

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ - ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ



ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΤΟΜΕΑΣ: ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Διπλωματική Εργασία

του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας
Υπολογιστών της Πολυτεχνικής Σχολής του Πανεπιστημίου Πατρών

ΑΓΓΕΛΟΥ ΚΑΡΔΟΥΤΣΟΥ ΤΟΥ ΑΠΟΣΤΟΛΟΥ

ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ: 1059372

Θέμα

Ανάπτυξη συστήματος μεικτής πραγματικότητας για
υποστήριξη ατόμων με προβλήματα όρασης

Επιβλέπων

Νικόλαος Αβούρης

Αριθμός Διπλωματικής Εργασίας: XXXX

Πάτρα, Φεβρουάριος 2024

ΠΙΣΤΟΠΟΙΗΣΗ

Πιστοποιείται ότι η διπλωματική εργασία με θέμα

**Ανάπτυξη συστήματος μεικτής πραγματικότητας για
υποστήριξη ατόμων με προβλήματα όρασης**

του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και
Τεχνολογίας Υπολογιστών

Άγγελου Καρδούτσου του Απόστολου

(Α.Μ.: 1059372)

παρουσιάτηκε δημόσια στο τμήμα Ηλεκτρολόγων Μηχανικών και
Τεχνολογίας Υπολογιστών στις

...../...../.....

και εξετάστηκε από την ακόλουθη εξεταστική επιτροπή:

Νικόλαος Αβούρης, Καθηγητής, ΗΜ&ΤΥ (Επιβλέπων)

Όνομα Επώνυμο, Βαθμίδα, ΗΜ&ΤΥ (Μέλος Επιτροπής)

Όνομα Επώνυμο, Βαθμίδα, ΗΜ&ΤΥ (Μέλος Επιτροπής)

Ο Επιβλέπων

Ο Διευθυντής του Τομέα

Νικόλαος Αβούρης
Καθηγητής

Γεώργιος Θεοδωρίρης
Καθηγητής

Στοιχεία διπλωματικής εργασίας

Θέμα: Ανάπτυξη συστήματος μεικτής πραγματικότητας για υποστήριξη ατόμων με προβλήματα όρασης

Φοιτητής: Αγγελος Καρδούτσος του Απόστολου

Ομάδα επίβλεψης

Νικόλαος Αβούρης
Γεώργιος Παπαδούλης

Περίοδος εκπόνησης της εργασίας:
Μάιος 2023 - Φεβρουάριος 2024

Η εργασία αυτή γράφτηκε στο X_EΛΤΕΧ και χρησιμοποιήθηκε η γραμματοσειρά GFS Didot του Greek Font Society.

Περίληψη

Παράγραφος 1: Περιγραφή σκοπού διπλωματικής

Παράγραφος 2: Περιγραφή του τι περιγράφουμε στο τεχνολογικό υπόβαθρο και για την ανάπτυξη της εφαρμογής

Παράγραφος 3: Περιγραφή του πειράματος (τόπος και τρόπος διεξαγωγής, συμμετέχοντες)

Παράγραφος 4: Περιγραφή του τελευταίου κεφαλαίου

Λέξεις-κλειδιά: Εκτεταμένη Πραγματικότητα, Επαυξημένη Πραγματικότητα, Μικτή Πραγματικότητα, Microsoft HoloLens 2, Microsoft Visual Studio, Unity, χωρική χαρτογράφηση, χωρικός ήχος, απώλεια όρασης, προσβασιμότητα

Extensive English Summary

Η εργασία αυτή ασχολείται με ένα ιδιαίτερα ενδιαφέρον ζήτημα στο χώρο της επεξεργασίας σημάτων και εικόνων, την ανάλυση (resolution). Παρόλο που σήμερα στον κόσμο μας έχουμε καταφέρει να δημιουργήσουμε συσκευές με μεγάλη ευαισθησία καταγραφής, μεγάλο χώρο αποθήκευσης καθώς και υψηλούς ρυθμούς μετάδοσης και επεξεργασίας δεδομένων, εντούτοις υπάρχουν εφαρμογές όπου η φύση τους είναι τέτοια που δε μας επιτρέπει να επωφεληθούμε σε μεγάλο βαθμό από την πρόοδο που έχει σημειωθεί. Μια τέτοια εφαρμογή είναι οι θερμικές εικόνες και θα δούμε στη συνέχεια της εργασίας τους λόγους εκείνους που την καθιστούν “ιδιαίτερη”.

Ευχαριστίες

Όσο κι αν φαίνεται σαν ατομική δουλειά η παρούσα εργασία, στην πραγματικότητα βοήθησαν αρκετοί άνθρωποι (ο καθένας με το δικό του τρόπο) για να ολοκληρωθεί.

ΠΕΡΙΕΧΟΜΕΝΑ

Κατάλογος σχημάτων	xv
Κατάλογος πινάκων	xvii
1 Εισαγωγή	1
1.1 Δομή Διπλωματικής Εργασίας	3
2 Θεωρητικό και Τεχνολογικό Υπόβαθρο	5
2.1 Όραση	6
2.1.1 Τρόπος Λειτουργίας	6
2.1.2 Απώλεια Όρασης: Κατηγοροποιήση και Αιτίες	7
2.1.3 Απώλεια Όρασης: Αντίκτυπος	7
2.1.4 Απώλεια Όρασης: Λύσεις	8
2.2 Εκτεταμένη Πραγματικότητα	10
2.2.1 Εικονική Πραγματικότητα	10
2.2.2 Επαυξημένη Πραγματικότητα	11
2.2.3 Μικτή Πραγματικότητα	12
2.3 Microsoft HoloLens 2	12
2.3.1 Τεχνικά Χαρακτηριστικά	13
2.3.2 Τρόποι Αλληλεπίδρασης	15
2.3.3 Χωρική Χαρτογράφηση (Spatial Mapping)	19
2.3.4 Χωρικός Ήχος (Spatial Audio)	21
2.4 Εργαλεία ανάπτυξης για HoloLens	21
2.4.1 Unity	21
2.4.2 Microsoft Visual Studio	27
2.4.3 Mixed Reality Toolkit	28

