Display είναι το Head Tracking. Επιτρέπει στο χρήστη να κοιτάζει ελεύθερα προς κάθε κατεύθυνση μέσα στον εικονικό κόσμο κινώντας απλά το κεφάλι του όπως θα έκανε και στον πραγματικό κόσμο χωρίς τη χρήση επιπλέον εξαρτημάτων, κερδίζοντας έτσι τη ψευδαίσθηση της παρουσίας του στο εικονικό περιβάλλον. Το σύστημα αυτό αποτελείται από εξαρτήματα όπως το επιταχυνσιόμετρο, το γυροσκόπιο και το μαγνητόμετρο, τα οποία χρησιμοποιούνται επίσης και στα σημερινά κινητά τηλέφωνα (smartphones). Επιπλέον μπορεί να χρησιμοποιηθεί και ένας σταθερός εξωτερικός αισθητήρας. Στην παρούσα εργασία, εκμεταλλευτήκαμε αυτή τη λειτουργία του HMD ώστε ο χρήστης να έχει τη δυνατότητα να εκτελεί περιστροφικές κινήσεις μέσα στον εικονικό κόσμο μόνο με τις κινήσεις της κεφαλής του.

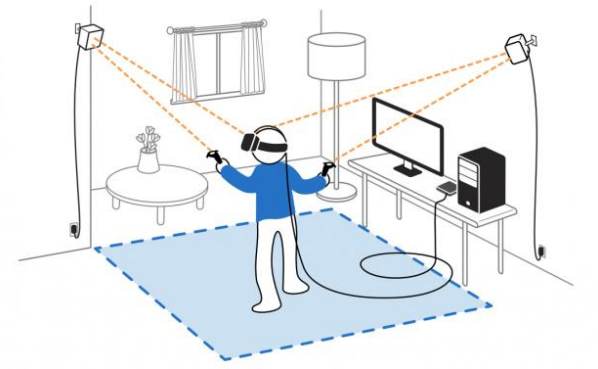


Εικόνα 11: Head Tracking

2.9.2 Positional Tracking

Είναι μια τεχνολογία που επιτρέπει σε μια συσκευή να εκτιμήσει τη θέση της σε σχέση με το περιβάλλον γύρω από αυτήν. Χρησιμοποιεί έναν συνδυασμό υλικού (hardware) και λογισμικού (software) ώστε να επιτύχει την ανίχνευση της θέσης του. Το Positional Tracking αποτελεί μια βασική τεχνολογία για την εικονική πραγματικότητα (VR), καθιστώντας δυνατή την παρακολούθηση της κίνησης με έξι βαθμούς ελευθερίας (Degree-Of-Freedom-6DOF[10][11].

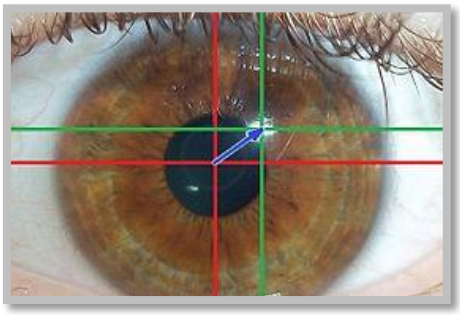
Η τεχνολογία αυτή επιτρέπει στο Head Mount Display ή τον υπολογιστή να γνωρίζει τη θέση του χρήστη στο πραγματικό χώρο και χρόνο, γεγονός που είναι πολύ χρήσιμο για ορισμένες εφαρμογές και παιχνίδια, καθώς ο χρήστης θα έχει τη δυνατότητα να κινηθεί στον εικονικό κόσμο με την κίνηση ή το περπάτημα που πραγματοποιεί στο πραγματικό περιβάλλον. Το Positional Tracking επιτυγχάνεται με τη συνεχή μέτρηση της απόστασης και των γωνιών, μεταξύ των εγκατεστημένων αισθητήρων εντός του Head Mount Display σε σχέση με ένα ή και περισσότερους σταθμούς βάσης (εξωτερικές κάμερες) που βρίσκονται στο χώρο. Στη συνέχεια μαθηματικοί υπολογισμοί μεταφράζουν τις μετρήσεις αυτές σε ακριβείς θέσεις του χρήστη μέσα στο εικονικό περιβάλλον. Με αυτήν την επέκταση του HMD επιτυγχάνουμε την ορθή κίνηση του χρήστη προς κάθε κατεύθυνση μέσα στο εικονικό περιβάλλον.



Εικόνα 12: Positional Tracking

2.9.3 Eye Tracking

Το Eye Tracking είναι η διαδικασία μέτρησης είτε του σημείου του βλέμματος είτε της κίνησης του ματιού σε σχέση με το κεφάλι. Είναι μια συσκευή μέτρησης της θέσης της κίνησης των ματιών. Υπάρχουν διάφοροι μέθοδοι μέτρησης, οι πιο δημοφιλείς χρησιμοποιούν video εικόνες από τις οποίες αναγνωρίζεται η θέση και η κίνηση των οφθαλμών [12].



Εικόνα 13: Eye Tracking. Κέντρο Ίριδας (κόκκινο), Ανάκλαση Κερατοειδούς (πράσινο) και Διάνυσμα Εξόδου (μπλε)

2.9.4 Hand Tracking

Είναι διαδικασία συνεχούς καταγραφής της θέσης και κίνησης των χεριών και δακτύλων ενός χρήστη στον πραγματικό χώρο. Σύμφωνα με αυτή τη τεχνολογία δίνεται η δυνατότητα στο χρήστη να βλέπει ένα τρισδιάστατο αντίγραφο των χεριών του στον εικονικό κόσμο ανάλογα με τη θέση και την περιστροφή τους στον πραγματικό. Θα μπορούσε να πιάσει δηλαδή ή να μετακινήσει εικονικά αντικείμενα, ή να κάνει μια χειρονομία ενεργοποιώντας ένα γεγονός στο εικονικό περιβάλλον. Οι δύο κύριες τεχνικές με τις οποίες επιτυγχάνεται το Hand Tracking είναι οι ακόλουθες. Στην πρώτη τεχνική ο χρήστης είναι αναγκαίο να κρατά στα χέρια του δύο ειδικά χειριστήρια που λειτουργούν ως συσκευές εντοπισμού θέσης για το κάθε χέρι ξεχωριστά, ενώ παράλληλα γίνεται μέτρηση μεταξύ των χειριστηρίων και των σταθμών βάσης του χώρου, παρόμοια διαδικασία με αυτή του Positional Tracking. Στη συνέχεια, η θέση κάθε χεριού μπορεί να υπολογιστεί σε σχέση με το κεφάλι του χρήστη και το δωμάτιο, ώστε να προβάλλεται στον εικονικό κόσμο. Η δεύτερη τεχνική αφορά στη λήψη εικόνων και την επεξεργασία αυτών. Ένας ή περισσότεροι αισθητήρες κάμερας είναι τοποθετημένοι σε κατάλληλο σημείο έχοντας επαφή με τα χέρια

του παίχτη καταγράφοντας τις κινήσεις τους σε αρκετά frames ανά δευτερόλεπτο. Στη συνέχεια, προηγμένοι μαθηματικοί αλγόριθμοι και αλγόριθμοι επεξεργασίας εικόνας αναλύουν κάθε εικόνα-frame παράγοντας μια τρισδιάστατη αναπαράσταση των χεριών στο εικονικό περιβάλλον όπως τα βλέπει και τα αντιλαμβάνεται η συσκευή. Το hand tracking αποτελεί θεμελιώδη πυλώνα της συγκεκριμένης εφαρμογής, μιας και ο χειρισμός της βασίζεται στον εντοπισμό χειρός, αποκλείοντας έτσι τις κλασικές συσκευές εισόδου (πληκτρολόγιο, ποντική) ώστε η εμπειρία της εικονικής πραγματικότητας να είναι όσο γίνεται πιο αληθινή

Εικόνα 14: Hand Tracking

Κεφάλαιο 3ο Γραφικά Υπολογιστών και Μηχανές Παιχνιδιών

3.1 Γραφικά Υπολογιστών και Μηχανές Παιχνιδιών

Με τον όρο Γραφικά Υπολογιστών εννοούμε την εικονική-οπτική αναπαράσταση δεδομένων τα οποία εμφανίζονται σε μια οθόνη από έναν υπολογιστή. Μπορεί να είναι απλές εικόνες ή ταινίες που δημιουργούνται με τη βοήθεια ειδικών προγραμμάτων υπολογιστή (software) και υλικού μηχανημάτων υπολογιστή (hardware) τα οποία είναι κατάλληλα για τη δημιουργία και απόδοση γραφικών. Επίσης, χρησιμοποιούνται για την επεξεργασία εικονικών δεδομένων (Data image) τα οποία λαμβάνονται από τον πραγματικό κόσμο. Η ανάπτυξη των γραφικών υπολογιστών είχε σημαντικό αντίκτυπο στα μέσα ενημέρωσης και έχουν φέρει επανάσταση σε κινούμενα γραφικά, ταινίες, διαφημίσεις, βιντεοπαιχνίδια και σχεδιασμό γραφικών γενικά [17].

3.2 Τρισδιάστατα Γραφικά Υπολογιστών (3D computer graphics)

Η συγκεκριμένη εργασία βασίζεται εξολοκλήρου σε 3D γραφικά. Με τον όρο Three-dimension computer graphics εννοούμε τα γραφικά που χρησιμοποιούν μια τρισδιάστατη αναπαράσταση των γεωμετρικών δεδομένων (καρτεσιανό) που είναι αποθηκευμένα σε ένα υπολογιστή με σκοπό αυτός να εκτελεί υπολογισμούς και να αποδίδει εικόνες δύο διαστάσεων. Η δημιουργία των 3D γραφικών χωρίζεται σε τρεις διαφορετικές φάσεις:

1. 3D modeling, διαδικασία κατά την οποία ένα μοντέλο υπολογιστή παίρνει το σχήμα του
2. Animation, δηλαδή κινούμενα σχέδια και διάταξη, διαδικασία κατά την οποία ορίζεται η θέση και η κίνηση ενός αντικειμένου μέσα σε μια σκηνή (scene)
3. 3D rendering, οι υπολογισμοί του υπολογιστή που πραγματοποιούνται με βάση την θέση του φωτός, τους τύπους των επιφανειών και άλλων ιδιοτήτων ώστε να δημιουργηθούν οι δισδιάστατες εικόνες

Πάνω σε αυτές τις τρεις φάσεις είναι βασισμένη ολόκληρη η ιδέα της διπλωματικής εργασίας αλλά και γενικότερα οποιασδήποτε άλλης εφαρμογής που σχετίζεται με την απόδοση γραφικών σε κάποια οθόνη. Οι παραπάνω διαδικασίες θα αναλυθούν στη συνέχεια εκτενέστερα [18].

3.2.1 Τρισδιάστατη Μοντελοποίηση (3D modeling)

Στον κόσμο των 3D γραφικών, 3D modeling είναι η διαδικασία της δημιουργίας μιας μαθηματικής αναπαράστασης της επιφάνειας ενός αντικειμένου μέσω ειδικών προγραμμάτων. Το προϊόν αυτής της διαδικασίας ονομάζεται 3D model (3D μοντέλο). Μπορεί να αποδοθεί σαν μια δισδιάστατη εικόνα μέσω της διαδικασίας που ονομάζεται 3D rendering ή να χρησιμοποιηθεί σε προσομοιώσεις υπολογιστών φυσικών φαινομένων. Τα 3D μοντέλα μπορούν να δημιουργηθούν αυτόματα ή με το χέρι. Η χειροκίνητη διαδικασία δημιουργίας 3D μοντέλων είναι παρόμοια τέχνη με την γλυπτική. Σχεδόν όλα τα 3D μοντέλα χωρίζονται σε δύο κατηγορίες:

- Solid, αυτά τα μοντέλα ορίζουν την πυκνότητα των αντικειμένων που αναπαριστούν. Τα συμπαγή (solid) μοντέλα χρησιμοποιούνται κυρίως για μηχανικές και ιατρικές προσομοιώσεις.
- Shell/boundary, αυτά τα μοντέλα αναπαριστούν την επιφάνεια, δηλαδή τα όρια ενός αντικειμένου και όχι την πυκνότητα. Σχεδόν όλα τα οπτικά μοντέλα που χρησιμοποιούνται στα βιντεοπαιχνίδια είναι shell models.

Υπάρχουν διάφορες τεχνικές δημιουργίας 3D μοντέλων. Οι τρεις πιο δημοφιλείς είναι οι εξής:

- Polygonal modeling: διάφορα σημεία στον τρισδιάστατο χώρο τα οποία ονομάζονται κορυφές ενώνονται με ευθείες γραμμές και σχηματίζουν ένα πλέγμα πολυγώνων. Η συντριπτική πλειοψηφία των μοντέλων είναι πολυγωνικά μοντέλα επειδή είναι ευέλικτα και οι υπολογιστές μπορούν να τα αποδώσουν με ταχύτητα. Παρ' όλα αυτά οι καμπύλες μπορούν να αποδοθούν μόνο χρησιμοποιώντας πολλά πολύγωνα.
- Curve modeling: οι επιφάνειες ορίζονται από καμπύλες οι οποίες επηρεάζονται από σημεία με ειδικό βάρος. Η καμπύλη ακολουθεί αυτά τα σημεία. Η αύξηση του ειδικού βάρους σε κάποιο σημείο θα έλξει την καμπύλη πιο κοντά του.
- Digital sculpting: πραγματοποιείται με τη χρήση ειδικού λογισμικού που προσφέρει εργαλεία για να ωθήσει, να τραβήξει, να εξομαλύνει, να πιάσει ή να χειριστεί ένα ψηφιακό αντικείμενο σαν να ήταν κατασκευασμένο από μια ουσία πραγματικής ζωής, όπως ο πηλός.

3.2.2 Animation-3D Animation

Animation είναι ένας τύπος κινούμενης εικόνας που έχει δημιουργηθεί από έναν υπολογιστή με σκοπό την απόδοση κινούμενων σκηνών. Σε σύγκριση με το 2D animation το 3D αποδίδει πολύ περισσότερο βάθος και φαίνεται πιο ρεαλιστικό. Μια κινούμενη σκηνή αρχίζει με μια εικόνα ενός frame. Το επόμενο frame είναι μια εικόνα ελαφρώς πιο προχωρημένη μερικά χιλιοστά του δευτερολέπτου στο χρόνο. Έτσι, εκατοντάδες ή χιλιάδες από αυτά τα frames δίνουν την ψευδαίσθηση μιας κινούμενης εικόνας [19].

3.2.3 Rendering-3D Rendering

Ένα αρχείο σκηνής περιλαμβάνει αντικείμενα σε μια αυστηρά καθορισμένη γλώσσα ή δομές δεδομένων. Δηλαδή, περιέχει γεωμετρία, textures, πληροφορίες σκίασης και φωτισμού κ.α. ως περιγραφή της εικονικής σκηνής. Τα δεδομένα που εμπεριέχονται στο αρχείο σκηνής διαβιβάζονται σε ένα πρόγραμμα απόδοσης (rendering) ώστε να τα επεξεργαστεί και στη συνέχεια τα εξάγει σαν μια εικόνα ή ένα αρχείο εικόνων. Αν και οι τεχνικές λεπτομέρειες των μεθόδων απόδοσης γραφικών ποικίλουν, υπάρχουν κάποια βασικά βήματα τα οποία πρέπει να πραγματοποιηθούν ώστε να αποδοθεί μια τρισδιάστατη σκηνή σε μια δισδιάστατη οθόνη. Αυτή η ακολουθία βημάτων ονομάζεται Graphic Pipeline και σχετίζεται άμεσα με συσκευές απόδοσης γραφικών όπως η κάρτα γραφικών GPU. Η GPU είναι μια συσκευή σχεδιασμένη έτσι ώστε να βοηθήσει τον επεξεργαστή CPU να εκτελέσει σύνθετους υπολογισμούς απόδοσης γραφικών. Μια εικόνα μπορεί να γίνει πιο κατανοητή από ένα σύνολό ορατών χαρακτηριστικών. Η συνεχής έρευνα και ανάπτυξη στο πεδίο της

απόδοσης γραφικών ενισχύεται από την ανεύρεση νέων μεθόδων ρεαλιστικότερης αναπαράστασης των χαρακτηριστικών της εικόνας. Τέτοιες μέθοδοι είναι οι εξής:

- Shading: μέθοδος με την οποία το χρώμα και η φωτεινότητα μιας επιφάνειας ποικίλει ανάλογα με τον φωτισμό
- Texture mapping: μέθοδος με την οποία αποδίδονται λεπτομέρειες στην επιφάνεια
- Bump mapping: είναι μια μέθοδος προσομοίωσης πρόσκρουσης στην επιφάνεια
- Forging/participating medium: μέθοδος που προσομοιώνει τη μεταβολή του φωτός όταν διέρχεται από θολή ατμόσφαιρα ή τον αέρα
- Shallows: προσομοίωση της παρεμπόδισης του φωτός
- Soft shallows: δημιουργία ενός ιδιαίτερου σκότους που παράγεται από "σκοτεινές" πηγές φωτός
- Reflection: μέθοδος αναπαράστασης αντανάκλασης «σαν καθρέφτη» ή πολύ αστραφτερής αντανάκλασης
- Transparency ή Opacity: διαπεραστική προσπέλαση φωτός μέσα από στερεά αντικείμενα
- Translucency: μέθοδος εξαιρετικά διάχυτης μετάδοσης του φωτός μέσα από στερεά αντικείμενα
- Refraction: μέθοδος κάμψης του φωτός που σχετίζεται με τη διαφάνεια
- Diffraction: μέθοδος κάμψης, εξάπλωσης και παρεμβολής του φωτός που διέρχεται από ένα αντικείμενο
- Indirect illumination: μέθοδος κατά την οποία οι επιφάνειες φωτίζονται από το φως που αντανακλάται από άλλες επιφάνειες και όχι απευθείας από μια πηγή φωτός (γνωστό και ως Global illumination)
- Caustics: μέθοδος αντανάκλασης του φωτός ενός γυαλιστερού αντικειμένου ή εστίαση του φωτός μέσα από ένα διαφανές αντικείμενο για τη δημιουργία φωτεινών σημείων πάνω σε ένα αντικείμενο.

3.3 Απόδοση Επιφάνειας ΜοντέλουTextures, Shader, και Materials

Shader (Σκιαστές)

Ένας shader θεωρείται ένα πρόγραμμα που εκτελείται κατά τη διαδικασία rendering και έχει σκοπό την απόδοση χρώματος μιας επιφάνειας στην τρισδιάστατη σκηνή με βάση τη γωνία και την απόστασή της από τις διάφορες πηγές φωτός ώστε να δημιουργηθεί ένα φωτορεαλιστικό αποτέλεσμα. Η σκίαση επίσης, εξαρτάται από τον ίδιο τον φωτισμό που υπάρχει στη σκηνή. Έτσι, οι shaders πιο συχνά χρησιμοποιούνται για τη διαχείριση των διαφόρων εφέ φωτισμού και σκίασης. Πέρα από αυτά, οι σκιαστές (shaders) έχουν και πιο πολύπλοκες χρήσεις όπως η αλλαγή της φωτεινότητας, της απόχρωσης ή της αντίθεσης μιας εικόνας προκαλώντας θόλωση (blur), παραμόρφωση (distortion), φωτεινή άνθηση (light bloom) και άλλα. Οι σκιαστές δηλαδή είναι μικρά script που περιέχουν μαθηματικούς

υπολογισμούς και αλγόριθμους για τον υπολογισμό του χρώματος κάθε pixel μιας επιφάνειας βασισμένοι στο φως που υπάρχει στη σκηνή και στη διαμόρφωση του υλικού (material) [30].

Texture

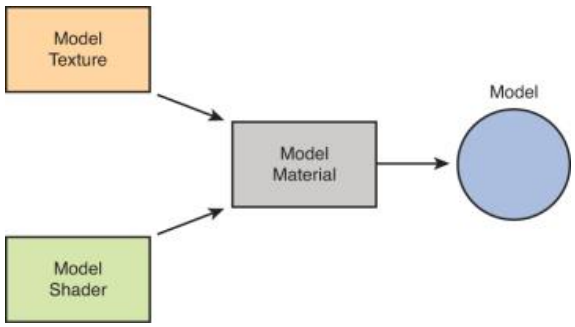
Texture (υφή) στα 3D γραφικά είναι μία δισδιάστατη εικόνα που εφαρμόζεται στην επιφάνεια ενός 3D μοντέλου. Είναι δηλαδή bitmap εικόνες που καθορίζουν το χρώμα και τη φωτεινότητα του αντικειμένου, όπως επίσης και το πόσο διαφανές και αντανακλαστικό είναι ένα αντικείμενο. Μόλις δημιουργηθεί ένα texture μπορεί να τυλιχτεί γύρω από οποιοδήποτε 3D gameobject [30].

Material

Το Material δεν είναι περισσότερο από ένα δοχείο μέσα για τους shaders και τα texture το οποίο μπορεί να εφαρμοστεί σε ένα μοντέλο.

Ένα material ορίζει έναν συγκεκριμένο shader που θα χρησιμοποιήσει και αυτός με τη σειρά του καθορίζει ποιες επιλογές του material είναι διαθέσιμες. Ένας shader μπορεί να προσδιορίζει μια ή περισσότερες texture μεταβλητές. Η Unity στη συνέχεια δίνει τη δυνατότητα στο χρήστη να αναθέσει στις texture μεταβλητές τα δικά του προσωπικά texture.

Οι υφές και οι σκιαστές εφαρμόζονται στα materials και τα materials εμφανίζονται στα μοντέλα. Με αυτό τον τρόπο η εμφάνιση ενός μοντέλου μπορεί να τροποποιηθεί γρήγορα και απλά [30].



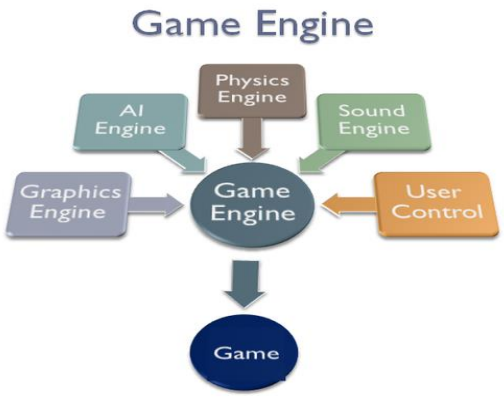
Εικόνα 15:Material, Texture, Shader

3.4 Μηχανές Παιχνιδιών

Σε αυτή την ενότητα του κεφαλαίου 2 θα αναλύσουμε τη βασική δομή των μηχανών παιχνιδιών. Υποκατηγορίες των μηχανών παιχνιδιού είναι η Rendering Engine, μηχανή φυσικής, μηχανή ήχου, σύστημα γραφικής διεπαφής χρήστη, τεχνητή νοημοσύνη και γλώσσα σεναρίων.

Μια Game Engine είναι ένα software framework (πλατφόρμα ανάπτυξης λογισμικού) σχεδιασμένη για τη δημιουργία και ανάπτυξη βιντεοπαιχνιδιών και γραφικών. Οι προγραμματιστές τις χρησιμοποιούν για να δημιουργούν παιχνίδια για κονσόλες, φορητές συσκευές και προσωπικούς υπολογιστές. Η βασική λειτουργικότητα που παρέχεται από τη

μηχανή παιχνιδιών οφείλεται στη μηχανή απόδοσης γραφικών 2D ή 3D (render) στη μηχανή φυσικής, στο μηχανισμό ανίχνευσης σύγκρουσης, στο μηχανισμό ήχου, στα κινούμενα σχέδια (animation), στην τεχνητή νοημοσύνη, scripting, networking, στη διαχείριση μνήμης, threating, localization support, scene graph. Συχνά διευκολύνεται σε μεγάλο βαθμό η δημιουργία βιντεοπαιχνιδιών από την επαναχρησιμοποίηση και προσαρμογή της ίδιας μηχανής ενώ δίνεται η δυνατότητα της συμβατικότητας με διαφορετικές πλατφόρμες [13].



Εικόνα 16: Ο Σκοπός της Game Engine

Rendering είναι η διαδικασία με την οποία δημιουργείται μια εικόνα από ένα 2D ή 3D μοντέλο μέσω ειδικών προγραμμάτων ηλεκτρονικών υπολογιστών. Η Rendering Engine, δημιουργεί 3D κινούμενα γραφικά χρησιμοποιώντας επιλεγμένες μεθόδους (rasterization, ray- tracking). Ο προγραμματισμός και η μεταγλώττιση αντί να εκτελείται από τη CPU ή τη GPU βασίζεται σε ένα ή περισσότερα API (application programming interface) όπως η Direct3D και OpenGL.

Μια μηχανή φυσικής είναι ένα λογισμικό υπολογιστή που παρέχει κατά προσέγγιση φυσική και πειστική συμπεριφορά, προσομοιώνοντας σε πραγματικές καταστάσεις και συμπεριφορές όπως είναι η δυναμική του άκαμπτου σώματος (συμπεριλαμβανομένης της ανίχνευσης σύγκρουσης) και η δυναμική του μαλακού σώματος και τη δυναμική του υγρού στοιχείου. Χρησιμοποιείται σε διάφορους τομείς όπως τα γραφικά υπολογιστών, σε βιντεοπαιχνίδια και ταινίες με στόχο τη προσομοίωση των νόμων της φυσικής μέσα σε μια εικονική εφαρμογή [14].

Η μηχανή ήχου είναι το σύστημα που αποτελείται από αλγορίθμους που έχουν σχέση με τον ήχο. Επιτρέπει να εισαχθούν στερεοφωνικά αλλά και μονοφωνικά αρχεία ήχου. Η χρήση των αρχείων ήχου μέσω της μηχανής συντελεί στη δημιουργία μιας ρεαλιστικότερης εφαρμογής.

Στο σύστημα γραφικής διεπαφής χρήστη δημιουργείται ένα γραφικό περιβάλλον διεπαφής χρήστη το οποίο βρίσκεται εμβυθισμένο μέσα στο χώρο. Περιλαμβάνει τόσο τα μενού της έναρξης του παιχνιδιού, όσο και όλο το διαδραστικό περιβάλλον κατά το χρόνο εκτέλεσής του.

Στα βιντεοπαιχνίδια η τεχνητή νοημοσύνη χρησιμοποιείται για να δημιουργήσει έξυπνες συμπεριφορές κυρίως σε χαρακτήρες που δεν τους χειρίζονται πραγματικοί χρήστες, συμπεριφορές που πολλές φορές τείνουν να προσεγγίσουν την ανθρώπινη νοημοσύνη. Το σύστημα τεχνητής νοημοσύνης αποτελεί ένα από τα σημαντικότερα αντικείμενα της πτυχιακής εργασίας καθώς προσδίδει διαδραστικότητα στο παιχνίδι [15].

Μια Scripting language είναι μια γλώσσα προγραμματισμού που επιτρέπει τον έλεγχο μιας ή περισσοτέρων εφαρμογών. Το script είναι μια λίστα εντολών που εκτελείται από συγκεκριμένο πρόγραμμα ή μια μηχανή scripting. Scripting languages θεωρούνται η C# και η JavaScript. Το Scripting αποτελεί βασικό χαρακτηριστικό των παιχνιδιών καθώς κάθε παιχνίδι θα χρειαστεί script για να ανταποκριθεί στην είσοδο του χρήστη αλλά και να φροντίσει ώστε τα διάφορα γεγονότα του game play να πραγματοποιηθούν την κατάλληλη στιγμή. Επίσης, μπορεί να χρησιμοποιηθεί για τη δημιουργία οπτικών εφέ, για τον έλεγχο της φυσικής συμπεριφοράς των αντικειμένων ή και για την υλοποίηση ενός προσαρμοσμένου συστήματος τεχνητής νοημοσύνης (AI) για τους χρήστες του παιχνιδιού [16].

3.5 Δημοφιλείς Μηχανές Παιχνιδιών

Οι δημοφιλείς μηχανές παιχνιδιών, οι οποίες αποτελούν ελεύθερο λογισμικό, είναι οι εξής:

- Cry Engine
- Amazon Lumberyard
- Unreal Engine (UE)
- Unity 3D

3.5.1 Cry Engine

Η Cry Engine είναι μια μηχανή παιχνιδιών η οποία δημιουργήθηκε από την εταιρία ανάπτυξης παιχνιδιών CryLeK και χρησιμοποιείται σε όλους τους τίτλους της. Κύριο χαρακτηριστικό της είναι τα υψηλού επιπέδου γραφικά σε συνδυασμό με τους προηγούμενους shaders (σκιαστές) και σύστημα φωτισμού που διαθέτει. Λόγω αυτών, η Cry Engine είναι κατάλληλη μόνο για ισχυρούς υπολογιστές και για κονσόλες υψηλής τεχνολογίας. Επίσης, υποστηρίζει την εικονική πραγματικότητα και διαθέτει πληθώρα προηγμένων οπτικών χαρακτηριστικών και εργαλείων, συστήματα ήχου και φυσικής και συστήματα χαρακτήρων και animation. Η Cry Engine αποτελεί ελεύθερο λογισμικό [20].

Εικόνα 17: CryEngine

3.5.2 Amazon Lumberyard

Η Amazon Lumberyard είναι μηχανή ελεύθερου λογισμικού που δημιουργήθηκε από την Amazon βασισμένη στην αρχιτεκτονική της Cry Engine. Έχει παρόμοιες δυνατότητες με την Cry Engine και είναι κατάλληλη για την ανάπτυξη παιχνιδιών με γραφικά υψηλής ποιότητας στοχεύοντας πλατφόρμες υψηλής τεχνολογίας. Αξιοσημείωτο αποτελεί το γεγονός ότι ολόκληρος ο

Εικόνα 18:Amazon Lumberyard

πηγαίος κώδικας είναι ορατός και μπορεί να δεχθεί τροποποιήσεις από τους προγραμματιστές αναλόγως των απαιτήσεων [21].

3.5.3 Unreal Engine

Η Unreal Engine κυκλοφόρησε το 1998. Είναι μια ολοκληρωμένη πλατφόρμα με πληθώρα εργαλείων ανάπτυξης παιχνιδιών από τις πιο προηγμένες μέχρι σήμερα ικανή να δημιουργήσει κορυφαία οπτικά εφέ. Λόγω των δυνατοτήτων της, την αποτελεσματικότητα στο σχεδιασμό και την εύχρηστη λειτουργία της είναι αποδεχτή ως μηχανή από τους ερασιτέχνες προγραμματιστές αλλά και από τα κορυφαία στούντιο ανάπτυξης παιχνιδιών. Οι προγραμματιστές μπορούν επίσης να κάνουν τα project τους συμβατά τόσο για κινητές συσκευές που χρησιμοποιούν ios όσο και για κινητές συσκευές που χρησιμοποιούν android. Η μηχανή αυτή όπως αναφέρεται και παραπάνω αποτελεί ελεύθερο λογισμικό [22].

Εικόνα 19: Unreal Engine

3.5.4 Unity3D

Η Unity 3D που αρχικά κυκλοφόρησε το 2005, είναι μια πλατφόρμα ανάπτυξης υψηλής ποιότητας 2D και 3D παιχνιδιών. Δίνοντας μεγάλη έμφαση στη συμβατότητα, η Unity 3D μπορεί να υποστηρίξει πάνω από είκοσι πλατφόρμες συμπεριλαμβανομένων των PC, κονσόλες, κινητές συσκευές (android και ios) και ιστοσελίδες. Επίσης, μπορούν να γίνουν διάφορες ρυθμίσεις ανάλογα την πλατφόρμα με σκοπό την συμβατότητα και την βελτιστοποίηση της εφαρμογής. Δηλαδή, η Unity 3D έχει την δυνατότητα να εντοπίσει την καταλληλότερη παραλλαγή των ρυθμίσεων όσων αφορά τα γραφικά, ανάλογα το hardware ή την πλατφόρμα που τρέχει η εφαρμογή, βελτιστοποιώντας έτσι αλλά και μειώνοντας την απόδοση των γραφικών εάν είναι απαραίτητο. Εκτός από τα υψηλού επιπέδου γραφικά, η Unity 3D έχει ενσωματωμένη κορυφαία μηχανή φυσικής η οποία ονομάζεται nVidia PhysX. Όπως η Unreal Engine, έτσι και η Unity 3D προσφέρει στους προγραμματιστές ένα διαδικτυακό κατάστημα, το Asset store ώστε να αγοράζουν επαναχρησιμοποιημένο υλικό και διάφορα άλλα assets (περιουσιακά στοιχεία) για να τα χρησιμοποιήσουν στην εφαρμογή τους. Εν κατακλείδι, χάρη στην ικανότητα της μηχανής αυτής να είναι συμβατή με μεγάλο αριθμό πλατφόρμων σε συνδυασμό με το φιλικό περιβάλλον που προσφέρει στο χρήστη, η Unity 3D θεωρείται η κατάλληλη επιλογή για πολλούς προγραμματιστές [23].

Εικόνα 20: Unity3D

3.6 Επιλογή της Κατάλληλης Μηχανής

Όπως έχουμε αναφέρει παραπάνω και οι 4 μηχανές παιχνιδιών ελεύθερου λογισμικού είναι εξίσου ισχυρές και κατάλληλες για την ανάπτυξη υψηλού επιπέδου εφαρμογών. Όμως η Unreal Engine και η Unity 3D έχουν το προβάδισμα σε σχέση με τις άλλες δύο, μιας και παρέχουν τεράστια ποικιλία χρήσιμων εργαλείων, είναι συμβατές με πολλές πλατφόρμες και

συσκευές χωρίς να περιορίζονται οι δυνατότητες τους και η απόδοση των γραφικών. Επίσης, προσφέρουν ένα πιο φιλικό περιβάλλον στον χρήστη σε συνδυασμό με την ύπαρξη τεράστιας διαδικτυακής κοινότητας και όχι μόνο όπου κάποιος χρήστης μπορεί να βρει απαντήσεις σχετικά εύκολα πάνω σε διάφορα ζητήματα που προκύπτουν κατά την ανάπτυξη μιας εφαρμογής. Έτσι, η Cry Engine και η Lumberyard παρά το γεγονός ότι και οι δύο είναι ισχυρές μηχανές ανάπτυξης παιχνιδιών, η πολυπλοκότητα της δομής τους και το όχι και τόσο φιλικό υπολογιστικό περιβάλλον που προσφέρουν σε συνδυασμό με την περιορισμένη ύπαρξη υποστηρικτικής διαδικτυακής κοινότητας, τις θέτουν εκτός συναγωνισμού. Συνοψίζοντας, για την υλοποίηση της συγκεκριμένης εργασίας τελικά προτιμήθηκε η χρήση Unity 3D έναντι της Unreal Engine λόγω ότι η Unity 3D δίνει την δυνατότητα σύνταξης κώδικα σε C#, Javascript και Boo ενώ η Unreal Engine σε C/C++.

Αν και οι δύο μηχανές έχουν τεράστιες διαδικτυακές κοινότητες υποστήριξης, αυτή της Unity3D είναι ελαφρώς μεγαλύτερη και πληρέστερη. Επίσης, το Asset store της Unity παρέχει αφθονία μοντέλων και περιουσιακών στοιχείων στους χρήστες σε αντίθεση με το Market place της Unreal. Λόγω λοιπόν, προσωπικής εξοικείωσης με την C#, επιλέχτηκε να χρησιμοποιηθεί η Unity3D στο συγκεκριμένο project.

Τέλος, παρότι η Unreal Engine θεωρείται καταλληλότερη για την ανάπτυξη 3D εφαρμογών προτιμήθηκε η Unity3D γιατί όπως προαναφέραμε παρέχει ένα πιο φιλικό και εύκολο στο χρήστη περιβάλλον ειδικά για όσους δεν έχουν εξοικείωση και μεγάλη εμπειρία στις μηχανές παιχνιδιών.

Κεφάλαιο 4ο Τεχνολογική βάση

4.1 Εισαγωγή

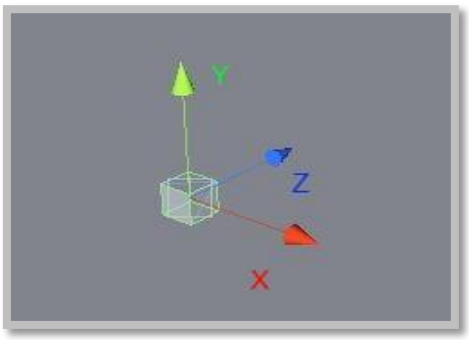
Στο κεφάλαιο αυτό αρχικά γίνεται μια εκτενής αναφορά στη μηχανή σχεδίασης παιχνιδιών Unity3D η οποία επιλέχθηκε να χρησιμοποιηθεί, αναλύοντας όλα τα βασικά της μέρη που είναι απαραίτητα για την δόμηση μιας ολοκληρωμένης εφαρμογής ή παιχνιδιού. Επίσης, αναλύεται ο όρος scripting languages, οι οποίες αποτελούν την «καρδιά» της συγκεκριμένης εργασίας καθώς οποιοδήποτε προγραμματιστικό εγχείρημα της εφαρμογής βασίζεται αυστηρά σε αυτές τις αρχές. Τέλος, παρουσιάζονται όλες οι βασικές λειτουργίες της συσκευής Leap Motion και γίνεται περιγραφή του Oculus Rift Consumer Version 1 που προσφέρει την εμπειρία του εικονικού περιβάλλοντος, αναλύοντας όλα τα τεχνικά χαρακτηριστικά του.

4.2 Βασικές Έννοιες της Unity

Οι βασικότερες έννοιες της Unity που θα μας απασχολήσουν στην συγκεκριμένη εργασία είναι το σύστημα συντεταγμένων, το GameObject, Component, και το Parenting. Κάθε εφαρμογή μπορεί να εμπεριέχει μία ή περισσότερες σκηνές. Κάθε σκηνή μπορεί να περιέχει ένα η περισσότερα Game Objects. Ένα Game Object μπορεί να αποτελείται από ένα ή περισσότερα components

Σύστημα Συντεταγμένων

Το σύστημα συντεταγμένων της Unity3D είναι όμοιο με το καρτεσιανό σύστημα, δηλαδή είναι απαραίτητο ώστε να προσδιοριστεί επαρκώς οποιοδήποτε σημείο στο χώρο ή στο επίπεδο. Επειδή όμως η συγκεκριμένη εφαρμογή έχει σκοπό την εμβύθιση του χρήστη σε ένα εικονικό τρισδιάστατο περιβάλλον, αναπόφευκτα χρησιμοποιούμε το σύστημα συντεταγμένων τριών διαστάσεων x,y,z. Το γενικό σύστημα συντεταγμένων της Unity3D είναι left handed και y-up. Αυτό σημαίνει ότι ο θετικός άξονας του x έχει φορά προς τα δεξιά, ο θετικός άξονας του y έχει φορά προς τα πάνω και ο θετικός άξονας του z κατευθύνεται προς το βάθος της οθόνης, έχοντας σημείο αναφοράς το (0,0,0) δηλαδή το μηδέν. Οποιοδήποτε αντικείμενο της εφαρμογής είναι τοποθετημένο στο χώρο σύμφωνα με αυτό σύστημα συντεταγμένων αλλιώς δε γίνεται να υφίσταται.



Εικόνα 21: Συστήμα Συντεταγμένων

GameObject

Το GameObject είναι η πιο σημαντική έννοια του Unity editor. Κάθε αντικείμενο που υπάρχει στην εφαρμογή καθώς και οτιδήποτε επιθυμεί κάποιος χρήστης να εισάγει σε αυτήν, αποτελεί ένα GameObject. Όμως, ένα GameObject δε διαθέτει αρχικά καμία ιδιότητα από μόνο του. Τα διάφορα χαρακτηριστικά και ιδιότητες καθορίζονται από τον δημιουργό του. Στην ουσία, ένα GameObject είναι σαν άδειο δοχείο. Δηλαδή, ο προγραμματιστής μπορεί να προσθέσει διάφορα κομμάτια σε αυτό, ώστε να το μετατρέψει σε έναν χαρακτήρα, σε φως, σε δέντρο, σε ήχο ή σε ότι άλλο επιθυμεί ο ίδιος. Ανάλογα με το είδος του αντικειμένου που θέλουμε να δημιουργήσουμε, μπορούμε να προσθέσουμε διαφορετικούς συνδυασμούς ειδικών κομματιών-χαρακτηριστικών στο GameObject. Κάθε κομμάτι που προστίθεται στο GameObject, ονομάζεται component (συστατικό) [24].