3 Η υλοποίηση	29
3.1 Σενάριο Εφαρμογής	30
3.2 Σχεδιασμός και Περιορισμοί	31
3.3 Γλοποίηση	32
3.3.1 Προετοιμασία και Εγκατάσταση	32
3.3.2 Οργάνωση του Project	35
3.3.3 Εντοπισμός Εμποδίων	38
3.3.4 Προειδοποίηση Χρήστη	46
3.3.5 Φωνητικές Εντολές	49
4 Αξιολόγηση	53
4.1 Διαδικασία αξιολόγησης	54
4.1.1 Πειριγραφή Πειράματος	54
4.1.2 Προδιαγραφές Αξιολόγησης	54
4.1.3 Διεξαγωγή Πειράματος	54
4.2 Αποτελέσματα	54
5 Προεκτάσεις και Επίλογος	55
5.1 Μελλοντικές προεκτάσεις	56
5.2 Επίλογος	56
Βιβλιογραφία	59
Παράρτημα Α'	69
Παράρτημα Β'	101

ΚΑΤΆΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

2.1	Η ανατομία του ματιού (Πηγή: dreamstime.com)	6
2.2	Απτική πλακόστρωση (Πηγή: vecteezy.com)	8
2.3	Κλίμακα Πραγματικού Κόσμου - Εικονικού Κόσμου [1] . .	10
2.4	Χρήστης φορά τα γυαλιά εικονικής πραγματικότητας Meta Quest 3	11
2.5	Η συσκεύη μικτής πραγματικότητας Microsoft HoloLens 2 (Πηγή: theverge.com)	13
2.6	Εξαρτήματα και αισθητήρες της συσκεύης Microsoft HoloLens 2 (Πηγή: learn.microsoft.com)	14
2.7	Αισθητήρες στη προσωπίδα (visor) της συσκεύης Microsoft HoloLens 2 (Πηγή: learn.microsoft.com)	15
2.8	Εικονική αναπαράσταση χεριού (Πηγή: learn.microsoft.com) . .	16
2.9	Τρόποι αλληλεπίδρασης με τους δείκτες των χεριών με 2D αντικείμενα (Πηγή: learn.microsoft.com)	17
2.10	Πίεση (Press) εικονικού κουμπιού (Πηγή: learn.microsoft.com) . .	17
2.11	Τρόποι αλληλεπίδρασης με το δείκτη και τον αντίχειρα με 3D αντικείμενα (Πηγή: learn.microsoft.com)	17
2.12	Αληγεπίδραση με εικονικό αντικείμενο σε μακρινή απόσταση (Πηγή: learn.microsoft.com)	18
2.13	Γύροδειξη διαθέσιμης φωνητικής εντολής μέσω ετικέτας (Πηγή: learn.microsoft.com)	19
2.14	Τριασδιάστατη ανάπαρασταση ένος χώρου (Πηγή: learn.microsoft.com)	20
2.15	Περιοχές (Quads) που σχηματίστηκαν από το Scene Understanding (Πηγή: learn.microsoft.com)	20
2.16	Το περιβάλλον ανάπτυξης (Editor) της Unity (Πηγή: docs.unity3d.com)	23
2.17	To Hierarchy window του Editor	24

2.18 To Game view του Editor	25
2.19 To Scene view του Editor	25
2.20 To Inspector window του Editor	26
2.21 To Project window του Editor	27
2.22 To Status bar του Editor	27
3.1 To Mixed Reality Feature Tool	33
3.2 To hierarchy window μετά την προσθήκη του MRTK	34
3.3 To Holographic Remoting Player σε αναμονή σύνδεσης	35
3.4 Η τελική μορφή του Hierarchy Window	36
3.5 To Inspector window για το variableAggObject	37
3.6 Οι ρυθμίσεις της λειτουργίας ‘Spatial Awareness’ και των ‘Spatial Surface Observers’	39
3.7 Απεικόνιση του εργαστηρίου με 3D μοντέλο	40
3.8 Χωρική χαρτογράφηση του χώρου του εργαστηρίου και απεικόνιση του mesh	41
3.9 Παράδειγμα εντοπισμού εμποδίων με τη μέθοδο BoxCast	45
3.10 Εντοπισμός εμποδίων με τη λειτουργία Hands Mode	46
3.11 To component Audio Source	47
3.12 To χειριστήριο Microsoft XBOX Series (Πηγή: microsoft.com)	49
3.13 To component SpeechInputHandler και οι φωνητικές εντολές της εφαρμογής	50

ΚΑΤΑΛΟΓΟΣ ΠΙΝ'ΑΚΩΝ

2.1 Τεχνικές προδιαγραφές του headset Microsoft HoloLens 2	16
2.2 Διαθέσιμα engines και APIs για την ανάπτυξη εφαρμογών για το Microsoft HoloLens 2	22
3.1 Packages που ενσωματώθηκαν στο project με τη χρήση του Mixed Reality Feature Tool	33

ΚΕΦΑΛΑΙΟ

1

ΕΙΣΑΓΩΓΗ

Ο άνθρωπος, από τη γέννησή του, διαθέτει πέντε βασικές αισθήσεις: την αφή, την όραση, την ακοή, την όσφρηση και την γεύση [2]. Οι αισθήσεις αυτές του δίνουν τη δυνάτοτητα να αντιλήφθει καλύτερα το περιβάλλον γύρω του και συμβάλλουν σημαντικά στην επιβίωσή του. Ωστόσο, εξαιτίας ποικίλων παραγόντων, είναι πιθανό να εμφανιστεί κάποια δυσλειτουργία στο αισθητήριο όργανο, οδηγώντας σε μερική ή πλήρη απώλεια της συγκεκριμένης αίσθησης. Στα πλαίσια της συγκεκριμένης διπλωματικής εργασίας, θα εστιάσουμε στο πρόβλημα της μερικής ή πλήρης απώλειας όρασης. Η απώλεια της συγκεκριμένης αίσθησης μπορεί να δυσχεράνει την υλοποίηση καθημερινών εργασιών από το άτομο, την πρόβαση και πλοήγηση του σε χώρους, να επηρέασε την ψυχολογία και την αυτοπεποίθησή του, οδηγώντας, τελικά, σε υποβάθμιση της ποιότητας ζώης του και έχοντας αρνητικό αντίκτυπο στην κοινωνική του ζωή και στην πνευματική του υγεία [3][4]. Επόμενως, κρίνεται απαραίτητη η ανάπτυξη και η κατασκευή συσκεύων, οι οποίες έχουν σκοπό να αναπληρώσουν την απούσα αίσθηση, εκμεταλλευόμενες τις ήδη υπάρχουσες αισθήσεις, με τελικό στόχο την διευκόλυνση της καθημερινότητας του ατόμου.

Παράλληλα, τα τελευταία χρόνια, η ανάπτυξη της τεχνολογία αποδεικνύεται να είναι ραγδαία. Ειδικότερα, η τεχνολογία της επαυξημένης (Augmented Reality, AR), εικόνικης (Virtual Reality, VR) και μικτής (Mixed Reality, MR) πραγματικότητας «εισβάλει» όλο και περισσότερο στην καθημερινότητα μας, βρίσκοντας εφαρμογή σε διάφορους τομείς αυτής, όπως είναι η διασκέδαση, η εκπαίδευση, η υγεία [5] κ.α. [6], διαθέτοντας συσκεύες και εφαρμογές προσβάσιμες στο μέσο χρήστη, λόγω της απλότητας χρήσης τους και του κόστους τους. Παραδείγματα αυτών αποτελούν οι συσκευές Meta Quest, Valve Index (γυαλιά Εικονικής Πραγματικότητας), Microsoft HoloLens και Google Glass (γυαλιά Επαυξημένης και Μικτής Πραγματικότητας).

Λαμβάνοντας υπόψην, λοιπόν, την τρέχουσα τεχνολογική πρόοδο και το πλήθιος συσκευών που βρίσκονται στη διάθεση του μέσου χρήστη, καθώς και τα προβλήματα προσβάσιμότητας που αντιμετωπίζουν άτομα με μερική ή πλήρη απώλεια όρασης, τότε εύλογα τίθεται το ερώτημα:

Είναι εφικτή η ανάπτυξη μιας εφαρμογής, η οποία αξιοποιεί τις διαθέσιμες τεχνολογίες και hardware επαυξημένης/μικτής πραγματικότητας και παράλληλα βοηθά άτομα με αναπηρία να πραγματοποιήσουν καθημερινές εργασίες με ευκολία;

Σκοπός της διπλωματικής εργασίας είναι η υλοποιήση μιας τέτοιας εφαρμογής, η οποία θα εξυπηρετεί ειδικότερα άτομα με μερική ή πλήρη απώλεια όρασης. Η εφαρμογή στοχεύει στο να προσφέρει βοήθεια κατά την πρόσβαση και περιήγηση ενός τέτοιου ατόμου σε χώρους, ειδοποιώντας τον για πιθανά εμπόδιο που μπορεί να συναντήσει στη διαδρομή του. Για τον εντοπισμό των εμποδίων και την ενημέρωση του χρήστη για αυτά, θα αναπτυχθεί λογισμικό, το οποίο θα αξιοποιεί τους αίσθητηρες και τις ενσωματωμένες τεχνολογίες της συσκευής HoloLens 2 της Microsoft.

1.1 Δομή Διπλωματικής Εργασίας

Η παρούσα διπλωματική εργασία χωρίζεται σε 5 κεφάλαια. Στο 2ο κεφάλαιο παρουσιάζεται σε βάθος το θεωρητικό και τεχνολογικό υπόβαθρο, το οποίο σχετίζεται με την ανάπτυξη της εφαρμογής. Έπειτα, στο 3ο κεφάλαιο, περιγράφεται αναλυτικά η διαδικασία σχεδιασμού και υλοποίησης της εφαρμογής, ενώ, στο 4ο κεφάλαιο, η εφαρμογή διατίθεται σε χρήστες, με σκοπό να την αξιολογήσουν. Παρατίθονται οι συνθήκες κάτω από τις οποίες έγινε η χρήση της εφαρμογής, ο τρόπος αξιολόγησης αυτής, καθώς και τα αποτελέσματα που προέκυψαν από τα πειράματα. Τέλος, στον 5ο κεφάλαιο, αναφέρονται ορισμένες από τις δυνατότητες και προεκτάσεις, που θα μπορούσε να αποκτήσει η εφαρμογή με την περαιτέρω ανάπτυξή της και την ενσωμάτωση επιπλέον τεχνολογιών.