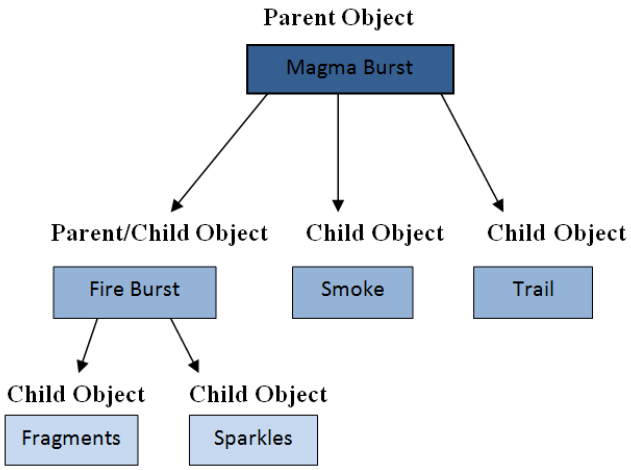
Component (Συστατικό)

Με τον όρο Component εννοούμε κάποιο "συστατικό" το οποίο παρέχεται σε ένα οποιαδήποτε GameObject με σκοπό τον καθορισμό της εμφάνισης, της συμπεριφοράς και διαφόρων λειτουργιών του. Η Unity3D έχοντας γνώμονα τη διευκόλυνση του χρήστη, παρέχει μια μεγάλη ποικιλία έτοιμων Component όπως RigidBody, φώτα, κάμερες κ.α. Για τη δημιουργία περαιτέρω διαδραστικών στοιχείων του παιχνιδιού, γράφουμε scripts, τα οποία αναγνωρίζονται ως Components με σκοπό την επέκταση ή τροποποίηση της υπάρχουσας διαθέσιμης λειτουργικότητας της Unity3D. Για παράδειγμα, κάθε GameObject περιλαμβάνει ένα "Transform Component". Όταν δημιουργείται κάποιο GameObject αυτόματα προστίθεται σε αυτό ένα Transform Component, το οποίο δε μπορεί να αφαιρεθεί.

Το Transform Component καθορίζει τη θέση του GameObject στο χώρο, την περιστροφή και το μέγεθός του βασισμένο στο σύστημα συντεταγμένων x,y,z της Unity3D [25].

Parenting (Ομαδοποίηση)

Το Parenting είναι μια ιδιότητα που χρησιμοποιήθηκε πάμπολλες φορές στη συγκεκριμένη εργασία. Όταν δημιουργείται μια συγκεκριμένη ομάδα GameObject, υπάρχει ένα αντικείμενο το οποίο βρίσκεται ιεραρχικά πάνω από τα υπόλοιπα και ονομάζεται parent object. Τα υπόλοιπα GameObject που είναι ομαδοποιημένα κάτω από αυτό ονομάζονται "child object" ή "children". Μια τέτοια ομαδοποίηση χρειάστηκε στην δημιουργία των εχθρών άλλα και άλλων GameObject του παιχνιδιού προκειμένου να έχουμε την δυνατότητα να τα προσπελαύνουμε προγραμματιστικά οποιαδήποτε στιγμή και να πραγματοποιούμε αλλαγές σε μεγάλο αριθμό αντικειμένων με ευκολία. Μπορούν επίσης να δημιουργηθούν εμφωλευμένα parent-child objects τα οποία ονομάζονται "απόγονοι" (descendants) του κορυφαίου parent object [26]. Αυτή η δυνατότητα που προσφέρεται από την Unity3D, χρησιμοποιήθηκε κατά κόρον στην συγκεκριμένη εφαρμογή αφού οι ομαδοποιήσεις που πραγματοποιήσαμε μας βοήθησαν στην εύκολη επεξεργασία μεγάλου όγκο δεδομένων.



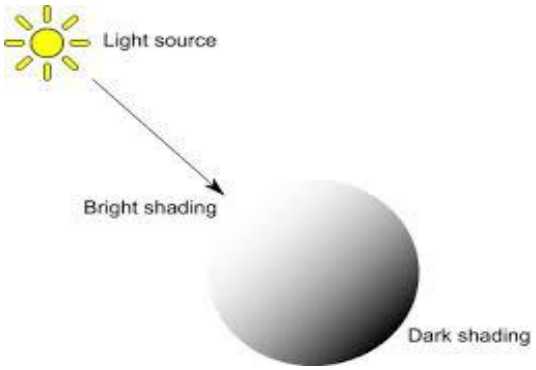
Εικόνα 22: Parenting

Σκηνή

Η σκηνή περιέχει όλα τα περιβάλλοντα και τα μενού της εφαρμογής. Ο προγραμματιστής σε κάθε σκηνή μπορεί να τοποθετήσει όσα και όποια Game Object επιθυμεί. Ουσιαστικά γίνεται σχεδιασμός και υλοποίηση του παιχνιδιού σε επί μέρους κομμάτια.

4.3 Φωτισμός (Lighting)

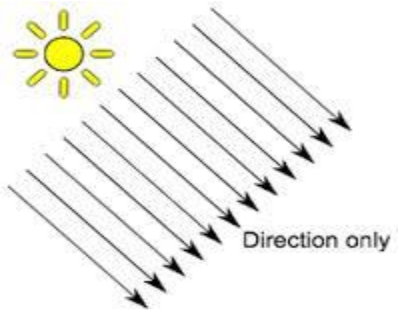
Προκειμένου να υπολογιστεί η σκιά ενός 3D αντικειμένου, η Unity πρέπει να γνωρίζει την ένταση, την κατεύθυνση και το χρώμα του φωτός που πέφτει σε αυτό. Αυτές οι ιδιότητες παρέχονται από τα Light Objects στη σκηνή (scene). Υπάρχουν διάφοροι τύποι φωτός με τον καθένα να παρέχει διαφορετικές ιδιότητες ώστε να καλύπτουν τις ανάγκες του προγραμματιστή ανάλογα την εφαρμογή. Οι πιο γνωστοί τύποι φωτός παραθέτονται παρακάτω και αναλύονται.



Εικόνα 23: Basic Lighting Diagram

Directional Lights

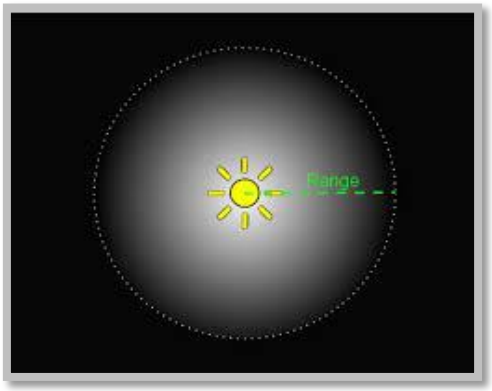
Τα Directional Lights είναι κατάλληλα για τη δημιουργία εφέ όπως το ηλιακό φως στη σκηνή. Στη παρούσα διπλωματική το Directional Light χρησιμοποιήθηκε για να προσομοιωθεί ο ήλιος και οι ακτίνες του για τη δημιουργία της μέρας στο παιχνίδι μας. Επειδή, έχουν την ικανότητα να συμπεριφέρονται όπως ο ήλιος, μπορούν να δώσουν την αίσθηση της απομακρυσμένης πηγής φωτός. Παρ' όλα αυτά τα Directional Lights δεν έχουν κάποια συγκεκριμένη θέση και άρα μπορούν να τοποθετηθούν οπουδήποτε στη σκηνή. Στη συνέχεια, όλα τα αντικείμενα που υπάρχουν στη σκηνή φωτίζονται σαν να έρχεται φως πάντα από την ίδια κατεύθυνση. Η απόσταση του φωτός από οποιοδήποτε αντικείμενο δεν ορίζεται και άρα η ένταση του δεν μειώνεται.



Εικόνα 24: Directional Light

Point Lights

Τα Point Lights βρίσκονται τοποθετημένα στο χώρο και εκπέμπουν ομοιόμορφα φως προς κάθε κατεύθυνση. Η κατεύθυνση του φωτός που χτυπάει την επιφάνεια ενός αντικειμένου είναι η ευθεία γραμμή από την εν λόγω εστία προς το σημείο πρόσκρουσης. Η ένταση του φωτός μειώνεται σύμφωνα με την απόσταση από το φωτεινό αντικείμενο φτάνοντας στο μηδέν εάν υπερβεί το κατώτερο όριο που έχει οριστεί. Πιο συγκεκριμένα, η ένταση του φωτός είναι αντιστρόφως ανάλογη προς το τετράγωνο της απόστασης από την πηγή. Αυτό το χαρακτηριστικό που είναι γνωστό ως "αντίστροφος τετραγωνικός νόμος", προσομοιώνει τη συμπεριφορά του φωτός στον πραγματικό κόσμο. Τα Point Lights χρησιμοποιούνται κυρίως για να αναπαραστήσουν λάμπες και άλλες τοπικές πηγές φωτός στη σκηνή όπως ακριβώς έγινε και στην παρούσα εργασία[27].

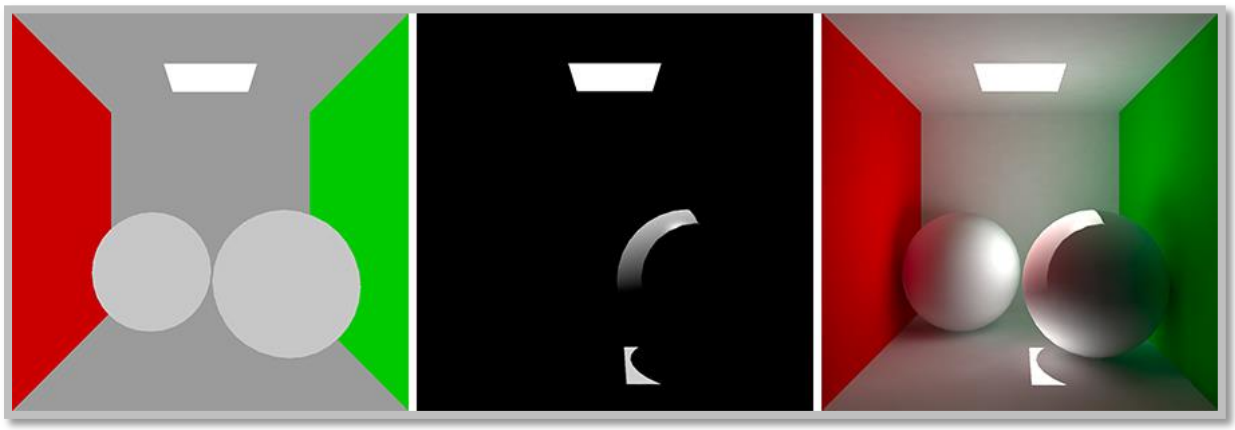


Εικόνα 25: Point Light

4.4 Τεχνικές Φωτισμού

Σε γενικές γραμμές, ο φωτισμός στη Unity μπορεί να θεωρηθεί ως "realtime" (πραγματικού χρόνου) ή "precomputed" (προϋπολογισμένος). Παρέχεται η δυνατότητα συνδυασμού των δύο παραπάνω τεχνικών με σκοπό τη δημιουργία σκηνής με όσο το δυνατόν πιο ρεαλιστικό φωτισμό.

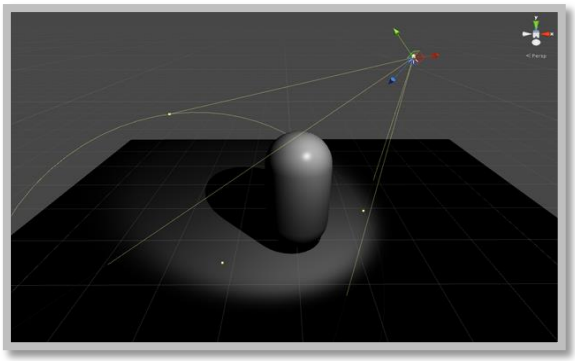
"Global Illumination" στο πεδίο των γραφικών, είναι ο όρος που αναφέρεται σε ένα εύρος από τεχνικές και μαθηματικά μοντέλα τα οποία επιχειρούν να προσομοιώσουν την περίπλοκη συμπεριφορά του φωτός καθώς αναπηδά και αλληλεπιδρά με τον κόσμο. Η αναπαράσταση του παγκόσμιου φωτισμού με ακρίβεια αποτελεί πρόκληση και είναι υπολογιστικά δαπανηρή [28].



Εικόνα 26: Η ίδια σκηνή: χωρίς lighting (αριστερά), μόνο με 'direct light' (κέντρο) & με έμεσο 'global illumination' (δεξία). Παρατηρήστε πως μεταφέρονται τα χρώματα καθώς το φως 'αναπηδά' πάνω στις επιφάνειες, δίνοντας πολύ περισσότερο ρεαλισμό

Τεχνική Πραγματικός χρόνος φωτισμού (RealTime Lighting)

Τα φώτα στη Unity όπως τα directional, point spot είναι ρυθμισμένα ως real time. Αυτό σημαίνει ότι συμβάλλουν άμεσα στη σκηνή και την ενημερώνουν σε κάθε frame. Καθώς τα φώτα και τα Gameobject κινούνται, ο φωτισμός ενημερώνεται αμέσως. Δηλαδή, παρατηρείται η επίδραση του φωτός σε πραγματικό χρόνο. Επιπρόσθετα, οι σκιές είναι εντελώς μαύρες καθώς δεν υπάρχει "αναπήδηση" (bounciness) του φωτός στις διάφορες επιφάνειες των αντικειμένων. Η τεχνική realtime lighting αποτελεί τον πιο βασικό τρόπο φωτισμού και είναι κατάλληλη για χαρακτήρες ή κινούμενα γεωμετρικά αντικείμενα. Όμως όπως αναφέραμε παραπάνω η τεχνική realtime lighting υστερεί σε ρεαλισμό όσον αφορά τη συμπεριφορά του φωτός, διότι δεν αναπαριστά την αναπήδηση του στις επιφάνειες των διαφόρων αντικειμένων. Για να επιτευχθεί ρεαλιστικότερος φωτισμός πρέπει να ενεργοποιηθεί το "Global Illumination" (παγκόσμιος φωτισμός) θυσιάζοντας ετσι υπολογιστικούς πόρους.



Εικόνα 27: Real Time Lighting

Τεχνική Baked Lightmaps Global Illumination

Η Unity έχει τη δυνατότητα να υπολογίσει σύνθετα εφέ φωτισμού χρησιμοποιώντας το υπολογιστικό μοντέλο Global Illumination και τα αποθηκεύει σε "texture" που ονομάζονται "lightmaps". Αυτή η διαδικασία αναφέρεται ως «baking» (ψήσιμο). Τα lightmaps τοποθετούνται στις επιφάνειες των αντικειμένων και έχουν τη δυνατότητα να περιλαμβάνουν την πληροφορία τόσο για τον άμεσο φωτισμό που χτυπάει μια επιφάνεια όσο και τον έμμεσο φωτισμό που αναπηδάει στις διάφορες επιφάνειες των αντικειμένων. Με τον «baking» φωτισμό, τα lightmaps δεν μπορούν να αλλάξουν κατά τη διάρκεια της χρήσης της εφαρμογής και γι' αυτό αναφέρονται ως «στατικά». Όμως μπορούν να χρησιμοποιηθούν προσεκτικά Realtime Lights σε μια σκηνή με lightmapped αντικείμενα. Με αυτή την προσέγγιση θυσιάζουμε την δυνατότητα της δυναμικής ενημέρωσης του φωτισμού στη σκηνή κατά τη διάρκεια της εφαρμογής με μια πιθανή αύξηση της υπολογιστικής απόδοσης, κάτι το οποίο είναι κατάλληλο για συσκευές που δεν χαρακτηρίζονται από τόσο ισχυρό hardware όπως είναι οι κινητές πλατφόρμες.

Τεχνική Precomputed Realtime Global Illumination

Είναι η διαδικασία προ-υπολογισμού αναπήδησης του φωτός γύρω από τη στατική γεωμετρία της σκηνής και αποθηκεύει τα δεδομένα για χρήση σε πραγματικό χρόνο. Αυτή η διαδικασία μειώνει τον αριθμό υπολογισμών για την απόδοση του φωτός που κανονικά θα έπρεπε να εκτελεστούν σε πραγματικό χρόνο επιτρέποντας έτσι τη διάχυση του φωτός διατηρώντας παράλληλα τη διαδραστικότητα με το ρυθμό μετάδοσης των frame.

Με τη συγκεκριμένη προσέγγιση μπορούμε να δημιουργήσουμε περιβάλλοντα που φωτίζονται βάσει του Global Illumination (παγκόσμιος φωτισμός) συν ότι ανταποκρίνονται στις διάφορες αλλαγές του φωτισμού σε πραγματικό χρόνο. Προκειμένου να επιτευχθούν αυτά τα αποτελέσματα σε κάθε frame θα πρέπει να μεταφέρουμε κάποιες από τις σύνθετες διαδικασίες υπολογισμού του φωτός από τον πραγματικό χρόνο σε χρονικά πλαίσια τα οποία δεν είναι κρίσιμα για την ομαλή έκβαση της εφαρμογής. Δηλαδή κάποιοι υπολογισμοί που εκτελούνταν σε πραγματικό χρόνο τώρα θα είναι precomputed (προϋπολογισμένοι). Αυτή η διαδικασία ονομάζεται «offline» (εκτός σύνδεσης) [29].

4.4.1 Επιλογή της Κατάλληλης Μεθόδου

Η επιλογή τεχνικής φωτισμού εξαρτάται από τη φύση της εφαρμογής και τις δυνατότητες απόδοσης του υπολογιστικού υλικού (hardware) που έχουμε στη διάθεση μας. Για παράδειγμα, σε κινητά όπου η μνήμη και η ισχύς επεξεργασίας είναι περιορισμένη, η τεχνική baked lightmaps θεωρείται η κατάλληλη. Αντίθετα, σε προσωπικούς υπολογιστές με εξειδικευμένο υλικό γραφικών ή σε κονσόλες παιχνιδιών, συνήθως χρησιμοποιείται η τεχνική precomputed realtime GI ή ακόμα και συνδυασμός αυτών των δύο. Στη συγκεκριμένη εφαρμογή χρησιμοποιείται η τεχνική Precomputed Realtime Global Illumination. Έτσι, αν και χρειαζόμαστε σχετικά μεγάλη υπολογιστική ισχύ, κάτι το οποίο είναι εφικτό αφού έχουμε στην κατοχή μας το κατάλληλο υλικό υψηλών προδιαγραφών, επιτυγχάνουμε να δημιουργήσουμε μια εφαρμογή ρεαλιστικότερη που σε συνδυασμό με το HMD και την εικονική πραγματικότητα, ο χρήστης θα μπορέσει να εμβυθιστεί απόλυτα στο εικονικό-ψηφιακό περιβάλλον.

4.5 Ray Casting

Μια ακτίνα είναι μία γραμμική συνιστώσα η οποία αρχίζει από ένα σημείο και συνεχίζει προς κάποια συγκεκριμένη κατεύθυνση έχοντας τη δυνατότητα να επεκτείνεται στο άπειρο εφόσον δεν καθοριστεί κάποιο όριο. Ένα Ray Cast περιλαμβάνει τη διασταύρωση μιας ακτίνας με τα αντικείμενα σ’ ένα περιβάλλον και μπορεί να μας δώσει πληροφορίες σχετικά με αυτά τα αντικείμενα όπως η απόσταση εντοπισμού, το όνομα, το σημείο πρόσκρουσης ακτίνας αντικειμένου και άλλα.

4.6 Φυσική Αντικειμένων, Στερεά Σώματα, Συγκρουστές (Physics, Rigidbody, Colliders)

Φυσική αντικειμένων

Για να έχουμε πειστική φυσική συμπεριφορά , ένα gameobject πρέπει να επιταχύνεται σωστά και να επηρεάζεται από συγκρούσεις, τη βαρύτητα και άλλες δυνάμεις. Η μηχανή φυσικής της Unity Nvidia PhysX παρέχει κατάλληλα components (συστατικά) ώστε να προσομοιώνουν τις φυσικές δυνάμεις των διαφόρων αντικειμένων. Αντίθετα, στις μηχανές παιχνιδιών δεν υπάρχει καμία δυνατότητα, από μόνα τους τα αντικείμενα να επηρεάζονται από τις διάφορες δυνάμεις. Για να γίνει αυτό θα πρέπει ο χρήστης να τοποθετήσει τα κατάλληλα components στα gameobject, σε αυτά δηλαδή που θέλει να ανταποκρίνονται στους νόμους της φυσικής και παράλληλα να πραγματοποιήσει και τις κατάλληλες ρυθμίσεις.

Στερεά Σώματα (Rigidbody components)

Το Rigidbody component επιτρέπει στο gameobject να συμπεριφέρεται και να δρα σύμφωνα με τους νόμους της φυσικής. Δηλαδή, το gameobject με την προσθήκη του Rigid body component, μπορεί να δεχθεί δυνάμεις και ροπές ώστε να κινείται με ρεαλιστικό τρόπο,

να προσομοιώνει τη βαρύτητα, να μπορεί να δρα κάτω από συμπληρωματικές δυνάμεις οι οποίες προστίθενται μέσω script και να αλληλεπιδρά με άλλα αντικείμενα [31]. Σε διάφορα αντικείμενα τα οποία χρησιμοποιήσαμε τοποθετήσαμε στερεά σώματα ώστε να ανταποκρίνονται σαν εμπόδια στην εφαρμογή όπως δέντρα, κτίρια κ.α.

Συγκρουστές (colliders)

Οι colliders είναι ένα άλλο είδος συστατικού (component) που πρέπει να προστεθεί μαζί με το Rigidbody component στο gameobject προκειμένου να μπορούν να πραγματοποιηθούν οι συγκρούσεις. Εάν δύο gameobject αλληλοχτυπηθούν, η μηχανή φυσικής δε θα υπολογίσει καμία σύγκρουση εκτός αν και τα δύο αντικείμενα διαθέτουν colliders. Εάν έστω κι ένα αντικείμενο δε διαθέτει, δε θα υπολογιστεί καμία σύγκρουση και θα περάσει το ένα μέσα από το άλλο [32]. Υπάρχουν τα εξής είδη collider:

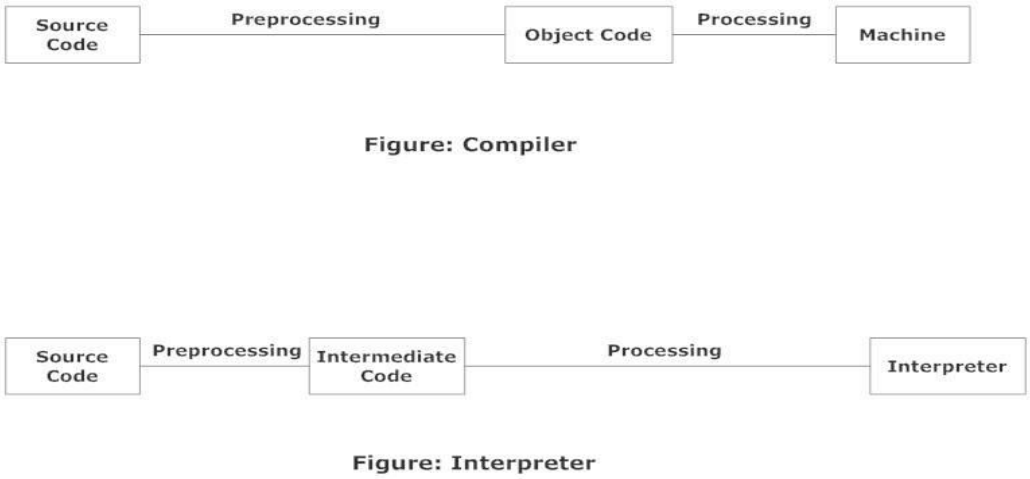
- σχήμα κύβου (box collider),
- σχήμα σφαίρας (sphere collider),
- σχήμα κάψουλας (capsule collider).
- Mesh collider, δημιουργεί ενιαίο collider από το δίχτυ mesh του αντικείμενου και δε μπορεί να συγκρουστεί με άλλα mesh collide.
- Wheel collider, ειδικό για τη δημιουργία αυτοκινήτων και άλλων κινούμενων οχημάτων.
- Terrain collider, υπεύθυνο για τις συγκρούσεις εδάφους.

Η δομή των περισσότερων αντικειμένων έχει ρυθμιστεί έτσι ώστε να υπακούν στους νόμους της φυσικής, όσον αφορά την ύπαρξη μάζας και την πραγματοποίηση συγκρούσεων όταν οι συνθήκες το απαιτούν. Όπως για παράδειγμα, η μπάλα της φωτιάς έχει δημιουργηθεί με βάση τις παραπάνω αρχές φυσικής, προκειμένου να πραγματοποιείται σύγκρουση όταν αυτή έρχεται σε επαφή με άλλα αντικείμενα που και αυτά με τη σειρά τους έχουν ρυθμιστεί να υπακούν στους νόμους της φυσικής. Αποτελεί βασικό κομμάτι της συγκεκριμένης πτυχιακή, αφού χρησιμοποιήθηκαν διάφορα ήδη colliders προκειμένου οι επιθέσεις του χρήστη, οι οποίες εμπεριέχουν sphere collider, να χτυπούν τους διάφορους στόχους που συνήθως εμπεριέχουν capsule collider. Το terrain εμπεριέχει terrain collider.

4.7 Scripting ή Script Language (Γλώσσα Σεναρίων)

Scripting ή Script Language είναι μία γλώσσα προγραμματισμού που υποστηρίζει τα scripts. Τα scripts είναι προγράμματα γραμμένα για ένα ειδικό run-time περιβάλλον που αυτοματοποιούν την εκτέλεση εργασιών οι οποίες θα έπρεπε διαφορετικά να εκτελεστούν μία προς μία από έναν ανθρώπινο φορέα[33]. Έχουν την ικανότητα να συνδυάζονται μεταξύ τους ώστε να δημιουργήσουν πιο σύνθετα προγράμματα. Τα περιβάλλοντα που μπορούν να αυτοματοποιηθούν μέσω scripting είναι εφαρμογές λογισμικού, ιστοσελίδες, ενσωματωμένα συστήματα (embedded systems), “shell” λειτουργικών συστημάτων καθώς και πολυάριθμα παιχνίδια. Οι γλώσσες σεναρίων, συχνά διερμηνεύονται (interpreted) αντί να μεταγλωττίζονται (compiled). Η διαφορά μεταξύ διερμηνέα (interpreter) και μεταγλωττιστή (compiler) είναι οι εξής: ο πρώτος, μεταφράζει ένα πρόγραμμα τη φορά και συνεχίζει μέχρι να συναντήσει το πρώτο λάθος, εάν υπάρχει, όπου και σταματά. Σε αυτή την περίπτωση το debugging (αποσφαλμάτωση) είναι εύκολο σε αντίθεση με τον compiler ο οποίος σαρώνει όλο το πρόγραμμα, μεταφράζει το σύνολό του σε κώδικα μηχανής και εμφανίζει μήνυμα

λάθους, εάν υπάρχουν σφάλματα, αφού σαρώσει όλο το πρόγραμμα, οπότε το debugging σε αυτήν την περίπτωση θεωρείται δύσκολο. Στον διερμηνέα (interpreter), η ανάλυση του πηγαίου κώδικα απαιτεί λιγότερο χρόνο, όμως ο συνολικός χρόνος εκτέλεσης είναι μεγαλύτερος ενώ στον μεταγλωττιστή (compiler), η ανάλυση απαιτεί περισσότερο χρόνο, αλλά ο συνολικός χρόνος εκτέλεσης είναι μικρότερος. Τέλος, μία σημαντική διαφορά ανάμεσα τους είναι ότι δημιουργείται ενδιάμεσο "object code" από τον compiler, άρα απαιτεί περισσότερη μνήμη, ενώ στην περίπτωση του interpreter δεν παράγεται τέτοιου είδους κώδικας, επομένως είναι πιο αποτελεσματικός για τη μνήμη.



Εικόνα 28: Διαφορά Compiler-Interpreter

Στα παιχνίδια, η χρήση των scripting languages είναι πιο συγκεκριμένη: είναι ενσωματωμένες στο παιχνίδι προκειμένου να τρέξουν εξωτερικά scripts. Αυτό είναι χρήσιμο για διάφορους λόγους, πρώτον μειώνεται ο χρόνος μεταγλώττισης του πηγαίου κώδικα, δεύτερον βοηθάει στο διαχωρισμό της μηχανής παιχνιδιού από το ίδιο το παιχνίδι, ενθαρρύνοντας έτσι τη διαδικασία υποδιαίρεσης ενός προγράμματος σε ξεχωριστά υποπρογράμματα και την χρήση επαναχρησιμοποιήσιμου κώδικα. Τέλος, επιτρέπει στους παίχτες να δημιουργούν τα δικά τους scripts με αποτέλεσμα να συνθέτουν πολύ πιο περίπλοκες και ενδιαφέροντες λειτουργίες.

Πιο συγκεκριμένα, στη Unity3D, τα scripts είναι συστατικά (components) τα οποία επεκτείνουν ή τροποποιούν τις υπάρχουσες διαθέσιμες λειτουργίες της. Δημιουργούνται και τοποθετούνται σαν υπομονάδες στα gameobject που επιθυμούμε, προσδίδοντας τους λειτουργικότητα δίχως να υπάρχει περιορισμός στον αριθμό των scripts που μπορούν να χρησιμοποιηθούν. Μπορούν να ελέγξουν τη φυσική συμπεριφορά των αντικειμένων, να συνθέσουν οπτικά εφέ, ακόμα και να υλοποιήσουν ένα προσαρμοσμένο σύστημα τεχνητής νοημοσύνης AI για τους χαρακτήρες. Επίσης, δίνει τη δυνατότητα να προγραμματιστούν είτε σε JavaScript είτε σε C# είτε σε Boo. Στη συγκεκριμένη εργασία όπως έχουμε ήδη αναφέρει και σε προηγούμενο κεφάλαιο χρησιμοποιούμε C#. Για την κατανόηση της λογικής του παιχνιδιού, των αλληλεπιδράσεων, των κινουμένων σχεδίων, των θέσεων της κάμερας, κ.α., προσφέρονται διάφοροι τρόποι που μπορεί αυτό να πραγματοποιηθεί. Το σύνηθες μοντέλο είναι να εκτελέσουμε τις περισσότερες διεργασίες στο εσωτερικό των συναρτήσεων Start() και Update(), ωστόσο υπάρχουν και άλλες λειτουργίες που μπορούν να χρησιμοποιηθούν [34].

- Start(): Η συνάρτηση Start() είναι το μέρος όπου πραγματοποιούνται οι διάφορες αρχικοποιήσεις και καλείται πριν από οποιαδήποτε άλλη συνάρτηση. Καλείται μόνο κατά την πρώτη εκτέλεση του κώδικα

- Update(): Η συνάρτηση Update() καλείται μια φορά ανά frame (καρέ). Είναι κατάλληλη για την κίνηση αντικειμένων τα οποία δεν υπακούν στους νόμους της φυσικής, για την ενημέρωση των διαφόρων χρονομετρητών που ενδεχομένως να χρησιμοποιήσαμε και την ανίχνευση πιθανών εισόδων. Είναι η πιο κοινώς διαδεδομένη συνάρτηση.

- FixedUpdate(): Η FixedUpdate() συνήθως καλείται πιο συχνά απ' ότι η Update(). Μπορεί να κληθεί πολλές φορές σε ένα frame (καρέ) εάν ο ρυθμός των frame είναι χαμηλός ή μπορεί να μη κληθεί καθόλου αν ο ρυθμός των frame είναι υψηλός. Βασικό χαρακτηριστικό της είναι ότι καλείται ανά συγκεκριμένα διαστήματα. Όλοι οι υπολογισμοί φυσικής και οι διάφορες ενημερώσεις συμβαίνουν αμέσως μετά το τέλος της. Είναι κατάλληλη για διαμόρφωση συμπεριφοράς αντικειμένων που υπακούν στους νόμους της φυσικής.

- LateUpdate(): Η LateUpdate() καλείται μια φορά σε κάθε frame, αφότου η συνάρτηση update έχει τελειώσει. Όλοι οι υπολογισμοί που εκτελούνται στην updated, θα πρέπει να έχουν ολοκληρωθεί όταν αρχίζει η LateUpdate().

Στη συγκεκριμένη διπλωματική εργασία χρειάστηκε να δημιουργηθούν πληθώρα script σε γλώσσα C# προκειμένου να επιτευχθεί η κατάλληλη λειτουργικότητα του παιχνιδιού σε διάφορα επίπεδα. Ήταν απαραίτητα για τη σωστή διαχείριση των συγκρούσεων, των ήχων, της τεχνητής νοημοσύνης καθώς επίσης και για την ενσωμάτωση του Oculus Rift και του Leap Motion ώστε να λειτουργεί αποτελεσματικά το head tracking και το hand tracking αντίστοιχα.

4.8 Κινούμενα Σχέδια (Animation)

Αλλαγή του σχήματος, της θέσης, της απόστασης ή του χρονισμού των αντικειμένων σε διαδοχικά frames.

Το σύστημα κινούμενων σχεδίων της Unity3D δίνει τη δυνατότητα στον χρήστη να δημιουργήσει και να διαμορφώσει τα δικά του κινούμενα σχέδια για την κατασκευή διαφόρων συμπεριφορών. Η Unity επιτρέπει στον χρήστη να διαμορφώνει κινούμενα σχέδια για κάθε gameobject. Δηλαδή όχι μόνο για ανθρωπόμορφους χαρακτήρες ή αντικείμενα που έχουν τη μορφή ζώων, άλλα να διαμορφώνει κινούμενα σχέδια για οποιαδήποτε αντικείμενα, όπως μια σφαίρα, αλλάζοντας το μέγεθος, το χρώμα, κι άλλα χαρακτηριστικά μέσα σε ένα καθορισμένο χρονικό περιθώριο. Κάθε αντικείμενο μπορεί να έχει ένα έως πολλά διαφορετικά κινούμενα σχέδια. Η σωστή διαχείριση τους γίνεται μέσω του μηχανισμού animator που παρέχει η Unity. Έχει μορφή "state machine" και οι διαφορετικές καταστάσεις αντιστοιχούν στα διαφορετικά κινούμενα σχέδια. Έτσι, για να μεταβεί το αντικείμενο από τη

μια κατάσταση στην άλλη, χρειάζεται να πληρούνται τα κατάλληλα κριτήρια. Η Unity με αυτόν τον τρόπο επιτυγχάνει εύκολη διαχείριση και δημιουργία κινούμενων σχεδίων άλλα και ρεαλισμό στις διάφορες κινήσεις, αφού η μετάβαση από τη μια συμπεριφορά στην επόμενη πραγματοποιείται πλέον με περισσότερη ομαλότητα. Στη συγκεκριμένη εφαρμογή, κινούμενα σχέδια χρησιμοποιήθηκαν σε όσα αντικείμενα χρειάστηκε να αποδοθεί η έννοια της κίνησης, από περπάτημα και τρέξιμο των χαρακτήρων μέχρι την εμφάνιση και αλλαγή μεγέθους ή χρώματος των μενού [35].

4.10 Πλοήγηση και Εύρεση Καλύτερου Μονοπατιού (Navigation and Path Finding)

Το σύστημα πλοήγησης επιτρέπει στον προγραμματιστή να δημιουργήσει χαρακτήρες και αντικείμενα που μπορούν να κινηθούν έξυπνα στον κόσμο του παιχνιδιού, γνωστοί ως πράκτορες (agents). Το σύστημα πλοήγησης χρησιμοποιεί πλέγματα πλοήγησης τα οποία δημιουργούνται αυτόματα από τη γεωμετρία της σκηνής μέσω της διαδικασίας baking (ψησίματος). Δυναμικά εμπόδια τροποποιούν επίσης την πλοήγηση των πρακτόρων κατά το χρόνο εκτέλεσης της εφαρμογής. Παράλληλα μπορούν να ρυθμιστούν διάφοροι άλλοι παράμετροι που καθορίζουν τον τρόπο δημιουργίας του πλέγματος πλοήγησης [39].

Βάσει αυτής της δυνατότητας που μας προσφέρει η Unity3D, μπορούμε να δημιουργήσουμε υποτυπώδη τεχνητή νοημοσύνη την οποία κατέχουν είτε αντικείμενα είτε χαρακτήρες. Δηλαδή, στην παρούσα διπλωματική, οι εχθροί έχουν ρυθμιστεί έτσι ώστε να κινούνται στο χώρο αυτόνομα, ικανοί να φτάνουν στον προορισμό που τους έχει δοθεί, διανύοντας το πιο γρήγορο μονοπάτι.

4.11 Διεπαφή Χρήστη (User interface)

Η Unity παρέχει ένα νέο σύστημα διεπαφής χρήστη (UI) που του επιτρέπει να δημιουργήσει διαισθητικά και γρήγορα ευπαρουσίαστες γραφικές διεπαφές. Το σύστημα διεπαφής χρήστη αποτελείται από έναν καμβά (canvas), μέσα στον οποίο τοποθετούνται όλα τα υπόλοιπα στοιχεία. Τα στοιχεία αυτά μπορεί να είναι κείμενα, εικόνες, κ.α. καθώς και κάποια πιο διαδραστικά, όπως κουμπιά και sliders. Υπερτερεί έναντι του παλαιότερου συστήματος γραφικών GUI της Unity3D καθώς ο χρήστης μπορεί να σχεδιάσει γρηγορότερα και αποδοτικότερα [40]. Έτσι και εμείς με τη σειρά μας, συνθέσαμε ευπαρουσίαστα μενού φιλικά προς το χρήστη ώστε ο ίδιος να μπορεί να περιηγηθεί με ευκολία μέσα στην εφαρμογή.

4.12 Ήχος (audio) και Κάμερα (camera)

Ένα παιχνίδι θα ήταν ημιτελές χωρίς την ύπαρξη κάποιων ήχων, όπως ηχητικών εφέ και μουσικής υπόκρουσης (background). Το σύστημα ήχου της Unity3D είναι ευέλικτο και ισχυρό. Έχει τη δυνατότητα να εισάγει τις περισσότερες από τις μορφές αρχείων ήχου και διαθέτει εξελιγμένες δυνατότητες αναπαραγωγής στον τρισδιάστατο χώρο. Επίσης, η Unity έχει τη δυνατότητα να εγγράφει και να εισάγει κάποιον ήχο από οποιοδήποτε διαθέσιμο μικρόφωνο στο μηχάνημα του χρήστη ώστε να χρησιμοποιηθεί κατά τη διάρκεια της εφαρμογής.

Στην πραγματική ζωή οι ήχοι εκπέμπονται από διάφορα αντικείμενα και ακούγονται από τους ακροατές. Ο τρόπος που γίνεται αντιληπτός ο ήχος εξαρτάται από διάφορους παράγοντες. Ένας ακροατής μπορεί να καταλάβει περίπου από ποια κατεύθυνση προέρχεται ένας ήχος όπως επίσης να έχει κάποια αίσθηση της απόστασης της πηγής βάσει της ποιότητας και της έντασης του. Μια τυχαίως κινούμενη πηγή θα αλλάξει καθώς κινείται ως αποτέλεσμα του φαινομένου Doppler. Επίσης, το περιβάλλον θα επηρεάσει τον τρόπο με τον οποίο αντανακλάται ο ήχος.

Εικόνα 29: Audio Source-Listener Diagram

Η επίτευξη της προσομοίωσης των ηχητικών εφέ πραγματοποιείται από τη Unity παράγοντας ήχους από Audio Sources (ηχητικές πηγές) οι οποίες είναι προσαρτημένες πάνω στα αντικείμενα. Οι ήχοι που εκπέμπονται εντοπίζονται από έναν audio listener (ηχητικό ακροατή) τοποθετημένο συνήθως σε ένα άλλο αντικείμενα όπως η κύρια κάμερα. Η Unity μπορεί να επηρεαστεί από την απόσταση και τη θέση της πηγής από το αντικείμενο- ακροατή και να αναπαράγει ρεαλιστικούς ήχους. Η Unity δε μπορεί όμως να υπολογίσει την ηχώ καθαρά με βάση τη γεωμετρική μορφολογία της σκηνής. Παρ' όλα αυτά μπορεί να την προσομοιώσει προσθέτοντας τα κατάλληλα φίλτρα ήχου (audio filters) στα αντικείμενα.

Επίσης, σε καταστάσεις όπου τα αντικείμενα κινούνται εντός και εκτός μιας περιοχής και πρέπει να εναλλάσεται αισθητά ο ήχος κάθε φόρα, τότε συνιστάται η χρήση της Reverb zone στη σκηνή ώστε να προσομοιωθεί αυτή η εναλλαγή των ήχων.

Οι πιο βασικές υπομονάδες ήχου οι οποίες αναφέρθηκαν και παραπάνω είναι οι εξής: Audio source, Audio clip, Audio listener.

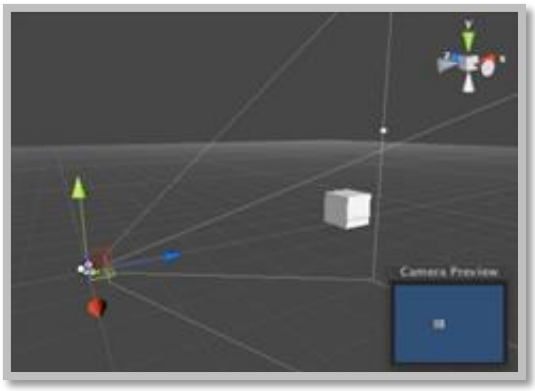
Audio source: αποτελεί την πηγή του ήχου και αναπαράγει ένα ηχητικό κλιπ το οποίο μπορεί να είναι δισδιάστατο ή τρισδιάστατο (spatial ή blend). Το component αυτό περιέχει ρυθμίσεις σχετικά με την ένταση του ήχου, την επαναληψιμότητα του, τον τόνο του, την προτεραιότητα του σε σχέση με άλλες πηγές καθώς και πληθώρα άλλων ρυθμίσεων και εφέ.