ΚΕΦΑΛΑΙΟ

2

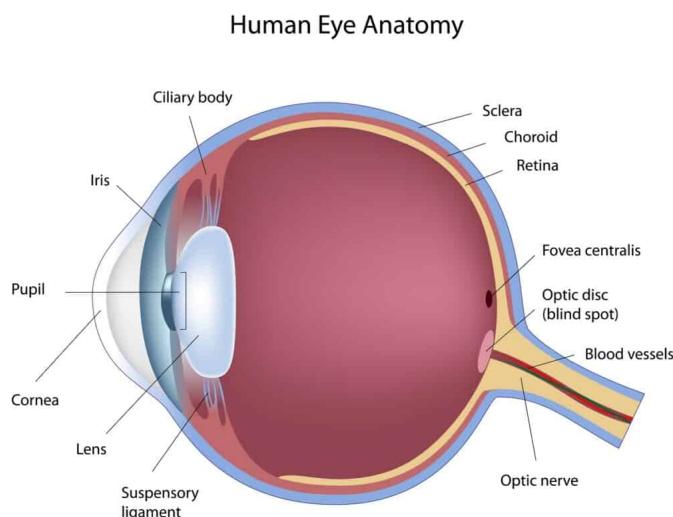
ΘΕΩΡΗΤΙΚΟ ΚΑΙ
ΤΕΧΝΟΛΟΓΙΚΟ ΥΠΟΒΑΘΡΟ

Στο παρών κεφάλαιο θα πραγματοποιηθεί μια αναλυτική παρουσίαση του θεωρητικού και τεχνολογικού υπόβαθρου, που αποτέλεσε βάση για την υλοποίηση της εφαρμογής. Αρχικά, θα δοθεί μια εξήγηση για το τι εστί απώλεια όρασης, τι δυσκολίες αντιμετωπίζουν τα άτομα με αυτό το είδος αναπηρίας, καθώς και τις τρέχουσες λύσεις για την καταπολέμηση δυσκολιών προσβασιμότητας (Κεφάλαιο 2.1). Στη συνέχεια, θα δοθεί ο ορισμός της εκτεταμένης πραγματικότητας, καθώς και τις εφαρμογές που βρίσκει στην καθημερινότητα του ανθρώπου (Κεφάλαιο 2.2). Τέλος, θα δοθεί μια περιγραφή της συσκευής Microsoft HoloLens 2 και των λειτουργιών της (Κεφάλαιο 2.3), και τα εργαλεία, τα οποία θα χρησιμοποιηθούν για την ανάπτυξη της εφαρμογής (Κεφάλαιο 2.4).

2.1 Όραση

2.1.1 Τρόπος Λειτουργίας

Η όραση αποτελεί μία από τις βασικές αισθήσεις του ανθρώπου. Η αίσθηση αυτή βασίζεται στη λειτουργία του ματιού, το οποίο αποτελεί το αισθητήριο όργανο και στο εσωτερικό του οποίου εισερχεται το φως, διαπερνώντας αρχικά το κερατωειδή χιτώνα και την κόρη και προσπίπτει, τελικά, στον αμφιβληστορειδή χιτώνα (Σχήμα 2.1). Αυτό οδηγεί σε διέγερση των οπτικών νεύρων και τα οπτικά σήματα, που αποστέλλονται στον εγκέφαλο, μετατρέπονται σε εικόνα [7][8].



Σχήμα 2.1: Η ανατομία του ματιού (Πηγή: dreamstime.com)

2.1.2 Απώλεια Όρασης: Κατηγοροποιήση και Αιτίες

Όταν η φυσιολογική λειτουργία του αισθητήριου οργάνου διαταραχθεί, τότε το άτομο έρχεται αντιμέτωπο με κάποιο τύπο προβλήματος όρασης. Με βάση τον Παγκόσμιο Οργανισμό Υγείας (Π.Ο.Υ.), τα άτομα αυτά κατηγοροποιούνται σε 6 κατηγορίες (Κατηγορία 0 έως Κατηγορία 5), ανάλογα την οπτική τους οξύτητα, δηλαδή της ικανότητάς του να διακρίνουν σχήματα και λεπτομέρειες από κάποια δεδομένη μακρυμένη απόσταση:

- Στην κατηγορία 0 ανήκουν άτομα με με πλήρως ή σχεδόν πλήρως λειτουργική όραση
- Στην κατηγορία 1 και 2 ανήκουν άτομα που έχουν υποστεί μερική απώλεια της όρασής τους
- Τέλος, στις κατηγορίες 3, 4 και 5 εντάσσονται τα άτομα με σχεδόν πλήρη ή πλήρη απώλεια όρασης

Αντιθέτως, για κοντινές αποστάσεις, τα άτομα με προβλήματα όρασης εντάσσονται σε μία μόνο κατηγορία [9].

Οι κυριότερες αιτίες, οι οποίες προκαλούν προβλήματα όρασης, απότελουν:

- τα διαθλαστικά σφάλματα
- ο καταράκτης
- η διαβητική αμφιβληστροειδοπάθεια
- το γλαύκωμα
- η ηλικιακή εκφύλιση της ωχράς κηλίδας

Από τα ανωτέρω, η κυριότερη αίτια πλήρης απώλειας όρασης σε άτομα ηλικίας 50 ετών και άνω αποτελεί ο καταράκτης, σε ποσοστό 46% των ατόμων που υπέστησαν πλήρη απώλεια όρασης το 2020. Αντιθέτως, για την ίδια ηλικία ομάδα, η οποία αντιμέτωπιζε μερική απώλεια όρασης, κύριες αιτίες αποτελούν τα υποδιορθωμένα διαθλαστικά σφάλματα και ο καταράκτης, σε ποσοστό 30% το καθένα για το ίδιο έτος [10].

2.1.3 Απώλεια Όρασης: Αντίκτυπος

Η απώλεια όρασης έχει ιδιαίτερο αντίκτυπο στην προσωπική, κοινωνική και οικονομική ζωή του ατόμου, ανεξαρτήτου της ηλικίας του. Η εκπλήρωση απλών καθημερινών εργασιών μπορεί να αποδειχθεί ιδιαίτερα δύσκολη και χρόνοβορα, επειδυνώνοντας ιδιαίτερα την ποιότητα ζωής του ατόμου [11][12], ακόμη και σε επίπεδο χαμηλότερο από αυτό ατόμων που αντιμετωπίζουν χρόνια νοσήματα [13].

Η εμφάνιση προβλημάτων όρασης σε αρκετά νεαρή ηλικία μπορεί να επηρέασει αισθητά την ανάπτυξη ικανοτήτων όπως είναι η κινητήρια και η γνωστική του ικανότητα και η κοινωνική του καλλιέργεια [3]. Αντιθέτως, στην ενήλικη ζωή του, μπορεί να δυσκολέψει την εύρεση εργασίας και να επηρεάσει την φυσική του υγεία, προκαλώντας μέχρι και άγχος και κατά-

θλιψη [12]. Τέλος, σε άτομα άρκετα μεγάλης ηλικίας, η απώλεια όρασης μπορεί να δυσχεράνει βασικές λειτουργίες, όπως είναι το περπατήμα με εγγενές κίνδυνο την πτώση και τον τραυματισμό [3].

2.1.4 Απώλεια Όρασης: Λύσεις

Πλήθος λύσεων είναι διαθέσιμες στο μέσο άνθρωπο, οι οποίες στοχεύουν στην πρόληψη της απώλειας της όρασης ή στην επιδιόρθωση αυτής. Σε απλές περιπτώσεις, όπως είναι η μυωπία, πρεσβυωπία, κ.λ.π., το πρόβλημα μπορεί να επιλύεται με χρήση διορθωτικών φακών ή εγχείρηση στο αισθητήριο όργανο [3]. Επίσης, για άτομα που δεν μπορούν επιδιορθώσουν το πρόβλημα όρασής του, έχουν αναπτυχθεί συσκευές και τεχνολογίες, οι οποίες συνεχώς εξελίσσονται και έχουν ως σκοπό την εξυπηρέτηση του ατόμου σε διάφορες πτυχές της καθημερινότητάς του. Επί δεκαετείες, το μπαστούνι αποτελεί μια αρκετά διαδεμένη συσκευή, που βοηθά ένα άτομο να αναγνωρίζει εμπόδια, καθώς περιηγείται σε έναν χώρο. Σε συνδυασμό με την απτική πλακόστρωση (Σχήμα 2.2), το άτομο μπορεί να περιηγηθεί σε δημοσίου χώρου, όντας συνεχώς ενήμερος για την ύπαρξη εμποδίων, όπως δρόμοι, διάβαση πεζών, γραμμές τραμ, στη διαδρομή του, ανάλογα με το μοτίβο των πλακών του πεζοδρομίου [14].



Σχήμα 2.2: Απτική πλακόστρωση (Πηγή: vecteezy.com)

Ευρέως διαδεδομένη είναι και η χρήση σκύλων-βοηθών, οι οποίοι είναι ειδικά εκπαιδευμένοι σχετικά με την αναπηρία του ιδιοκτήτη τους με σκοπό να τους εξυπηρετήσουν στις ιδιαίτερες ανάγκες που μπορούν να έχουν. Στην περίπτωση των ατόμων με προβλήματα όρασης, σκοπός τους είναι να κατευθύνουν τον ιδιοκτήτη τους στο προορισμό τους, ειδοποιώντας τον για

πιθανά εμπόδια [15]. Τέλος, αναπτύχθηκε το σύστημα γραφής Braille, ώστε να είναι εφικτή η ανάγνωση κειμένων με τη βοήθεια της αφής.

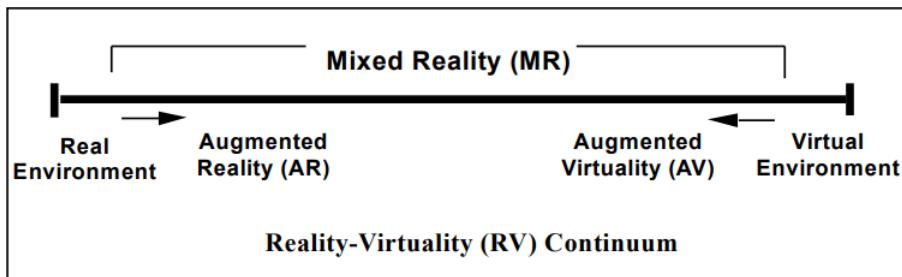
Στη σύγχρονή εποχή, όλο και περισσότερες συσκευές κατασκευάζονται λαμβάνοντας προκαταβολικά υπόψη τη δυνατότητα χρήσης αυτών από άτομα με περιορισμένη όραση. Οι κινητές συσκευές διαθέτουν λογισμικό screen reader (π.χ. TalkBack σε Android συσκεύες ή VoiceOver σε iOS συσκευές), διευκολύνοντας την πλοιήγηση του χρήστη στις εφαρμογές του κινητού, καθώς πραγματοποιούν καταγραφή και αναγνώριση των κειμένων και των λειτουργιών, που βρίσκονται στη οθόνη τη δεδομένη χρονική στιγμή, και ο χρήστης, με συγκεκριμένες χειρονομίες, επιλέγει αν επιθυμεί την ανάγνωση κάποιο κείμενου με χρήση text-to-speech ή την πραγματοποίηση κάποιας ενέργειας [16]. Επιπλέον, η γλώσσα προγραμματισμού HTML δίνει τη δυνανότητα στους προγραμματιστές να κατασκευάσουν ιστοσελίδες, οι οποίες θα είναι προσβάσιμες από άτομα με περιορισμένη όραση, καθώς οι screen readers θα μπορούν να αναγνωρίζουν τμήματα της ιστοσελίδας, όπως επικεφαλίδες, υπερσυνδέσμους, κουμπιά και το σκοπό που επιτελούν [17].