Audio clip: εμπεριέχει όλα τα ηχητικά δεδομένα που χρησιμοποιούνται από την πηγή ήχου (audio source). Η Unity υποστηρίζει μονοφωνικά, στερεοφωνικά και πολυκαναλικά στοιχεία ήχου. Η μορφές ήχου που μπορεί να εισάγει η Unity είναι aif, wav, mp3,και gg.

Audio listener: λειτουργεί σαν συσκευή μικροφώνου λαμβάνοντας είσοδο από τις πηγές ήχου της σκηνής και αναπαράγει ήχο από τα ηχεία της συσκευής. Πρέπει να υπάρχει πάντα ένας listener ενεργοποιημένος μια δεδομένη χρονική στιγμή καθ'όλη τη διάρκεια της εφαρμογής. Όπως αναφέραμε και παραπάνω συνήθως τοποθετείται στη κύρια κάμερα καθώς από εκει ο χρήστης ακούει και βλέπει [41].

Η **Camera** είναι το μέσον με το οποίο ο χρήστης βλέπει το ψηφιακό κόσμο. Μπορούν να ρυθμιστούν, να επεκτείνουν τις λειτουργίες τους μέσω script και να ομαδοποιηθούν

ιεραρχικά προκειμένου να επιτευχθεί οποιοδήποτε οπτικό αποτέλεσμα. Τοποθετείται σε οποιοδήποτε σημείο του χώρου και μπορεί να συνδεθεί με χαρακτήρες ή αντικείμενα της εφαρμογής. Είναι δυνατόν να υπάρχουν μια ή και περισσότερες κάμερες σε κάποια σκηνή, μία όμως εξ αυτών θεωρείται ως κύρια και θα προβεί στη διαδικασία του rendering [42]. Στη δική μας περίπτωση η κάμερα είναι μία και βρίσκεται πάνω στο αντικείμενο το οποίο κινείται στο χώρο. Με αυτόν τον τρόπο δίνεται η αίσθηση στον χρήστη προς τα που να κινηθεί. Επίσης, στην κάμερα έχει προστεθεί ένας audio listener μέσω του οποίου ο χρήστης μπορεί να ακούει τα διάφορα ηχητικά εφέ που υπάρχουν στον εικονικό κόσμο.



Εικόνα 30: Camera

4.13 Πλέγμα Πολυγώνων

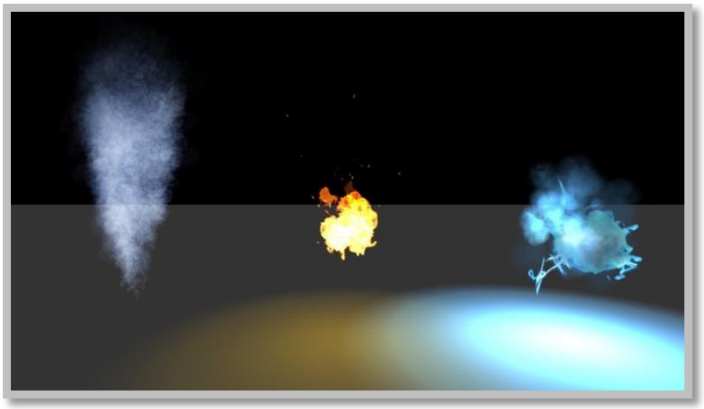
Ένα πλέγμα πολυγώνων αποτελείται από κορυφές (vertices), ακμές (edges) και όψεις (faces) ορίζοντας το σχήμα ενός 3D αντικειμένου. Οι κορυφές είναι σημεία στον τρισδιάστατο χώρο παρέχοντας πληροφορίες σχετικά με τη θέση, το χρώμα και την υφή τους (texture). Η ακμή είναι μια σύνδεση μεταξύ δύο κορυφών και μια όψη είναι ένα κλειστό σύνολο ακμών. Οι όψεις μπορούν να σχηματίζονται από τρίγωνα (3 ακμές), τετράγωνα (4 ακμές) ή κυρτά πολύγωνα (πέντε ή και περισσότερες ακμές). Μια κορυφή μπορεί να χρησιμοποιείται από πολλές γειτονικές όψεις ενώ μια ακμή από δύο μόνο. Οι όψεις μοιράζονται τις κορυφές σχηματίζοντας έτσι την τρισδιάστατη επιφάνεια ενός αντικειμένου. Ένα πλέγμα πολυγώνων είναι αρκετά εύκολο να αποδοθεί, ωστόσο τα περισσότερα υπολογιστικά υλικά απόδοσης (rendering, hardware) υποστηρίζουν μόνο τρίγωνα ή τετράγωνα. Θα πρέπει να σημειωθεί ότι τα περισσότερα λογισμικά γραφικών απαιτούν τα αντικείμενα να αποτελούνται εξ ολοκλήρου από τρίγωνα ή τετράγωνα ώστε να εξοικονομείται υπολογιστική ισχύς.

4.14 Particle System, Prefab και Assets

Particle System

Είναι μηχανισμός παραγωγής σωματιδίων (shuriken). Τα σωματίδια (particle) είναι μικρές εικόνες ή πλέγματα, που κινούνται ή απεικονίζονται κατά μεγάλες ποσότητες, όπως αυτά καθορίζονται από το σύστημα σωματιδίων. Κάθε σωματίδιο έχει προκαθορισμένη διάρκεια ζωής τυπικά λίγων δευτερολέπτων κατά την οποία μπορεί να υποστεί διάφορες αλλαγές. Τα σωματίδια αυτά μπορούν να ρυθμιστούν έτσι ώστε να επιτευχθεί το κατάλληλο

οπτικό εφέ επεμβαίνοντας στην ταχύτητα, στο χρώμα και το σχήμα που παίρνει το σύνολο των σωματιδίων. Το σύστημα σωματιδίων προσομοιώνει τις ρευστές οντότητες όπως υγρά σύννεφα, φλόγες και άλλα. Η μαζική παραγωγή των σωματιδίων αυτών δημιουργεί τη ψευδαίσθηση μη στερεών (ρευστών) οντοτήτων ή αντικειμένων [43].



Εικόνα 31: Particle System

Prefab

Τα prefabs αποτελούν προκαθορισμένες και αποθηκευμένες εκδόσεις ενός αντικειμένου ή σύνολο αντικειμένων με δυνατότητα εύκολης επαναχρησιμοποίησης σε διάφορα μέρη της εφαρμογής. Με τη χρήση των prefabs πολύπλοκα αντικείμενα με διάφορα στοιχεία και ρυθμίσεις μπορούν να χρησιμοποιηθούν ανά πάσα στιγμή, επιτρέποντας σε κάθε ένα ξεχωριστά να τροποποιηθεί όποτε και όταν αυτό απαιτείται [44].

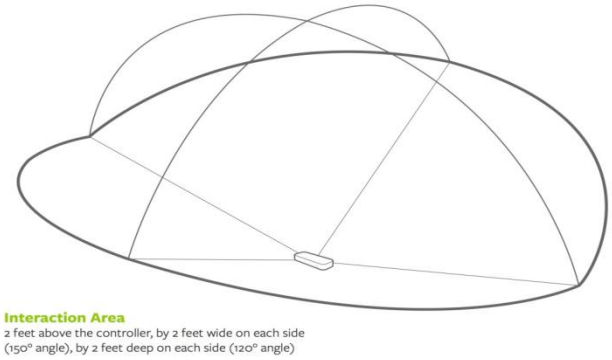
Assets

Είναι τα δομικά στοιχεία κάθε project της Unity και αναφέρονται σε όλα τα αρχεία που θα χρησιμοποιηθούν για τη δημιουργία μιας εφαρμογής όπως τα μοντέλα ήχου και άλλα.

4.15 Leap Motion

Το σύστημα Leap Motion αναγνωρίζει και παρακολουθεί τα χέρια και τα δάχτυλα του χρήστη. Η συσκευή λειτουργεί σε κοντινή απόσταση με υψηλή ακρίβεια και υψηλό ρυθμό παρακολούθησης και αναφέρει διακριτές θέσεις και κίνηση.

Από πλευράς υλικού (hardware), η συσκευή Leap Motion είναι σχετικά απλή. Αποτελείται από δύο κάμερες και τρία υπέρυθρα LED. Το υπέρυθρο φως που παράγουν έχει μήκος κύματος 850 νανομέτρων και βρίσκεται εκτός φάσματος ορατού φωτός.



Εικόνα 32: Πεδίο Hand Tracking του Leap Motion

Χάρη στους φακούς ευρείας γωνίας, η συσκευή διαθέτει ένα μεγάλο χώρο αλληλεπίδρασης οκτώ κυβικών ποδιών (ή 226.5348 λίτρα), το οποίο παίρνεις σχήμα ανεστραμμένης πυραμίδας. Το πεδίο προβολής του Leap Motion περιοριζόταν σε απόσταση 60cm από τη συσκευή. Με το λογισμικό Orion beta, κάτι το οποίο χρησιμοποιήσαμε, το πεδίο προβολής έχει επεκταθεί στα 80cm. Ο περιορισμός του εύρους προβολής οφείλεται στη διάδοση του υπέρυθρου φωτός στο χώρο, καθώς καθίσταται δυσκολότερο να αποδώσει τη θέση του χεριού σε τρεις διαστάσεις πέρα από μία ορισμένη απόσταση. Η ένταση του φωτός LED περιορίζεται τελικά από τη μέγιστη τιμή ρεύματος που μπορεί να περάσει μέσα από τη θύρα USB. Σε αυτό το σημείο, ο ελεγκτής USB (USB controller) της συσκευής, διαβάζει τα δεδομένα του αισθητήρα στην δική του τοπική μνήμη και πραγματοποιεί τις κατάλληλες ρυθμίσεις ανάλυσης. Αυτά τα δεδομένα μεταδίδονται στη συνέχεια μέσω του USB στο λογισμικό παρακολούθησης κίνησης του Leap Motion.

Εικόνα 33:Δεδομένα αισθητήρα σε μορφή στερεοσκοπικής εικόνας γκρι φάσματος

Τα δεδομένα παίρνουν τη μορφή στερεοσκοπικής εικόνας γκρι φάσματος χωρισμένα στις κάμερες αριστερά και δεξιά. Τυπικά, τα μοναδικά αντικείμενα που μπορούμε να δούμε είναι αυτά που φωτίζονται άμεσα από τις λυχνίες LED του Leap Motion.

Από άποψη λογισμικού, μόλις τα δεδομένα της εικόνας μεταδοθούν στον υπολογιστή, πραγματοποιούνται κάποιες πολύπλοκες μαθηματικές πράξεις. Το λογισμικό του Leap Motion εφαρμόζει προηγμένους αλγόριθμους στα ακατέργαστα δεδομένα του αισθητήρα. Το «Leap Motion Service» είναι το λογισμικό στον υπολογιστή που επεξεργάζεται τις εικόνες.

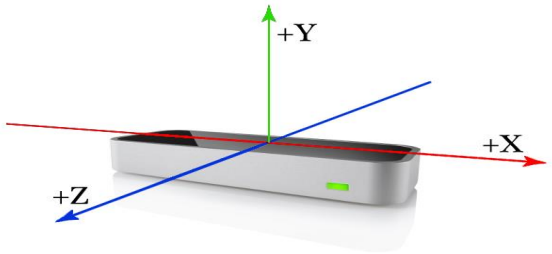
Μετά την αντιστάθμιση των αντικειμένων που βρίσκονται στο φόντο και του φωτός που διαχέεται στο περιβάλλον, οι εικόνες αναλύονται για να αναδημιουργήσουν μία τρισδιάστατη αναπαράσταση του τι βλέπει η συσκευή. Στη συνέχεια, ο «tracking layer» συσχετίζει τα δεδομένα έτσι ώστε να εξάγει διάφορες πληροφορίες παρακολούθησης, όπως τα δάχτυλα. Οι αλγόριθμοι παρακολούθησης αποκωδικοποιούν τα δεδομένα τριών διαστάσεων και συμπεραίνουν τις θέσεις των αντικειμένων. Χρησιμοποιούνται τεχνικές φιλτραρίσματος για την εξασφάλιση ομαλής χρονικής συνοχής των δεδομένων. Το «Leap Motion Service» μεταφέρει τα αποτελέσματα, εκφρασμένα ως μια σειρά από frames ή snapshots που περιέχουν όλα τα δεδομένα παρακολούθησης, σε ένα πρωτόκολλο μεταφοράς.

Μέσω αυτού του πρωτοκόλλου, το «Leap Motion Service» επικοινωνεί με το Control Panel (πίνακας ελέγχου) του Leap Motion, καθώς και με τις web client libraries (βιβλιοθήκες), μέσω τοπικής σύνδεσης socket. Η client library οργανώνει τα δεδομένα σε μία αντικειμενοστραφής API δομή, η οποία διαχειρίζεται το ιστορικό των frame και παρέχει βοηθητικές λειτουργίες. Από εδώ και πέρα, η λογική αυτή, εφαρμόζεται στην είσοδο του Leap Motion επιτυγχάνοντας μία διαδραστική εμπειρία κίνησης.

Εικόνα 34: Η άποψη του Motion Leap για τα χέρια σας

Η ανίχνευση και η παρακολούθηση αποδίδουν καλύτερα όταν το Leap Motion έχει μια καθαρή εικόνα της σιλουέτας ενός αντικειμένου. Μπορεί να καταγράψει έως 200 frames/second. Το λογισμικό του Leap Motion συνδυάζει τα δεδομένα του αισθητήρα με ένα εσωτερικό μοντέλο του ανθρώπινου χεριού για να αντιμετωπίσει τις διάφορες δυσκολίες της παρακολούθησης. Το οπτικό πεδίο του Leap Motion έχει 150^{o} πλάτος και 120^{o} βάθος. Άρα κατά μέσο όρο έχει FoV 135^{o}.

Το σύστημα Leap Motion χρησιμοποιεί ένα καρτεσιανό σύστημα συντεταγμένων δεξιού χεριού. Το καρτεσιανό σύστημα είναι κεντραρισμένο στο επάνω μέρος του Leap Motion. Οι άξονες x και z βρίσκονται στο οριζόντιο επίπεδο, με τον άξονα x να είναι παράλληλος προς το επίμηκες άκρο της συσκευής. Ο άξονας y είναι κάθετος, με τις θετικές τιμές να αυξάνονται προς τα πάνω (σε αντίθεση με τον προς τα κάτω προσανατολισμό των περισσότερων συστημάτων συντεταγμένων γραφικών). Ο άξονας z έχει θετικές τιμές που αυξάνονται προς το χρήστη.



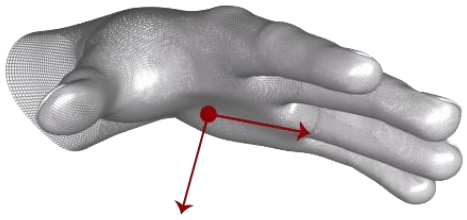
Εικόνα 35: Το σωστό σύστημα συντεταγμένων Leap Motion

Το API Leap Motion μετράει τις φυσικές ποσότητες με τις ακόλουθες μονάδες:

Απόσταση:	Χιλιοστά
Χρόνος:	μικροδευτερόλεπτα (εκτός αν αναφέρεται διαφορετικά)
Ταχύτητα:	χιλιοστά/δευτερόλεπτο
Γωνία:	Ακτίνια

Καθώς το Leap Motion παρακολουθεί τα χέρια και τα δάχτυλά του χρήστη στο οπτικό του πεδίο, παρέχει ενημερώσεις ως σύνολο ή πλαίσιο δεδομένων. Κάθε Frame Object που αντιπροσωπεύει ένα πλαίσιο, περιέχει τυχόντα χέρια που παρακολουθούνται, με λεπτομερή περιγραφή των ιδιοτήτων τους σε μία μόνο χρονική στιγμή. Το Frame Object (αντικείμενο Πλαίσιο) είναι ουσιαστικά η ρίζα του μοντέλου δεδομένων Leap Motion.

Το μοντέλο χειρός παρέχει πληροφορίες σχετικά με την ταυτότητα, τη θέση και άλλα χαρακτηριστικά ενός χεριού που ανιχνεύεται όπως τον βραχίονα στον οποίο είναι συνδεδεμένο το χέρι και διάφορες άλλες πληροφορίες σχετικές με τα δάκτυλα του συγκεκριμένου χεριού. Τα χέρια αντιπροσωπεύονται από την κατηγορία Hand.



Εικόνα 36: Το "χέρι" PalmNormal και η κατεύθυνση του διανύσματος (Direction vectors) ορίζουν τον προσανατολισμό του χεριού

Το λογισμικό Leap Motion χρησιμοποιεί ένα εσωτερικό μοντέλο ανθρώπινου χεριού για την «προβλεπόμενη παρακολούθηση» ακόμη και όταν δεν είναι ορατά όλα τα τμήματα ενός χεριού. Το μοντέλο χειρός παρέχει πάντα θέσεις για πέντε δάχτυλα, αν και η ανίχνευση είναι βέλτιστη όταν η μορφή ενός χεριού και όλων των δακτύλων του είναι ξεκάθαρα ορατά.

Το λογισμικό χρησιμοποιεί τα ορατά μέρη του χεριού, το εσωτερικό του μοντέλο και παρελθούσες παρατηρήσεις για τον υπολογισμό των πιθανότερων θέσεων των τμημάτων που δεν είναι ορατά. Οι κινήσεις των δακτύλων που σφίγγουν το χέρι απλώς δεν είναι ανιχνεύσιμες. Η βαθμολογία Hand.Confidence φανερώνει πόσο καλά εναρμονίζονται τα παρατηρούμενα δεδομένα στο εσωτερικό μοντέλο.

Περισσότερα από δύο χέρια μπορούν να εμφανιστούν στη λίστα χεριών που αντιστοιχεί σε ένα frame, εάν υπάρχουν παραπάνω από ένας χρήστης. Ωστόσο, συνιστάται να υπάρχουν το πολύ δύο χέρια στο οπτικό πεδίο του Leap Motion για βέλτιστη ποιότητα παρακολούθησης κίνησης.

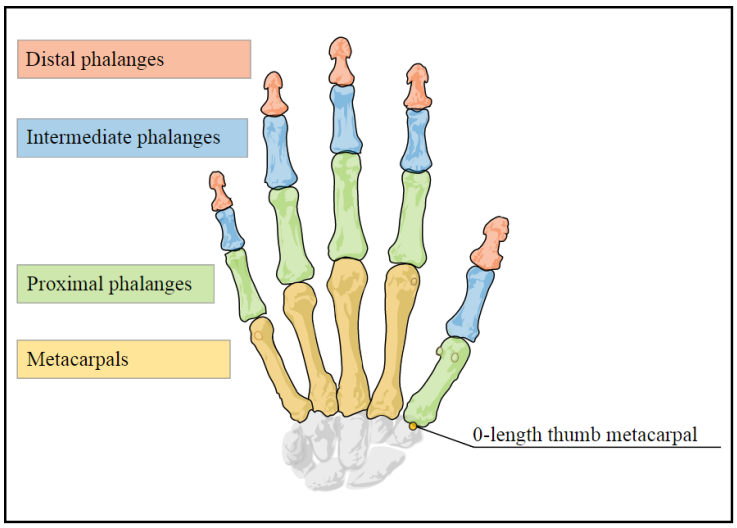
Ένας βραχίονας είναι ένα αντικείμενο που μοιάζει με οστό και παρέχει τον προσανατολισμό, το μήκος, το πλάτος και τα τελικά σημεία ενός βραχίονα. Όταν ο αγκώνας δεν είναι ορατός, το Leap Motion υπολογίζει τη θέση του βάσει προηγούμενων παρατηρήσεων καθώς και της τυπικής ανθρώπινης αναλογίας.

Το Leap Motion παρέχει πληροφορίες για κάθε δάχτυλο ενός χεριού. Εάν ολόκληρο ή μέρος του δακτύλου δεν είναι ορατό, τα χαρακτηριστικά των δακτύλων υπολογίζονται με βάση τις πρόσφατες παρατηρήσεις και το ανατομικό μοντέλο του χεριού. Τα δάχτυλα αναγνωρίζονται από το όνομα τους, δηλαδή αντίχειρας, δείκτης, μέσος, παράμεσος και μικρό δάκτυλο. Τα δάχτυλα αντιπροσωπεύονται από την κλάση Finger.

Εικόνα 37: Τα σύμβολα "TipPosition" και "Direction" του δάκτυλου παρέχουν τη θέση ενός άκρου δακτύλου και τη γενική κατεύθυνση στην οποία δείχνει ένα δάχτυλο.

Ένα αντικείμενο Finger παρέχει ένα αντικείμενο Bone που περιγράφει τη θέση και τον προσανατολισμό κάθε ανατομικού δακτύλου. Κάθε δάχτυλο σχηματίζεται από τέσσερα οστά που ξεκινάνε από τη βάση, από το τέλος του βραχίονα έως το άκρον, εκτός του αντίχειρα ο οποίος περιλαμβάνει τρία οστά. Τα οστά αναγνωρίζονται ως:

- μετακάρπιο - οστό της παλάμης που συνδέει το δάκτυλο με τον καρπό (εκτός από τον αντίχειρα)
- άνω φάλαγγα - οστό στη βάση του δακτύλου που συνδέεται με την παλάμη
- ενδιάμεση φάλαγγα- το μέσο οστό του δακτύλου μεταξύ άκρης και βάσης
- ονυχοφόρα φάλαγγα - το τερματικό οστό στο τέλος του δακτύλου



Εικόνα 38: Ανατομία χεριού

Αυτό το μοντέλο για τον αντίχειρα δεν ταιριάζει αρκετά με το τυπικό ανατομικό σύστημα ονομασίας. Ένας πραγματικός αντίχειρας έχει ένα λιγότερο οστό από τα άλλα δάχτυλα. Ωστόσο, για ευκολία στον προγραμματισμό, το μοντέλο αντίχειρα Leap Motion περιλαμβάνει μετακαρπικό οστό μηδενικού μήκους έτσι ώστε ο αντίχειρας να έχει τον ίδιο αριθμό οστών με τα άλλα δάχτυλα. Ως αποτέλεσμα, το ανατομικό μετακάρπιο οστό του αντίχειρα επισημαίνεται ως εγγύς φάλαγγα και η ανατομική εγγύς φαλάγγα επισημαίνεται ως η ενδιάμεση φάλαγγα στο μοντέλο των δακτύλων. Μαζί με τα υπολογισμένα δεδομένα παρακολούθησης μπορούμε να πάρουμε τις πρώτες εικόνες αισθητήρα από τις κάμερες Leap Motion [45].

Επίσης, δίνεται η δυνατότητα στο Leap Motion να τοποθετηθεί πάνω στο Head Mount Display ώστε να ακολουθάει οποιαδήποτε κίνηση της κεφαλής με απαραίτητη προϋπόθεση την εισαγωγή των κατάλληλων βιβλιοθηκών από την επίσημη ιστοσελίδα. Με αυτήν την τεχνολογία επιτυγχάνεται η σχεδόν μηδενική καθυστέρηση (latency) και χαμηλή υπολογιστική ισχύς από πλευράς Leap Motion, μεγαλύτερη εμβέλεια 180x180 βαθμών του πεδίου εντοπισμού πέρα από τον βραχίονα, συμβατότητα με όλα τα νέα μοντέλα κινητών τηλεφώνων που περιέχουν Android λογισμικό.

4.16 Oculus Rift

4.16.1 Εισαγωγή

Στη συγκεκριμένη εργασία το HMD που χρησιμοποιήθηκε είναι το Oculus Rift consumer version 1. Παρέχει στο χρήστη τη δυνατότητα εμβύθισης (immersion) σ’ ένα εικονικό περιβάλλον προσφέροντας ταυτόχρονα μια πολύ έντονη εμπειρία ρεαλιστικότερη απ’ αυτή μιας οθόνης. Οι λόγοι που επιλέχθηκε το Oculus Rift Consumer Version 1 είναι ότι θεωρείται το πλέον προηγμένο μοντέλο μέχρι σήμερα της εταιρίας Oculus μετά και τις αναβαθμίσεις που έχουν προηγηθεί. Παρέχει απλή και γρήγορη ενσωμάτωση (integration) της Unity3D προσφέροντας συνάμα εντυπωσιακά αποτελέσματα. Δυνατότητα χρήσης μέσα από τα εργαστήρια της σχολής.

Εικόνα 39: Oculus Rift Consumer Version 1

4.16.2 Binocular Vision

Το Binocular Vision περιγράφει τον τρόπο με τον οποίο βλέπουμε δύο όψεις του κόσμου ταυτόχρονα. Η θέα από κάθε μάτι είναι ελαφρώς διαφορετική όμως ο εγκέφαλος τις συνδυάζει σε μια ενιαία τρισδιάστατη στερεοσκοπική εικόνα, μια εμπειρία γνωστή ως στερέοψη (stereopsis). Η διαφορά ανάμεσα σε αυτό που βλέπουμε από το αριστερό μάτι μας και αυτό που βλέπουμε από το δεξί παράγει τη διόφθαλμη διασπορά (binocular disparity).

Το Oculus Rift παρέχει δύο εικόνες, μια για κάθε μάτι, οι οποίες παράγονται από δύο εικονικές κάμερες και οι οποίες χωρίζονται από μια μικρή απόσταση. Η απόσταση μεταξύ των δύο οφθαλών μας καλείται inter-pupilarity distance (IPD), και η απόσταση μεταξύ των δύο καμερών που συλλαμβάνουν το εικονικό περιβάλλον inter-camera distance (ICD). Παρά το γεγονός ότι η IPD μπορεί να κυμαίνεται περίπου από 52 έως 78 χιλιοστά, κατά μέσο όρο είναι περίπου 63,5, το ίδιο όπως η απόσταση μεταξύ των κέντρων των φακών του Rift.

Αλλαγή της απόσταση μεταξύ των καμερών, μπορεί να επηρεάσει τους χρήστες σημαντικά. Αν η απόσταση μεταξύ των καμερών είναι αυξημένη, δημιουργεί μια εμπειρία γνωστή ως hyperstereo στην οποία το βάθος αυξάνεται. Αν η απόσταση μειωθεί τότε, θα μειωθεί και το βάθος, μια κατάσταση γνωστή ως hypostereo. Η αλλαγή στην απόσταση μεταξύ καμερών επιφέρει δύο επιπλέον συνέπειες στο χρήστη: Πρώτον, αλλάζει ο βαθμός στον οποίον τα μάτια πρέπει να συγκλίνουν για να δουν ένα συγκεκριμένο αντικείμενο καθώς όσο αυξάνεται η απόσταση μεταξύ των καμερών, οι χρήστες πρέπει να συγκλίνουν περισσότερο τους οφθαλμούς τους για να δουν το ίδιο αντικείμενο γεγονός που μπορεί να οδηγήσει σε καταπόνηση των ματιών. Δεύτερον, μπορεί να αλλάξει την αίσθηση του μεγέθους του χρήστη μέσα στο εικονικό περιβάλλον [46].

4.16.3 Field of View & Scale

Το οπτικό πεδίο Field of View μπορεί να αναφέρεται σε διαφορετικές περιπτώσεις που θα πρέπει πρώτα να αποσαφηνιστούν. Υπάρχουν δύο ειδών οπτικά πεδία, το display Field of View (dFOV) και το camera Field of View (cFOV).

Το dFOV αναφέρεται στο φυσικό οπτικό πεδίο του χρήστη το οποίο καταλαμβάνεται από περιεχόμενο εικονικής πραγματικότητας. Είναι ένα φυσικό χαρακτηριστικό του υλικού (hardware) και των οπτικών. Στην περίπτωση του Oculus Rift CV1 έχουμε 110^{o} FOV (80^{o} οριζόντια, 90^{o} κάθετα, 120^{o} διαγώνια).

Το cFOV αναφέρεται στο εύρος του εικονικού κόσμου, που φαίνεται από την απόδοση των καμερών σε οποιαδήποτε δεδομένη στιγμή. Δηλαδή είναι το εύρος του οπτικού πεδίου των καμερών μέσα στον εικονικό κόσμο και μπορεί να ρυθμιστεί μέσα από την εφαρμογή ανάλογα τις ανάγκες του χρήστη. Όλα τα οπτικά πεδία FOVs ορίζονται από μια γωνιακή μέτρηση των κάθετων, οριζόντιων και / ή διαγώνιων διαστάσεων.

Στην εικονική πραγματικότητα, δεν υπάρχει επαφή του εξωτερικού περιβάλλοντος και ο εικονικός κόσμος γεμίζει ένα μεγάλο μέρος της περιφερειακής όρασης. Συνεπώς, είναι πολύ σημαντικό το cFOV και το dFOV να ταιριάζουν ακριβώς. Η αναλογία μεταξύ αυτών των δύο τιμών αναφέρεται ως κλίμακα (scale), και στην εικονική πραγματικότητα η κλίμακα αυτή θα πρέπει πάντα να είναι ίση με ένα. Στο Oculus Rift CV1, το μέγιστο dFOV καθορίζεται από την οθόνη, τους φακούς, και το πόσο κοντά ο χρήστης τοποθετεί τους φακούς στα μάτια του (σε γενικές γραμμές, όσο πιο κοντά είναι τα μάτια στους φακούς, τόσο ευρύτερο είναι το dFOV). Το βοηθητικό πρόγραμμα διαμόρφωσης μετρά το μέγιστο dFOV που οι χρήστες μπορούν να δουν, αποθηκεύει αυτές τις πληροφορίες και το SDK θα συστήσει το κατάλληλο cFOV. Έτσι το SDK επιτρέπει τον χειρισμό του cFOV και του dFOV χωρίς αλλαγή της κλίμακας και επιτυγχάνεται δε αυτό με την προσθήκη μαύρων πλαισίων γύρω από την ορατή εικόνα [47].

4.16.4 Rendering Techniques

Παρακάτω αναλύονται βασικές τεχνικές απόδοσης (Rendering Techniques) καθώς και τα χαρακτηριστικά τους, τις οποίες χρησιμοποιεί το Oculus Rift για να αποδώσει με όσο το δυνατόν καλύτερο και ρεαλιστικότερο τρόπο το εικονικό περιβάλλον στο χρήστη. Αξίζει να σημειωθεί ότι στο Oculus Rift consumer version 1 έχουν επέλθει σημαντικές βελτιώσεις σε σχέση με το Oculus Rift DK2 όπως λεπτομερέστερη ανάλυση, από 1920x1080 σε 2160x1200 (1080x1200 ανά μάτι), καθώς και ταχύτερος ρυθμός ανανέωσης εικόνας, από 75hz σε 90hz, ο οποίος έχει σαν αποτέλεσμα τη μείωση του θαμπώματος (motion blur) και του flickering (τρεμόσβημα) κατά τη μετακίνηση της κεφαλής. Το πρώτο μοντέλο Oculus Rift DK1 είχε ανάλυση 1280x720 και ρυθμό ανανέωσης 60hz.

Rendering resolution: Όπως αναφέρθηκε και παραπάνω, το Oculus Rift consumer version 1 έχει ανάλυση οθόνης 2160x1200 (1080x1200 ανά μάτι), αλλά η παραμόρφωση των φακών σημαίνει ότι η εικόνα στην οθόνη πρέπει να μετασχηματιστεί ώστε να εμφανιστεί κανονική στο θεατή. Προκειμένου να παρασχεθεί επαρκής πυκνότητα εικονοστοιχείων για τον μετασχηματισμό, κάθε μάτι απαιτεί εικόνα η οποία να είναι μεγαλύτερη από την ανάλυση του μισού της οθόνης. Αυτό μπορεί να αποτελέσει πρόβλημα απόδοσης για ορισμένες κάρτες γραφικών και η μείωση του ρυθμού των καρέ παράγει μια κακή εμπειρία εικονικής πραγματικότητας. Ρίχνοντας την ανάλυση της οθόνης, μπορεί να έχει επίδραση και να δημιουργήσει μορφώματα. Ρίχνοντας όμως την ανάλυση των ρυθμιστών στο μάτι, μπορεί να βελτιωθούν οι επιδόσεις, διατηρώντας την οπτική ποιότητα[48].

4.16.5 Παρακολούθηση (Tracking)

Το Oculus Rift περιέχει ένα γυροσκόπιο, ένα επιταχυνσιόμετρο και ένα μαγνητόμετρο και συνδυάζει αυτές τις πληροφορίες που λαμβάνει από τους αισθητήρες μέσω μιας διαδικασίας που ονομάζεται «sensor fusion» για να καθορίσει τον προσανατολισμό του κεφαλιού του χρήστη στον πραγματικό κόσμο και να συγχρονίσει την εικονική πλευρά με τον πραγματικό περιβάλλον. Αυτοί οι αισθητήρες παρέχουν δεδομένα για να παρακολουθείται με ακρίβεια η γωνιακή παρέκκλιση του κεφαλιού (pitch, yaw, roll). Έτσι λοιπόν, έχει δημιουργηθεί ένα απλό μοντέλο κεφαλής και λαιμού του χρήστη το οποίο είναι χρήσιμο για την ακριβή αποκωδικοποίηση των πληροφοριών των αισθητήρων για την μετατροπή των κινήσεων της κεφαλής σε κινήσεις κάμερας.

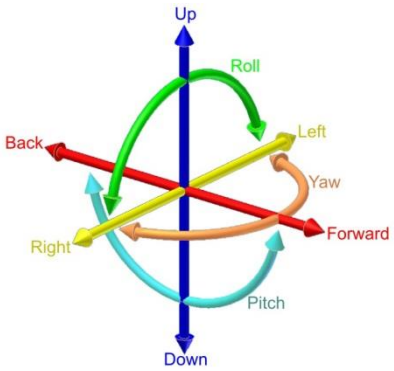
Κάτω από το ημιδιαφανές εξωτερικό περίβλημα του Oculus Rift consumer version 1, υπάρχει μια συστοιχία υπέρυθρων micro-LEDs, που παρακολουθούνται από μία κάμερα υπέρυθρων που βρίσκεται στο χώρο. Έχει 360^{o} δυνατότητα εντοπισμού χρήστη. Το εύρος του πεδίου εντοπισμού δεν έχει γνωστοποιηθεί ακόμα από την εταιρία. Η παρακολούθηση θέσης (Positional tracking) θα πρέπει να αντιστοιχεί απόλυτα με τις κινήσεις του χρήστη ειδάλλως μπορεί να προκληθεί δυσφορία. Ένας βασικός παράγοντας του tracking που θα πρέπει να λαμβάνεται σοβαρά υπόψιν είναι η λανθάνουσα κατάσταση (latency), η οποία ορίζεται ως ο συνολικός χρόνος μεταξύ της κίνησης του κεφαλιού του χρήστη και της ανανεωμένης εικόνας που εμφανίζεται στην οθόνη και περιλαμβάνει τους χρόνους για την απόκριση του αισθητήρα, τη σύντηξη, την απόδοση, τη μετάδοση της εικόνας και την απόκριση προβολής. Η ελαχιστοποίηση λοιπόν, της λανθάνουσας κατάστασης είναι ζωτικής σημασίας για τον χρήστη, για μια καλή εμπειρία εμβύθισης σε εικονικό περιβάλλον[50].

Δηλαδή, το Oculus Rift consumer version 1 εισάγει έξι βαθμούς ελευθερίας εντοπισμού θέσης. Ο όρος 6DoF αναφέρεται στην ελεύθερη κίνηση ενός άκαμπτου σώματος στον τρισδιάστατο χώρο. Πιο συγκεκριμένα, το σώμα είναι ελεύθερο να αλλάζει θέση πάνω στους τρεις κάθετους άξονες προς τα εμπρός/προς τα πίσω, προς τα αριστερά/προς τα δεξιά, προς τα πάνω/προς τα κάτω σε συνδυασμό με αλλαγές στον προσανατολισμό μέσω περιστροφής γύρω από αυτούς τους κάθετους άξονες (aircraft principal axes).

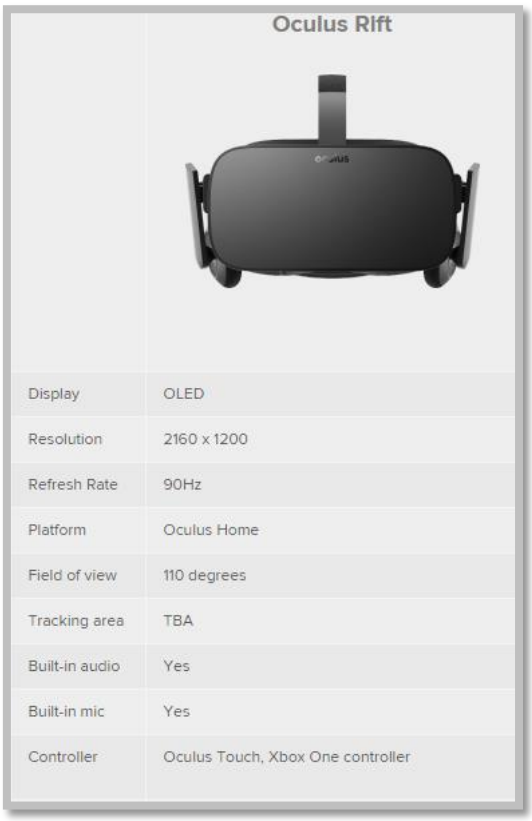
Yaw: περιστροφή γύρω από πάνω/κάτω άξονα,

Pitch: περιστροφή γύρω από αριστερά/δεξιά άξονα

Roll: περιστροφή γύρω από εμπρός/πίσω άξονα



Εικόνα 40: 6DoF



Εικόνα 41: Oculus Rift CV1 Features

4.17 Ασθένεια Εικονικής Πραγματικότητας (Virtual Reality Sickness)

Η ασθένεια εικονικής πραγματικότητας εμφανίζεται όταν η έκθεση σε ένα εικονικό περιβάλλον προκαλεί συμπτώματα παρόμοια με αυτά της ασθένειας της κίνησης (travel sickness). Η ναυτία της προσομοίωσης (ζάλη) συμβαίνει όταν η οπτική πληροφορία από ένα περιβάλλον προσομοίωσης δημιουργεί σύγχυση, μπερδεύοντας τον χρήστη ως προς τη θέση, την ισορροπία και τον τρόπο κίνησης του. Εκτός της ζάλης τα πιο συνηθισμένα συμπτώματα είναι η γενική δυσφορία, ο πονοκέφαλος, η εφίδρωση, η κόπωση, η υπνηλία, ο αποπροσανατολισμός, η στομαχική διαταραχή, ο εμετός και η κόπωση των ματιών. Οι λόγοι που προκαλούν τα ανωτέρω περιγράφονται παρακάτω.

Η ταχύτητα κίνησης και επιτάχυνσης (speed of movement and acceleration) είναι ένας από τους λόγους που ο χρήστης αντιμετωπίζει κάποια από τα παραπάνω συμπτώματα. Η ταχύτητα της κίνησης είναι άμεσα ανάλογη της κίνησης εμφάνισης της ασθένειας. Αν και οι πιο αργές ταχύτητες γενικά κάνουν τον χρήστη να αισθάνεται άνετα, το πιο σημαντικό θέμα είναι η επιτάχυνση, ερέθισμα στο οποίο ανταποκρίνονται τα αιθουσαία όργανα στο εσωτερικό του αυτιού. Η εικονική επιτάχυνση, η οποία γίνεται αντιληπτή οπτικά αλλά όχι και από τα αιθουσαία όργανα, μπορεί να οδηγήσει σε σύγκρουση των αισθήσεων προκαλώντας έτσι διαταραχές όπως η δυσφορία. Μια άλλη αιτία εκδήλωσης της ασθένειας της εικονικής πραγματικότητας είναι ο βαθμός ελέγχου (Degree of control). Η αφαίρεση του ελέγχου της εικονικής κάμερας ή πρόκληση κινήσεων οι οποίες δεν πραγματοποιούνται από το χρήστη μπορεί να οδηγήσουν στην εκδήλωση της ασθένειας. Ακόμα, η διάρκεια

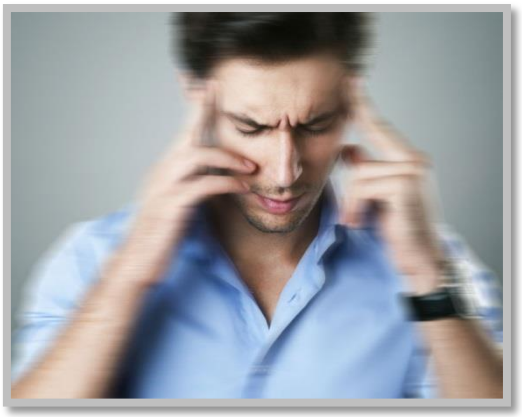
(Duration) αλληλεπίδρασης του χρήστη με το εικονικό περιβάλλον επηρεάζει τον οργανισμό του. Όσο περισσότερο παραμένει ο χρήστης στο εικονικό περιβάλλον τόσο πιθανότερο είναι να προκληθεί η ασθένεια προσομοίωσης. Επομένως οι χρήστες πρέπει πάντα να έχουν την ελευθερία να «παγώνουν» το παιχνίδι τους και στη συνέχεια να επιστρέφουν στο ακριβές σημείο όπου βρίσκονταν.