Όσον αφορά την περιήγηση ατόμων σε ανοιχτούς ή κλειστούς χώρους, πληθώρα συσκευών έχουν κατασκευαστεί, οι οποίες βελτιώνουν ήδη υπάρχουσες ή προσφέρουν έναν εναλλακτικό τρόπο λειτουργίας και παροχής βοήθειας. Παραδείγματα τέτοιων συσκευών αποτελούν:

- **biped:** Συσκευή, η οποία φοριέται γύρω από το λαιμό του χρήστη. Διαθέτει 3 κάμερες, που προσφέρουν πεδίο ορατότητας 170°, και ενσωματώνουν λογισμικό για τον εντοπισμό και αναγνώριση εμποδίων. Ο χρήστης προειδοποιείται για εμπόδια με μια ηχητική προειδοποίηση. [18]
- **OrCam MyEye:** Φορητή συσκευή, η οποία προσφέρει τη δυνατότητα ανάγνωσης κειμένων, αναγνώρισης αντικειμένων, χρωμάτων και προσώπων. [19]
- **BlindSquare:** Εφαρμογή πλοιήγησης, η οποία παρέχει αναλύτικες οδηγίες στο χρήστη, ώστε να κατευθυνθεί στο προόρισμα του, παρέχοντας παράλληλα πληροφορίες για το περιβάλλον του χρήστη και για σημεία ενδιαφέροντος. [20]
- **Be My Eyes:** Εφαρμογή, η οποία επιτρέπει σε χρήστες να έρθουν σε επικοινωνία με εθελοντές μέσω βιντεοκλήσης, με σκοπό να ζητήσουν βοήθεια. [21]
- **WeWALK:** Αποτελεί ένα εξάρτημα για μπαστούνια, το οποίο έχει τη δυνατότητα να εντοπίζει εμπόδια σε χαμηλό ύψος με χρήση υπερήχων και να ειδοποιεί το χρήστη με δονήσεις και ήχο. [22]
- **Seeing AI:** Εφαρμογή της Microsoft, η οποία, με χρήση τεχνητής νοημοσύνης, παρέχει περιγραφές των αντικειμένων, χρωμάτων, ανθρώπων και κειμένων, τα οποία στοχεύει ο χρήστης με την κάμερα του κινητού του τηλεφώνου. [23]

2.2 Εκτεταμένη Πραγματικότητα

Η εκτεταμένη πραγματικότητα (Extended Reality - XR) αποτελεί μια ιδιαιτέρως ευρεία έννοια, η οποία χρησιμοποιήθηκε πρώτη φορά το 1991 από τους Steve Mann και Charles Wyckoff, κατά την προσπάθεια κατασκευής συσκευών εικονικής/επαυξημένης πραγματικότητας [24][25]. Αποτελεί 'ομπρέλα' εννοιών και περιλαμβάνεις τις έννοιες της Εικονικής (Virtual Reality - VR, Κεφάλαιο 2.2.3), της Επαυξημένης (Augmented Reality - AR, Κεφάλαιο 2.2.2) και της Μικτής Πραγματικότητας (Mixed Reality - MR, Κεφάλαιο 2.2.3) [1]. Το 1994, οι Paul Milgram και Fumio Kishino άρισαν ένα συνεχές μεταξύ πραγματικού και εικονικού κόσμου (Σχήμα 2.3). Αυτοί οι δύο κόσμοι αποτελούν τα άκρα της κλίμακας και μεταξύ αυτών υπάρχουν οι έννοιες της επαυξημένης και μικτής πραγματικότητας, καθώς και της επαυξημένης εικονικότητας.



Σχήμα 2.3: Κλίμακα Πραγματικού Κόσμου - Εικονικού Κόσμου [1]

2.2.1 Εικονική Πραγματικότητα

Η εικονική πραγματικότητα αποτελεί μια προσομοίωση, η οποία εντάσσει τον χρήστη σε έναν εικονικό κόσμο, κατασκευάσμενος από υπολογιστή, διεγείροντας αισθήσεις όπως η όραση (μπορεί να δει τον κόσμο αυτόν), η ακοή (μπορεί να ακούσει ήχους που προέρχονται από τον κόσμο αυτό) και η αφή (μπορεί να αντιληφθεί την επαφή με εικονικά αντικείμενα λαμβάνοντας απτική ανάδραση [26][27]), δίνοντάς του την δυνατότητα να αλληλεπιδράσει με αυτόν [28]. Τα ανωτέρω επιτυγχάνονται με χρήση ειδικών συσκευών, όπως είναι τα γυαλιά εικονικής πραγματικότητας (VR headsets), τα οποία είτε ενσωματώνουν ειδικούς αισθητήρες με σκοπό να ανιχνεύσουν την κίνηση του κεφαλιού, είτε η ανίχνευση αυτή γίνεται με χρήση εξωτερικών αισθητήρων, που τοποθετούνται σε διάφορα σημεία στον περιβάλλοντα χώρο. Για την ανίχνευση της κίνησης των χέριων, που επιτρέπουν και την αλληλεπίδραση με εικονικά αντικείμενα, απαιτείται η χρήση χειριστηρίων. Παραδείγματα τέτοιων συσκευών αποτελούν το Meta Quest (Σχήμα 2.4) και το Valve Index. Η VR τεχνολογία έχει εισβάλει πλέον και στην καθημερινότητα του μέσου χρήστη βρίσκοντας εφαρμογή σε ποικίλους τομείς:

- Στην ψυχαγωγία και στη διασκέδαση, μέσω των βιντεοπαιχνιδιών που έχουν αναπτυχθεί, την δυνατότητα παρακολούθησης πραγματικών αθλητικών αγώνων, καθώς και εικονικών ξεναγήσεων σε χώρους πολιτιστικού ενδιαφέροντους [29][30][31]
- Στην εκπαίδευση, προσφέροντας έναν εναλλακτικό και διαδραστικό τρόπο εκμάθησης [32][33][34]
- Στην ιατρική, με σκοπό την εκπαίδευση και προετοιμασία ιατρών, καθώς και για να προσφέρει βιοήθεια σε ασθενείς [35][32][36][37]



Σχήμα 2.4: Χρήστης φορά τα γυαλιά εικονικής πραγματικότητας Meta Quest 3

2.2.2 Επαυξημένη Πραγματικότητα

Η επαυξημένη πραγματικότητα αποτελεί μια τεχνολογία, η οποία ‘ενισχύει’ τον πραγματικό κόσμο, ενσωματώνοντας τεχνητά δημιουργήμένα, από υπολογιστή, στοιχεία (εικονικά αντικείμενα, ήχους) σε αυτόν. Τα στοιχεία αυτά βρίσκονται εντός ενός ‘στρώματος’ (layer), το οποίο τοποθετείται ‘πάνω’ από τον πραγματικό κόσμο, ο οποίος μπορεί να είναι μια φωτογραφία, ένα βίντεο ή ένα βίντεο πραγματικού χρόνου [38][39]. Λόγω της εξέλιξης των επεξεργαστών και την απλότητας της τεχνολογίας (σε σχέση με την εικονική πραγματικότητα), ένας χρήστης μπορεί να βιώσει την εμπειρία της επαυξημένης πραγματικότητας χρησιμοποιώντας μια απλή, έξυπνη κινητή συσκευή (smartphone) [40]. Παράλληλα, είναι διαθέσιμα και γυαλιά επαυξημένης πραγματικότητας (AR headsets), όπως είναι τα Xreal Air AR Glasses και Magic Leap 2.