Το ύψος του χρήστη, δηλαδή το ύψος της οπτικής γωνίας αυτού (POV) μπορεί να είναι ένας έμμεσος παράγοντας που μπορεί να προκαλέσει την ασθένεια προσομοίωσης. Όσο χαμηλότερα είναι η οπτική γωνία του χρήστη τόσο πιο γρήγορα γίνονται οι αλλαγές στο επίπεδο του εδάφους, προκαλώντας εντονότερα την οπτική ροή, γεγονός που μπορεί επίσης να προκαλέσει την ασθένεια. Μια άλλη αιτία εμφάνισης της ασθένειας είναι η διόφθαλμη διασπορά. Αν και η διόφθαλμη διασπορά είναι βασικός παράγοντας για την ρεαλιστικότερη απεικόνιση του βάθους, δεν πρέπει παρ' όλα αυτά να αγνοήσουμε τις αρνητικές επιπτώσεις που προκύπτουν απ' αυτήν. Οι στερεοσκοπικές εικόνες μπορεί να ωθήσουν τους οφθαλμούς να συγκλίνουν σ' ένα σημείο σε κάποιο βάθος ενώ οι φακοί να εστιάσουν σε κάποιο άλλο σημείο. Έτσι ορισμένοι άνθρωποι παρατηρώντας στερεοσκοπικές εικόνες αισθάνονται δυσφορία, γεγονός που οδήγησε τους ερευνητές να προτείνουν μείωση της απόστασης μεταξύ των δύο εικονικών καμερών ή και εξάλειψη αυτής (μονοσκοπική απεικόνιση).

Ένας άλλος λόγος που προκαλείται η ασθένεια είναι το πεδίο προβολής (field of view). Το πεδίο προβολής αναφέρεται σε δύο είδη οπτικών πεδίων. Το πρώτο είναι η περιοχή του οφθαλμικού πεδίου που καλύπτεται από την οθόνη (FOV ή dFOV) και η περιοχή του εικονικού περιβάλλοντος που απεικονίζει η μηχανή γραφικών στην οθόνη (camera FOV ή cFOV). Ένα ευρύ dFOV είναι πιθανότερο να συμβάλλει στην ασθένεια προσομοίωσης για δύο λόγους που σχετίζονται με την αντίληψη της κίνησης. Πρώτον, η αντίληψη της κίνησης είναι πιο ευαίσθητη στην περιφέρεια της όρασης και δεύτερον όταν μια μεγαλύτερη οθόνη FOV χρησιμοποιείται στο σύνολο της και υποδεικνύει στο χρήστη ότι κινείται, μπορεί να προξενήσει σύγχυση μεταξύ των αισθήσεων τού, οδηγώντας τον έτσι σε δυσφορία. Ο άσκοπος χειρισμός του cFOV μπορεί να οδηγήσει σε αφύσικη μετακίνηση του εικονικού περιβάλλοντος σε σχέση με τις κινήσεις του κεφαλιού και των οφθαλμών. Οι οφθαλμοί και το αιθουσαίο σύστημα συνεργάζονται για να καθορίσουν πόσο πρέπει να κινούνται τα μάτια κατά την κίνηση του κεφαλιού, ώστε να διατηρείται σταθερή η οπτική εστίαση σ' ένα αντικείμενο. Αν το εικονικό περιβάλλον οδηγήσει σε αποτυχία τότε θα προκληθεί ασθένεια.

Παρόλο που οι προγραμματιστές δεν έχουν τον έλεγχο σε πολλές πτυχές της λανθάνουσας κατάστασης (Latency and lag), είναι σημαντικό να διασφαλίζεται ότι η εφαρμογή στο εικονικό περιβάλλον δεν θα κολλάει (lag) ή δεν θα μειώνει τα frames σ' ένα σύστημα που πληροί τις ελάχιστες τεχνικές προδιαγραφές. Πολλά παιχνίδια μπορεί να επιβραδυνθούν εξαιτίας πολλών ή πιο συνθετών στοιχείων που επεξεργάζονται και αποδίδονται στην οθόνη. Αν και αυτό είναι μια μικρή ενόχληση στα παραδοσιακά βιντεοπαιχνίδια, μπορεί να αποτελέσει μια δυσάρεστη εμπειρία για τους χρήστες της εικονικής πραγματικότητας. Οι φακοί του Oculus Rift στρεβλώνουν την εικόνα που εμφανίζεται στην οθόνη. Η διόρθωση επιτυγχάνεται με τη δυνατότητα επεξεργασίας που παρέχεται από το SDK. Η εσφαλμένη παραμόρφωση μπορεί να "φαίνεται" αρκετά σωστή αλλά εξακολουθεί να είναι αποπροσανατολιστική και άβολη για το χρήστη οπότε χρειάζεται ιδιαίτερη προσοχή στις λεπτομέρειες. Όλες οι τιμές διόρθωσης προσομοίωσης πρέπει να ταιριάζουν με τη συγκεκριμένη συσκευή και καμία από αυτές τις τιμές δεν πρέπει να ρυθμίζεται από το χρήστη. Όλοι οι προγραμματιστές πρέπει να χρησιμοποιούν τις επίσημες ρυθμίσεις παραμόρφωσης του Oculus VR προκειμένου να εμφανιστεί το σωστό αποτέλεσμα. Συνήθως χρησιμοποιούνται οι σκιαστές παραμόρφωσης του Oculus VR.

Το τρεμόσβημα μπορεί να γίνει εντονότερο εξαιτίας των υψηλών επιπέδων φωτεινότητας. Γίνεται δε περισσότερο αντιληπτό στην περιφέρεια του οφθαλμού. Παρά το γεγονός ότι το τρεμόσβημα μπορεί να γίνει λιγότερα συνειδητό με την πάροδο του χρόνου ενδέχεται να οδηγήσει σε πονοκεφάλους και κόπωση των ματιών [51].



Εικόνα 42: Virtual Reality Sickness

Κεφάλαιο 5ο Σχεδιασμός Παιχνιδιού

5.1 Εισαγωγή

Στην ενότητα αυτή γίνεται περιγραφή του παιχνιδιού ως προς τη σχεδίαση και το χειρισμό του όπως και άλλων βασικών χαρακτηριστικών προκειμένου να προετοιμαστεί κατάλληλα ο χρήστης ώστε να ανταπεξέλθει στις απαιτήσεις της εφαρμογής. Επίσης, περιγράφεται η συμπεριφορά των διαφόρων εχθρών και χαρακτήρων η οποία είναι βασισμένοι στην τεχνητή νοημοσύνη. Τέλος, παρουσιάζονται οι διάφορες διεπαφές χρήστη και τα ηχητικά εφέ που εμπλουτίσανε την εφαρμογή.

5.2 Σενάριο και Σύντομη Περιγραφή του Παιχνιδιού

Ο τίτλος της εφαρμογής είναι Eternal Conflict. Πρόκειται για Gaming (παιχνίδι) εφαρμογής First Person Shooter (πρώτου προσώπου) στο οποίο δε χρησιμοποιούνται όπλα αλλά μαγείες και ενεργειακά κύματα που θα εκτοξεύονται από τα χέρια του παίχτη χωρίς τη χρήση χειριστηρίου, πληκτρολογίου και ποντικιού. Όλες οι ενέργειες θα πραγματοποιούνται με τη βοήθεια των τεχνολογικών μέσων Oculus Rift και Leap Motion.

Πιο συγκεκριμένα, ο χρήστης έχει τη δυνατότητα να γυρίζει προς κάθε κατεύθυνση μέσα στο εικονικό περιβάλλον εκμεταλλευόμενος τις περιστροφικές κινήσεις της κεφαλής του στον πραγματικό κόσμο, κάτι το οποίο επιτυγχάνεται μέσω του Oculus Rift και του γυροσκοπίου το οποίο είναι ενσωματωμένο σε αυτο. Επιπλέον, με συγκεκριμένες κινήσεις του αριστερού χεριού που ορίσαμε εμείς μέσω του Leap Motion, οι οποίες περιγράφονται αναλυτικά παρακάτω, δίνεται η δυνατότητα στο χρήστη να εκτελεί ευθύγραμμες κινήσεις με σταθερή ταχύτητα (έμπροσθεν, αριστερή δεξιά, όπισθεν και διαγώνιες). Έτσι, συνδυάζοντας τα δυο παραπάνω είδη κινήσεων, επιτυγχάνεται ευχερής ελευθερία κινήσεων μέσα στο εικονικό περιβάλλον.

Επιπρόσθετα, ο χρήστης έχει τη δυνατότητα να εκτελεί συγκεκριμένες επιθέσεις σε μορφή μαγείας. Οι επιθέσεις διακρίνονται σε τρεις κατηγορίες με διαφορετικές ιδιότητες η κάθε μία (μπάλα φωτιάς, μπάλα πάγου, πράσινη μπάλα). Για την επιτυχή εκτέλεση κάποιας εκ των επιθέσεων, εκμεταλλευτήκαμε την ταχύτητα κίνησης της παλάμης προς ευθύγραμμη έμπροσθεν κατεύθυνση, γεγονός που ήταν δυνατόν να επιτευχθεί μόνο μέσω του Leap Motion. Τέλος, πρέπει να σημειωθεί ότι μπορεί να γίνει συνδυασμός όλων των παραπάνω κινήσεων.

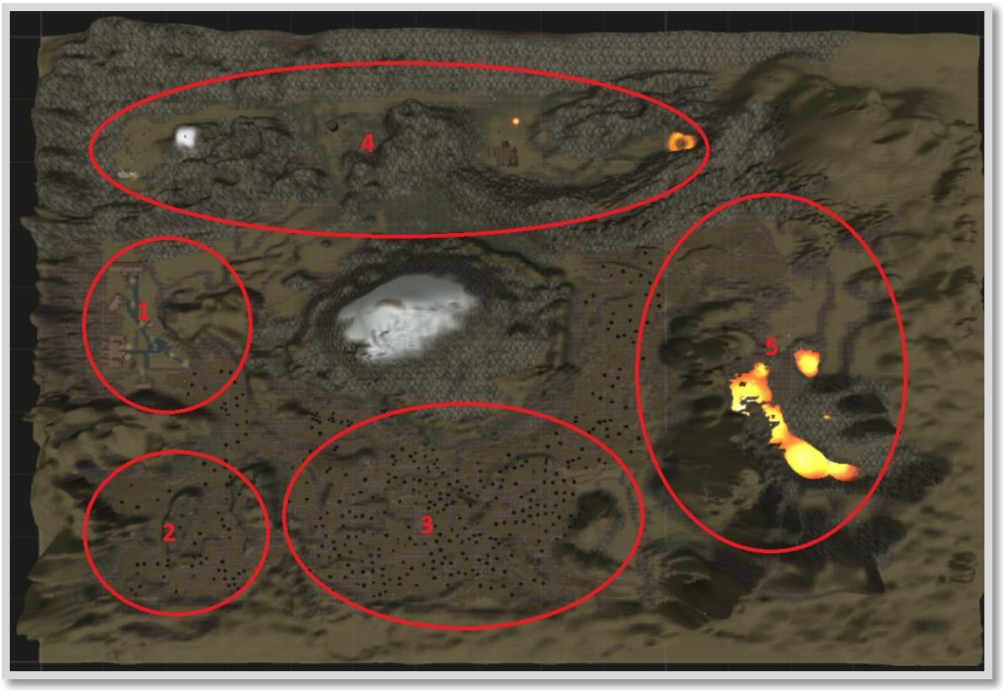
Η πλοκή του σεναρίου εκτυλίσσεται ως ακολούθως: ο παίχτης ταυτίζεται με το μάγο (wizzard) Aidunghast. Ο μάγος αυτός βρίσκεται στη γη πάνω από 1000 έτη προκειμένου να την προστατεύει από επιδρομές αιρετικών μειονοτήτων που έχουν σκοπό την επαναφορά του δαίμονα Baphomet. Η τελευταία σύγκρουση μεταξύ Baphomet και μάγου έγινε πριν περίπου πεντακόσια χρόνια στην οποία μετά από τιτάνια μάχη σκοτώθηκε ο μέντορας και δάσκαλος του Aidunghast, ο αρχιμάγος Azeras. Ο μάγος του παιχνιδιού μας δεν μπόρεσε να νικήσει ολοκληρωτικά το δαίμονα καθώς υπήρχε μεταξύ τους τεράστιο χάσμα δυνάμεων, κατάφερε όμως να τον φυλακίσει σε μια άλλη διάσταση ώστε να μην προκαλεί δυστυχία και καταστροφή στη Γη και σε ολόκληρο το σύμπαν. Όμως επειδή, γνωρίζει (ο μάγος) ότι κάποια μέρα ο αντίπαλος του θα πάρει ξανά σάρκα και οστά, εξασκείται και βρίσκεται σε εγρήγορση ώστε να μπορέσει να τον αντιμετωπίσει, αποτελεσματικότερα αυτή τη φορά.

Αυτή η μέρα λοιπόν έχει φτάσει.

Το παιχνίδι περιέχει ένα βασικό terrain πάνω στο οποίο εκτυλίσσεται όλη η δράση, τέσσερις αποστολές και ένα αρχικό μενού. Κάθε αποστολή παρουσιάζει, τη δική της ιδιαιτερότητα, δυσκολία, εχθρούς ενώ στην τέταρτη αποστολή εμφανίζεται ο Baphomet σαν τελευταίος αρχηγός (Last Boss). Κάθε είδος εχθρού έχει ξεχωριστή συμπεριφορά η οποία βασίζεται στις αρχές της τεχνητής νοημοσύνης. Ακόμα, ο χρήστης έχει τη δυνατότητα να παγώσει το παιχνίδι ή και να ξεκινήσει από την αρχή μεσώ ενός δεύτερου μενού(Pause menu) που μπορεί να καλέσει κατά τη διάρκεια της δράσης (Game Play). Ηχητικά και οπτικά εφέ εμπλουτίζουν την εφαρμογή με ρεαλισμό και ένταση ενώ η αίσθηση της μαγείας και του μυστηρίου είναι διάχυτη καθ' όλη τη διάρκεια του παιχνιδιού, συντελώντας έτσι στην αποτελεσματικότερη εμβύθιση του χρήστη στο εικονικό περιβάλλον. Στα επόμενα κεφάλαια θα αναλυθούν περαιτέρω τα διάφορα χαρακτηριστικά της εφαρμογής.

5.3 Terrain

Στη συγκεκριμένη εργασία δόθηκε ιδιαίτερη έμφαση στη δημιουργία του terrain (Εικόνα 43), καθώς το παιχνίδι εξελίσσεται μόνο σε μία πίστα. Πρόκειται, για ένα terrain ορθογώνιου σχήματος, με διαστάσεις 1500x1000m και μέγιστο ύψος 600m. Διαθέτει ποικιλόμορφα τοπία και πληθώρα υφών τοποθετημένες με καλλιτεχνικό τρόπο πάνω σε αυτό, καθώς και πλούσια γεωμορφολογία εδάφους ενώ φωτίζεται συνολικά από μία ενιαία πηγή φωτος directional light, πηγή που προσομοιώνει τις ακτίνες του ήλιου. Η γεωμορφολογία του terrain όπως αυτή αποτυπώνεται στην πίστα του παιχνιδιού παρουσιάζει τα ακόλουθα χαρακτηριστικά. Στο κέντρο της πίστας αναπτύσσεται ένας ορεινός όγκος χωρίζοντάς την σε δύο επί μέρους πεδία που με τη σειρά τους υποδιαιρούνται και αυτά σε μικρότερης έκτασης επίπεδα με τη χωροθέτηση χαμηλότερων βουνών και λόφων. Μέσα σε αυτό το τερέν υπάρχουν ταυτόχρονα πέντε νοητές περιοχές. Αυτές είναι: το χωριό (εικόνα 43), το δάσος αρκούδων (εικόνα 44), το δάσος λύκων (εικόνα 45), το κοιμητήριο (εικόνες 46,47,48) και η περιοχή του βωμού (εικόνα 49). Στόχος της γεωμορφολογίας αυτής είναι η κατά το δυνατόν απρόσκοπτη πλοήγηση του παίχτη μέσα από περάσματα και μονοπάτια αλλά και η αίσθηση της εξερεύνησης και της περιπέτειας σε άγνωστες-απάτητες περιοχές. Συνάμα, δεν πρέπει να αγνοείται ότι η ποικιλότητα του ανάγλυφου χαρίζει στον παίχτη επαφή με τη φύση όπως αυτή τουλάχιστον αναπτύσσεται στη πίστα του παιχνιδιού. Τέλος, για τη ρεαλιστικότερη απεικόνιση της πίστας εμπλουτίστηκε το παιχνίδι από ένα σύνολο εικαστικών λεπτομερειών όπως γρασίδι, δέντρα, συστήματα σωματιδίων, κτίρια, διάφορα στατικά μοντέλα, φωτισμό κ.α. οι οποίες θα περιγραφούν παρακάτω αναλυτικά.



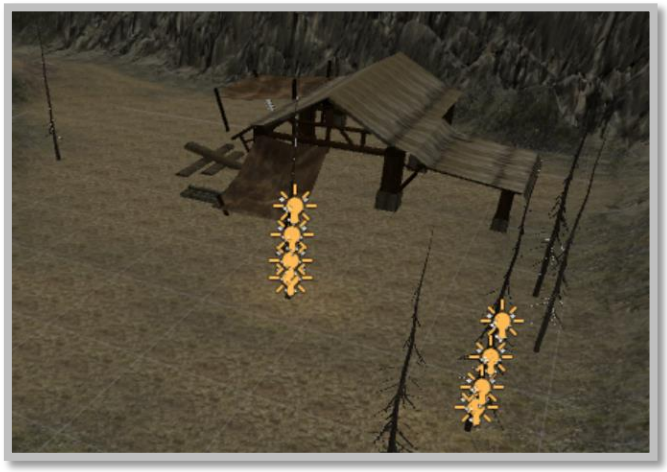
Εικόνα 43: Terrain Περιοχή 1: Χωριό(σημείο εκκινησης), Περιοχή 2: Δάσος Αρκούδων, Περιοχή 3: Δάσος Λύκων Περιοχή 4: Κοιμητήριο, Περιοχή 5: Βωμός

Εικόνα 44: Village

Εικόνα 45: Δάσος αρκούδων

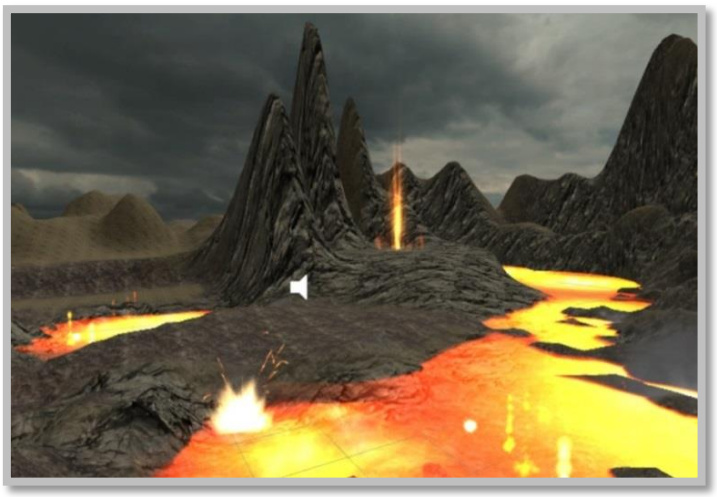
Εικόνα 46: Δάσος λύκων

Εικόνα 47: Cemetery 1



Εικόνα 48: Cemetery 2

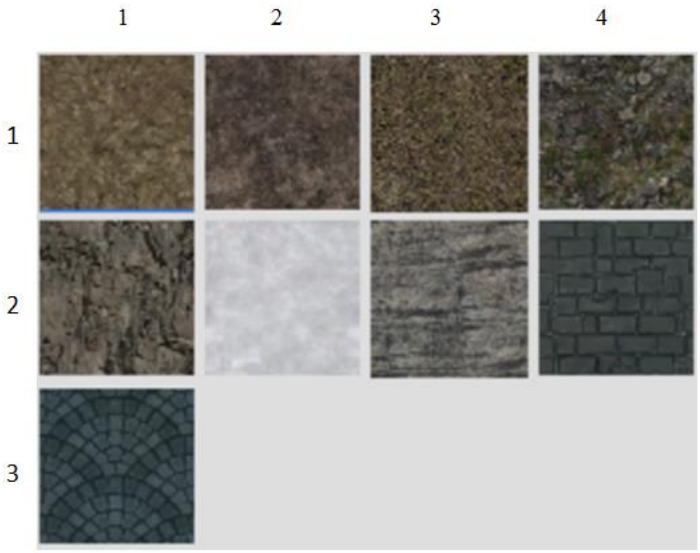
Εικόνα 49: Cemetery 3



Εικόνα 50: Βωμός

5.3.1 Texture (υφές)

Στη συγκεκριμένη εφαρμογή χρησιμοποιήθηκαν οχτώ διαφορετικά texture ανάλογα με τις ανάγκες κάθε περιοχής. Η διαδικασία που ακολουθήθηκε προκειμένου να καλυφθεί όλη η πίστα με texture είναι παρόμοια με αυτή της ζωγραφικής. Αρχικά, επιλέξαμε το texture που βρίσκεται στη θέση [1,1] της εικόνας 50 ώστε να καλυφθεί όλη η περιοχή εξ' ολοκλήρου με αυτό. Στη συνέχεια, για κάθε γεωλογική ιδιομορφία χρησιμοποιήσαμε άλλο ή άλλα είδη texture πάνω από το ήδη υπάρχον, χρωματίζοντας ξανά κατά κάποιο τρόπο το έδαφος. Ξεκινήσαμε το βάψιμο στην αρχή των γεωλογικών ιδιομορφιών με απαλές κινήσεις και στη συνέχεια το υπόλοιπο σώμα τους με πιο έντονες και δυναμικές, επιδιώκοντας έτσι την κατά το δυνατόν απόλυτη ψευδαίσθηση του ωραίου και αληθινού τοπίου αλλά και την ευχάριστη, αποτελεσματική και ρεαλιστική εμπειρία του παίχτη. Στην περιοχή του χωριού χρησιμοποιήσαμε τα texture των θέσεων [2,4] και [3,1] της εικόνας 50 με απεικόνιση πλακόστρωτο για να επικαλύψει και να οριοθετήσει τη μικρή πλατεία ενώ στην περιοχή του βωμού το texture της θέσης [2,1] της ίδιας εικόνας, όπου αποτυπώνονται σκουρόχρωμες πέτρες για να αποδοθεί οπτικά και αισθητικά η αλλοίωση που έχει υποστεί το περιβάλλον του βωμού από την επέλαση της λάβας. Επίσης, στις περιοχές όπου συχνάζουν λύκοι και αρκούδες σε δασικές κυρίως εκτάσεις, δουλέψαμε με το texture της θέσης [1,3] όπου τα φύλλα, η χλόη και άλλα στοιχεία των υφών αυτών αποδίδουν ρεαλιστικά τη χλωρίδα των περιοχών αυτών. Όσον αφορά το ψηλό βουνό στη μέση του τερέν, το texture της θέσης [2,2] απεικονίζει το χιόνι που έχει επικαλύψει την κορυφή του. Τα υπόλοιπα τρία texture [1,4], [2,2] και [2,3] μας βοήθησαν να αποδώσουμε επιφάνειες και σημεία της πίστας επιτυγχάνοντας έτσι ποικιλομορφία όπως και να παράξουμε αισθητικά μορφολογικά και ογκοπλαστικά αποτελέσματα λίαν ικανοποιητικά λαμβάνοντας υπόψη τα σκληρά γεωλογικά χαρακτηριστικά και το έντονο ανάγλυφο της περιοχής του παιχνιδιού. Τέλος, η εικόνα 46 είναι ένα δείγμα στο οποίο φαίνεται η τεχνική της επικάλυψης των υφών [1,1], [1,2] και [1,3] πάνω στο τερέν ώστε να αποδοθεί όσο το δυνατόν πιο ρεαλιστικό αποτέλεσμα.



Εικόνα 51: Υφές Εφαρμογής

Εικόνα 52: Επικάλυψη Τερέν με Υφές

5.3.2 Βλάστηση, Άνεμος, Δέντρα

Πέρα από τα texture που χρησιμοποιήθηκαν για να αποδοθεί η γεωμορφολογία του εδάφους, θα πρέπει παράλληλα να εμπλουτιστεί το υπόλοιπο περιβάλλον και με άλλα στοιχεία προκειμένου να επιτευχθεί η ψευδαίσθηση του πραγματικού χώρου. Τέτοια είναι τα δέντρα, οι θάμνοι, το γρασίδι, ο άνεμος.

Δέντρα: στη συγκεκριμένη εργασία χρησιμοποιήθηκαν εφτά διαφορετικά μοντέλα δέντρων και δύο είδη θάμνων. Κάθε δέντρο της πίστας περιέχει collider ώστε να μπορεί να προσομοιώνει διάφορες συγκρούσεις όποτε αυτές προκύπτουν, λειτουργώντας δηλαδή σαν ένα φυσικό εμπόδιο προς τον χρήστη. Αντίθετα, οι θάμνοι προγραμματίστηκαν έτσι ώστε να μη χρησιμοποιούν τη μηχανική των colliders, απλά η παρουσία τους στην πίστα εξυπηρετεί μόνο εικαστικούς λόγους. Με την ισορροπημένη χωροθέτηση των παραπάνω μοντέλων δημιουργήσαμε δασικές εκτάσεις με ποικιλία βλάστησης που καλύπτουν μάλιστα μεγάλες επιφάνειες της πίστας. Χρησιμοποιήθηκαν όλα τα είδη δέντρων και θάμνων που εμφανίζονται στην εικόνα 53. Πιο συγκεκριμένα, το ένατο μοντέλο δέντρου της εικόνας 53,

χρησιμοποιήθηκε κυρίως στις περιοχές του κοιμητηρίου και του βωμού συμβολίζοντας τη μη γονιμότητα εκείνων των εδαφών, καθώς πρόκειται για μοντέλο νεκρού δέντρου. Παράλληλα πρέπει να αναφερθεί ότι κατά την εισαγωγή κάθε τύπου δέντρου, η Unity3D δίνει τη δυνατότητα διαμόρφωσης κάποιων χαρακτηριστικών του, όπως του ύψους, του μεγέθους και του χρώματος του, λειτουργία την οποία χρησιμοποιήσαμε.

Εικόνα 53: Δέντρα

Γρασίδι: στο πραγματικό έδαφός μπορεί να αποτυπώνονται συστάδες γρασιδιού και άλλων μικρών φυσικών σχηματισμών. Για να αποδοθούν ψηφιακά οι μεμονωμένες συστάδες γρασιδιού, χρησιμοποιούνται εικόνες 2D (δύο διαστάσεων). Επίσης, κατά την εισαγωγή του γρασιδιού, σύμφωνα με τη δυνατότητα που παρέχει η Unity3D, καθορίζονται και τα χαρακτηριστικά του όπως το μέγεθος του, το ύψος του, το χρώμα του και η διασπορά του ήχου των φυλλωμάτων του.

Εικόνα 54: Γρασίδι

Skybox: είναι ένα περιτύλιγμα γύρω από ολόκληρη τη σκηνή. Έτσι η σκηνή δίνει η εντύπωση ενός σύνθετου σκηνικού στον ορίζοντα. Το πλέγμα που χρησιμοποιείται για την επίτευξη του στόχου αυτού είναι είτε ένα "κουτί" με έξι πλευρές είτε μία σφαίρα με ψηφιδωτή επιφάνεια. Το Skybox που χρησιμοποιήθηκε για τη δημιουργία του ορίζοντα της παρούσας εργασίας είναι αυτό όπως εμφανίζεται στην ακόλουθη απεικόνιση.

Εικόνα 55: Skybox

Ζώνη Ανέμου: δίνει την εντύπωση της πνοής του ανέμου στα φυλλώματα των δέντρων και στο γρασίδι ενώ επιτυγχάνεται με την προσθήκη ενός ή περισσότερων αντικειμένων που περιέχουν το συστατικό (component) Wind Zone (Ζώνη ανέμου). Τα δέντρα, η χλόη και το γρασίδι μέσα σε μια ζώνη ανέμου λυγίζουν και κινούνται με ρεαλισμό και φυσικότητα καθώς ο άνεμος πνέει προς αυτά και εισχωρεί μέσα από τα φυλλώματα τους είτε κατά ρίπους είτε ηπιότερα με επαναλαμβανόμενες πνοές.

5.3.3 Στατικά Μοντέλα, Σύστημα Σωματιδίων

Για τη φυσικότερη απόδοση του τοπίου εκτός των παραπάνω, παρουσιάζονται στατικά μοντέλα και σωματίδια τα οποία και θα αναλυθούν κατωτέρω, διαφορετικά για κάθε υπό-περιοχή. Τα περισσότερα απ' αυτά έχουν εισαχθεί από το Asset Store της Unity.

Στη περιοχή χωριού (Village) έχουμε δημιουργήσει μια ψευδαίσθηση ενός υπαρκτού χωρίου με κτίσματα. Έχουμε επιλέξει πέντε τύπους κτισμάτων οι οποίοι διαφοροποιούνται μεταξύ τους όσον αφορά το μέγεθος, τον όγκο και την πλαστικότητα τους. Στο συγκεκριμένο χωριό έχουμε δημιουργήσει ένα πηγάδι, έναν πάγκο αγοράς όπου έχουμε τοποθετήσει τελάρα με κόκκινα και πράσινα μήλα για να φαίνεται πιο ρεαλιστικό. Προχωρώντας βλέπουμε μια πινακίδα κτηρίου και ένα κάρο με δύο τεμάχια σανό. Στο πηγάδι δίπλα έχουμε τοποθετήσει ένα κουβά και λίγο πιο πέρα υπάρχουν βαρέλια και κάποια καυσόξυλα. Τέλος, το χωριό φωτίζεται με φανάρια φωτισμού (Lantern-post) που είναι στατικά μοντέλα και στα οποίο έχει προστεθεί φωτισμός τύπου Point Light. Σε κάποιες περιοχές του δάσους πλην της βλάστησης που έχει αναφερθεί ανωτέρω, χρησιμοποιήθηκε και το σωματίδιο sparkle (σπίθα) που ίπταται σε συγκεκριμένα σημεία της δασικής περιοχής προκειμένου να προσδώσει ατμοσφαιρικότητα στο παιχνίδι.

Στην περιοχή του κοιμητηρίου υπάρχει μια εκκλησία (Church), δύο ξύλινα φέρετρα κλειστά (Coffin-01-cemetery) και ένα κατεστραμμένο φέρετρο (Coffin-01) μέσα στο οποίο κείτεται ένας ανθρώπινος σκελετός. Επίσης, στο χώρο του κοιμητηρίου είναι τοποθετημένα επτά ταφικά μνημεία. Μέσα στην ευρύτερη περιοχή του κοιμητηρίου βρίσκονται οι κίονες (1,2,3), ο μικρός ναός, η εξοχική καλύβα, δύο ιεροί βράχοι και ο ανθρώπινος σκελετός. Αξίζει να σημειωθεί ότι στα παραπάνω στατικά μοντέλα ο μηχανισμός σύγκρουσης έχει προστεθεί χειροκίνητα.

Έχουμε προσθέσει και διάφορα συστήματα σωματιδίων ώστε να επιτύχουμε εξωπραγματικά εφέ. Αυτά τα συστήματα περιγράφονται παρακάτω. Το σύννεφο φωτιάς (Fire Cloud) που εμφανίζεται όταν ο παίχτης εκτοξεύει την μπάλα της φωτιάς. Στον ιερό βράχο

υπάρχει φωτιά η οποία για να δημιουργηθεί χρησιμοποιήθηκαν τρία διαφορετικά σωματίδια και είναι τα εξής: ακτίνες (Ray), σπίθες (sparkles) και δακτυλίδι φωτιάς (Ring). Τον τοίχο φωτιάς (Fire Wall), όπου για τη σύνθεση του συγκεκριμένου σωματιδίου χρησιμοποιήθηκαν δύο επιπλέον σωματίδια, του καπνού (smoke) και της σπίθας (sparkle). Τη φωτιά δέντρου (tree fire). Τη φωτιά δέντρου απέδωσαν τέσσερα εφέ ανάφλεξης (ignite effect) όμοια μεταξύ τους, κάθε ένα από αυτά περιέχει: σύστημα σωματιδίων παραμόρφωσης της ατμόσφαιρας λόγω διαφοράς θερμοκρασίας (Particle Distortion), σωματίδια καπνού (smoke particles) και φωτισμό (point light) όπως επίσης και δύο συστήματα σωματιδίων παραμόρφωσης. Εφέ χιονοθύελλας (snow storm), που εμφανίζεται όταν εκτοξευθεί η μπάλα πάγου και για τη δημιουργία της χρειαζόμαστε τρία διαφορετικά είδη σωματιδίων και είναι τα εξής: σύστημα σωματιδίων χιονιού (snow storm), σύστημα σωματιδίου ομίχλης , σύστημα σωματιδίου χιονιού επί εδάφους (snow).

Στην περιοχή του βωμού, πλην της ιδιαίτερης γεωμορφολογίας που παρουσιάζει, έχει εμπλουτιστεί με συστήματα σωματιδίων πλούσιων οπτικών εφέ. Τέσσερα εφέ σωματιδίων δημιουργούν την ψευδαίσθηση της λάβας. Το πρώτο σύστημα σωματιδίων είναι τα εφέ έκρηξης μάγματος (magma burst), τα οποία για να επιτευχθεί χρειάστηκαν έξι διαφορετικά συστήματα σωματιδίων: έκρηξης μάγματος (magma burst), ανάφλεξη φωτιάς (fire burst), θραύσματα (fragments), σπίθες (sparkles), μονοπάτια φωτιάς (trail), καπνός (smoke). Το δεύτερο εφέ είναι αυτό της λάβας (lava) που αποτελείται από δύο είδη συστημάτων σωματιδίων, το σύστημα σωματιδίων λάβας και καπνού. Το τρίτο εφέ, είναι το εφέ έκρηξης νάρκης (Land mine) όπου δημιουργείται από την ένωση τεσσάρων συστημάτων σωματιδίων, ένα σύστημα σωματιδίων έκρηξης νάρκης, δύο φωτιάς, φλεγόμενης φούσκας (buff) και τέσσερις σπινθήρες (sparkles). Τέλος, το εφέ αιώνιας φλόγας (Eternal Flame), που είναι το τέταρτο και το τελευταίο εφέ, αποτελείται από δύο συστήματα σωματιδίων, την αιώνια φλόγα και τους σπινθήρες.

Ο χρήστης μπορεί να πραγματοποιήσει τρεις διαφορετικές επιθέσεις παράγοντας έτσι τρία διαφορετικά ενεργειακά κύματα.

Ενεργειακό κύμα 1:Μπάλα φωτιάς (Fireball)

Η μπάλα φωτιάς είναι μια σφαίρα της οποίας έχει ενεργοποιηθεί ο μηχανισμός ανίχνευσης συγκρούσεων, χωρίς όμως να αποδίδεται οπτικά η επιφάνεια της. Γίνεται αντιληπτή από την παρουσία γύρω της ενός συνόλου συστημάτων σωματιδίων γεγονός που ισχύει και για τα υπόλοιπα δύο ενεργειακά κύματα. Η μπάλα φωτιάς σαν σύστημα σωματιδίων περιέχει: Flash, Smoke, Sparkle, Particle Distortion, παράγοντας παράλληλα κατά την πορείας της μια φλεγόμενη φωτεινή ουρά. Όταν η μπάλα φωτιάς ανιχνεύσει σύγκρουση ή όταν διανύσει ένα συγκεκριμένο αριθμό μέτρων χωρίς να συγκρουστεί, τότε αυτή καταστρέφεται προκαλώντας έκρηξη στο συγκεκριμένο σημείο. Η έκρηξη διαμορφώνεται και αυτή από ένα σύνολο συστημάτων σωματιδίων όπως το Line up, το smoke, το flash, το particle distortion και το point light. Η έκρηξη της Fire ball ονομάζεται “Explosion 9”.

Εικόνα 56: Μπάλα Φωτιάς

Ενεργειακό κύμα 2:Μπάλα πάγου (Ice ball)

Ισχύουν τα ίδια όπως και στην μπάλα φωτιάς με εξαίρεση τη χρωματική διαφοροποίησης ορισμένων σωματιδίων ώστε να αποδίδεται η εικόνα του πάγου. Η μπάλα πάγου μπορεί να προκαλέσει δύο διαφορετικές εκρήξεις, μία κατά τη σύγκρουση της στο έδαφος ή σε σταθερό και άκαμπτο αντικείμενο δημιουργώντας έτσι σταλαγμίτες πάγου που εξαφανίζονται μετά από μικρό χρονικό διάστημα και η δεύτερη, όταν η μπάλα χτυπήσει εχθρό ή διανύσει μια συγκεκριμένη απόσταση χωρίς σύγκρουση, όπου εκρήγνυται με όμοιο τρόπο όπως αυτό της μπάλας φωτιάς, παράγοντας ταυτόχρονα ιδιαίτερες χρωματικές αποχρώσεις.

Εικόνα 57: Μπάλα Πάγου

Ενεργειακό κύμα 3: Πράσινη μπάλα (Green ball)

Η πράσινη μπάλα διαφοροποιείται από τις άλλες δύο ως προς το χρώμα των σωματιδίων που την περιβάλλουν αλλά και στον τρόπο έκρηξης.

Εικόνα 58: Πράσινη Μπάλα

Μαγική μπάλα (Magic ball)

Είναι μικρότερη σε όγκο και παρουσιάζει ως προς τη μορφή της την ίδια φιλοσοφία όπως και οι προηγούμενες. Η μπάλα αυτή δεν αποτελεί μέσον επίθεσης, αποτελεί όμως μέρος της τελευταίας αποστολής. (Οι αποστολές θα περιγραφούν σε επόμενο κεφάλαιο) Η μαγική μπάλα περιέχει δύο συστήματα σωματιδίων (spark ereki smoke) και ένα point light.

Εικόνα 59: Μαγική Μπάλα

5.4 Χειρισμός Παιχνιδιού

Όπως έχουμε ήδη αναφέρει, πρόκειται για ένα παιχνίδι πρώτου προσώπου, όπου ο χρήστης χωρίς να γνωρίζει χειρίζεται ένα αντικείμενο σε μορφή κάψουλας πάνω στο οποίο είναι τοποθετημένη η κάμερα και ο HandControler του Leap Motion. Προκειμένου ο χρήστης να περιπλανηθεί στο εικονικό περιβάλλον, είναι απαραίτητο να κινείται, να συνδυάζει κινήσεις και να εκτελεί διάφορους ελιγμούς. Αυτό επιτυγχάνεται με το σύστημα tracking του Oculus Rift. Δηλαδή, ο παίχτης πραγματοποιεί τι περιστροφικές κινήσεις όπως ακριβώς θα τις εκτελούσε και στον πραγματικό κόσμο. Έτσι, η κίνηση αυτή στο εικονικό περιβάλλον είναι άρρηκτα συνδεδεμένη και εξαρτώμενη από τις κινήσεις της κεφαλής στο πραγματικό περιβάλλον.

5.4.1 Είδη Κινήσεων

Επειδή στη συγκεκριμένη εργασία δεν χρησιμοποιούνται χειριστήρια και κουμπιά για την πραγματοποίηση ευθύγραμμων κινήσεων, το ρόλο αυτό έχει αναλάβει το Leap Motion εντοπίζοντας (hand tracking) τις κινήσεις των χεριών ώστε στη συνέχεια να εκτελούνται οι

επιθυμητές ενέργειες στο εικονικό περιβάλλον. Βασική προϋπόθεση είναι η ορθή κίνηση του χρήστη που θα πρέπει να αντιστοιχεί στις χειρονομίες που δημιουργήσαμε και προγραμματίσαμε για την εξέλιξη του παιχνιδιού. Το παιχνίδι περιέχει οκτώ ευθύγραμμες κινήσεις, τις οποίες θα παρουσιάσουμε παρακάτω. Οι κινήσεις πραγματοποιούνται βάσει της κατεύθυνσης του οπτικού πεδίου της κάμερας.

Πρόσθια κίνηση

Η κίνηση αυτή πραγματοποιείται στο εικονικό περιβάλλον με την προϋπόθεση ότι ο χρήστης θα έχει την παλάμη του αριστερού χεριού κλειστή (γροθιά) και χωρίς να προεξέχει κανένα δάκτυλο ενώ η κάτω πλευρά της γροθιάς θα πρέπει να κοιτάζει προς τα κάτω .

Εικόνα 60: Πρόσθια κίνηση προς την κατεύθυνση του οπτικού πεδίου του χρήστη

Όπισθεν κίνηση

Για την πραγματοποίηση της θα πρέπει τα τρία δάκτυλα της αριστερής παλάμης, μικρός, παράμεσος και ο μέσος να προεξέχουν ενώ ο αντίχειρας και ο δείκτης να είναι κλειστοί ενώ η εσωτερική πλευρά της παλάμης θα πρέπει να κοιτάζει προς τα κάτω.

Εικόνα 61: Όπισθεν Κίνηση

Δεξιά κίνηση

Για την κίνηση αυτή η παλάμη του αριστερού χεριού του παίχτη θα πρέπει να είναι ανοιχτή με την εσωτερική επιφάνεια της στραμμένη προς τα δεξιά.

Εικόνα 62: Δεξιά Κίνηση

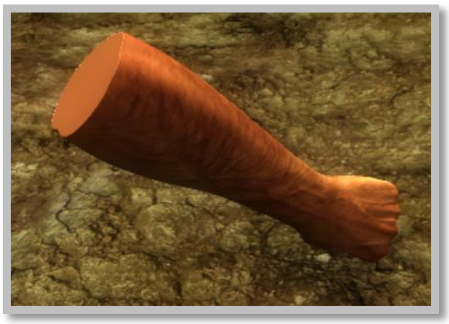
Αριστερή κίνηση

Η παλάμη του αριστερού χεριού του παίχτη θα πρέπει να είναι ανοιχτή με την εσωτερική επιφάνεια της στραμμένη προς τα αριστερά.

Εικόνα 63: Αριστερή Κίνηση

Διαγώνια κίνηση με φορά έμπροσθεν και δεξιά

Αυτή η κίνηση πραγματοποιείται σε συνδυασμό αριστερής κλειστής παλάμης και κατεύθυνση της εσωτερικής πλευράς προς τα δεξιά.



Εικόνα 64: Διαγώνια Κίνηση με φορά Έμπροσθεν και Δεξιά

Διαγώνια κίνηση με φορά έμπροσθεν και αριστερά

Και αυτή η κίνηση πραγματοποιείται σε συνδυασμό αριστερής κλειστής παλάμης και κατεύθυνση της εσωτερικής πλευράς προς τα αριστερά.

Εικόνα 65: Διαγώνια Κίνηση με φορά Έμπροσθεν και Αριστερά

Όπισθεν δεξιά κίνηση

Πραγματοποιείται όταν η παλάμη του αριστερού χεριού με προτεταμένα το μεσαίο, το παράμεσο και το μικρό δάκτυλο είναι στραμμένα προς τα δεξιά (εσωτερική πλευρά).

Εικόνα 66: Όπισθεν Δεξιά Κίνηση

Όπισθεν αριστερή κίνηση

Όπως και προηγουμένως άλλα με στροφή της αριστερής παλάμης προς τα αριστερά.

Εικόνα 67: Όπισθεν Αριστερή Κίνηση

Οι κινήσεις 1 (πρόσθια κίνηση), 3 (δεξιά κίνηση), 4 (αριστερή κίνηση) αλλά και ο συνδυασμός αυτών, επιλέχτηκαν και δημιουργήθηκαν σύμφωνα με το paper [52] το οποίο έχει σαν αντικείμενο μελέτης τις αποδοτικές χειρονομίες για κινήσεις πρώτου προσώπου σε τεχνολογίες εικονικής πραγματικότητας.

5.4.2 Επιθέσεις

Ο χρήστης μπορεί να πραγματοποιήσει τρεις επιθέσεις για να μπορέσει να εξολοθρεύσει και προστατευθεί από τους εχθρούς που του επιτίθενται κατά τη διάρκεια του παιχνιδιού. Οι επιθέσεις που μπορεί να πραγματοποιήσει ο χρήστης είναι η μπάλα της φωτιάς, η μπάλα του πάγου και η πράσινη μπάλα.

Μπάλα φωτιάς

Για την πραγματοποίηση αυτής της επίθεσης θα πρέπει η ανοιχτή παλάμη του δεξιού χεριού να στοχεύει ευθεία και έμπροσθεν με ταυτόχρονη γρήγορη κίνηση εκτίναξης της προς αυτή την κατεύθυνση. Η μπάλα που προκύπτει από την κίνηση αυτή είναι μια μπάλα φωτιάς που κινείται σε ευθύγραμμη πορεία προς την οπτική γωνία του χρήστη που αυτός είχε τη στιγμή της εκτόξευσης της μπάλας. Χαρακτηριστικό της επίθεσης αυτής είναι ότι προκαλεί μεγάλη φθορά στον αντίπαλο. Για την επίθεση αυτή απαιτείται κατανάλωση 20 mana από τα 100 ενώ ανά δευτερόλεπτο το mana αναπληρώνεται κατά 5 mana.

Εικόνα 68: Επίθεση Μπάλας Φωτιάς

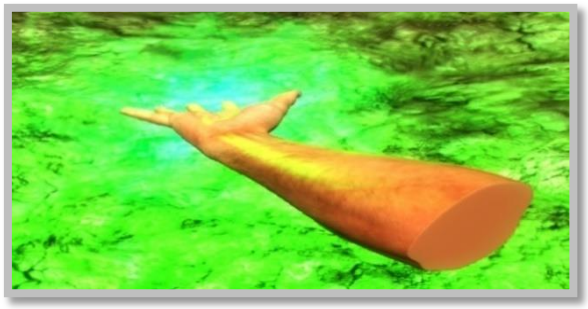
Μπάλα πάγου

Λειτουργεί με όμοια φιλοσοφία όπως αυτής της μπάλας φωτιάς με διαφορά τη χρήση του αριστερού χεριού και όχι του δεξιού. Επιπλέον, η βλάβη που προκαλεί στον αντίπαλο δεν είναι τόσο καταστροφική, απλώς επιβραδύνει τις κινήσεις του εχθρού για διάστημα 3 δευτερολέπτων. Όπως και στη μπάλα φωτιάς η επίθεση αυτή "κοστίζει" στον παίχτη 20 mana.

Εικόνα 69: Επίθεση Μπάλας Πάγου

Πράσινη μπάλα

Η υλοποίηση της επίθεσης αυτής γίνεται σε δύο φάσης. Στην πρώτη, θα πρέπει η δεξιά παλάμη του παίχτη να είναι ανοιχτή και να "κοιτάζει" προς τα πάνω για 5 δευτερόλεπτα. Παράλληλα, εμφανίζονται οπτικά και ηχητικά εφέ ώστε ο χρήστης καταλάβει ότι εκτελεί σωστά τη διαδικασία. Με τη λήξη του χρόνου αυτού, περνάμε στη δεύτερη φάση όπου η επίθεση με την πράσινη μπάλα μπορεί να πραγματοποιηθεί ανά πάσα στιγμή με όμοιο τρόπο όπως και η μπάλα φωτιάς αλλά για μόνο μια φορά. Δηλαδή, με το πέρας των 5 δευτερολέπτων η αμέσως επόμενη επίθεση που θα μπορεί να υλοποιήσει ο παίχτης με το δεξί χέρι είναι αυτή της πράσινης μπάλας και όχι της μπάλας φωτιάς, ενώ αν εντός των 5 δευτερολέπτων αλλάξει η κίνηση του δεξιού χεριού του χρήστη τότε ακυρώνεται αυτομάτως η διαδικασία της πρώτης φάσης η οποία στη συνέχεια θα πρέπει να επαναληφθεί από την αρχή. Επίσης, εάν έχουν εξαντληθεί τα 5 δευτερόλεπτα και δεν έχει πραγματοποιηθεί η επίθεση με το δεξί χέρι και για οποιοδήποτε λόγο αυτό εξαφανιστεί από το οπτικό πεδίο του Leap Motion, τότε η διαδικασία για την πραγματοποίηση της πράσινης μπάλας ακυρώνεται και θα πρέπει να επαναληφθεί όλη από την αρχή. Η επίθεση αυτή δεν καταναλώνει mana και είναι ισχυρότερη από τις δύο προηγούμενες.



Εικόνα 70: Επίθεση Πράσινης Μπάλας φάση 1

Εικόνα 71: Επίθεση Πράσινης Μπάλας φάση 2

Τέλος, θα πρέπει να αναφερθεί ότι η ενέργεια του παίχτη είναι 100 Hit Point και αφορά τη ζωή που έχει ο παίχτης η οποία καταναλώνεται από τις επιθέσεις που δέχεται κατά τη διάρκεια του παιχνιδιού. Όπως έχουμε ήδη αναφέρει έχει 100 mana point που αντιστοιχούν στην ενέργεια που διαθέτει και είναι απαραίτητη για την πραγματοποίηση των επιθέσεων του εναντίων των εχθρών του.

5.5 Κίνηση Χαρακτήρων στο Χώρο

Παρακάτω, αναλύονται λεπτομερέστατα οι οντότητες που συμμετέχουν στο παιχνίδι ώστε να γίνει κατανοητός ο τρόπος λειτουργίας τους μέσα σε αυτό.

5.5.1 Αρκούδα

Η αρκούδα έχει δημιουργηθεί έτσι ώστε να μπορεί να πραγματοποιεί τις βασικότερες κινήσεις που θα έκανε και μια πραγματική. Αυτές είναι οι εξής: να παραμένει αδρανής στην ίδια θέση (κίνηση idle), να περιπλανείται στο χώρο περπατώντας στα τέσσερα πόδια (κίνηση Walk), να τρέχει για να κυνηγήσει τον εχθρό της (κίνηση Run), να ανασηκώνεται στα δύο πόδια και να βρυχάται (κίνηση Roar) όταν εντοπίσει τον εχθρό της και να επιτίθεται όταν πλησιάσει τον εχθρό της. Η επίθεση της αρκούδας μπορεί να γίνει με τρεις διαφορετικές κινήσεις. Στη $1^{η}$ επίθεση η αρκούδα πετυχαίνει χτύπημα με το δεξί μπροστινό πόδι, ενώ στην $2^{η}$ επίθεση με το αριστερό μπροστινό πόδι ενώ το $3^{ο}$ της χτύπημα είναι δάγκωμα. Τέλος, αν ο χρήστης καταφέρει να σκοτώσει την αρκούδα, αυτή πραγματοποιεί κίνηση θανάτου.

Εικόνα 72: Αρκούδα

Στην αρχή, η αρκούδα περιφέρεται μεταξύ δύο σημείων στο χώρο (patrol) χρησιμοποιώντας παράλληλα το animation (κινούμενο σχέδιο) walk. Αν ο χρήστης την πλησιάσει σε μικρότερη απόσταση των 30 μέτρων και μεγαλύτερη ή ίση των 15, τότε αυτή θα κατευθυνθεί περπατώντας προς αυτόν (κίνηση walking). Αν ο χρήστης την πλησιάσει σε απόσταση μικρότερη των 30 μέτρων αλλά μεγαλύτερη των 15 και στη συνέχεια ο χρήστης απομακρυνθεί πάνω από 30 μέτρα, τότε η αρκούδα θα σταματήσει να τον ακολουθεί και θα επαναλάβει την patrol κίνηση. Εάν για πρώτη φορά την πλησιάσει σε απόσταση μικρότερη των 15 μέτρων, τότε η αρκούδα βρυχάται ακίνητη στο σημείο που βρίσκεται κάνοντας χρήση του animation roar. Με το πέρας του βρυχηθμού κινείται γρηγορότερα και τρέχει εναντίον του παίχτη χρησιμοποιώντας το animation run. Από εκείνο το σημείο και μετά, η αρκούδα δε σταματά να ακολουθεί ποτέ το χρήστη δηλαδή τον κλειδώνει, ενώ εκείνος δεν μπορεί να της ξεφύγει (κατάσταση danger). Επίσης, εάν ο παίχτης χτυπήσει την αρκούδα από μεγαλύτερη απόσταση των 15 μέτρων και πριν αυτή βρυχηθεί, αντιδρά με βρυχηθμό ενώ είναι ακόμα εν στάση και αμέσως μετά αρχίζει τρέχοντας να τον κυνηγά παντού ανεξάρτητα από την μεταξύ τους απόσταση. Από εκείνο το σημείο και μετά, ο παίχτης δεν μπορεί να της ξεφύγει. Εάν η αρκούδα τον πλησιάσει στα τέσσερα μέτρα ή λιγότερο, τότε αυτή σταματά και του εξαπολύει επίθεση αφαιρώντας του έτσι 20 HP (Hit Point) από τα 100 που διαθέτει συνολικά ο παίχτης.

Όπως αναφέρθηκε και παραπάνω η αρκούδα μπορεί να χρησιμοποιήσει τρία διαφορετικά animation κατά την επίθεση. Η επιλογή του animation είναι θέμα τύχης καθώς και τα τρία έχουν τις ίδιες πιθανότητες να εκτελεστούν. Μετά από κάθε επίθεση, η αρκούδα παραμένει στη συγκεκριμένη θέση για μικρό χρονικό διάστημα εκτελώντας το animation idle, δίνοντας έτσι τη δυνατότητα στον παίχτη να αντιδράσει. Η "ζωή" της αρκούδας είναι 120 HP. Κάθε επίθεση του χρήστη έχει διαφορετική επίδραση πάνω της. Η πράσινη μπάλα της αφαιρεί 100 HP, η μπάλα φωτιάς 30 HP, ενώ η μπάλα του πάγου αν και αφαιρεί μόνο 10 HP επιβραδύνει τις κινήσεις της για τρία δευτερόλεπτα. Εάν η αρκούδα απολέσει ίσα ή περισσότερα από 120 HP τότε πεθαίνει προσομοιώνοντας το θάνατο της με το animation death. Εάν ο θάνατος της προκληθεί από μπάλα πάγου (δηλαδή, εάν το τελευταίο χτύπημα προέρχεται από iceball), τότε ακινητοποιείται πλήρως στο σημείο που βρίσκεται και στη στάση που έχει, ενώ δημιουργείται γύρω της ένας κρύσταλλος πάγου. Μετά το θάνατο της και με την παρέλευση πέντε δευτερολέπτων, η σωρός της αρκούδας εξαφανίζεται. Το μοντέλο πάγου που χρησιμοποιήθηκε είναι αυτό που φαίνεται στην παρακάτω εικόνα, και επιτεύχθηκε μέσω του προγράμματος 3DStudioMax το οποίο ενδείκνυται για τη δημιουργία 3D γραφικών μοντέλων.

Εικόνα 73: Πάγος

5.5.2 Λύκος

Οι κινήσεις που μπορεί να πραγματοποιήσει είναι το Idle, το Walk, το Howl, το Run, το Attack και το Death. Με το Idle ο λύκος δεν κινείται παρά εκτελεί ορισμένες κινήσεις εν στάση. Το περπάτημα του λύκου στην πίστα προσομοιώνει το κινούμενο σχέδιο Walk ενώ το τρέξιμο το κινούμενο σχέδιο Run. Το ουρλιαχτό του προσομοιώνεται με το κινούμενο σχέδιο Howl και η επίθεση πραγματοποιείται με το κινούμενο σχέδιο Attack. Τέλος, ο θάνατος του προσομοιώνεται με το κινούμενο σχέδιο Death.

Εικόνα 74: Λύκος

Ο λύκος αρχικά περιφέρεται μεταξύ δύο σημείων στο χώρο (patrol) χρησιμοποιώντας παράλληλα το animation walk. Όταν ο παίχτης τον πλησιάσει για πρώτη φορά και η μεταξύ τους απόσταση είναι μικρότερη των 25 μέτρων, τότε ο λύκος αντιδρά με ουρλιαχτά και παραμένει σταθερός στη θέση του χρησιμοποιώντας το animation howl. Το ουρλιαχτό του είναι κάλεσμα βοήθειας προς όλους τους λύκους που βρίσκονται σε ακτίνα 40 μέτρων. Έπειτα από το ουρλιαχτό, ο συγκεκριμένος λύκος αλλά και όσοι βρέθηκαν εντός της προκαθορισμένης ακτίνας (40 μέτρων) κινούνται ταχύτερα προς τον παίχτη χρησιμοποιώντας το animation run. Συνεπώς, ο παίχτης σε αυτή την περίπτωση δεν μπορεί πλέον να ξεφύγει από τη συνεχή καταδίωξή τους, δηλαδή τον κλειδώσανε.

Εάν ο λύκος τρέχοντας πλησιάσει το χρήστη σε απόσταση μικρότερη των 3.5 μέτρων, τότε σταματά και του εξαπολύει επίθεση χρησιμοποιώντας το animation attack, αφαιρώντας του 15 HP. Μετά την επίθεση, ο λύκος παραμένει στη θέση του για σύντομο χρονικό διάστημα εκτελώντας το animation idle, δίνοντας ταυτόχρονα τη δυνατότητα στον παίχτη να αντιδράσει. Ο λύκος διαθέτει 70 HP. Η πράσινη μπάλα του αφαιρεί 100 HP, η μπάλα φωτιάς 30 HP και η μπάλα πάγου 10 HP επιβραδύνοντας όμως παράλληλα τις κινήσεις του για 3 δευτερόλεπτα. Εάν ο λύκος χάσει 70 HP και άνω πεθαίνει προσομοιώνοντας το θάνατο του με το animation death. Εάν ο θάνατος προκληθεί από μπάλα πάγου, τότε ο λύκος ακινητοποιείται εντελώς στο σημείο που βρίσκεται και στη στάση που έχει ενώ γύρω του δημιουργείται κρύσταλλος πάγου όπως ακριβώς και στην αρκούδα. Μετά το θάνατο του και με παρέλευση 5 δευτερολέπτων, η σωρός του εξαφανίζεται.

5.5.3 Σκελετός

Ο σκελετός δεν περιφέρεται σε αντίθεση με τις δύο προηγούμενες περιπτώσεις αλλά παραμένει όρθιος σε κάποιο σημείο της πίστας. Μπορεί να πραγματοποιήσει τα παρακάτω animation: Idle, Run, Attack, Death. Με το κινούμενο σχέδιο Idle ο σκελετός δεν κινείται στο χώρο εκτελώντας μόνο ορισμένες στατικές κινήσεις. Με τα κινούμενα σχέδια Run και Attack τρέχει και επιτίθεται όπως συμβαίνει αντίστοιχα και στους προηγούμενους χαρακτήρες (αρκούδα λύκος). Τέλος, ο θάνατος προσομοιώνεται από το κινούμενο σχέδιο Death.

Εικόνα 75: Σκελετός

Ο σκελετός πριν εντοπίσει τον παίχτη βρίσκεται σε κάποιο σημείο χωρίς να περιφέρεται στην πίστα χρησιμοποιώντας το animation idle. Εάν ο χρήστης πλησιάσει τον σκελετό σε απόσταση λιγότερη των 15 μέτρων τότε αυτός αρχίζει να κινείται προς το μέρος του παίχτη δια του animation run και δε θα σταματήσει ποτέ να τον ακολουθεί δηλαδή κλειδώνει τον στόχο. Εάν ο χρήστης χτυπήσει με οποιαδήποτε μπάλα και από οποιαδήποτε απόσταση το σκελετό, τότε αυτός πεθαίνει προσομοιώνοντας τον θάνατο του με το animation death. 2.5 δευτερόλεπτα μετά το θάνατο του σηκώνεται και εξακολουθεί να κινείται προς τον παίχτη τρέχοντας. Θεωρείται αθάνατος και πεθαίνει μόνο με το πέρας μιας συγκεκριμένης αποστολής που θα περιγράψουμε παρακάτω. Εάν ο σκελετός βρεθεί σε απόσταση μικρότερη των τεσσάρων μέτρων από το χρήστη, εξαπολύει επίθεση χρησιμοποιώντας το animation attack προκαλώντας φθορά 10 HP. Με το τέλος της επίθεσης, ο σκελετός βρίσκεται σε κατάσταση idle για μικρό χρονικό διάστημα δίνοντας έτσι τη δυνατότητα στο χρήστη να αντιδράσει.

5.5.4 Αρχηγός Δαίμονας

Ο δαίμονας είναι ο τελευταίος εχθρός που πρέπει να αντιμετωπίσει ο παίχτης ώστε να τερματιστεί το παιχνίδι. Στην πρώτη φάση, ο αρχηγός περιβάλλεται από μια ασπίδα (εικόνα 76) που τον προστατεύει από τις επιθέσεις του χρήστη, δηλαδή προσωρινά είναι άτρωτος. Ο παίχτης πρέπει να εκτελέσει μια συγκεκριμένη διαδικασία, η οποία θα περιγραφεί παρακάτω, προκειμένου να την εξαφανίσει. Αφότου απενεργοποιηθεί η ασπίδα, ο δαίμονας μπορεί να ηττηθεί μόνο εάν χτυπηθεί δύο φορές από επίθεση πράσινης μπάλας. Τα animation που

μπορεί να πραγματοποιήσει ο δαίμονας είναι: Roar, Walk, Attack, Death, Idle. Το animation Roar προσομοιώνει την κίνηση κατά τη στιγμή του βρυχηθμού ενώ το animation Walk το περπάτημα του. Με το animation Attack προσομοιώνεται η επίθεση και με το Death θάνατός του. Τέλος, με το animation Idle ο δαίμονας δεν κινείται στο χώρο εκτελώντας μόνο ορισμένες στατικές κινήσεις.

Εικόνα 76: Τελικός Αρχηγός Δαίμονας με Ασπίδα

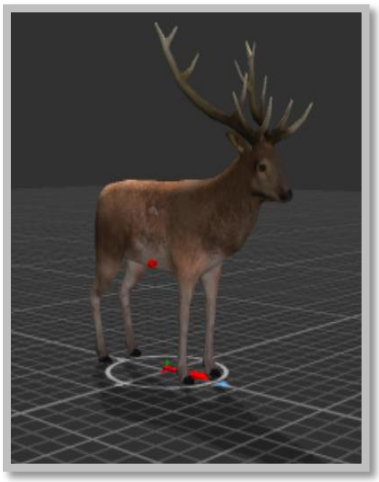
Εικόνα 77: Τελικός Αρχηγός Δαίμονας χωρίς Ασπίδα

Μόλις ο δαίμονας εμφανιστεί στην πίστα βρυχάται ενώ μένει ακίνητος χρησιμοποιώντας το αντίστοιχο animation. Στη συνέχεια ακολουθεί συνεχώς το χρήστη σε όλη την πίστα, δηλαδή τον έχει κλειδώσει, κάνοντας χρήση του animation walk. Εάν ο δαίμονας βρεθεί σε μικρότερη απόσταση του ενός μέτρου από το χρήστη, τότε εξαπολύει εναντίον του επίθεση χρησιμοποιώντας το animation attack προκαλώντας του "ζημία" 30 HP κάθε φορά. Μετά την επίθεση, ο δαίμονας παραμένει στη συγκεκριμένη θέση για μικρό χρονικό διάστημα εκτελώντας παράλληλα το animation idle δίνοντας έτσι τη δυνατότητα στον παίχτη να αντιδράσει. Ο θάνατος του προσομοιώνεται με το animation death αφού προηγουμένως έχει

χτυπηθεί δύο φορές από επίθεση πράσινης μπάλας όπως αναφέραμε παραπάνω. Οι άλλες δύο επιθέσεις (iceball, fireball) δεν του προκαλούν καμία βλάβη και σε καμία φάση.

5.5.5 Ελάφι

Το ελάφι είναι βασικό στοιχείο της δεύτερης αποστολής. Τα animation που χρησιμοποιούνται για το ελάφι είναι το Idle1, Idle2, Walk. Με το animation Idle1 προσομοιώνονται ορισμένες στατικές κινήσεις του όταν το ελάφι δεν κινείται στο χώρο. Επίσης, με το animation idle2 το ελάφι εκτελεί πάλι στατικές κινήσεις, διαφορετικές όμως αυτή τη φορά από το Idle1. Τέλος, το animation Walk προσομοιώνει το περπάτημα του ελαφιού. Το ελάφι βρίσκεται αρχικά σε μια συγκεκριμένη θέση μέσα στο δάσος εκτελώντας διαδοχικά και σε επανάληψη το Idle1 και Idle2.



Εικόνα 78: Ελάφι

Εάν ο χρήστης το πλησιάσει σε μικρότερη απόσταση των πέντε μέτρων, τότε το ελάφι "κλειδώνει" τον παίχτη σαν στόχο και κινείται προς αυτόν κάνοντας χρήση του animation walk. Ενώ είναι κλειδωμένος ο στόχος και η απόσταση μεταξύ παίχτη και ελαφιού είναι μεγαλύτερη των 4,5 μέτρων και μικρότερη των 20 μέτρων τότε αυτό κινείται προς το χρήστη. Εφόσον ο στόχος είναι κλειδωμένος και η μεταξύ τους απόσταση γίνει μεγαλύτερη από 20 μέτρα, τότε ο στόχος ξεκλειδώνεται και το ελάφι ακινητοποιείται εκτελώντας διαδοχικά και επαναλαμβανόμενα το Idle1 και Idle 2. Για να κλειδωθεί ξανά ο στόχος θα πρέπει ο χρήστης να πλησιάσει ξανά το ελάφι στα 5 μέτρα.

Οι δράσεις των παραπάνω χαρακτήρων συνοδεύονται από ηχητικά εφέ τα ενεργοποιούνται σε κάθε διαφορετικό κινούμενο σχέδιο που εκτελείται από αυτούς. Τα ονόματα των ηχητικών εφέ παραθέτονται σε επόμενο υποκεφάλαιο.

5.6 Αποστολές

Για να ολοκληρωθεί το παιχνίδι θα πρέπει να πραγματοποιηθούν τέσσερις συγκεκριμένες αποστολές διαδοχικά η μια μετά την άλλη. Οι αποστολές δεν εκτελούνται παράλληλα και με οποιαδήποτε σειρά αλλά έχουν καθορισμένη ροή. Κάθε φορά που ολοκληρώνεται μια ξεκλειδώνει η επόμενη και γίνεται αυτόματη αποθήκευση της προόδου ενώ ταυτόχρονα ανανεώνεται η ζωή του παίχτη. Έτσι, ο χρήστης έχει τη δυνατότητα να επανεκκινήσει το παιχνίδι αμέσως μετά την τελευταία ολοκληρωμένη αποστολή. Εάν ο χρήστης χάσει, μπορεί να αρχίσει και πάλι από την αποστολή στην όποια βρισκόταν ή ξεκινώντας το παιχνίδι από την αρχή.

Αποστολή 1

Η πρώτη αποστολή είναι απλή ώστε ο χρήστης να συνειδητοποιήσει ότι έχει εισέρθει σε ένα εικονικό περιβάλλον, να εξοικειωθεί με αυτό και γενικότερα με τον χειρισμό του παιχνιδιού. Στόχος της πρώτης αποστολής είναι η εύρεση και η εξόντωση των τεσσάρων αρκούδων που υπάρχουν.

Αποστολή 2

Ο χρήστης σε αυτή την αποστολή πρέπει να εξερευνήσει την περιοχή του δάσους ώστε να βρει το ελάφι που βρίσκεται κοντά στην περιοχή των λύκων και να το οδηγήσει μέχρι το πηγάδι του χωριού.

Αποστολή 3

Σ’ αυτή την αποστολή ο παίχτης πρέπει να πάει στην ευρύτερη περιοχή του κοιμητηρίου για να βρει και να ενεργοποιήσει τους δύο ιερούς βράχους. Ο ένας ιερός βράχος ενεργοποιείται με την μπάλα φωτιάς ενώ ο δεύτερος με την μπάλα του πάγου. Οι εχθροί αυτής της αποστολής είναι σκελετοί που εξοντώνονται και καταστρέφονται μόνο με την ενεργοποίηση των ιερών βράχων.

Αποστολή 4

Στη περιοχή του μικρού ναού που βρίσκεται στην είσοδο του κοιμητηρίου, εμφανίζεται η μαγική μπάλα που θα πρέπει να μεταφερθεί στη περιοχή του βωμού. Η μεταφορά της γίνεται ως εξής: ο χρήστης θα πρέπει να την πλησιάσει σε μικρή απόσταση και εκτείνοντας ταυτόχρονα το δείκτη του δεξιού χεριού, χωρίς να προεξέχουν τα υπόλοιπα δάκτυλα. Η μπάλα τότε προσκολλάται στο δάκτυλο για όσο χρονικό διάστημα το χέρι παραμένει σε αυτή τη στάση. Εάν όμως αλλάξει στάση το χέρι ενώ είναι προσκολλημένη η μπάλα σ’ αυτό, τότε αυτή παραμένει στάσιμη στη θέση όπου άλλαξε η στάση του χεριού αναμένοντας την επανάληψη της ορθής κίνησης. Μόλις γίνει η πρώτη επαφή μαγικής μπάλας και δείκτη, εμφανίζεται ο τελικός αρχηγός στην περιοχή του μικρού ναού, με ενεργοποιημένη την ασπίδα γύρω του. Έτσι, ο παίχτης δεν μπορεί προς το παρόν να χτυπήσει τον αρχηγό ο οποίος και τον ακολουθεί το χρήστη συνεχώς. Για να απενεργοποιηθεί η ασπίδα του αρχηγού θα πρέπει η μαγική μπάλα να μεταφερθεί στο βωμό ο οποίος έτσι παύει

να είναι άτρωτος. Όπως αναφέραμε και παραπάνω, ο αρχηγός για να ηττηθεί πρέπει να χτυπηθεί από δύο πράσινες μπάλες οπότε και ολοκληρώνεται η τέταρτη αποστολή και τερματίζεται το παιχνίδι.

5.7 Γραφική Διεπαφή Χρήστη

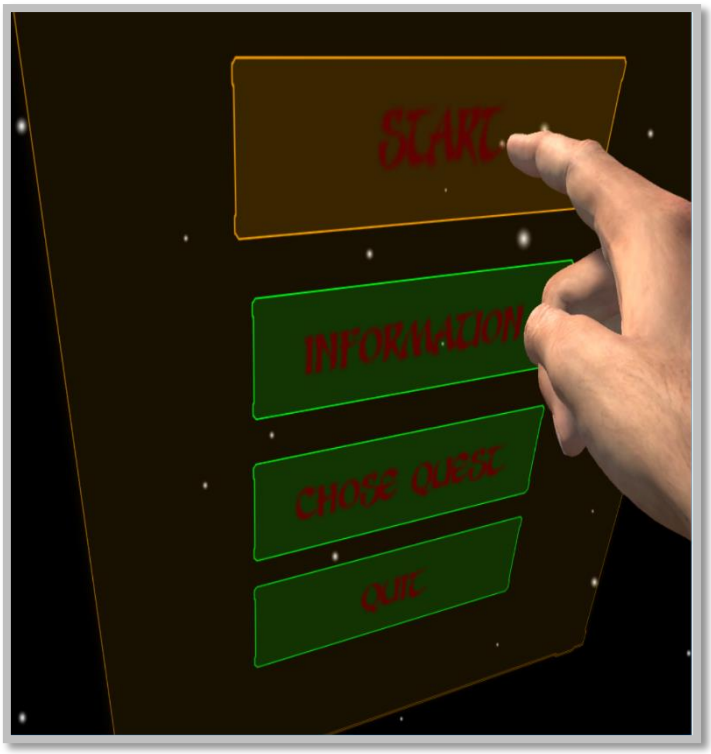
Όπως ήδη αναφέρθηκε και σε προηγούμενες ενότητες, στο συγκεκριμένο παιχνίδι δεν γίνεται χρήση πληκτρολογίου ή ποντικιού. Έτσι, τα διάφορα μενού που εμφανίζονται κατά τη διάρκεια του παιχνιδιού μπορούν να ενεργοποιηθούν μόνο εάν ο δείκτης του δεξιού χεριού τα ακουμπήσει για σύντομο χρονικό διάστημα. Μετά το χρόνο αυτό πραγματοποιείται η αντίστοιχη δράση σύμφωνα με την επιλογή. Υπάρχουν τρία μενού στο παιχνίδι με τα οποία μπορεί να αλληλεπιδράσει ο χρήστης. Το $1^{ο}$ εξ αυτών είναι το αρχικό, το οποίο εμφανίζεται αυτόματα με την έναρξη της εφαρμογής σε διαφορετική αρχική σκηνή από εκείνη που λαμβάνει χώρα η κατ' εξοχήν δράση. Το $2^{ο}$ είναι το μενού παύσης το οποίο μπορεί να εμφανιστεί μόνο κατά τη διάρκεια της κύριας σκηνής του παιχνιδιού και μόνο όταν αυτό κληθεί από τον παίχτη με συγκεκριμένη κίνηση των χεριών. Το $3^{ο}$ είναι το μενού θανάτου. Αυτό εμφανίζεται αυτόματα μόλις ο παίχτης απολέσει όλη του την ενέργεια, "ζωή". Πέρα από τα μενού που περιγράψαμε ανωτέρω, χρησιμοποιούμε και γραφικές διεπαφές προκειμένου να παρέχονται στο χρήστη χρήσιμες πληροφορίες που τον διευκολύνουν για την ομαλή εξέλιξη του παιχνιδιού. Οι γραφικές διεπαφές τέτοιου είδους ονομάζονται head-up display (HUD).

Εκτός από τα HUD και τα μενού που αναφέραμε υπάρχουν και άλλες γραφικές διεπαφές που στοχεύουν και αυτές στη διευκόλυνση του χρήστη αλλά και στη ρεαλιστικότερη απόδοση του παιχνιδιού. Στη συνέχεια θα περιγράψουμε αναλυτικότερα τι περιέχουν τα μενού που αναφέραμε παραπάνω.

5.8 Μενού Εφαρμογής

5.8.1 Αρχικό Μενού

Το αρχικό μενού του παιχνιδιού αποτελείται από τέσσερα κουμπιά τα οποία μπορεί να επιλέξει ο χρήστης ανάλογα με την ενέργεια που θέλει να εκτελέσει κάθε φορά. Οι εντολές αυτές είναι το κουμπί "START" με την επιλογή του οποίου πραγματοποιείται η έναρξη του παιχνιδιού και η είσοδος μας στο περιβάλλον της πίστας. Το "QUIT" με την πίεση του οποίου απενεργοποιείται η εφαρμογή, δηλαδή εξερχόμαστε από το εικονικό περιβάλλον. Το "INFORMATION" button, με την επιλογή του ανοίγει ένα άλλο υπομενού το οποίο παρέχει γενικές πληροφορίες για τη πορεία του παιχνιδιού (χειρισμός, αποστολές, κ.α.), ενώ αυτόματα απενεργοποιείται το ήδη υπάρχον μενού. Επιπροσθέτως, αυτό το υπομενού περιέχει ένα μόνο κουμπί το οποίο είναι το "BACK" button. Το "BACK" button είναι το κουμπί επιστροφής στο αρχικό μενού με παράλληλη απενεργοποίηση του υπομενού. Η τέταρτη επιλογή του αρχικού μενού είναι το "CHOSE QUEST", με το πάτημα του οποίου ανοίγει ένα άλλο υπομενού το "QUEST MENU" ενώ συντελείται ταυτόχρονη έξοδος από το προηγούμενο. Το "QUEST MENU" περιέχει 4 κουμπιά: "QUEST 2", "QUEST 3", "QUEST 4", "BACK BUTTON".



Εικόνα 79: Αρχικό Μενού

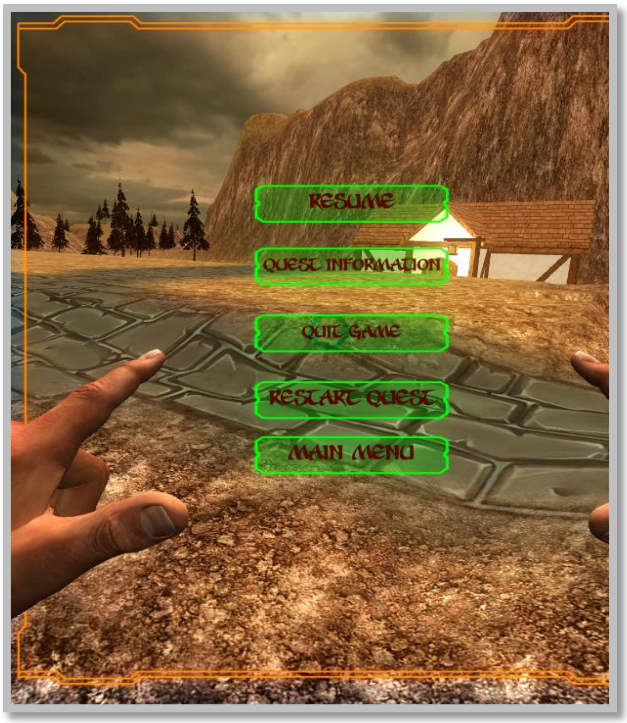
Οι επιλογές που παρέχει το "QUEST MENU" έχουν σχέση με την επιθυμία του χρήστη να μεταφερθεί σε επόμενες ή και προηγούμενες αποστολές. Δηλαδή, επιλέγοντας το "QUEST 2" μεταφερόμαστε στη σκηνή του κυρίως παιχνιδιού ξεκινώντας από την $2^{η}$ αποστολή χωρίς απαραίτητα να έχουμε ολοκληρώσει την $1^{η}$. Πατώντας το "QUEST 3" η επιλογή αυτή μας μεταφέρει στην $3^{η}$ αποστολή και παρομοίως το "QUEST 4" μας μεταφέρει στο ξεκίνημα της $4^{ης}$ και τελευταίας αποστολής.

5.8.2 Μενού Παύσης

Στο μενού παύσης ο παίχτης μπορεί να παγώσει το παιχνίδι του οποιαδήποτε στιγμή χωρίς όμως να χαθεί η πρόοδος και η προσπάθεια του μέχρι τότε. Για να ενεργοποιηθεί αυτό το μενού θα πρέπει ο δείκτης και ο αντίχειρας των δύο χεριών του χρήστη να βρίσκονται σε έκταση ενώ τα υπόλοιπα δάχτυλα κάθε παλάμης να είναι κλειστά. Αυτή η χειρονομία προσωρινής παύσης της δράσης του παιχνιδιού σχεδιάστηκε έτσι ώστε ο χρήστης να μπορεί οποιαδήποτε στιγμή με ευκολία να παγώσει το παιχνίδι κυρίως λόγω πιθανής αδιαθεσίας του εξαιτίας των αρνητικών επιπτώσεων που μπορεί να προκαλέσει η εικονική πραγματικότητα (simulation sickness).

Όπως διακρίνουμε και στη φωτογραφική απεικόνιση, το μενού παύσης περιέχει πέντε διαφορετικές επιλογές. Την επιλογή "RESUME" όπου πατώντας το κουμπί αυτό απενεργοποιείται το μενού παύσης και συνεχίζεται κανονικά η ροή του παιχνιδιού από το σημείο που είχε σταματήσει. Η επιλογή "QUITE GAME" που δίνει τη δυνατότητα εξόδου από το εικονικό περιβάλλον. Η επιλογή "QUSET INFORMATION" που απενεργοποιεί το υπάρχον μενού και ανάλογα με την αποστολή που εκτελεί εκείνη τη στιγμή ο παίχτης, ενεργοποιείται ένα άλλο υπομενού, το οποίο περιέχει κείμενο με πληροφορίες σχετικές με

την τρέχουσα αποστολή καθώς και ένα “BACK BUTTON”. Επιλέγοντας το “RESTART QUEST” ο χρήστης ξεκινά από την αρχή την αποστολή την οποία ήδη εκτελούσε, γεμάτος ενέργεια και mana. Τέλος, με την επιλογή του “MAIN MENU” ο παίχτης επιστρέφει στο αρχικό μενού εγκαταλείποντας το παιχνίδι.



Εικόνα 80: Μενού Παύσης

5.8.3 Μενού Θανάτου

Το μενού θανάτου θα εμφανιστεί στην οθόνη όταν ο παίχτης ηττηθεί από τους αντιπάλους παρέχοντας του τη δυνατότητα επιλογής ορισμένων ενεργειών προκειμένου να συνεχίσει ή να εγκαταλείψει την εφαρμογή. Οι επιλογές του μενού θανάτου είναι το “RESTART QUEST”, το “MAIN MENU” και το “QUIT GAME” τα οποία έχουν περιγραφεί ανωτέρω και εκτελούν τις ίδιες ενέργειες όπως και στα προηγούμενα μενού.

Εικόνα 81: Μενού Θανάτου

5.8.4 HUD και Γραφικές Διεπαφές

Σε όλα τα παιχνίδια δράσης υπάρχουν μπάρες που ενημερώνουν τους χρήστες για την «ζωή» τους και την ενέργεια που έχουν καταναλώσει εκτελώντας τις διάφορες επιθέσεις. Έτσι και στο συγκεκριμένο παιχνίδι υπάρχουν οι μπάρες αυτές προκειμένου να γνωρίζουμε το ποσοστό «ζωής» που μας έχει απομείνει άλλα και αν έχουμε ενέργεια για να εκτελέσουμε κάποια επίθεση. Στην εφαρμογή χρησιμοποιούμε δύο HUD τα οποία εμφανίζονται στην κυρίως σκηνή στην πάνω αριστερή γωνία της οθόνης. Η μπάρα ενέργειας εμφανίζει πόση ενέργεια απομένει στο χρήστη και η μπάρα mana δείχνει πόσο mana απομένει στον παίχτη για την πραγματοποίηση των επιθέσεων του.

Εικόνα 82: HUD, Η πράσινη μπάρα απεικονίζει την ενέργεια του παίχτη και η μπλε το mana.

Άλλες γραφικές διεπαφές είναι το αίμα, η οθόνη θανάτου και ο στόχος. Όταν ο χρήστης χάσει ενέργεια τότε εμφανίζεται στην οθόνη για μικρό χρονικό διάστημα μια ημιδιαφανής εικόνα αίματος δηλώνοντας ότι ο χρήστης δέχτηκε κάποιο αιματηρό χτύπημα.