Η επαυξημένη πραγματικότητα βρίσκει εφαρμογή σε παρόμοιους τομείς με αυτούς της εικονικής πραγματικότητας:

- Στην εκπαίδευση [41][42]
- Στον εργασιακό χώρο, αποτελώντας ένα επιπλέον εργαλείο για τον εργαζόμενο με σκοπό να διευκολύνει το έργο και να βελτιώσει την απόδοσή του [43][44][45]
- Στην υγεία [46][47][37][48]

- Στην ψυχαγωγία [49]
- Στον τουρισμό, παρέχοντας ξεναγήσεις, όπου ο χρήστης μπορεί να μάθει περισσότερες πληροφορίες για μνημεία απλά στοχεύοντας την κάμερα του κινητού του σε αυτά [50][51]

2.2.3 Μικτή Πραγματικότητα

Η ‘μικτή πραγματικότητα’ αποτελεί μια εννοιά, η οποία, μέχρι και πρόσφατα, δεν έχει προσδιοριστεί ξεκαθάρα και επιδέχεται πληθώρα ορισμών, οι οποίοι, ωστόσο διαθέτουν μια κοινή βάση [52]. Βάση του ορισμού που δόθηκε από τους Milgram και Kishino με το Συνεχές Πραγματικού-Εικονικού Κόσμου (Σχήμα 2.3), η μικτή πραγματικότητα αποτελεί οτιδήποτε ανήκει μεταξύ των δύο άκρων της αλιμακας, όντας η επαυξημένη πραγματικότητα και η επαυξημένη εικονικότητα. Με τον όρο επαυξημένη εικονικότητα εννοείται ένας εικονικός κόσμος, στον οποίο ενσωματώνονται πραγματικά αντικείμενα του περιβάλλοντα χώρου [1]. Ορισμένοι ακόμη δημοφιλείς ορισμοί της μικτής πραγματικότητας είναι:

1. Η μικτή πραγματικότητα αποτελεί συνώνυμο της επαυξημένης πραγματικότητας
2. Η τεχνολογίας μικτής πραγματικότητας αποτελεί μια ενισχυμένη μορφή της τεχνολογίας επαυξημένης πραγματικότητας, όπου τα εικονικά αντικείμενα αλληλεπιδρούν με το πραγματικό περιβάλλον και τα αντικείμενα σε αυτόν
3. Η μικτή πραγματικότητα αποτελεί συνδυασμός της επαυξημένης και της εικονικής πραγματικότητας, ενσωματώνοντας στοιχεία και από τις δύο τεχνολογίες

Παράδειγμα συσκευών που αξιοποιούν την τεχνολογία μικτής πραγματικότητας αποτελούν τα Microsoft HoloLens, Apple Vision Pro και Meta Quest Pro. Η μικτή πραματικότητα βρίσκει και αυτή ευρεία εφαρμογή σε αρκετούς τομείς της σύγχρονης καθημερινότητας, όπως είναι η εκπαίδευση [53][54], στην υγεία [55][56], στην αρχιτεκτονική [57][58] κ.α.

2.3 Microsoft HoloLens 2

Το 2015, η Microsoft παρουσίασε τα γυαλιά μικτής πραγματικότητας Microsoft HoloLens, όντας το πρώτο ‘ασύρματο’ ολογραφικό υπολογιστικό σύστημα, δηλαδή δεν ήταν απαραίτητη η χρήση καλωδίων για τη λειτουργία του, καθιστώντας το πλήρως φορητό [59]. Η πρώτη έκδοση του headset, Development Edition, έγινε διαθέσιμη το 2016. Η συσκεύη αυτή, καθώς και οι εκδόσεις που ακολούθησαν, ενσωματώνουν την πλατφόρμα Windows Mixed Reality, η οποία βασίζεται στο λειτουργικό σύστημα Windows 10 [60]. Η συσκεύη εντάχθηκε σε version ‘μακροχρόνιας υποστήριξης’ (Long-term support, LTS) λαμβάνοντας την τελευταία ενημέρωση τον Οκτώβριο του 2018 [61][62]. Τρία χρόνια μετά την κυκλοφορία

των Microsoft HoloLens, το 2019, παρουσιάζονται σε συνέδριο της Βαρκελώνης τα Microsoft HoloLens 2 (Σχήμα 2.5), τα οποία και τίθονται σε διαθεσιμότητα την ίδια χρονιά (Νοέμβριος 2019) [63]. Είναι διαθέσιμες προς αγορά τρεις εκδόσεις της συσκεύης, ανάλογα με το λόγο και το χώρο χρήσης της [64]:

- **HoloLens 2**, για απλή, καθημερινή χρήση
- **HoloLens 2 Industrial Edition**, για χρήση σε ελεγχόμενους χώρους, όπως ένα αποστειρωμένο εργαστήριο
- **Trimble XR10 with HoloLens 2**, ένα προστατευτικό κράνος που ενσωματώνει τη συσκευή HoloLens για χώρους, όπου η ασφάλεια κρίνεται απαραίτητη