Η οθόνη θανάτου εμφανίζεται λίγο πριν εμφανιστεί το μενού θανάτου, όταν ο χρήστης χάσει όλη του την ενέργεια και καλύπτεται τότε όλη η οθόνη με ένα ημιδιαφανές

κόκκινο χρώμα. Έτσι, δημιουργείται η ψευδαίσθηση στα μάτια του χρήστη πως όλος ο ψηφιακός κόσμος αποδίδεται οπτικά με κόκκινες αποχρώσεις.

Όταν ο χρήστης ολοκληρώσει μια από τις τρεις πρώτες αποστολές, εμφανίζεται κάτω αριστερά στην οθόνη και για σύντομο χρονικό διάστημα οι λέξεις ‘‘check point’’, υποδηλώνοντας έτσι στο χρήστη το σημείο από το οποίο μπορεί να επανακινηθεί το παιχνίδι.

Τέλος, ο στόχος βρίσκεται στο κέντρο του οπτικού πεδίου του χρήστη και τον βοηθά για να στοχεύει με επιτυχία όταν εκτελεί τις επιθέσεις του.

5.9 Ήχοι (Audio)

Ένα από τα βασικότερα στοιχεία μιας gaming εφαρμογής είναι η επένδυση της με διάφορα ηχητικά εφέ, συντελώντας στην αποτελεσματικότερη εμβύθιση του χρήστη στο εικονικό περιβάλλον. Οι ήχοι και οι μελωδίες επιλέχθηκαν από το διαδίκτυο και διαμορφώθηκαν κατάλληλα για να καλύψουν τις ηχητικές ανάγκες του παιχνιδιού. Οι ήχοι και τα σημεία που αυτοί χρησιμοποιήθηκαν παρουσιάζονται παρακάτω.

- Exploring space 1: αποτελεί τον βασικό ήχο περιβάλλοντος των τριών πρώτων αποστολών
- Tirion fording theme song: βασικός ήχος περιβάλλοντος αρχικού μενού
- Bloodborne soundtrack OST- Father Gascoigne, the Hunter: βασικός ήχος περιβάλλοντος που ακούγεται από ένα σημείο και μετά της τέταρτης αποστολής
- Demon Voice Sound Effect: βασικός ήχος περιβάλλοντος τέταρτης αποστολής ο οποίος ακούγεται από την έναρξη της αποστολής μέχρι ένα συγκεκριμένο σημείο
- Flickering Sci Electricity Sound Effect: ηχητικό εφέ το οποίο γίνεται αντιληπτό όταν ο χρήστης πλησιάσει τη μαγική μπάλα της τέταρτης αποστολής
- Super Saiyan Rose Aura Sound Effect: ηχητικό εφέ το οποίο ακούγεται από την αρχή της φόρτισης της πράσινης μπάλας μέχρι τη στιγμή της εκτόξευσης
- Sound Effect Pack #4: ηχητικό εφέ το οποίο ενημερώνει το χρήστη πως η πράσινη μπάλα είναι έτοιμη για εκτόξευση
- Fireball Sound effect: ηχητικό εφέ το οποίο ακούγεται μόλις εκτοξευθεί από το χρήστη μια μπάλα φωτιάς
- Big Explosion effect Video: ηχητικό εφέ το οποίο παράγεται κατά την έκρηξη της μπάλας φωτιάς και πράσινης μπάλας
- 2 greenspellcast: ηχητικό εφέ που παράγεται κατά την εκτόξευσης της πράσινης
- Ice Attack: ηχητικό εφέ το οποίο ακούγεται κατά την εκτόξευση της μπάλας πάγου
- Ice expl3: ηχητικό εφέ το οποίο παράγεται κατά την έκρηξη της μπάλας πάγου εκτός επιφάνειας εδάφους
- Ice expl2: ηχητικό εφέ το οποίο παράγεται κατά την έκρηξη της μπάλας πάγου εξαιτίας σύγκρουσης με το έδαφος
- Roar 2 Grizzlg Bear Sound Effect: ηχητικό εφέ βρυχηθμού το οποίο ακούγεται μόλις η αρκούδα εντοπίσει τον χρήστη
- Breath bear: ηχητικό εφέ ανάσας της αρκούδας το οποίο ακούγεται καθώς η αρκούδα τρέχει
- Attackbear2: ηχητικό εφέ επίθεσης αρκούδας με δάγκωμα
- Attackbear: ηχητικό εφέ επίθεσης αρκούδας με αριστερό ή δεξί πόδι μπροστά
- Death bear: ηχητικό εφέ θανάτου αρκούδας
- Wolf Howling2: ηχητικό εφέ ουρλιαχτού λύκου

- Dog Refriever Fast: ηχητικό εφέ λαχανιάσματος λύκου που ακούγεται κατά τη διάρκεια του τρεξίματος
- Dog Bite- Sound Effect: ηχητικό εφέ επίθεσης λύκου με δάγκωμα
- Dogwhine: ηχητικό εφέ θανάτου λύκου
- Skeleton scream: ηχητικό εφέ το οποίο παράγεται από το σκελετό μόλις αυτός αντιληφθεί τον χρήστη
- Death skeleton: ηχητικό εφέ θορύβου οστών το οποίο παράγεται κατά τον θάνατο του σκελετού
- Death skeleton: αντεστραμμένο, ηχητικό εφέ θανάτου σκελετού το οποίο παράγεται όταν αυτός σηκώνεται από το έδαφος
- Knife hit1: ηχητικό εφέ επίθεσης σκελετού
- Skeleton walk: ηχητικό εφέ περπατήματος σκελετού
- Walking Fast on Grass: ηχητικό εφέ βαδίσματος που παράγεται καθώς ο παίχτης κινείται στην ευρύτερη περιοχή της πίστας (κοιμητήριο, βωμός, δάσος)
- Walking Fast on Dirt: ηχητικό εφέ βαδίσματος που παράγεται καθώς ο χρήστης κινείται στη περιοχή του χωριού
- Bacon Frying Sound Effect: ηχητικό εφέ που παράγεται καθώς ο παίχτης κινείται μέσα στη λάβα
- Gas Stove: ηχητικό εφέ το οποίο ακούγεται καθώς ο χρήστης πλησιάζει το σύννεφο φωτιάς
- Wind Sound Effect: ηχητικό εφέ έντονων καιρικών φαινομένων το οποίο γίνεται αντιληπτό καθώς ο χρήστης πλησιάζει τον ιερό βράχο πάγου
- Forest on Fire #1: ηχητικό εφέ πυρκαγιάς δάσους το οποίο γίνεται αντιληπτό καθώς ο χρήστης πλησιάζει το σύστημα σωματιδίων "Τοίχος φωτιάς"
- Star Wars Lightshaber activate sound: ηχητικό εφέ το οποίο ακούγεται μια φορά κατά την ενεργοποίηση κάθε ιερού βράχου
- Lava Flow Sound Effect: ηχητικό εφέ ροής λάβας το οποίο γίνεται αντιληπτό καθώς ο χρήστης πλησιάζει
- System Shutdown: ηχητικό εφέ που υποδηλώνει την απενεργοποίηση της ασπίδας του τελικού αρχηγού
- Monster Roar: ηχητικό εφέ επίθεσης βρυχηθμού τελικού αρχηγού το οποίο παράγεται μόλις αυτός εμφανιστεί για πρώτη φόρα
- Monster Footsteps: ηχητικό εφέ βαδίσματος τελικού αρχηγού
- Demon hit 1, Demon hit 2, Demon hit 3: ηχητικά εφέ επίθεσης τελικού αρχηγού. Η επιλογή του ηχητικού εφέ κάθε φορά που επιτίθεται ο τελικός αρχηγός αποτελεί μια διαδικασία που βασίζεται στη τύχη και στις ίδιες πιθανότητες
- Monster Pain Death: ηχητικό εφέ θανάτου τελικού αρχηγού
- Menu change: ηχητικό εφέ το οποίο παράγεται με την εμφάνιση κάποιου μενού στην οθόνη
- Menu change2: ηχητικό εφέ που ακούγεται μόλις ο δείκτης του δεξιού χεριού ακουμπήσει κάποιο από τα κουμπιά ενός μενού
- Menu1: ηχητικό εφέ το οποίο παράγεται καθώς το δάκτυλο του χρήστη είναι σε επαφή με κάποιο κουμπί, και το απομακρύνει προτού πραγματοποιηθεί η δράση του κουμπιού αυτού
- Death (SGO sound effect): ηχητικό εφέ θανάτου του χρήστη το οποίο παράγεται μόλις αυτός χάσει όλη του την ενέργεια.

Για την διαμόρφωση και την "κοπή" των διαφόρων ήχων της εφαρμογής χρησιμοποιήθηκαν διαδικτυακοί audio cutters.

Κεφάλαιο 6ο Υλοποίηση Παιχνιδιού

6.1 Εισαγωγή

Στο κεφάλαιο αυτό γίνεται αναλυτική περιγραφή των σταδίων υλοποίησης του παιχνιδιού. Θα αναλυθούν τα τεχνικά χαρακτηριστικά και το προγραμματιστικό κομμάτι βασικών λειτουργιών καθοριστικών για τη σύνθεση και σχεδιασμό της εφαρμογής.

6.2 Χειρισμός Παίχτη

6.2.1 Μοντέλο Παίχτη Κατάλληλο για Κίνηση στο Χώρο

Όπως ήδη έχουμε αναφέρει και σε προηγούμενο κεφάλαιο, ο χρήστης χειρίζεται ένα αντικείμενο μορφής κάψουλας όπου είναι έμμεσα ενσωματωμένη σε αυτό μέσω script (όχι parenting) η κυρίως κάμερα που παρέχεται από το API του Oculus Rift μέσω της οποίας αποδίδεται ο εικονικός κόσμος στα μάτια του παίχτη. Επειδή όμως υπάρχει στην κάψουλα συστατικό στερεού σώματος, ώστε να ανιχνεύει τις συγκρούσεις με άλλα στερεά αντικείμενα, ήταν αναπόφευκτο να ισχύουν για αυτήν οι νόμοι της βαρύτητας και γενικότερα της φυσικής με αποτέλεσμα να προκύψει ένα πρόβλημα το οποίο παρουσιάζουμε παρακάτω.

Η κάψουλα είχε την τάση να κατρακύλα και να στριφογυρίζει προς τυχαίες κατευθύνσεις πάνω στο έδαφος με συνέπεια να παρασύρει ταυτόχρονα και την κυρίως κάμερα δημιουργώντας μια ανεξέλεγκτη κατάσταση όπως η απώλεια χειρισμού και έντονη δυσφορία στο χρήστη. Εν μέρει το πρόβλημα αυτό λύθηκε “παγώνοντας” μέσω του συστατικού στερεού σώματος της κάψουλας την περιστροφή της κατά τους άξονες x y z. Έτσι η περιστροφή μπορεί να πραγματοποιηθεί μόνο μέσω του χειρισμού του χρήστη.

Παρ’ όλα αυτά, ένα άλλο πρόβλημα ενέκυψε μετά τη διόρθωση που αναφέρθηκε. Αν και η κάψουλα εκινείτο χωρίς κανένα πρόβλημα πάνω σε επίπεδο εδάφος, κατά τη διαδρομή της σε κεκλιμένο επίπεδο η πορεία της δεν εξελισσόταν ομαλά αφού έκανε αλλεπάλληλες μικρές πτώσεις, δίνοντας την εντύπωση πως ο παίχτης κατέβαινε σκαλοπάτια. Το γεγονός αυτό ήταν αποτέλεσμα πολλών παραγόντων όπως αυτό της βαρύτητας σε συνδυασμό με την ταχύτητα κίνησης του παίχτη, της γωνίας κλίσης της επιφάνειας του εδάφους, του παγώματος περιστροφής των αξόνων της κάψουλας x y z και του σχήματος της κάψουλας η βάση της οποίας είναι ημισφαιρική και ακουμπούσε διακοπτόμενα το έδαφος προκαλώντας έτσι την μερική πτώση της. Προκειμένου να αποφύγουμε την αρνητική αυτή κάθετη πτώση, επεξεργασθήκαμε την ακόλουθη λύση. Θεωρούμε ότι η κάψουλα αιωρείται συνεχώς πάνω από το έδαφος 1.50 μέτρα, ανεξαρτήτως της μορφολογίας του και πάντα σε σταθερή απόσταση από αυτό. Η λύση αυτή απέδωσε. Για να υλοποιηθεί η αιώρηση της κάψουλας (hovering) σε σταθερή πάντα απόσταση από το έδαφος, χρησιμοποιήσαμε μια βασική δυνατότητα της Unity3D το Ray Casting (ρίψη ακτίνας).

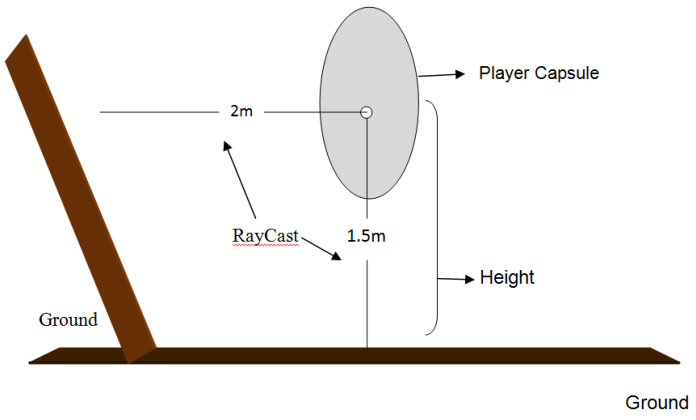
Στη δική μας εφαρμογή εξάγουμε μια ακτίνα από το κέντρο της κάψουλας προς το έδαφος. Έτσι, λαμβάνουμε πληροφορίες σχετικά με το αντικείμενο που προσκρούει η ακτίνα και την απόσταση της κάψουλας από αυτό (συνήθως το αντικείμενο πρόσκρουσης είναι το έδαφος). Εάν η διαφορά της θέσης κέντρου κάψουλας με το σημείο πρόσκρουσης της ακτίνας ως προς τον άξονα y δεν είναι 1.50 μ., τότε υπολογίζουμε πόση απόσταση πρέπει να

προστεθεί κάθε φόρα στον άξονα y της κάψουλας προκειμένου να βρίσκεται το κέντρο της κάθε στιγμή 1,50 μ. πάνω από το έδαφος. Για την εύρεση αυτού του αριθμού αφαιρούμε από τα 1.50 μ. την διαφορά θέσης κάψουλας στον άξονα y με τη θέση του σημείου πρόσκρουσης της ακτίνας στον άξονα y. Η διαδικασία αυτή επαναλαμβάνεται κάθε frame, επιτυγχάνοντας έτσι τη σταθερή και τη μόνιμη απόσταση της κάψουλας από το έδαφος συν την αίσθηση της αιώρησης και της ομαλής κίνησης του χρήστη στο χώρο. Για τον προγραμματισμό της λειτουργίας που περιγράψαμε, δημιουργήσαμε ένα Script με την ονομασία movement.cs το οποίο βρίσκεται προσκολλημένο ως συστατικό του αντικειμένου κάψουλας που αντιπροσωπεύει τον παίχτη.

```
5
6   float dist = transform.position.y - hit.point.y;
7
8   if (hit.transform.tag == ("terrain"))
9       {
10        if (transform.position.y - hit.point.y != 1.5)
11            {
12              float dist2 = 1.5f - dist;
13              transform.position = new Vector3(transform.position.x, transform.position.y + dist2, transform.position.z);
14            }
15         Physics.SphereCast(transform.position, 1.0f, -Vector3.up, out slopehit, 10);
16      }
17
```

Εικόνα 83: Κώδικας Floating του PlayerCapsule

Αν και η υλοποίηση αυτής της μεθόδου απέδωσε με επιτυχία στη συγκεκριμένη εφαρμογή, στη συνέχεια εμφανίστηκε ένα δευτερεύουσας σημασίας πρόβλημα που έπρεπε με τη σειρά του να επιλυθεί. Συγκεκριμένα, το πρόβλημα ήταν πως ο χρήστης μπορούσε να ανεβαίνει και να σκαρφαλώνει σε οποιαδήποτε κεκλιμένο επίπεδο ανεξάρτητα της γωνίας κλίσης. Αυτό βέβαια είναι γνωστό πρόβλημα στο χώρο των βιντεοπαιχνιδιών. Καταφέραμε να αντεπεξέλθουμε με επιτυχία στο αδιέξοδο αυτό χρησιμοποιώντας την τεχνική του Ray Casting με τον ακόλουθο τρόπο: Από το κέντρο της κάψουλας εκτοξεύεται μια ακτίνα 2,0 μέτρων προς την κατεύθυνση κίνησης του παίχτη. Εάν η ακτίνα εντοπίσει αντικείμενο σε μικρότερη απόσταση από αυτή των 0,55 μέτρων τότε ο χρήστης δεν μπορεί να κινηθεί άλλο προς αυτή την κατεύθυνση. Ο κώδικας που ευθύνεται για τη λειτουργία αυτή εμπεριέχεται στο script movement.cs.



Εικόνα 84: Floating and Sloaping

6.2.2 Ενσωμάτωση Oculus Rift και Mounted Leap Motion

Μετά τη δημιουργία του κατάλληλου μοντέλου κίνησης στο χώρο, επόμενο βήμα είναι ο προγραμματισμός του έτσι ώστε να μπορεί πλέον να εκτελεί τις εντολές που δίνονται από το χρήστη. Αρχικά έγινε εισαγωγή των κατάλληλων πακέτων στη Unity3D έτσι ώστε να είναι εφικτή η πρόσβαση στους μετασχηματισμούς του Oculus Rift και στην ενεργοποίηση του συστήματος εντοπισμού του Mounted Leap Motion. Επίσης, με το πακέτο ανάπτυξης του Leap Motion που εισάγαμε, δίνεται η δυνατότητα χρησιμοποίησης μιας ποικιλίας ειδικών εντολών και συναρτήσεων που αφορούν αποκλειστικά στη λειτουργία και στον προγραμματισμό του. Στη συνέχεια, αφαιρέθηκε η βασική κάμερα του παιχνιδιού και στη θέση της προστέθηκε το Prefab του Oculus Rift το οποίο περιέχει δύο κάμερες με διαφορετική οπτική γωνία η κάθε μια. Οι εικόνες των δύο αυτών καμερών μεταφέρονται στις δύο οθόνες του Oculus Rift. Οι εικόνες που παράγονται στις δύο διαφορετικές οθόνες συνδυάζονται στον ανθρώπινο εγκέφαλο που με τη σειρά του δημιουργεί μια ενιαία τρισδιάστατη στερεοσκοπική εικόνα. Όσον αφορά το Leap Motion, προστέθηκε το κατάλληλο Hand Controller στη Unity3D προκειμένου αυτό να μπορεί να εντοπίζει με επιτυχία τα χέρια (hand tracking) ενώ βρίσκεται τοποθετημένο πάνω στο Oculus Rift. Το Leap Motion παρέχει μεγάλη ποικιλία επιλογής γραφικών μοντέλων για την απεικόνιση των χεριών του χρήστη σε μορφή Perfab, καθώς και τη φυσική αυτών. Τα μοντέλα χεριών που επιλέξαμε έχουν όνομα “PepperMediumFullLeftHand” για το αριστερό χέρι και “PepperMediumFullRightHand” αντίστοιχα για το δεξί. Βασικό χαρακτηριστικό αυτών είναι το λευκό χρώμα τους και η προέκτασή τους μέχρι το ύψος του αγκώνα. Φυσικό μοντέλο εισάγαμε μόνο για το δεξί χέρι και αυτό γιατί μας είναι απαραίτητο για τη λειτουργικότητα των μενού. Οι μαγικές μπάλες επίθεσης προγραμματίστηκαν έτσι ώστε να αγνοούν την ύπαρξη του φυσικού μοντέλου του δεξιού χεριού. Αυτό έχει σαν αποτέλεσμα να μην ενεργοποιείται ο μηχανισμός σύγκρουσης τους και να μην εκρήγνυνται όταν έρχονται σε επαφή με το χέρι.

6.2.3 Χειρισμός Παίχτη

Όπως αναφέρθηκε και παραπάνω, ο χειρισμός του παίχτη γίνεται μέσω συνδυασμού του Leap Motion και του Oculus Rift. Στην περίπτωση του Oculus Rift εκτός της εμβύθισης που επιτυγχάνεται στο ψηφιακό κόσμο και την εικονική πραγματικότητα, γίνεται παράλληλα χρήση και των διαφόρων τιμών που προσδίδονται από το γυροσκόπιο που φέρει. ως προς τους άξονες x y z. Έτσι, αυτές οι τιμές ενημερώνουν αυτόματα το Rotation του transform της κύριας κάμερας. Μαζί με την κύρια κάμερα περιστρέφεται και η κάψουλα του παίχτη την οποία προσκολλήσαμε σε αυτή. Η αλληλεξάρτηση κύριας κάμερας και κάψουλας παίχτη είναι το κλειδί μιας ολοκληρωμένης τρισδιάστατης κίνησης στο ψηφιακό περιβάλλον. Η κύρια κάμερα είναι υπεύθυνη για τις περιστροφικές κινήσεις ενώ η κάψουλα για τις ευθύγραμμες. Επομένως, για να επιτευχτεί η τρισδιάστατη κίνηση, θα πρέπει το ένα αντικείμενο να ακολουθεί τις κινήσεις του άλλου και αντίστροφα.

Η σύνδεση κύριας κάμερας και κάψουλας παίχτη γίνεται με συνδυασμό δύο λειτουργιών

Α) **Συγχρονισμός Περιστροφής:** Αξιοποιώντας τη λειτουργία του Head Tracking, γίνεται άμεση ενημέρωση της κύριας κάμερας από την περιστροφική κίνηση της κεφαλής του χρήστη. Οι συντεταγμένες της περιστροφικής κίνησης της κύριας κάμερα αντλούνται από την κάψουλα παίχτη μέσω του movement.sc που βρίσκεται προσκολλημένο πάνω στην κάψουλα. Πιο συγκεκριμένη, η πληροφορία περιστροφής κάμερας αντλείται ως προς άξονα y και εφαρμόζεται στις συντεταγμένες της κάψουλας σε κάθε frame. Αν δε γινότανε αυτό, τότε η κατεύθυνση του οπτικού πεδίου της κάμερας δε θα συσχετιζότανε με το μπροστινό μέρος της κάψουλας. Αυτό θα είχε ως αποτέλεσμα να μην συγχρονίζονταν αυτά τα δύο αντικείμενα. Η κάψουλα θα ακολουθούσε ευθύγραμμη κίνηση σύμφωνα με την αρχική παραδοχή του μπροστινού της μέρος κατά την τοποθέτησή της στον ψηφιακό κόσμο ανεξάρτητα την κάμερα. Εντολή άντλησης συντεταγμένων περιστροφής:

```
transform.localEulerAngles = new Vector3(0, GameObject.Find("LMHeadMountedRig/TrackingSpace/CenterEyeAnchor").transform.localEulerAngles.y, 0);
```

Β) **Συγχρονισμός Κίνησης:** Αξιοποιώντας τη λειτουργία Hand Tracking, γίνεται άμεση ενημέρωση της κάψουλας παίχτη από τις χειρονομίες του χρήστη. Οι συντεταγμένες της θέσης του parent object κάτω από το οποίο ιεραρχικά βρίσκεται η κάμερα, ενημερώνονται μέσω του movement.sc που είναι προσκολλημένο στον αντικείμενο “Camera” της κάψουλα του παίχτη το οποίο δε λειτουργεί σαν σημείο αναφοράς για την κύρια κάμερα. Πιο συγκεκριμένα, η πληροφορία θέσης κάψουλας προσδίδεται στην κάμερα μέσω εντολής. Αν δε γινότανε αυτό, η κάμερα θα παρέμενε στο ίδιο σημείο ενώ η κάψουλα θα απομακρυνότανε. Εντολή αποστολής συντεταγμένων θέσης:

```
GameObject.Find("LMHeadMountedRig").transform.position = GameObject.Find("PlayerCapsule/Camera").transform.position;
```

Έτσι, για να πραγματοποιηθεί μια ολοκληρωμένη κίνηση στο εικονικό περιβάλλον της εφαρμογής μας θα χρειαστεί συνδυασμό των δύο παραπάνω λειτουργιών

6.2.4 Κινήσεις Χεριών

Σε αυτό το υποκεφάλαιο θα αναλυθούν με τεχνικούς όρους και με βάση κώδικα οι κινήσεις των χεριών που γίνονται αντιληπτές από το Leap Motion και οι οποίες έχουν περιγραφεί εν μέρει στο προηγούμενο κεφάλαιο. Ένα πολύ σημαντικό στοιχείο το οποίο θα μας βοηθήσει να αντιληφθούμε πληρέστερα τη λειτουργία του Leap Motion είναι η βαθύτερη κατανόηση κάποιων συγκεκριμένων κλάσεων.

Κλάση τύπου frame: αντιπροσωπεύει ένα σύνολο δεδομένων εντοπισμού χεριών και δακτύλων που ανιχνεύονται σε ένα μοναδικό frame. Το λογισμικό του Leap Motion ανιχνεύει τα χέρια, τα δάκτυλα και τα εργαλεία μέσα στην περιοχή παρακολούθησης, αναφέροντας τις θέσεις, τις χειρονομίες και τις κινήσεις τους σε κάθε frame.

Κλάση τύπου Hand: αναφέρει τα φυσικά χαρακτηριστικά κάθε χεριού που ανιχνεύεται. Τα δεδομένα παρακολούθησης χεριού περιλαμβάνουν τη θέση, τη ταχύτητα της παλάμης, το διάνυσμα κατεύθυνσης της προς τα δάκτυλα (Palm direction), το διάνυσμα έξωθεν και κάθετα αυτής (Palm normal), λίστες που εμπεριέχουν τα δάκτυλα που προεξέχουν από την παλάμη και τις ιδιότητες μιας σφαίρας που χωράει στο χέρι.

Κλάση τύπου Finger: περιέχει πληροφορίες για κάθε δάκτυλο όπως την ονομασία, τον τύπο, τα οστά και άλλες.

Ένα αντικείμενο τύπου Hand μπορεί να εισαχθεί μέσω ενός αντικειμένου τύπου Frame. Η λίστα τύπου FingerList αντιπροσωπεύει μια λίστα με αντικείμενα του τύπου Finger. Τα Finger είναι μια κλάση που αντιπροσωπεύει ένα εντοπισμένο δάκτυλο. Ένα αντικείμενο τύπου Finger μπορεί να αποκτηθεί μόνο μέσω ενός αντικειμένου τύπου Frame ή ενός αντικειμένου τύπου Hand.

Κλειστή παλάμη αριστερού χεριού

Η τεχνική υλοποίησης αυτής της χειρονομίας στηρίζεται στην ιδέα ότι η FingerList του αριστερού χεριού θα πρέπει να μην περιέχει κανένα στοιχείο. Για να είναι κενή η λίστα σημαίνει ότι δεν προεξέχει κανένα δάκτυλο.

```
1
2
3
4   FingerList extendedFingerList;
5
6   Hand Lefthand= frame.Hands.Leftmost;
7   extendedFingerList=lefthand.fingers.extended();
8 ▸ if (extendedFingerListsEmpty){...}
11
12
```

Εικόνα 85: Συνθήκη Εντοπισμού Κλειστής Αριστερής Παλάμης

Στροφή αριστερής παλάμης προς τα δεξιά ή προς τα αριστερά

Για να εντοπιστεί αυτή η κίνηση χρησιμοποιούμε τη συνάρτηση της κλάσης Hand, την PalmNormal. Η συνάρτηση αυτή δίνει ένα διάνυσμα κάθετο που ξεκινάει από την εσωτερική επιφάνεια της παλάμης. Το διάνυσμα "δείχνει" προς τα κάτω, έξω από την παλάμη και έτσι μπορούμε να ελέγξουμε προς ποια κατεύθυνση είναι αυτό στραμμένο και κατ' επέκταση προς τα πού είναι στραμμένη η παλάμη. Για να ελέγξουμε την περιστροφή της παλάμης αντλούμε από την κλάση διανύσματος τη γωνία περιστροφής γύρω από τον άξονα εμπρός/πίσω με τη συνάρτηση Roll, μετρούμενη σε ακτίνια. Το ακτίνιο (Rad) είναι μονάδα μέτρησης γωνίας. Ένα ακτίνιο (1 Rad) είναι η επίπεδη γωνία που όταν γίνει επίκεντρη ορίζει τόξο σε οποιοδήποτε κύκλο με μήκος ίσο με την ακτίνα του. Έτσι, αν η τιμή που λαμβάνεται από τη συνάρτηση Roll βρίσκεται στα επιθυμητά όρια, υποδηλώνεται ότι η παλάμη εκτέλεσε την ζητούμενη περιστροφική κίνηση και κατ' επέκταση θα πραγματοποιηθεί επιτυχώς η ενέργεια που σχετίζεται με αυτήν για όσα frame παραμένει σε αυτή τη θέση η παλάμη. Στην στροφή αριστερής παλάμης προς τα αριστερά ισχύει ό,τι και στην στροφή αριστερής παλάμης προς τα δεξιά με μόνη διαφορά ότι αλλάζουν τα όρια των ακτινίων.

```
4   float roll = Lefthand.PalmNormal.Roll;
5 ▾ if (roll > 1.5f && roll <= 2.5f ){
```

Εικόνα 86: Όρια Δεξιάς Στροφής Καρπού

```
10 if (roll < -0.5 && roll >= -2.5f){
```

Εικόνα 87: Όρια Αριστερής Στροφής Καρπού

Προεξοχή τριών δακτύλων αριστερού χεριού

Ακολουθούνται τα ίδια βήματα όπως αυτά της κλειστής παλάμης που ήδη αναφέρθηκε με τη διαφορά ότι πρέπει να ελεγχθεί αν το FingerList περιέχει τρία στοιχεία τα οποία ονομαστικά αντιστοιχούν στο μέσο, παράμεσο και μικρό δάκτυλο. Κάθε στοιχείο της λίστας αντιστοιχεί σε ένα τεντωμένο δάκτυλο. Αν το δάκτυλο δεν προεξέχει, δεν ανιχνεύεται και άρα δεν καταχωρείται στη λίστα.

Κλειστή αριστερή παλάμη με δεξιά ή αριστερή στροφή

Δεν υπάρχει συγκεκριμένος κώδικας για αυτή την κίνηση απλώς γίνεται αυτόματος συνδυασμός των εντολών μέσω script των κινήσεων της κλειστής παλάμης αριστερού χεριού με την στροφή αριστερής παλάμης προς τα δεξιά ή προς τα αριστερά.

Προεξέχοντα τρία δάκτυλα αριστερού χεριού με δεξιά ή αριστερή στροφή

Πραγματοποιείται με αυτόματο συνδυασμό εντολών μέσω script. Οι κινήσεις που συνδυάζονται είναι τα προεξέχοντα τρία δάκτυλα αριστερού χεριού με στροφή αριστερής παλάμης προς τα δεξιά ή προς τα αριστερά.

Ανοιχτή παλάμη δεξιού χεριού στοχεύοντας έμπροσθεν και με παράλληλη σύντομη και απότομη εκτίναξη προς την κατεύθυνση αυτή

Πρόκειται για συνδυασμό κινήσεων σε προγραμματιστικό επίπεδο. Για την επικύρωση της ορθότητας της κίνησης του χρήστη απαιτείται και εκτελείται σειρά ελέγχων. Πρώτα πρέπει να ελεγχθεί εάν η παλάμη του δεξιού χεριού είναι ανοιχτή και η κίνηση της προς το άξονα y (y γιατί είναι Mounted Leap Motion) κατακτά μια συγκεκριμένη κατώτερη ταχύτητα που μετριέται σε millimeters/second. Η μέτρηση της ταχύτητας της παλάμης κατά τον άξονα y γίνεται κατόπιν εντολής του Palmvelocity.y. Στη συνέχεια ελέγχονται οι συντεταγμένες του διανύσματος που "φεύγει" από την εσωτερική πλευρά της παλάμης και προς τα έξω, ως προς τους άξονες z και y ώστε να εντοπίσουμε τις επιθυμητές τιμές του διανύσματος στο χώρο σε σχέση με την ταχύτητα του πάνω στο άξονα z. Επίσης κατά τη διάρκεια των ελέγχων χρησιμοποιούμε ορισμένες σημαίες (flag) για να αποφευχθούν ανεπιθύμητες ενέργειες που θα επηρεάσουν την ομαλή εξέλιξη του παιχνιδιού. Με την επιτυχή ολοκλήρωση της χειρονομίας γεννάται ένας κλώνος μπάλας φωτιάς ή πράσινης μπάλας μέσω της συνάρτησης instantiate. Η συνάρτηση αυτή όταν εκτελεστεί παράγει ένα κλώνο του Prefab που της έχει δοθεί ως είσοδος μέσω script. Οι συντεταγμένες του σημείου δίνονται και αυτές ως είσοδος της συνάρτησης. Οι συνθήκες που σχετίζονται με τον έλεγχο της περιστροφής του διανύσματος θα μπορούσαν να είχαν υλοποιηθεί και με το σύστημα αξόνων pitch, yaw, roll πράγμα που δεν έγινε γιατί δεν ήμασταν εξοικειωμένοι τότε με αυτό.

```
extendFingers = RightHand.Fingers.Extended();
if (extendFingers.Count >= 4 && controller.Frame().Hands.Leftmost.PalmVelocity.y >700 && fireflag == 0 && thunderflag == 0)
{
    if (RightHand.PalmNormal.z <= 0.65 && RightHand.PalmNormal.z >= -0.30)
    {

        if (RightHand.PalmNormal.y >= 0.6)
        {
```

Εικόνα 88: Έλεγχοι για Εκτόξευση Πρασινης Μπάλας ή Φωτιάς

Ανοιχτή παλάμη αριστερού χεριού με στόχο έμπροσθεν και με παράλληλη σύντομη και απότομη κίνηση προς την κατεύθυνση αυτή

Η φιλοσοφία είναι ίδια όπως και προηγουμένως απλώς ελέγχεται εάν η κίνηση γίνεται με το αριστερό χέρι. Υπάρχουν επίσης μικρές διαφοροποιήσεις σε κάποια όρια που αναφέρονται στο PalmNormal.

```
extendFingers = LeftHand.Fingers.Extended();
    if (GameObject.Find("PlayerCapsule").GetComponent<GamePlay>().death == 0 && pause ==0 && extendFingers.Count >= 4
    && controller.Frame().Hands.Rightmost.PalmVelocity.y > 700 && iceFlag == 0 && controller.Frame().Hands.Rightmost.IsLeft)
    {

        if (LeftHand.PalmNormal.z <= 0.70 && LeftHand.PalmNormal.z >= -0.30)
        {
            iceFlag = 1;
            if (LeftHand.PalmNormal.y >= 0.4)
            {



                if (iceFlag == 1)
                {

                    if (GameObject.Find("PlayerCapsule").GetComponent<GamePlay>().PlayerMP >= 20) {
```

Εικόνα 89: Κώδικας Επίθεσης Αριστερού Χεριού

Ανοιχτή δεξιά παλάμη και στροφή προς τα πάνω για πέντε δευτερόλεπτα

Η συγκεκριμένη κίνηση χαρακτηρίζεται από πολυπλοκότητα ελέγχων καθώς κατά την εξέλιξη της κίνησης ενεργοποιούνται και απενεργοποιούνται συστήματα σωματιδίων ανά συγκεκριμένα χρονικά διαστήματα ως και ηχητικά εφέ. Γίνονται έλεγχοι σχετικά με τον αριθμό των δαχτύλων του δεξιού χεριού και του διανύσματος PalmNormal ως προς z του ίδιου χεριού ώστε αυτό να δείχνει προς τα πάνω. Αν η κίνηση διακοπεί πριν το πέρας των πέντε δευτερολέπτων θα απενεργοποιηθούν τα συστήματα σωματιδίων όπως επίσης και οι ήχοι. Αν ολοκληρωθούν τα πέντε δευτερόλεπτα τα συστήματα σωματιδίων και οι ήχοι εξακολουθούν να είναι ενεργοί μέχρις ότου ο χρήστης εκτελέσει την κίνηση ανοιχτής

παλάμης δεξιού χεριού στοχεύοντας έμπροσθεν με παράλληλη σύντομη και απότομη κίνηση. Πραγματοποιώντας αυτή την κίνηση αντί κλώνου μπάλας φωτιάς δημιουργείται κλώνος πράσινης μπάλας αφού ναι μεν εκτελείται ξανά η συνάρτηση instantiate αλλά σε άλλο σημείο του κώδικα και με άλλο prefab ως είσοδο. Ο κώδικας που αναφέρεται στις κινήσεις του δεξιού χεριού βρίσκεται στο script με ονομασία HandVelocity.cs, ενώ ο κώδικας που αναφέρεται στην κίνηση εκτόξευσης μπάλας πάγου στο script IceAttack.cs.

Έκταση δείκτη και αντίχειρα ταυτόχρονα και των δύο χεριών

Είναι χειρονομία μενού παύσης. Για την κίνηση αυτή αρχικά δημιουργήσαμε δύο FingerList, η μια εκ των δύο αντιστοιχεί στο δεξί χέρι και η άλλη στο αριστερό. Σε κάθε μια από αυτές τοποθετούνται τα προτεταμένα δάκτυλα του χεριού στο οποίο αναλογεί. Στη συνέχεια, γίνεται έλεγχος προκείμενου να εντοπίσουμε εάν υπάρχουν δύο μόνο στοιχεία σε κάθε λίστα και εξετάζουμε εάν τα ονόματα των στοιχείων αυτών αντιστοιχούν στον αντίχειρα και το δείκτη. Για να εμφανιστεί το μενού παύσης θα πρέπει η ανωτέρω χειρονομία να έχει διάρκεια 1.5 δευτερόλεπτα.

```
4   if (GameObject.Find("PlayerCapsule").GetComponent<GamePlay>().death == 0
5   && pause == 0
6   && RH.IsRight
7   &
8   && LeftHand.IsLeft
9   && RightFingers.Count <= 3 && LeftFingers.Count <= 3
10  && RightFingers[0].Type.ToString() == "TYPE_THUMB"
11  && LeftFingers[0].Type.ToString() == "TYPE_THUMB"
12  && LeftFingers[1].Type.ToString() == "TYPE_INDEX"
13  && RightFingers[1].Type.ToString() == "TYPE_INDEX")
```

Εικόνα 90: Κώδικας Προεξέχοντα Αντίχειρα και Δείκτη Δεξιού και Αριστερού Χεριού

6.2.5 Τεχνικά Στοιχεία Μπάλας Φωτιάς

Η μπάλα φωτιάς είναι μια σφαίρα της οποίας έχει απενεργοποιηθεί η οπτική απόδοση της επιφάνειας της ενώ αντίστοιχα έχει ενεργοποιηθεί ο μηχανισμό φυσικής της ώστε να μπορεί να αντιλαμβάνεται συγκρούσεις με άλλα στερεά αντικείμενα. Την περιβάλλουν διάφορα συστήματα σωματιδίων που είναι προσκολλημένα σε αυτή και ιεραρχικά κάτω από αυτήν.

Δημιουργήσαμε ένα script με ονομασία velocityscript το οποίο καθορίζει τη συμπεριφορά και την κίνηση της μπάλας. Από το κέντρο της κύριας κάμερας εκχέεται πάντα μια ακτίνα (Ray cast). Στο velocityscript εισάγουμε τις συντεταγμένες του σημείου που θα προσκρούει ή θα σταματά η ακτίνα της κύριας κάμερας. Αυτό πραγματοποιείται μόνο μια φορά μέσα στη συνάρτηση void start() με την εξής εντολή

```
vector3 targetDirection=GameObject.Find(“camera”).GetComponent< cameraAim>().HitPoint;
```

Στη συνέχεια, θα πρέπει να μετακινήσουμε την μπάλα από το σημείο δημιουργίας της με κατεύθυνση προς τις συντεταγμένες του σημείου που αντλήσαμε από την παραπάνω εντολή εκτελώντας αυστηρά ευθύγραμμη κίνηση. Για να γίνει αυτό θα εκτελείται η εξής συνάρτηση σε κάθε frame ώστε η ίδια να ενημερώνεται συνεχώς

```
transform.position=vector3.MoveTowards(transform.position,targetDirection,step);
```

Η συνάρτηση αυτή μετακινεί ένα αντικείμενο από ένα σημείο προς ένα άλλο τα οποία της δίνονται σαν είσοδος, εκτελώντας ευθύγραμμη κίνηση με συγκεκριμένο ρυθμό που καθορίζεται από τη μεταβλητή step. Η μπάλα δεν μπορεί να κινείται επ' αόριστον. Σε περίπτωση που αυτή συγκρουστεί με κάποιο στερεό αντικείμενο πριν φθάσει στο τερματικό σημείο, παράγει μέσω της συνάρτησης instatiate ένα κλώνο του prefab που έχει δοθεί σαν είσοδος στην instatiate. Στην συγκεκριμένη περίπτωση πρόκειται για ένα prefab έκρηξης. Αφού παραχθεί ο κλώνος της έκρηξης, η μπάλα αυτοκαταστρέφεται με την εντολή Destroy(gameobject). Αν πάλι η μπάλα δε συγκρουστεί με κάποιο στερεό αντικείμενο, για ένα συγκεκριμένο χρονικό διάστημα πυροδοτείται έκρηξη και η μπάλα αυτοκαταστρέφεται με τον τρόπο που περιγράψαμε παραπάνω πριν φθάσει στο τερματικό σημείο. Για να ελέγξουμε τα χρονικά περιθώρια, χρησιμοποιούμε τις εντολές

```
3   float timeleft=0.45f;
4   If (timeleft<=0){[redacted]}
```

Εικόνα 91: Κώδικας Ελέγχου Χρονικών Περιθωρίων Μπάλας Επίθεσης

Έτσι σε κάθε frame ενημερώνεται το χρονόμετρο και ελέγχουμε εάν έχει εξαντληθεί το χρονικό διάστημα που ορίσαμε σαν όριο διάρκειας ζωής της μπάλας φωτιάς. Με την ίδια λογική συμπεριφέρονται και οι υπόλοιπες δύο μπάλες (πράσινη και πάγου) με τη διαφορά ότι η μπάλα πάγου εκτελεί έναν επιπλέον έλεγχο προκειμένου να πραγματοποιήσει την κατάλληλη έκρηξη ανάλογα με το αντικείμενο με το οποίο προσκρούει. Η μπάλα πάγου χρησιμοποιεί ένα script με όνομα IceVelocityScript όπου γίνονται οι αντίστοιχες ενέργειες και έλεγχοι όπως και στις άλλες δύο μπάλες. Η διαφορά έγκειται στον έλεγχο του ονόματος του αντικειμένου με το οποίο προσκρούει. Αν αυτό ονομάζεται Terrain, τότε μέσω της εντολής instantiate παράγεται κλώνος του prefab έκρηξη σταλαγμιτών. Σε αντίθετη περίπτωση προκύπτει κλώνος prefab που είχε δοθεί σαν δεύτερη επιλογή έκρηξης.

6.3 Συστήματα Σωματιδίων και Χειρονομίες

Τα συστήματα σωματιδίων, όπως περιγράψαμε και σε προηγούμενο κεφάλαιο, χρησιμοποιούνται συνεχώς στα σύγχρονα βιντεοπαιχνίδια καθώς παρέχουν σπουδαία οπτικά εφέ καταναλώνοντας σχετικά λίγους υπολογιστικούς πόρους. Η συγκεκριμένη εργασία εμπεριέχει πληθώρα συστημάτων σωματιδίων, το μεγαλύτερο ποσοστό των οποίων λειτουργεί αυτόματα μέσα στο παιχνίδι χωρίς να απαιτούνται παράλληλα διάφορες ρυθμίσεις μέσα από κάποιο script. Η Unity3D ονομάζει το ανανεωμένο σύστημα σωματιδίων "shuriken". Στην περίπτωση της χειρονομίας ανοιχτής δεξιάς παλάμης με στροφή προς τα πάνω προγραμματίσαμε την ενεργοποίηση διαφόρων συστημάτων σωματιδίων που είναι προσκολλημένα πάνω στη δεξιά παλάμη σύμφωνα με κάποια χρονικά όρια που θέσαμε.

Ενδεικτική δήλωση, ενεργοποίηση και απενεργοποίηση συστημάτων σωματιδίων.

```
3
4  ParticleSystem ps;
5      ParticleSystem.EmissionMadule psemit;
6
```

Εικόνα 92: Δήλωση Συστήματος Σωματιδίου

```
3
4  ps=GameObject.particleSystem;
5     psemit=ps.emission;
6     psemit.enabled=true;
7     ps.play;
8
```

Εικόνα 93: Ενεργοποίηση Συστήματος Σωματιδίου

```
3
4    psemit.enabled=false;
5       ps.stop;
6
```

Εικόνα 94: Απενεργοποίηση Συστήματος Σωματιδίου

Πιο συγκεκριμένα, μετά το πέρας του ενός δευτερολέπτου από τα πέντε που απαιτούνται για τη διάρκεια της χειρονομίας, ενεργοποιείται η εκπομπή συστημάτων σωματιδίων προκειμένου να αρχίσουν τα κατάλληλα οπτικά εφέ.

Το αντικείμενο PS είναι τύπου Particle System στο οποίο τοποθετήσαμε το σύστημα σωματιδίων που θέλουμε να επεξεργαστούμε. Το psemit είναι τύπου μονάδας εκπομπής σωματιδίων και εκχωρούμε σε αυτό την μονάδα εκπομπής σωματιδίων του συστήματος σωματιδίων που τοποθετήσαμε στο αντικείμενο. Έτσι ενεργοποιούμε την μονάδα εκπομπής με την εντολή psemit.enabled = true;. Κατόπιν, για να λειτουργήσει το σύστημα σωματιδίων ώστε στη συνέχεια να ενεργοποιηθεί και η εκπομπή των σωματιδίων, απαιτείται η εντολή ps.play;. Με την παρέλευση και των πέντε δευτερολέπτων ενεργοποιούνται άλλα τρία συστήματα σωματιδίων με τον ίδιο τρόπο.

```
3
4  If (time left<=4){
5     Ps=GameObject.find(pepper/energyBlast).GetComponent<ParticleSystem>();
6     Psemit=ps.emission;
7     Psemit.enabled-true;
8     Ps.play();
9  }
10
```

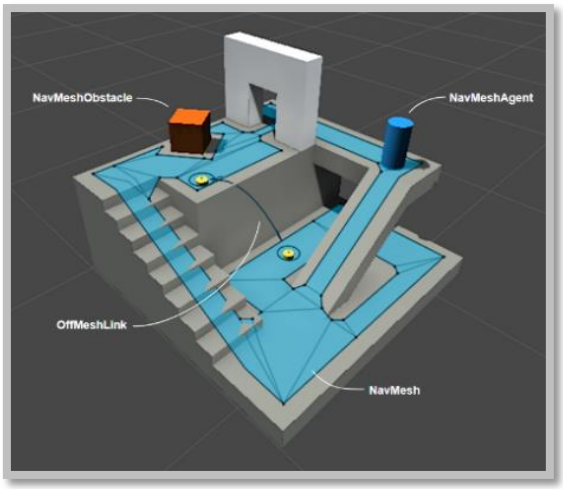
Εικόνα 95: Κώδικας Ενεργοποίησης Συστήματος Σωματιδίου σε Σχέση με το Χρόνο

Η απενεργοποίηση των συστημάτων σωματιδίων γίνεται με μια σειρά παρόμοιων εντολών. Ο κώδικας που αναφέραμε σχετικά με την ενεργοποίηση και απενεργοποίηση των σωματιδίων βρίσκεται στο script HandVelocity.cs. Σε όλες τις υπόλοιπες περιπτώσεις που προκύπτουν συστήματα σωματιδίων, η λειτουργία τους γίνεται αυτόματα χωρίς να χρειάζεται να παρέμβουμε μέσω κώδικα.

6.4 Τεχνητή Νοημοσύνη και Χαρακτήρες

Προκειμένου να δημιουργήσουμε και να εισάγουμε μια μορφή τεχνητής νοημοσύνης στο παιχνίδι είναι απαραίτητη η χρήση του Navigation System της Unity μέσω του οποίου δημιουργούμε «ψευδοσυμπεριφορά» σε χαρακτήρες της εφαρμογής . Μέσω του συστήματος αυτού γίνεται ανάγνωση του εδάφους δημιουργώντας πλέγματα πλοήγησης, τα navigation meshes ή αλλιώς navmeshes. Τα πλέγματα πλοήγησης προκύπτουν έπειτα από τη διαδικασία Baking (ψησίματος) της επιφάνειας ώστε να καθοριστούν ποιες περιοχές είναι πλοηγήσιμες. Πιο αναλυτικά, τα βήματα που πρέπει να ακολουθήσουμε για τη δημιουργία πλεγμάτων πλοήγησης είναι τα εξής: Επιλέγουμε την περιοχή όπου θέλουμε να εφαρμοστεί η διαδικασία της πλοήγησης και ενεργοποιούμε τη στατική πλοήγηση ώστε να συμπεριληφθούν τα επιλεγμένα αντικείμενα στη διαδικασία του ψησίματος. Έπειτα, τροποποιούμε τις ρυθμίσεις ψησίματος ώστε να ταιριάζουν με το μέγεθος του πράκτορα που θέλουμε να δημιουργήσουμε. Με τον όρο πράκτορα θεωρούμε το αντικείμενο που μπορεί να πλοηγηθεί στη σκηνή. Η ακτίνα του πράκτορα καθορίζει πόσο κοντά αυτός μπορεί να πλησιάσει στερεά αντικείμενα ενώ το ύψος του καθορίζει το μέγιστο ύψος κάτω από το οποίο μπορεί να διέλθει. Η μέγιστη κλήση ενός πράκτορα σηματοδοτεί τη μέγιστη κλήση πάνω στην οποία μπορεί να πλοηγηθεί. Τέλος, μπορεί να καθοριστεί το μέγιστο ύψος εμποδίου που μπορεί ο πράκτορας να υπερπηδήσει.

Αφού δημιουργηθεί πλέγμα πλοήγησης της περιοχής που επιθυμούμε, το επόμενο βήμα για τη δημιουργία αυτόνομων πρακτόρων που περιπλανιούνται στη περιοχή είναι να διαβαστούν και να επεξεργαστούν τα δεδομένα αυτά από τους διάφορους χαρακτήρες που έχουν προστεθεί στην πίστα. Για να είναι δυνατή η ανάγνωση των δεδομένων από κάποιο χαρακτήρα θα πρέπει να προσθέσουμε σε αυτόν το συστατικό NavMeshAgent. Ο NavMeshAgent είναι ένας νοητός κύλινδρος γύρω από τον εκάστοτε χαρακτήρα. Διάφορα χαρακτηριστικά του αυτόνομου πράκτορα μπορούν να ρυθμιστούν μέσα από το συστατικό NavMeshAgent όπως η ακτίνα του κυλίνδρου, η ταχύτητα, η επιτάχυνση και άλλα.



Εικόνα 96: Εύρεση του Καλύτερου Μονοπατιού

6.5 Χαρακτήρες – Αυτόνομοι Πράκτορες και Animation

Όπως αναφέραμε, οι χαρακτήρες που χρησιμοποιήσαμε και τους προσδώσαμε την ιδιότητα του αυτόνομου πράκτορα είναι οι εξής πέντε: αρκούδα, λύκος, ελάφι, σκελετός και τελικός αρχηγός. Κάθε ένας αυτόνομος πράκτορας έχει τη δική του ξεχωριστή συμπεριφορά, η οποία διαμορφώνεται με το συστατικό NavmeshAgent σε συνδυασμό με κάποιο script. Για να δημιουργηθεί ένας αυτόνομος πράκτορας δεν αρκεί μόνο η προσκόλληση του συστατικού NavmeshAgent σε κάποιο χαρακτήρα αλλά είναι απαραίτητη και η δημιουργία ενός κατάλληλου script το οποίο θα εισαχθεί σε αυτόν. Μέσα στο script θα υπάρχουν διάφορες εντολές οι οποίες θα παρεμβαίνουν στην κίνηση του χαρακτήρα ρυθμίζοντας μέσα από αυτό τα διάφορα πεδία του συστατικού NavmeshAgent. Επίσης, μέσα από το script μπορούμε να παρέμβουμε στον ήχο του χαρακτήρα αλλά και στα κινούμενα σχέδιά του.

Όσον αφορά τα κινούμενα σχέδια, η νέα έκδοση της Unity3D χρησιμοποιεί Animator Controller (ελεγκτής κινουμένων σχεδίων). Ένας Animator Controller δημιουργείται μέσα στη Unity και μας επιτρέπει να οργανώσουμε και να καθορίσουμε ένα σύνολο κινουμένων σχεδίων για έναν χαρακτήρα ή ένα αντικείμενο. Στις περισσότερες περιπτώσεις είναι φυσιολογικό να έχουμε πολλαπλά κινούμενα σχέδια και να εναλλάσσονται μεταξύ τους υπό κατάλληλες πάντα συνθήκες. Ο Animator Controller επηρεάζει τα διάφορα clip κινουμένων σχεδίων που χρησιμοποιούνται μέσα σε αυτόν, διαχειρίζοντας τις διάφορες καταστάσεις κινουμένων σχεδίων όπως και τις μεταβάσεις μεταξύ τους χρησιμοποιώντας την γνωστή μηχανή σε όλους “μηχανή καταστάσεων” (state machine).

6.5.1 Αρκούδα

Για να αποδώσουμε με ρεαλιστικότητα τη συμπεριφορά της αρκούδας ήταν απαραίτητο να εισάγουμε σε αυτήν έξι διαφορετικά συστατικά. Έναν animator, δύο script το BearMovement και το BearCollision, ένα audio source και ένα capsule collider. Το Animator εισάγει μια state machine σύμφωνα με την οποία η αρκούδα πραγματοποιεί κινούμενα σχέδια ανάλογα των καταστάσεων. Το audio source είναι υπεύθυνο για την παραγωγή και διάδοση του ήχου ενώ το capsule collider για την ανίχνευση διαφόρων συγκρούσεων. Το script

BearCollision μειώνει την ενέργεια της αρκούδας ανάλογα το είδος της μπάλας επίθεσης με την οποία συγκρούεται, δηλαδή επεξεργάζεται την πληροφορία της σύγκρουσης την οποία λαμβάνει από το capsule collider. Το script BearMovement αναλαμβάνει τη διαχείριση όλων των παραπάνω συστατικών. Σ αυτό το script δίνονται σαν είσοδοι δύο σημεία στο χώρο δια μέσου των οποίων πρέπει να κινείται η αρκούδα και ορίζονται ως στόχος του NavmeshAgent εναλλάξ το κάθε ένα από αυτά. Ο στόχος εναλλάσσεται μεταξύ αυτών των δύο σημείων όταν η αρκούδα τον πλησιάσει και εφόσον η απόσταση του παίχτη από αυτή είναι μεγαλύτερη των 30 μέτρων. Εάν ο παίχτης βρίσκεται σε απόσταση μικρότερη των 30 μέτρων, ο στόχος του πράκτορα-αρκούδα αλλάζει και είναι πλέον ο ίδιος ο παίχτης.

Επίσης, εάν ο παίχτης δεν έχει χτυπήσει την αρκούδα και παράλληλα δεν την έχει πλησιάσει λιγότερο από 15 μέτρα, τότε αν απομακρυνθεί σε απόσταση μεγαλύτερη των 30 μέτρων από αυτήν, ο στόχος του πράκτορα θα γίνει πάλι κάποιο από τα δύο αρχικά σημεία.

Όταν η απόσταση παίχτη αρκούδας για πρώτη φορά είναι μικρότερη των 15 μέτρων ή εάν η αρκούδα χτυπηθεί από οποιαδήποτε μαγική μπάλα του χρήστη ενώ η μεταξύ τους απόσταση δεν ήταν ποτέ έως τότε μικρότερη των 15 μέτρων, τότε η αρκούδα πραγματοποιεί κινούμενο σχέδιο “Roar” και παραμένει στην ίδια θέση μέχρι αυτό να ολοκληρωθεί. Αυτό συμβαίνει γιατί η απόσταση μεταξύ παίχτη και αρκούδας σε κάθε frame δίνεται σαν είσοδος από το script BearMovement στον animator controller μέσω της εντολής `anim.SetFloat("Dist", dist);`.

Έτσι, μόλις δοθεί απόσταση μικρότερη των 15 μέτρων ή η αρκούδα χτυπηθεί από απόσταση μεγαλύτερη των 15 μέτρων, τότε αυτή μεταβαίνει από την κατάσταση walk στην κατάσταση roar. Παράλληλα γίνεται έλεγχος στο BearMovement script σχετικά με ποιο κινούμενο σχέδιο πραγματοποιεί η αρκούδα το συγκεκριμένο frame. Εάν η αρκούδα εκτελεί το “roar” τότε δίνεται εντολή στο συστατικό NavMeshAgent να μηδενίσει την ταχύτητα του ενώ αν εκτελεί το κινούμενο σχέδιο walk η run τότε αντίστοιχα δίνεται εντολή στο συστατικό NavMeshAgent να αυξήσει την ταχύτητά του. Εάν η απόσταση μεταξύ παίχτη και αρκούδας είναι 4 μέτρα ή λιγότερο, τότε στον animator controller δίνεται σαν είσοδος εκτός από την απόσταση και η τυχαία τιμή της συνάρτηση Random.Value. Σύμφωνα με την τιμή αυτή ο animator controller θα επιλέξει το αντίστοιχο clip επίθεσης ανάμεσα σε τρία διαφορετικά ώστε να εκτελεστεί το αντίστοιχο animation. Το BearMovement script εντοπίζει το κινούμενο σχέδιο της επίθεσης και παράλληλα δίνει εντολή στο συστατικό NavMeshAgent να μηδενιστεί η ταχύτητα κινήσεως της αρκούδας ως αυτόνομος πράκτορας. Ο animator controller μεταβαίνει στο clip idle ρυθμίζοντας έτσι ώστε ο πράκτορας να έχει μηδενική ταχύτητα. Η κατάσταση Idle διαρκεί ελάχιστα δευτερόλεπτα. Μόλις τελειώσει η κατάσταση αυτή πραγματοποιούνται ξανά έλεγχοι από τον animator controller ως προς το κινούμενο σχέδιο που θα πρέπει να εκτελεστεί. Ανάλογα λοιπόν αυτό, το BearMovement script ρυθμίζει την ταχύτητα του πράκτορα.

Στην περίπτωση που η αρκούδα χτυπηθεί από μπάλα πάγου, το script BearMovement αντλεί την πληροφορία από το script BearCollision. Εάν η αρκούδα εκτελεί κινούμενο σχέδιο run, μειώνεται η ταχύτητα του κινούμενου σχεδίου μέσω του animator controller αλλά και η ταχύτητα κίνησης της αρκούδας προς τον παίχτη μέσω της ενημέρωσης του συστατικού NavMeshAgent για τρία δευτερόλεπτα. Η μείωση της ταχύτητας κινούμενων σχεδίων πραγματοποιείται μόνο για το clip run ενώ τα υπόλοιπα animation πραγματοποιούνται κανονικά μέσα σε αυτά τα τρία δευτερόλεπτα. Αφού εκλείψει η επήρεια της καθυστέρησης τότε η ροή του animation clip και η ταχύτητα κίνησης της αρκούδας επανέρχονται στα φυσιολογικά επίπεδα.

Τέλος, σε περίπτωση που η αρκούδα χάσει όλη της την ενέργεια, γεγονός που ελέγχεται από το script BearCollision, το BearMovement δέχεται την πληροφορία αυτή και στη συνέχεια τη μεταβιβάζει στον animator controller προκειμένου να εκτελεστεί το αντίστοιχο clip διακόπτοντας οποιοδήποτε άλλο animation βρίσκεται σε εξέλιξη εκείνη τη στιγμή. Εάν ο θάνατος της αρκούδας προκληθεί από μπάλα πάγου τότε μέσα από το script BearMovement παγώνει η ταχύτητα κίνησης του πράκτορα αλλά και η ταχύτητα κίνησης του κινουμένου σχεδίου ακριβώς στο σημείο πριν πεθάνει ώστε να προσομοιώνεται η ακαμψία των μυών λόγω ψύξης ενώ ταυτόχρονα ενεργοποιείται το αντικείμενο "ice" που βρίσκεται εντός της αρκούδας. Το αντικείμενο "ice" περιέχει έναν animator controller με ένα βασικό clip το οποίο δημιουργήσαμε εμείς και εκτελείται αυτόματα με την ενεργοποίηση του αντικειμένου. Το κινούμενο σχέδιο είναι η μεγέθυνση του αντικειμένου "ice" αναλογικά με το χρόνο μέχρι να φτάσει στο σημείο όπου θα περιβάλει ολόκληρη την αρκούδα. Σε άλλη περίπτωση όπου ο θάνατός της προκληθεί από οποιαδήποτε άλλη επίθεση, τότε με την ίδια σειρά όπως περιγράψαμε παραπάνω μηδενίζεται η ταχύτητα κίνησης του πράκτορα ενώ παράλληλα το animator controller διακόπτει οποιαδήποτε άλλο κινούμενο σχέδιο βρίσκεται σε εξέλιξη ώστε να εκτελεστεί το clip που προσομοιώνει την κίνηση θανάτου. Το αντικείμενο αρκούδα και στις δύο περιπτώσεις καταστρέφεται με το πέρας των πέντε δευτερολέπτων από το θάνατό της.

Η επιλογή του ηχητικού clip που θα αναπαραχθεί από το συστατικό audio source της αρκούδας πραγματοποιείται εξολοκλήρου στο script BearMovement. Στο script αυτό έχουμε δώσει σαν είσοδο πέντε διαφορετικά ηχητικά clips. Το ηχητικό clip που θα δοθεί στο συστατικό audio source για αναπαραγωγή, καθορίζεται από το εκάστοτε κινούμενο σχέδιο μέσα από το BearMovement. Χρησιμοποιούμε μια δική μας συνάρτηση, την playsound η οποία βρίσκεται μέσα στο script BearMovement και δέχεται σαν είσοδο τον αριθμό του clip που θέλουμε να αναπαραχθεί. Το κυρίως σώμα της συνάρτησης βρίσκεται στο ίδιο script μέσα στο οποίο πραγματοποιούνται και οι κατάλληλοι έλεγχοι αναπαραγωγής των ήχων. Η δυσκολία στη συγκεκριμένη περίπτωση ήταν ότι κάθε animation διαρκούσε πολλά frames. Σε κάθε frame γινόταν παράλληλα κλήση της συνάρτησης playsound με αποτέλεσμα να επαναλαμβάνεται ο σύντομος ήχος του ηχητικού clip που αντιστοιχούσε στο πρώτου frame, δημιουργώντας ένα περίεργο θόρυβο. Αυτό το εμπόδιο προσπεράστηκε χρησιμοποιώντας με ορθό τρόπο τη συνάρτηση audioSource.isPlaying με προσθήκη ορισμένων flags ώστε να αποτραπεί η αναπαραγωγή κάποιου ήχου εφόσον το clip που ακούγεται δεν έχει ολοκληρωθεί

```
void playSound(int clip) {
        if (!audioSource.isPlaying && clip != 2 && clip != 3 && clip!=4)
        {
            audioSource.clip = audioClip[clip];
            audioSource.Play();
        }
        else if ((clip == 2 || clip == 3 || clip==4) && hitsoundflag==0 ) {

            if (clip == 2 || clip == 3) {
                GameObject.Find("PlayerCapsule").SendMessage("ApplyDamage", 1.0F);
            }
            audioSource.clip = audioClip[clip];
            audioSource.Play();
            hitsoundflag = 1;
        }
    }
```

Εικόνα 97: Κώδικας Αναπαραγωγής Ήχου

6.5.2 Λύκος

Όπως με την αρκούδα έτσι και στον λύκο προσθέσαμε πέντε συστατικά τα οποία είναι NavMeshAgent, Capsule collider, wolfmovement script, BearCollision script, και audio source. Μπορούμε να πούμε ότι λειτουργούν με την ίδια φιλοσοφία όπως αυτή της αρκούδας με εξαίρεση κάποιων μικρών διαφορών. Το βασικό script του λύκου μέσα από το οποίο γίνεται διαχείριση και συνδυασμός των υπολοίπων συστατικών είναι το wolfmovement. Κατέχει τον ίδιο ρόλο στον λύκο με το αντίστοιχο bearmovement της αρκούδας. Ο λύκος πράκτορας κινείται και αυτός μεταξύ δύο σημείων στο χώρο που εναλλάσσονται ως στόχοι του πράκτορα, κάθε φορά που πλησιάζει κάποιον από τους δύο. Το κύριο χαρακτηριστικό του λύκου είναι ότι όταν ο χρήστης τον πλησιάσει για πρώτη φορά σε απόσταση μικρότερη των 25 μέτρων, δίνεται το γεγονός αυτό ως πληροφορία από το wolfmovement.sc στον animator controller μέσω της εντολής anim.SetFloat("Dist", dist);. Τότε ο animator controller προχωρεί στην εκτέλεση του clip “howl” ενώ μέσα από το script wolfmovement σταματά η κίνηση του λύκου ως πράκτορας, εκτελώντας τις απαιτούμενες εντολές (αντίστοιχες με αυτές της αρκούδας) και μέχρι να ολοκληρωθεί το animation “howl”. Παράλληλα, μέσω της συνάρτησης Physics.OverlapSphere(gameObject.transform.position, 40); δημιουργείται γύρω από τον λύκο μια νοητή σφαίρα με μέγεθος ακτίνας που καθορίζεται από τον προγραμματιστή ενώ ταυτόχρονα επιστρέφει σαν αποτέλεσμα έναν πίνακα με όλους τους colliders που άγγιξαν ή βρέθηκαν μέσα σε αυτήν. Στη συνέχεια, χρησιμοποιώντας τον επαναληπτικό βρόγχο “while” ελέγχουμε ποιοι από τους colliders (συγκρουστές) που βρίσκονται μέσα στον πίνακα, έχουν όνομα που αρχίζει από “STANDARD_W”. Έτσι εντοπίζονται όλοι οι λύκοι που βρίσκονται εντός του πεδίου καλέσματος, προσομοιώνοντας με αυτόν τον τρόπο το κάλεσμα της αγέλης.

```
3
4   else if (this.anim.GetCurrentAnimatorStateInfo(0).IsName("howl"))
5           {
6               playSound(0);
7               Collider[] hitColliders = Physics.OverlapSphere(gameObject.transform.position, 40);
8               int i = 0;
9               while (i < hitColliders.Length)
10              {
11                  if (hitColliders[i].name.Contains("STANDARD_W") && !hitColliders[i].name.Equals(this.name))
12                  {
13                      GameObject.Find(hitColliders[i].name).GetComponent<wolfMovement>().wolfAttack = 1;
14                  }
15
16                  i++;
17              }
18              this.agent.velocity = Vector3.zero;
19              this.agent.Stop();
20          }
21
```

Εικόνα 98: Κώδικας Πεδίου Καλέσματος Λύκου

Κάθε λύκος που εντοπίζεται ενημερώνει την μεταβλητή i στο script του wolfmovement ενώ με τη σειρά του το script αυτό μεταφέρει την πληροφορία στον animator controller και αλλάζει το στόχο του λύκου – πράκτορα ώστε αυτός να είναι πλέον ο χρήστης. Ο animator controller, αφού του δοθεί η κατάλληλη πληροφορία, μεταβαίνει για ένα μικρό χρονικό διάστημα στην κατάσταση idle και αμέσως μετά στην κατάσταση run. Δηλαδή, ο λύκος που έστειλε το κάλεσμα, περνά μέσω του animator, από την κατάσταση howl στην κατάσταση run. Σε κάθε frame ενημερώνεται η απόσταση λύκου-παίχτη μέσω του script wolfmovement. Εάν η απόσταση λύκου και παίχτη γίνει μικρότερη των 3 μέτρων, τότε ο animator controller μεταβαίνει στο clip stand bite, για να πραγματοποιήσει την κίνηση δαγκώματος ενώ μέσα από το script wolfmovement εκτελούνται οι διάφορες εντολές σε συνδυασμό με το συστατικό NavMeshAgent προκειμένου να κινείται ή να ακινητοποιείται ο λύκος ως πράκτορας.

Αφού ολοκληρωθεί η επίθεση, ο λύκος μεταβαίνει στην κατάσταση idle διατηρώντας τη θέση του ενώ στη συνέχεια γίνεται επανέλεγχος της απόστασης παίχτη λύκου ώστε να εκτελεστεί η ανάλογη κίνηση. Τα συστατικά wolfmovement, NavMeshAgent και hand controller συνεργάζονται πολύ στενά για τον καθορισμό της ταχύτητας κίνησης του λύκου μέσα στον εικονικό κόσμο ανάλογα με τις πληροφορίες που αυτός δέχεται από το περιβάλλον του.

Όσον αφορά τη διαχείριση των ηχητικών εφέ, είναι σχεδόν όμοια με αυτά της αρκούδας. Χρησιμοποιούμε την τοπική συνάρτηση playsound που δημιουργήσαμε μέσα στο script wolfmovement, καλώντας την κάθε φορά που αλλάζει το animation προκειμένου να αναπαραχθεί το κατάλληλο ηχητικό clip. Επίσης και οι υπόλοιπες λειτουργίες (πάγωμα, θάνατος) του λύκου είναι ακριβώς ίδιες με αυτές της αρκούδας με τη διαφορά ότι ο λύκος διαθέτει λιγότερη ενέργεια (70 HP ο λύκος, 120HP η αρκούδα).

6.5.3 Ελάφι

Το ελάφι, επειδή δεν αποτελεί εχθρό, παρουσιάζει μειωμένη προγραμματιστική πολυπλοκότητα. Για το λόγο αυτό χρειάστηκε να προσθέσουμε μόνο τρία συστατικά: το NavMeshAgent, τον Animator controller και το script stagmovement. Το stagmovement είναι το κεντρικό συστατικό μέσα από το οποίο γίνεται διαχείριση των υπολοίπων δύο, αναλόγως των συνθηκών που επικρατούν. Σε αυτό το script γίνεται έλεγχος σε κάθε frame εάν η απόσταση του ελαφιού από το πηγάδι είναι λιγότερη από 10 μέτρα. Εάν η απόσταση είναι

μεγαλύτερη από 10 μέτρα, δίνεται σαν είσοδος στον animator controller του η απόσταση μεταξύ του ελαφιού και του παίχτη. Εάν η απόσταση αυτή είναι μικρότερη των 5 μέτρων, ενημερώνεται το συστατικό NavMeshAgent από το stagmovement script ώστε το ελάφι να αναπτύξει κάποια ταχύτητα για να ακολουθήσει το χρήστη. Με τη σειρά του, ο animator controller εκτελεί το κατάλληλο animation clip ανάλογα με την απόσταση του χρήστη από το ελάφι. Εφόσον ο χρήστης απομακρυνθεί από το ελάφι πάνω από 20 μέτρα ενώ αυτό είχε ήδη αρχίσει να τον ακολουθεί, δίνεται η εντολή στον NavMeshAgent να μηδενίσει τη ταχύτητα του ελαφιού-πράκτορα ώστε να σταματήσει, ενώ ο animator controller μεταβαίνει από την κατάσταση walk στην κατάσταση idle. Το ίδιο θα συμβεί αν φτάσει το ελάφι σε απόσταση 10 μέτρων από το πηγάδι με μ διαφορά ότι αυτό δε θα ακολουθήσει ξανά το χρήστη.

```
2
3   if (Vector3.Distance(GameObject.Find("Well").transform.position, this.transform.position) > 10)
4           {
5               float dist = Vector3.Distance(GameObject.Find("PlayerCapsule").transform.position, this.transform.position);
6               anim.SetFloat("Dist", dist);
7               if (dist <= 5.0f || this.anim.GetCurrentAnimatorStateInfo(0).IsName("walk"))
8               {
9                   agent.speed = 5.0f;
10                  agent.destination = GameObject.Find("PlayerCapsule").transform.position;
11                  this.agent.Resume();
12                  if (dist <= 4)
13                  {
14                      this.agent.velocity = Vector3.zero;
15                      this.agent.Stop();
16                  }
17
18              }
19              else if (dist >= 20.0f)
20              {
21                  this.agent.velocity = Vector3.zero;
22                  this.agent.Stop();
23              }
24          }
25
```

Εικόνα 99: Κώδικας Συμπεριφοράς Ελαφιού

6.5.4 Σκελετός

Αν και η γενική φιλοσοφία της συμπεριφοράς του σκελετού είναι παρόμοια με αυτή της αρκούδας και του λύκου, υπάρχουν βασικές διαφορές στις οποίες αντικατοπτρίζεται και ο ξεχωριστός τρόπος δημιουργίας του. Χρειάστηκε να προσθέσουμε σε αυτόν πέντε συστατικά τα οποία είναι ο Animator Controller, το NavMeshAgent, το undeadMovement script και Audio Source. Κεντρικό συστατικό διαχείριση συμπεριφοράς αποτελεί το undeadMovement script.

Ο σκελετός βρίσκεται ακίνητος σε μια αρχική θέση στη πίστα εκτελώντας μόνο το κινούμενο σχέδιο idle. Το undeadMovement script ενημερώνεται σε κάθε frame σχετικά με την απόσταση του σκελετού από το χρήστη. Εάν ο χρήστης πλησιάσει τον σκελετό σε λιγότερο από 15 μέτρα, τότε το script undeadMovement δίνει εντολή στο συστατικό NavMeshAgent του σκελετού ώστε να αναπτύξει ταχύτητα και να αρχίσει να ακολουθεί τον παίχτη ασταμάτητα. Ο animator controller, αφού δεχτεί σαν είσοδο την απόσταση παίχτη-σκελετού, μεταβαίνει από την κατάσταση idle στην κατάσταση run. Έτσι ο σκελετός κινείται προς τον χρήστη, πραγματοποιώντας παράλληλα το κινούμενο σχέδιο run προσομοιώνοντας την κίνηση του τρεξίματος. Εάν η απόσταση μεταξύ παίχτη και σκελετού είναι μικρότερη των 4 μέτρων, τότε ο animator controller μεταβαίνει στο κινούμενο σχέδιο attack. Το script

undeadMovement με τη σειρά του μηδενίζει την ταχύτητα κίνησης του σκελετού ως πράκτορα μέσω του συστατικού NavMeshAgent. Αφού ολοκληρωθεί το κινούμενο σχέδιο επίθεσης, ο animator μεταβαίνει στην κατάσταση waitingforbattle, η οποία είναι όμοια με το animation clip idle και ο σκελετός παραμένει εκεί για μικρό χρονικό διάστημα. Εάν η απόσταση εξακολουθεί να είναι μικρότερη των 4 μέτρων, επαναλαμβάνεται η κίνηση της επίθεσης του σκελετού, διαφορετικά αποκτά σαν πράκτορας ταχύτητα κίνησης και ακολουθεί τον χρήστη καθώς εκτελεί το κινούμενο σχέδιο run.

Στην περίπτωση όπου ο σκελετός χτυπηθεί από οποιαδήποτε επίθεση του χρήστη, προς στιγμήν πέφτει στο έδαφος για μερικά δευτερόλεπτα ανεξαρτήτως της αρχικής του κατάστασης. Δηλαδή, ο animator controller ενημερώνεται μέσω του undeadMovement script που περιέχει τη συνάρτηση εντοπισμού συγκρούσεων και η οποία δέχεται την πληροφορία από το συστατικό capsule collider. Ο animator controller στη συνέχεια μεταβαίνει από την κατάσταση που βρίσκεται αυτόματα στην κατάσταση die προσομοιώνοντας τον θάνατο του. Το script undeadMovement αντιλαμβάνεται το κινούμενο σχέδιο και ενημερώνει το συστατικό NavMeshAgent μηδενίζοντας την ταχύτητα προκειμένου να ακινητοποιηθεί ο σκελετός. Έπειτα από μικρό χρονικό διάστημα, ο σκελετός ζωντανεύει, ακολουθώντας την ίδια συμπεριφορά.

Όσον αφορά την ηχητική επένδυση, η επιλογή γίνεται στο undeadMovement script μεταξύ πέντε διαφορετικών προτάσεων σύμφωνα με το κινούμενο σχέδιο που εκτελείται. Και σε αυτή την περίπτωση χρησιμοποιείται η συνάρτηση audioSource.isPlaying προκειμένου να εκτελεστεί το κατάλληλο clip ήχου αλλά και να αποτρέψει την επαναλαμβανόμενη αναπαραγωγή του ήχου σε κάθε frame. Το σημείο το οποίο χρειάστηκε ιδιαίτερη διαχείριση ήταν η περίπτωση του σύντομου θανάτου του σκελετού. Ενώ ο σκελετός πέφτει ή βρίσκεται στο έδαφος εκτελώντας το αντίστοιχο κινούμενο σχέδιο, ο capsule collider του παρέμενε όρθιος εξακολουθώντας να εντοπίζει τις διάφορες συγκρούσεις, παρεμποδίζοντας έτσι, σαν ασπίδα, μια πιθανή επίθεση του παίχτη προς εκείνη την κατεύθυνση. Αυτό το πρόβλημα λύθηκε απενεργοποιώντας το συστατικό capsule collider του σκελετού όσο χρόνο αυτός κείτονταν στο έδαφος μέσω της εντολής `gameobject.getcomponent<capsule>().enable=false;`

6.5.5 Τελικός Αρχηγός, Δαίμονας

Στον τελικό αρχηγό έχουμε προσθέσει πέντε επιπλέον συστατικά, ένα NavMeshAgent, ένα capsule collider, ένα audio source, ένα animator controller και ιδιαίτερα το κεντρικό script demonMovement που συντελεί στο συγχρονισμό των ήχων με τα κινούμενα σχέδια και την κίνηση στο χώρο. Μόλις εμφανιστεί ο αρχηγός στην πίστα, ο animator controller εκτελεί το κινούμενο σχέδιο roar ενώ παράλληλα το demonMovement script μηδενίζει την ταχύτητα κίνησης του αρχηγού μέσω του συστατικού NavMeshAgent. Ταυτόχρονα, μέσω της συνάρτησης playsound που βρίσκεται στο demonMovement script, δίνεται η εντολή να παραχθεί ο κατάλληλος ήχος. Σε κάθε frame, στην αρχή της συνάρτησης update του script demonMovement υπολογίζεται η απόσταση μεταξύ παίχτη και τελικού αρχηγού με την παρακάτω εντολή

```
dist = Vector3.Distance(GameObject.Find("PlayerCapsule").transform.position, gameObject.transform.position);
```

Αφού ολοκληρωθεί το animation roar, ο animator controller μεταβαίνει αυτόματα στο επόμενο clip το οποίο είναι το walk. Έτσι, ο αρχηγός εκτελεί το κινούμενο σχέδιο walk, το οποίο εντοπίζεται από το κεντρικό script, παρέχοντας στον τελικό αρχηγό ταχύτητα κίνησης με κατεύθυνση προς το στόχο μέσω του NavMeshAgent. Εάν η απόσταση μεταξύ παίχτη και

αρχηγού γίνει μικρότερη από 6 μέτρα τότε δίνεται από το κεντρικό script στον animator controller η κατάλληλη είσοδος ώστε αυτός να μεταβεί στο clip attack. Στη συνέχεια, μέσω του κεντρικού script μηδενίζεται η ταχύτητα κίνησης του συστατικού NavMeshAgent με αποτέλεσμα ο αρχηγός να παραμένει στη θέση του χωρίς να κατευθύνεται πλέον προς το στόχο εκτελώντας το κινούμενο σχέδιο attack2. Αφού ολοκληρωθεί το κινούμενο σχέδιο, ο animator controller προχωρεί στο clip idle παραμένοντας εκεί για μικρό χρονικό διάστημα χωρίς να κινείται ο αρχηγός προς τον παίχτη, δίνοντας τη δυνατότητα στο χρήστη να αντιδράσει. Με την ολοκλήρωση του κινούμενου σχεδίου idle γίνεται επανέλεγχος της απόστασης αρχηγού-παίχτη από το κεντρικό script. Αν η απόσταση είναι μεγαλύτερη των έξι μέτρων, επανέρχεται η ταχύτητα κίνησης του αρχηγού ως αυτόνομου πράκτορα, εκτελώντας το animation walk. Σε διαφορετική περίπτωση, εξακολουθεί το κεντρικό script να μηδενίζει την ταχύτητα κίνησης του συστατικού NavMeshAgent παραμένοντας ακίνητος ο αρχηγός, εκτελώντας το κινούμενο σχέδιο attack2, εντολή που δίνει ο animator αφού ενημερωθεί για την απόσταση. Ο animator controller ενημερώνεται σχετικά με την απόσταση αρχηγού-παίχτη μέσω των εντολών που φαίνονται στην παρακάτω

```
2
3   if (dist <= 6)
4               {
5                   anim.SetInteger("dist", 1);
6                   this.agent.velocity = Vector3.zero;
7                   this.agent.Stop();
8               }
9   else if (dist > 6)
10              {
11                  anim.SetInteger("dist", 0);
12              }
13
```

Εικόνα 100: Κώδικας Ενημέρωσης Animation Controller σύμφωνα με συνθήκη if

Όσον αφορά τον εντοπισμό συγκρούσεων, πρωταρχικό ρόλο κατέχει το συστατικό capsule collider, ενώ η συνάρτηση OncollisionEnter, η οποία βρίσκεται μέσα στο κεντρικό script, δίνει το όνομα του αντικειμένου που χτύπησε τον αρχηγό. Αν ο αρχηγός χτυπηθεί δύο φορές από πράσινη μπάλα, τότε ο animator controller δέχεται την κατάλληλη είσοδο και αυτόματα, σε οποιαδήποτε κατάσταση και αν βρίσκεται μεταβαίνει στο clip death. Με τη σειρά του το κεντρικό script εντοπίζει ότι εκτελείται το συγκεκριμένο animation και μηδενίζει αμέσως την ταχύτητα κίνησης του δαίμονα με τις κατάλληλες εντολές μέσω του συστατικού NavMeshAgent.

Ανάλογα με το κινούμενο σχέδιο που πραγματοποιεί ο αρχηγός, επιλέγεται και το κατάλληλο ηχητικό clip από το κεντρικό script στο οποίο δόθηκε ένα σύνολο διαφορετικών ήχων χρησιμοποιώντας την τοπική συνάρτηση playsound() όπως ακριβώς και στις προηγούμενες περιπτώσεις.

Όλοι οι παραπάνω αυτόνομοι πράκτορες-εχθροί έχουν ένα κοινό χαρακτηριστικό. Κάθε φορά που εκτελούν το δικός τους κινούμενο σχέδιο επίθεσης, καλούν τη συνάρτηση ApplyDamage που βρίσκεται στο script GamePlay το οποίο με τη σειρά του είναι προσκολλημένο στο αντικείμενο PlayerCapsule που αντιπροσωπεύει το χρήστη με

αποτέλεσμα τη μείωση των πόντων ζωής του. Η κλήση της συνάρτησης γίνεται με την εντολή SendMessage ακολουθούμενη από το όνομα της συνάρτησης και το μήνυμα που θέλουμε να στείλουμε. Η σύνταξη της εντολής παρουσιάζεται στο παρακάτω παράδειγμα.

```
GameObject.Find("PlayerCapsule").SendMessage("ApplyDamage",7.0F);
```

Με την εντολή Find βρίσκουμε το αντικείμενο που περιέχει script με τη συνάρτηση στην οποία θέλουμε να στείλουμε μήνυμα. Η ApplyDamage συμβολίζει το όνομα της εκάστοτε συνάρτησης που θέλουμε να στείλουμε το μήνυμα ενώ το 7.0f αποτελεί το μήνυμα.

6.6 Διαχείριση Ενεργειών Μέσω Script

Για την ομαλή διεξαγωγή του παιχνιδιού και για την επίτευξη της διαδραστικότητας μεταξύ χρήστη και εικονικού περιβάλλοντος, δημιουργήσαμε ένα κεντρικό script με ονομασία GamePlay το οποίο είναι προσκολλημένο σαν συστατικό στο αντικείμενο PlayerCapsule που χειρίζεται ο παίχτης.

Το script αυτό είναι υπεύθυνο για τη σωστή ροή των αποστολών, για τους πόντους ζωής, το mana του παίχτη και για άλλες δευτερεύουσες λειτουργίες που αφορούν τις γραφικές διεπαφές. Βασικό χαρακτηριστικό του είναι η αφαίρεση πόντων ζωής του χρήστη από οποιοδήποτε εχθρό που εκτελεί κινούμενο σχέδιο επίθεσης. Όταν κάποιος εχθρός εκτελεί κινούμενο σχέδιο επίθεσης, τότε το κεντρικό του script αποστέλλει, το κατάλληλο μήνυμα προκειμένου να αφαιρεθούν οι αντίστοιχοι πόντοι ζωής του χρήστη, ανάλογα πάντα τον κάθε αντίπαλο. Ακόμα, όταν εκτελείται από το χρήστη επίθεση μπάλας φωτιάς ή μπάλας πάγου, τότε τα script που είναι υπεύθυνα γι' αυτές τις δύο κινήσεις δηλαδή, το HandVelocity.cs και το IceAttack.cs αντίστοιχα αποστέλλουν και αυτά με τη σειρά τους μήνυμα στο κεντρικό script του παίχτη προκειμένου να ενημερώσουν για την απώλεια mana.

Επίσης, το script GamePlay είναι υπεύθυνο για την ομαλή διεξαγωγή κάθε αποστολής. Συγκεκριμένα, όσον αφορά την πρώτη αποστολή, υπολογίζει πόσες αρκούδες έχουν εξολοθρευτεί. Κάθε αρκούδα που πεθαίνει στέλνει μήνυμα μέσω της εντολής SendMessage στο script GamePlay για τον τελικό απολογισμό. Αν ανιχνευτούν 4 νεκρές αρκούδες τότε ενεργοποιείται το πρώτο σε ιεραρχία αντικείμενο (πατέρας) της δεύτερης αποστολής μέσα στο οποίο εμπεριέχονται όλα τα gameObject (parenting) που σχετίζονται με αυτήν. Αφού ολοκληρωθεί η δεύτερη αποστολή, με τα ίδια βήματα εκτελούνται και ολοκληρώνονται και οι επόμενες. Ταυτόχρονα, καταστρέφονται όλα τα άχρηστα αντικείμενα της προγενέστερης αποστολής ώστε να εξοικονομήσουμε υπολογιστικούς πόρους.

Η ενεργοποίηση των αντικείμενων "πατέρων" γίνεται με την εντολή:
```
GameObject.Find("WolfArea1").transform.GetChild(0).gameObject.SetActive(true);
```
Απαραίτητη προϋπόθεση για τη λειτουργία της παραπάνω εντολής είναι να βρίσκονται ανενεργά στη σκηνή τα προς ενεργοποίηση αντικείμενα ιεραρχικά κάτω από ένα άλλο ενεργό αντικείμενο προκειμένου να είναι δυνατή η ανίχνευση τους. Το πεδίο `Find("WolfArea1")` περιέχει το όνομα του ενεργού αντικειμένου μέσω του οποίου θα ανιχνευθεί και θα ενεργοποιηθεί το αντικείμενο που επιθυμούμε. Στο πεδίο `GetChild(0).gameObject.SetActive(true)` ανιχνεύεται και ενεργοποιείται αυτό το αντικείμενο.

Ακολουθήσαμε την ανωτέρω διαδικασία γιατί είναι ο μοναδικός τρόπος εντοπισμού μη ενεργών αντικειμένων. Αυτή η τεχνική καταστροφής άχρηστων αντικειμένων και η παράλληλη ενεργοποίηση των απολύτως αναγκαίων, εφαρμόστηκε προκειμένου να έχουμε

κάθε στιγμή στη σκηνή το μικρότερο αριθμό απαραίτητων αντικειμένων και κατ' επέκταση την ελάχιστη κατανάλωση και δέσμευση υπολογιστικών πόρων.

6.7 Γραφική Διεπαφή Χρήστη

Για την υλοποίηση γραφικής διεπαφής χρησιμοποιήθηκε το σύστημα UI (User Interface) της Unity3D το οποίο μας επιτρέπει να δημιουργήσουμε γρήγορα και εύκολα καλαίσθητες διεπαφές χρήστη. Τα στοιχεία διεπαφής χρήστη που χρησιμοποιήσαμε είναι τα εξής:

Canvas: είναι η περιοχή μέσα στην οποία πρέπει να βρίσκονται όλα τα στοιχεία του U.I. Ο καμβάς είναι ένα αντικείμενο παιχνιδιού με ένα συστατικό canvas όπου όλα τα U.I. στοιχεία πρέπει να είναι παιδιά του. Κατάλληλο για επεξεργασία UI στοιχείων αφού στο scene view φαίνεται σαν τετράγωνο και έτσι δε χρειάζεται να χρησιμοποιεί κάποιος να χρησιμοποιεί το game.

Panel: είναι μια γραφική διεπαφή που τοποθετείται ιεραρχικά κάτω από τον καμβά και χρησιμοποιείται ως εργαλείο ομαδοποίησης άλλων γραφικών διεπαφών. Στην παρούσα εργασία, έχει προστεθεί στο panel ένας animator controller που χρησιμοποιεί ένα κινούμενο σχέδιο το οποίο δημιουργήσαμε εμείς, έτσι ώστε η εκάστοτε γραφική διεπαφή να παρουσιάζεται ομαλά στην οθόνη όταν έρχεται ώρα της εμφάνισής της.

Button: είναι ένα κουμπί που υπό κανονικές συνθήκες ανταποκρίνεται στο πάτημα του χρήστη μέσω πληκτρολογίου ή ποντικιού ώστε να εκτελεί τις εντολές που του έχουν ανατεθεί. Στη συγκεκριμένη εργασία επειδή δεν χρησιμοποιείται πληκτρολόγιο ή ποντίκι, το προγραμματίσαμε εμείς έτσι ώστε να ανταποκρίνεται στο άγγιγμα του δείκτη του δεξιού χεριού. Επιλέξαμε την αναγνώριση μόνο ενός εκ των δύο χεριών γιατί αντιμετωπίσαμε δυσκολίες σχετικά με την λειτουργία ανίχνευσης σύγκρουσης του κουμπιού με τον δείκτη. Προσθέσαμε δηλαδή σ' αυτό ένα συστατικό ανίχνευσής συγκρούσεων Box collider. Στη συνέχεια δημιουργήσαμε ένα script με ονομασία ButtonScript μέσα από το οποίο αντλούμε την πληροφορία σχετικά με το όνομα του αντικείμενου που συγκρούεται με το κουμπί. Εάν το όνομα αυτό συμπίπτει με το "Bone 3" δηλαδή ένα συγκεκριμένο οστό του δείκτη του δεξιού χεριού, τότε ενεργοποιείται το Button. Επίσης, στο κουμπί έχουμε προσθέσει το συστατικό animator controller προσδίδοντας του έτσι κινούμενα σχέδια τα οποία δημιουργήσαμε εμείς. Όταν ανιχνευτεί η σύγκρουση με το συγκεκριμένο οστό, τότε η πληροφορία αυτή μεταφέρεται από το script στον animator controller ώστε να εκτελεστεί το κινούμενο σχέδιο υποδηλώνοντας έτσι στο χρήστη ότι το κουμπί έχει πατηθεί. Όταν αυτό πατιέται, αυξάνεται το μέγεθος του και διαφοροποιείται το χρώμα του. Για να διεξαχθεί η εντολή που του έχει ανατεθεί πρέπει να είναι πατημένο για 1,5 δευτερόλεπτα.

Slider: επιτρέπει στο χρήστη να επιλέξει μια αριθμητική τιμή μέσα από ένα προκαθορισμένο εύρος, σέρνοντας γενικά για το σκοπό αυτό το ποντίκι. Στη δική μας περίπτωση χρησιμοποιήσαμε αυτό το στοιχείο ώστε να εμφανίζονται στην οθόνη η ενέργεια και το mana του παίχτη, χωρίς βέβαια τη χρήση ποντικιού. Για την επίτευξη αυτού του στόχου δώσαμε δύο sliders ως εισόδους στο script GamePlay. Οι δύο sliders είναι ισομεγέθεις αλλά διαφοροποιημένοι χρωματικά, ο ένας εκ των δύο είναι πράσινος με κόκκινο φόντο και αντιστοιχεί στην ενέργεια του παίχτη, ενώ ο άλλος είναι γαλάζιος με μωβ φόντο και αναφέρεται στην ποσότητα mana του παίχτη. Από τους δύο sliders αφαιρέσαμε το αντικείμενο εκείνο που αλληλεπιδρά με το ποντίκι. Μέσα από το script GamePlay ρυθμίζονται οι τιμές των sliders που απεικονίζονται στην πάνω αριστερή γωνία της οθόνης. Οι εντολές με τις οποίες διαχειριζόμαστε τους sliders μέσα στο script είναι οι εξής:

```
1   healthbar.value=playerHP;
2   manabar.value=playerMp;
```

Εικόνα 101: Εντολές Ενημέρωσης των Slider

Οι μεταβλητές healthbar και manabar είναι κλάσης τύπου slider στις οποίες δόθηκαν σαν είσοδο οι δύο sliders που περιγράψαμε. Με τη συνάρτηση .value αναφερόμαστε στη τιμή του εκάστοτε slider η οποία ενημερώνεται ανάλογα με την αλλαγή της τιμής των μεταβλητών PlayerHp και PlayerMp.

Κείμενο και εικόνα, εμφάνιση κάποιου κειμένου ή εικόνας στην οθόνη: Όπως έχει ήδη αναφερθεί και σε προηγούμενο κεφάλαιο, στο παιχνίδι αυτό έχουν δημιουργηθεί τρία διαφορετικά menu, το αρχικό μενού, το μενού παύσης και το μενού θανάτου. Κοινά στοιχεία και των τριών είναι η χρήση κουμπιού με τον ίδιο μηχανισμό και με τα ίδια συστατικά σε κάθε μενού αλλά με διαφορετικές ονομασίες. Δηλαδή, το αντικείμενο κουμπί χρησιμοποιείται με διαφορετικά ονόματα. Αυτό επιτυγχάνεται μέσω του κοινού script ButtonScript, που ανάλογα με την ονομασία του κουμπιού στο οποίο είναι επικολλημένο εκτελεί και την αντίστοιχη ενέργεια. Οι ενέργειες που μπορεί να εκτελέσει κάποιο κουμπί αναλόγως της ονομασίας του είναι είτε αλλαγή σκηνής είτε εμφάνιση υπομενού.

6.8 Αλλαγή Σκηνής και Υπομενού

6.8.1 Αλλαγή Σκηνής

Αλλαγή σκηνή είναι η διαδικασία κατά την οποία είναι εφικτή η μετάβαση από την ενεργή σκηνή που βρίσκεται ο χρήστης σε κάποια άλλη που επιθυμεί μέσω της εντολής. SceneManager.LoadScene(“”);. Απαραίτητη προϋπόθεση για την ομαλή εκτέλεση της εντολής είναι η φόρτωση της σκηνής που θέλουμε να μεταβούμε στα Build Settings της Unity. Για να φορτώσουμε μια σκηνή πρέπει να ακολουθηθεί η εξής διαδικασία: μέσω του File→Build settings θα ανοίξει το παράθυρο “Build settings”. Μέσα σε αυτό υπάρχει ένα πεδίο με όνομα “Scenes In Build στο οποίο θα πρέπει να σύρουμε την σκηνή την οποία θέλουμε να φορτώσουμε μέσω της εντολής που περιγράψαμε ανωτέρω.

6.8.2 Εμφάνιση Υπομενού

Για την εμφάνιση του υπομενού, κατόπιν πιέσεως του απαραίτητου κουμπιού, ακολουθείται η παρακάτω διαδικασία η οποία λαμβάνει χώρα στο ButtonScript που είναι προσκολλημένο σε κάθε κουμπί. Απενεργοποίηση του υπάρχοντος υπομενού και παράλληλη ενεργοποίηση εκείνου που θέλουμε να μεταβούμε. Η απενεργοποίηση ενός υπομενού είναι σχετικά εύκολη διαδικασία, καθώς εκείνο που πρέπει να γνωρίζουμε είναι το όνομα του Panel που αντιστοιχεί σε αυτό μέσω της εντολής

```
GameObject.Find("Canvas/menus").SetActive(false);
```

Αντίθετα, η διαδίκασία για την ενεργοποίηση ενός υπομενού είναι πολυπλοκότερη και εξελίσσεται όπως περιγράφεται παρακάτω.

Το ανενεργό αντικείμενο-υπομενού που θέλουμε να ενεργοποιήσουμε θα πρέπει να είναι ιεραρχικά "παιδί" ενός ενεργού αντικειμένου "πατέρα". Επίσης, θα πρέπει να γνωρίζουμε τη σειρά κατάταξής του σε σχέση με τα άλλα "παιδιά" αλλά και το όνομα του αντικειμένου "πατέρα". Πρέπει να σημειωθεί ότι με τον όρο υπομενού αναφερόμαστε σε ένα panel που περιέχει ένα σύνολο γραφικών διεπαφών. Η εντολή για ενεργοποίηση ενός αδρανούς υπομενού είναι η ακόλουθη

```
GameObject.Find("canvas").transform.GetChild(1).gameObject.SetActive(true);
```

Για τη μετάβαση σε κάποια από τις αποστολές μέσω των τριών μενού, δημιουργήσαμε τέσσερις διαφορετικές σκηνές. Κάθε μια από αυτές έχει ως αφετηρία μια από τις τέσσερις αποστολές. Έτσι, αν κάποιος χρήστης βρίσκεται στο αρχικό μενού και θέλει να μεταπηδήσει στη δεύτερη αποστολή, τότε μέσω της εντολής που περιγράψαμε στην αλλαγή σκηνής, φορτώνεται η αντίστοιχη σκηνή που έχει σαν αφετηρία τη δεύτερη αποστολή ενώ στη συνέχεια η ροή του παιχνιδιού εξελίσσεται κανονικά.

Επίσης, με την ολοκλήρωση μια αποστολής, η μεταβλητή "Quest" που βρίσκεται στο script GamePlay ενημερώνεται με τον αριθμό της αποστολής που ακολουθεί. Όταν βρισκόμαστε στο μενού παύσης ή στο μενού θανάτου και θέλουμε να ξεκινήσουμε από την αρχή την αποστολή που βρισκόμαστε, πατάμε το κουμπί "RESTART". Έτσι, το ButtonScript αντλεί την πληροφορία του σειριακού αριθμού της αποστολής που βρίσκεται σε εξέλιξη μέσω του script GamePlay ο οποίος αντιστοιχεί στη σκηνή με αφετηρία αυτή την αποστολή. Έπειτα, γίνεται φόρτωση της σκηνής με τη διαδικασία που ήδη έχουμε περιγράψει. Ακόμα, εάν θελήσουμε να επιστρέψουμε στο αρχικό μενού ενώ βρισκόμαστε στο μενού παύσης ή θανάτου, θα πρέπει να πατήσουμε το κουμπί "MAIN MENU" το οποίο δίνει εντολή στο ButtonScript να προχωρήσει στη φόρτωση της αντίστοιχης σκηνής.

6.9 Μενού Παύσης

Όταν καλούμε το μενού παύσης θεωρητικά θα πρέπει να παγώνει ο χρόνος για ολόκληρη την εφαρμογή εκτός από τα χέρια του χρήστη και το μενού. Αυτό δεν είναι εφικτό καθώς δεν παρέχεται η δυνατότητα από τη Unity3D να σταματάει ο χρόνος για μεμονωμένα αντικείμενα.

Όταν αναφερόμαστε στο χρόνο εννοούμε τα frame/second. Δηλαδή, όταν λέμε ότι παγώνουμε το χρόνο σημαίνει ότι σταματάμε τη ροή των frames στην εφαρμογή. Έτσι δεν μπορούν μεμονωμένα αντικείμενα να ενεργούν ανεπηρέαστα από την παύση του χρόνου. Όσον αφορά τη δική μας περίπτωση, εάν άνοιγε το μενού παύσης ενώ παράλληλα η εφαρμογή πάγωνε, δε θα ήταν δυνατή η επιλογή των διαφόρων κουμπιών καθώς λειτουργούν με βάση τη ροή των frames την οποία είχαμε ανακόψει, γεγονός που θα επέφερε σοβαρό πρόβλημα καθώς το παιχνίδι θα παρέμενε στάσιμο για πάντα.

Για τη λύση του προβλήματος ακολουθήσαμε την παρακάτω διαδικασία. Σε κάθε ένα από τα βασικά αντικείμενα του παιχνιδιού (εχθροί, μπάλες επίθεσης και παίχτης) προσθέσαμε μια λειτουργία παγώματος. Η λειτουργία αυτή είναι διαφορετική στους εχθρούς και διαφορετική στις μπάλες επίθεσης. Με την ενεργοποίηση του μενού παύσης, ενημερώνεται η Public μεταβλητή Pause από 0 σε 1 η οποία βρίσκεται στο script iceattack που είναι προσκολλημένο στο αντικείμενο PlayerCapsule του παίχτη. Όταν δηλώνουμε ένα στοιχείο ως Public αποκτά την ιδιότητα να είναι ορατό και προσπελάσιμο από οποιοδήποτε

σημείο της σκηνής. Κάθε εχθρός και κάθε χαρακτήρας μέσω του κεντρικού του script ενημερώνεται σχετικά με τη μεταβλητή Pause σε κάθε frame. Εάν η μεταβλητή είναι 0, τότε αποθηκεύεται η ταχύτητα κίνησης του χαρακτήρα και η ταχύτητα κίνησης του κινούμενου σχεδίου του σε δύο προσωρινές μεταβλητές. Αν στο επόμενο frame η τιμή του Pause γίνει από 0 σε 1 τότε η ταχύτητα κίνησης του χαρακτήρα στο χώρο και η ταχύτητα κίνησης του κινουμένου σχεδίου μηδενίζονται και διατηρούν τη τιμή 0 για όσο η μεταβλητή Pause είναι ίση με 1. Με αυτό τον τρόπο επιτυγχάνεται το πάγωμα του χαρακτήρα σε συγκεκριμένη στάση.

Για να αλλάξει η μεταβλητή Pause από 1 σε 0 θα πρέπει ο χρήστης να πατήσει το κουμπί Resume. Πατώντας το κουμπί αυτό μέσω του ButtonScript στέλνουμε μήνυμα στη συνάρτηση UnPause που βρίσκεται στο script iceattack δια της εντολής SendMessage που έχει περιγραφεί, επαναφέροντας έτσι την ταχύτητα κίνησης του χρήστη και την ταχύτητα κίνησης του κινουμένου σχεδίου προ παύσης. Η ροή και η λειτουργικότητα του χαρακτήρα συνεχίζεται ομαλά από εκείνο το σημείο και μετά.

Όσον αφορά τις κινήσεις και τις επιθέσεις που μπορεί να κάνει ο χρήστης με τα χέρια, αυτές αδρανοποιούνται όσο η μεταβλητή pause παραμένει 1. Επομένως ο χρήστης δεν μπορεί να κινηθεί στο χώρο αλλά ούτε να εκτελέσει επίθεση όσο το μενού παύσης είναι ενεργό με εξαίρεση τις κινήσεις κεφαλής γύρω από τον εαυτό του και τις τυχαίες κινήσεις χεριών. Αυτό πραγματοποιείται ελέγχοντας από κάθε script που σχετίζεται με τις παραπάνω κινήσεις εάν η μεταβλητή Pause είναι 1. Η πληροφορία αυτή αντλείται μέσω της εντολής `Gameobject.Find("PlayerCapsoule").GetComponent<IceAttack>().pause;` και χρησιμοποιείται σε συνθήκες ελέγχου οι οποίες έχουν εμφωλευμένες διάφορες εντολές κίνησης και επίθεσης του παίχτη.

Επίσης, εάν μια μπάλα επίθεσης κατευθύνεται προς το στόχο και πριν τη σύγκρουση ενεργοποιηθεί το μενού παύσης, τότε η μπάλα παραμένει ακίνητη στον αέρα. Αυτό επιτυγχάνεται μέσω του κεντρικού script κάθε μπάλας επίθεσης που είναι υπεύθυνο για την κίνηση της προς το στόχο. Στην αρχή του script χρησιμοποιούμε μια συνθήκη ελέγχου if αντλώντας και ελέγχοντας με παρόμοια εντολή όπως την παραπάνω, αν η μεταβλητή pause είναι 0 σε κάθε frame. Η κίνηση της μπάλας πραγματοποιείται με την εντολή `vector3.Movetowards(transform.position,targetdDirection,step);` όπου step ο ρυθμός κίνησης της μπάλας προς το στόχο με step = 10 * Time.deltaTime;. Αν η μεταβλητή pause είναι 0, η μεταβλητή step ενημερώνεται ώστε να προσδίδει στη μπάλα σταθερή ομαλή ταχύτητα προς το στόχο μέσω της εντολής που αναφέραμε. Στην περίπτωση κατά την οποία η μεταβλητή pause γίνει 1, η μεταβλητή step μηδενίζεται και με αυτό το τρόπο η μπάλα έχει μηδενικό ρυθμό κίνησης προς το στόχο και επομένως παραμένει ακίνητη στον αέρα για όσο διάστημα είναι ενεργό το μενού παύσης.

Η μπάλα επίθεσης δεν καταστρέφεται λόγω λήξης του χρόνου διάρκειας ζωής της όσο το παιχνίδι βρίσκεται σε κατάσταση παύσης, αλλά παραμένει ακίνητη στη θέση της μέχρι αυτό να ξεπαγώσει. Όταν η εφαρμογή ξεπαγώσει, τότε η μεταβλητή step ενημερώνεται με τον αρχικό ρυθμό της και δίνεται σαν είσοδος ως ρυθμός κίνησης στην μπάλα. Η μπάλα επίθεσης συνεχίζει κανονικά την πορεία της μέχρι να φτάσει τον στόχο της.

Κεφάλαιο 7^{ο} Αποτελέσματα, Εκτίμηση και Μελλοντικές Επεκτάσεις

7.1 Εισαγωγή

Στο κεφάλαιο αυτό γίνεται αναφορά των στόχων που επετεύχθησαν αλλά καταγράφονται και οι αρνητικές επιπτώσεις της εφαρμογής. Όλο αυτό πραγματοποιήθηκε με τη βοήθεια δεκαπέντε εθελοντών που δοκίμασαν την εφαρμογή και συμπλήρωσαν το ερωτηματολόγιο το οποίο τους δόθηκε από εμάς. Τέλος, αναφέρουμε κάποιες ιδέες για μελλοντικές επεκτάσεις της εφαρμογής.

7.2 Αποτελέσματα-Εκτίμηση

Στόχος όπως ήδη αναφέραμε ήταν η δημιουργία ενός παιχνιδιού εικονικής πραγματικότητας συνδυάζοντας το Oculus Rift και το Leap Motion ώστε ο χρήστης να εμβυθιστεί με επιτυχία στο εικονικό περιβάλλον.

Ο στόχος αυτός σε μεγάλος βαθμό επιτεύχθηκε αποδίδοντας τη ζητούμενη ρεαλιστικότατα και εμβύθιση. Αυτό μπορεί να θεωρηθεί και η κυρίαρχη θετική πλευρά του όλου εγχειρήματος, δηλαδή η απόλυτη εμβύθιση του χρήστη στο εικονικό περιβάλλον με παράλληλη απόλαυση και ψυχαγωγία αλλά και πρόκληση για την ολοκλήρωση του παιχνιδιού.

Η εφαρμογή εμπεριέχει ένα ποικιλόμορφο και καλαίσθητο terrain με αρμονικές εναλλαγές τοπίων και κατάλληλη μουσική επένδυση. Οι κινήσεις του χρήστη είναι ευέλικτες και η διάρκεια του παιχνιδιού εντός των αποδεκτών χρονικών ορίων. Οι επιθέσεις του χρήστη χαρακτηρίζονται από επιδεξιότητα και άνεση ενώ παράλληλα νιώθει ότι εκτελεί διάφορες μαγικές τεχνικές, πράγμα που προκαλεί τον θαυμασμό του ως προς το αποτέλεσμα αφού τα εφέ παραπέμπουν σε γνήσια μαγεία. Παράλληλα, η ύπαρξη εχθρών με τεχνητή νοημοσύνη συντελεί στην αυθεντικότητα του παιχνιδιού. Επίσης, η εφαρμογή προγραμματίστηκε με βάση τις εκτιμήσεις και των δεκαπέντε εθελοντών έτσι ώστε να αποφευχθεί η ασθένεια προσομοίωσης (simulation sickness). Τα παραπάνω, όπως και όσα θα αναφερθούν στη συνέχεια, προκύπτουν τόσο από τα συμπεράσματα και τις δοκιμές που έγιναν από εμάς όσο και από τα πορίσματα των εντυπώσεων των εθελοντών που δέχτηκαν να δοκιμάσουν την εφαρμογή.

Λάβαμε πολύ σοβαρά υπόψη τις παρατηρήσεις των εθελοντών που δοκίμασαν το παιχνίδι και ιδιαίτερα σχετικά με την ασθένεια εικονικής πραγματικότητας, παρατηρήσεις που συνέβαλαν στη βελτίωση της εφαρμογής στον τομέα του virtual reality sickness. Επίσης, μέσα από αυτές τις παρατηρήσεις προέκυψαν βελτιώσεις σχετικές με την κίνηση του χρήστη στον εικονικό κόσμο. Οι βελτιώσεις αυτές οδήγησαν σε πιο ομαλές και απελευθερωμένες κινήσεις του χρήστη.

Στις αρνητικές επιπτώσεις πλην της ταχύτητας της κίνησης η οποία ρυθμίστηκε έτσι ώστε να αποφευχθεί η ασθένεια προσομοίωσης, παρατηρήθηκε μια σχετική δυσχέρεια στον χρήστη κατά τον χειρισμό του Leap Motion για μικρό χρονικό διάστημα μέχρι να αποκτήσει οικειότητα με το σύστημα και τον τρόπο χειρισμού της εφαρμογής. Οι χρήστες που έρχονται για πρώτη φορά σε επαφή με το εικονικό περιβάλλον, αρχικά αποπροσανατολίζονται, όμως με την πρόοδο του παιχνιδιού αποκαθίσταται η οικειότητα και επιτυγχάνεται η επαφή και η ανάδραση με το παιχνίδι.

Υπάρχει πάντα περιθώριο για βελτιώσεις, όμως στην παρούσα χρονική στιγμή όσον αφορά το Hand Tracking δεν είναι δυνατόν να επιτευχθεί σε πάρα πολύ υψηλό βαθμό ανεξάρτητα από το κομμάτι που μας αναλογεί σαν προγραμματιστές και αυτό κυρίως για δύο λόγους. Οι αισθητήρες του Leap Motion δεν είναι απόλυτα ακριβείς, με αποτέλεσμα οι κινήσεις που απαιτούν υψηλή ακρίβεια να είναι μία πρόκληση. Επίσης, η εικονική πραγματικότητα αυτή καθεαυτή, παρά τις εποικοδομητικές βελτιώσεις των τελευταίων ετών, πάντα θα προκαλεί οπτικά θέματα τα οποία θα πρέπει να βελτιωθούν, όπως η ποιότητα της εικόνας ως και η αίσθηση της κλίμακας της απόστασης.

7.3 Μελλοντικές Επεκτάσεις και Βελτιώσεις

Ολοκληρώνοντας το παιχνίδι και λαμβάνοντας αναπληροφόρηση από τους χρήστες, διαπιστώσαμε ότι μπορούν να γίνουν διάφορες τροποποιήσεις και επεκτάσεις. Έτσι καταλήξαμε στις εξής τέσσερις πιθανές τροποποιήσεις:

1) Ταυτόχρονη χρήση της εφαρμογής στην ίδια πίστα, την ίδια χρονική στιγμή από περισσότερους από έναν χρήστες είτε ως συμπαίκτες είτε ως αντίπαλοι. Αυτό προϋποθέτει την ύπαρξη επιπλέον του ενός υπολογιστή, Oculus Rift και Leap Motion, δηλαδή μια τέτοια τριάδα για κάθε χρήστη. Για να πραγματοποιηθεί το multi-player θα πρέπει τα δεδομένα που παράγονται από κάθε χρήστη στους διάφορους υπολογιστές με Oculus Rift και Leap Motion, να ανεβαίνουν (upload) σε ένα κοινό διαδικτυακό αποθηκευτικό νέφος (cloud) όπου θα γίνεται η επεξεργασία των δεδομένων. Στη συνέχεια, θα πραγματοποιείται το κατάλληλο feedback των επεξεργασμένων δεδομένων από το cloud στον εκάστοτε υπολογιστή. Βασική ασφαλώς προϋπόθεση είναι η ύπαρξη διαδικτύου.

2) Ύπαρξη πιο προηγμένης τεχνητής νοημοσύνης σε σχέση με την ήδη υπάρχουσα στην εφαρμογή. Αυτό θα μπορούσε να πραγματοποιηθεί για παράδειγμα με την ύπαρξη εχθρών ή και χαρακτήρων με πολυπλοκότερη και πιο σύνθετη συμπεριφορά οι οποίοι ενδεχομένως να είναι και πιο ευαίσθητοι στα ερεθίσματα του περιβάλλοντος.

3) Δημιουργία επιπλέον χειρονομιών ώστε ο χρήστης να μπορεί να εκτελεί περισσότερες και πιο σύνθετες ενέργειες μέσω του Leap Motion.

4) Εμπλουτισμός του παιχνιδιού με περισσότερες πίστες και αυξημένος αριθμός αποστολών προκειμένου να έχει τη δυνατότητα ο χρήστης να εμβυθίζεται μέσω της εικονικής πραγματικότητας σε διαφορετικά περιβάλλοντα πχ βυθός θάλασσας, σύμπαν κλπ.

Κεφάλαιο 8ο Επίλογος

Μέσω της πτυχιακής αυτής εργασίας, μου δόθηκε η ευκαιρία να σχεδιάσω και να αναπτύξω ένα τρισδιάστατο παιχνίδι για υπολογιστές κάνοντας χρήση τεχνολογιών που θα παίξουν ιδιαίτερα καθοριστικό ρόλο τα επόμενα χρόνια. Απαιτήθηκαν γνώσεις χειρισμού πολλών επαγγελματικών προγραμμάτων και εργαλείων που χρησιμοποιούνται για την παραγωγή παιχνιδιών και ταινιών όπως της Unity3D, 3DStudioMax, το Leap Motion και το Oculus Rift.

Παρόλο που η διαδικασία υλοποίησης του παιχνιδιού ήταν αρκετά δύσκολη και χρονοβόρα, καθώς η δημιουργία εικονικών περιβαλλόντων για τρισδιάστατα βιντεοπαιχνίδια απαιτεί δημιουργικότητα φαντασία και μηχανική, εν τούτοις το όφελος είναι σημαντικό, καθώς η ικανοποίηση και η γνώση που αποκτά κάποιος ο οποίος ολοκληρώνει μια τέτοια εφαρμογή είναι ανεκτίμητα εφόδια για το μέλλον

Είναι πλέον γεγονός ότι η εικονική πραγματικότητα είναι ένα από τα σημαντικότερα επιτεύγματα της τεχνολογίας και της μηχανικής, παρέχοντας εκτός των άλλων και την ευκαιρία στους χρήστες να επισκέπτονται και να εμβυθίζονται σε κόσμους που ποτέ δε θα είχαν τη δυνατότητα να βιώσουν στην πραγματική τους ζωή. Όμως ο δρόμος είναι ακόμα μακρύς για τη βελτίωση και την τελειοποίηση της με απώτερο στόχο τη δημιουργία εικονικών κόσμων, όσο το δυνατόν δυσδιάκριτων από την πραγματικότητα.

Μέσα λοιπόν σε αυτήν την αυστηρή εφαρμογή κανόνων, συνδυασμό σκέψεων, τεχνολογιών αιχμής, φυσικής και την επιθυμία για περαιτέρω ενασχόληση ας κάνουμε και ένα γλαφυρό διάλειμμα, παρακολουθώντας την κωμωδία «National Lampoon’s Last Resort» όπου η εικονική πραγματικότητα και η αληθινή ζωή αλληλεπικαλύπτονται ή ακόμα και την βρετανική κωμωδία «Red Dwarf» όπου η ζωή είναι ένα παιχνίδι εικονικής πραγματικότητας.

Βιβλιογραφία

1. ***Mark J. P. Wolf, The video game explosion: a history from PONG to Playstation and beyond, 2007***

2. Lanier, Jaron. ***Virtual reality: The promise of the future.*** **s.l. : Interactive Learning International, σσ. 275-79, 1992**

3. ***Virtual Art: From Illusion to Immersion.*** **Grau, Oliver. Cambridge/Massachussetts: MIT-Press : s.n., 2003.**

4. Nikolaidis, Dimitris. ***Auhmented Reality.*** **s.l. : Periscopio tis Epistimis, 2003.**

5. Caudell T. P. and Mizell, D. W., "*Augmented Reality: An Application of Heads-Up Display Technology to Manual Manufacturing Processes*," in Proceedings IEEE Hawaii International Conference on Systems Sciences, Page(s) :659-669, 1992.

6. e Silva, Adriana de Souza και Sutko, Daniel M. ***Digital Cityscapes: Merging digital and urban playspaces.*** **s.l. : Peter Lang, 2009.**

7. Milgram, Paul, και συν. ***Augmented reality: A class of displays on the reality-virtuality continuum.*** **s.l. : Telemanipulator and Telepresence Technologies, 1994.**

8. Read, Paul και Meyer, Mark-Paul. ***Restoration of motion picture film.*** **s.l. : Butterworth-Heinemann, 2000.**

9. ***Head-Mounted Diplay Systems.*** **Rolland, Jannick και Hong, Hua. s.l. : New York: Marcel Dekker, 2005.**

10. StereoLabs. PositionalTracking. [Ηλεκτρονικό] retrieved from

https://www.stereolabs.com/documentation/overview/positional-tracking/introduction.html.

11. Lang, B. An introduction to positional tracking and degrees of freedom (DOF), 2013.

12. Top 8 Eye Tracking. [Ηλεκτρονικό] https://imotions.com/blog/top-8-applications-eye-tracking-research/.

13. ***Designing a PC game engine.*** **Bishop, Lars and Eberly, Dave and Whitted, Turner and Finch, Mark and Shantz, Michael. s.l. : IEEE, IEEE Computer Graphics and Applications, 1998.**

14. Erleben, Kenny and Sporring, Jon and Henriksen, Knud and Dohlman, Kenrik. ***Physics-based Animation (Graphics Series), Charles River Media,*** **2005.**

15. Schreiner, Tim. ***Artificial intelligence in game design,*** **2003.**

16. ***Scripting languages.*** **Morin, Rich και Brown, Vicki. s.l. : http://www. mactech. com/articles/mactech, 1999.**

17. ***Schaum's Outline of Computer Graphics 2/E.*** **Xiang, Zhigang και Plastock , Roy. McGraw Hill Professional, 2000.**

18. 3D Computer Graphics. [Ηλεκτρονικό]

http://www.comphist.org/computing_history/new_page_6.htm

19. ***Moving innovation: a history of computer animation.*** **Sito, Tom. s.l. : MIT Press, 2013.**

20. CryEngine. [Ηλεκτρονικό] https://www.cryengine.com/.

21. Amazon Lumberyard. [Ηλεκτρονικό] https://aws.amazon.com/lumberyard/faq/.

22. Unreal Engine . [Ηλεκτρονικό] https://www.unrealengine.com/unreal-engine-4.

23. Unity . [Ηλεκτρονικό] https://unity3d.com/unity.

24. Unity GameObject. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/class-GameObject.html.

25. Unity Component. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/Components.html.

26. Unity Parenting. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/Hierarchy.html.

27. Unity Lighting. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/Lighting.html.

28. Unity . [Ηλεκτρονικό] https://unity3d.com/learn/tutorials/topics/graphics/introduction-lighting-and-rendering?playlist=17102.

29. Uniy Lighting technique. [Ηλεκτρονικό] https://unity3d.com/learn/tutorials/topics/graphics/choosing-lighting-technique?playlist=17102.

30. *Sams Teach Yourself Unity Game Development in 24 Hours.* Geig, Mike. s.l. : Pearson Education, 2014.

31. Unity . [Ηλεκτρονικό] https://docs.unity3d.com/Manual/RigidbodiesOverview.html.

32. Unity. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/CollidersOverview.html.

33. Brown, Vicki. «"Scripting Languages"»

34. Unity Scripting. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/ScriptingSection.html.

35. Unity Animation. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/AnimationOverview.html.

36. https://dictionary.cambridge.org/dictionary/english/artificial-intelligence

37. Russell, Stuart J.; Norvig, Peter. Artificial Intelligence: A Modern Approach (3rd ed.). Upper Saddle River, New Jersey: Prentice Hall, 2009.

38. Definition of AI as the study of intelligent agents:

- ***Poole, David; Mackworth, Alan; Goebel, Randy . Computational Intelligence: A Logical Approach. New York: Oxford University Press, which provides the version that is used in this article. Note that they use the term "computational intelligence" as a synonym for artificial intelligence, 1998.***
- ***Russell, Stuart J.; Norvig, Peter (2003), Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, New Jersey: Prentice Hall (who prefer the term "rational agent") and write "The whole-agent view is now widely accepted in the field" (Russell & Norvig 2003, p. 55).***
- ***Nilsson, Nils (1998). Artificial Intelligence: A New Synthesis. Morgan Kaufmann Legg & Hutter 2007.***
- ***Legg, Shane; Hutter, Marcus. A Collection of Definitions of Intelligence (Technical report), (15 June 2007)***

39. Unity. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/nav-NavigationSystem.html.

40. Unity User Interface. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/UISystem.html.

41. Unity Audio . [Ηλεκτρονικό] https://docs.unity3d.com/Manual/AudioOverview.html.

42. Unity Camera. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/class-Camera.html.

43. Unity Particle System. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/ParticleSystems.html.

44. Unity Prefab. [Ηλεκτρονικό] https://docs.unity3d.com/Manual/Prefabs.html.

45. Leap Motion Developer. [Ηλεκτρονικό]
https://developer.leapmotion.com/documentation/cpp/devguide/Leap_Overview.html.

46. Oculus Developer. [Ηλεκτρονικό]
https://developer.oculus.com/design/latest/concepts/bp_app_imaging/.

47. Oculus Developer Field of View. [Ηλεκτρονικό]
https://developer.oculus.com/design/latest/concepts/bp_app_fov_scale/.

48.Oculus Developer Rendering. [Ηλεκτρονικό]
https://developer.oculus.com/design/latest/concepts/bp_app_rendering/.

49. Feng Zhu School of Design – Field of View in Games

50. Oculus Developer Tracking. [Ηλεκτρονικό]
https://developer.oculus.com/design/latest/concepts/bp_app_tracking/.

51. Oculus Developer Simulation Sickness. [Ηλεκτρονικό]
https://developer.oculus.com/design/latest/concepts/bp_app_simulator_sickness/.

52. Khundam, Chaowanan. First person movement control with palm normal and hand gesture interaction in virtual reality. *Computer Science and Software Engineering (JCSSE), 2015 12th International Joint Conference on.* s.l. : IEEE, 2015.