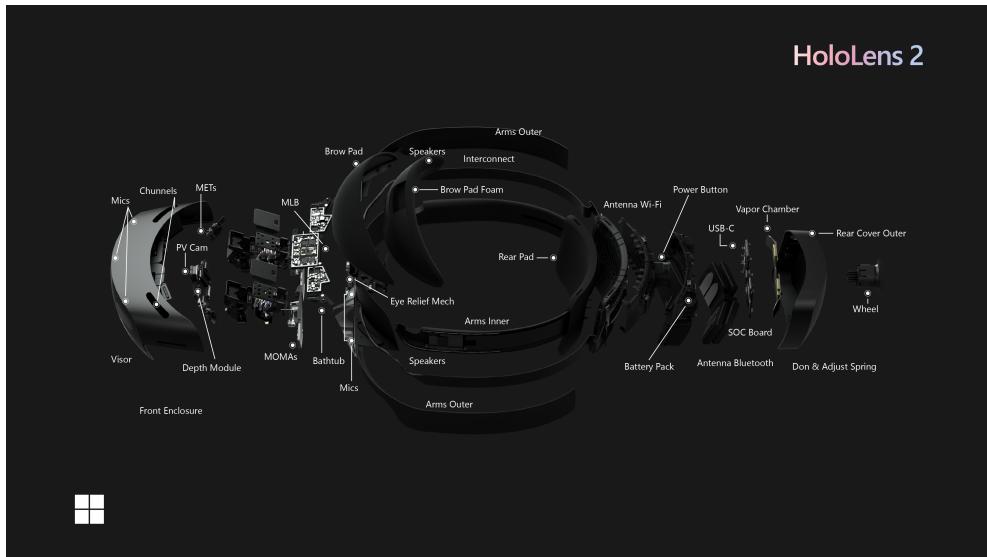
Σχήμα 2.5: Η συσκεύη μικτής πραγματικότητας Microsoft HoloLens 2 (Πηγή: [theverge.com](https://www.theverge.com))

2.3.1 Τεχνικά Χαρακτηριστικά

Η συσκεύη Microsoft HoloLens 2, όπως και ο προκάτοχός της, αποτελεί ένα 'ασύρματο' ολογραφικό υπολογιστικό σύστημα. Το λειτουργικό του σύστημα είναι το Windows Holographic OS, το οποίο βασίζεται στο λειτουργικό σύστημα Windows 10 [65]. Ωστόσο, από το πρώτο εξάμηνο του 2023, η Microsoft δίνει τη δυνατότητα στους χρήστες της συσκεύης να την αναβαθμίσουν στο λειτουργικό σύστημα Windows 11 [66]. Το headset αποτελείται από πολλά τμήματα/έξαρτηματα και αισθητήρες, όπως μπορούμε να παρατηρήσουμε και στο Σχήμα 2.6.

Αρχικά, τα εξαρτήματα από τα οποία αποτελείται η συσκευή είναι τα κατώθι [65][67]:

- **Visor** (Προσωπίδα): Η προσωπίδα διαθέτει τους αισθητήρες και τις



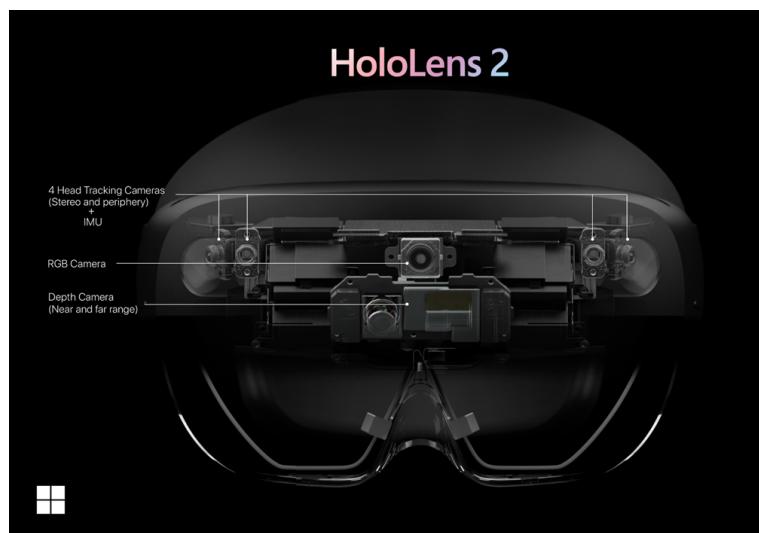
Σχήμα 2.6: Εξαρτήματα και αισθητήρες της συσκεύης Microsoft HoloLens 2 (Πηγή: learn.microsoft.com)

οθόνες. Επιπλέον, μπορεί να περιστραφεί προς τα πάνω κατά της χρήσης της συσκεύης, σε περίπτωση που ο χρήστης δεν χρησιμοποιεί τις οθόνες και θέλει να παρατηρήσει κάτι στον πραγματικό περιβάλλοντα χώρο.

- **Headband** (Ζώνη κεφαλής): Η ζώνη περιβάλλει το κεφάλι του χρήστη και εξασφαλίζει τη σωστή και σταθερή τοποθέτησή της. Με χρήση μιας ροδέλας, στο πίσω μέρος της συσκευής, ο χρήστης μπορεί να ρυθμίσει πόσο ‘σφιχτή’ ή ‘χαλαρή’ είναι η ζώνη, κατά την αρεσκεία του.
- **Brightness Buttons** (Κουμπιά φωτεινότητας): Με τα κουμπιά ρύθμισης φωτεινότητας, τα οποία βρίσκονται στην αριστερή πλευρά της προσωπίδας, ο χρήστης μπορεί να προσαρμόσει την φωτεινότητα των οθονών ανάλογα με το φως του περιβάλλοντα χώρου.
- **Volume Buttons** (Κουμπιά ήχου): Τα κούμπια ρύθμισης της έντασης του ήχου βρίσκονται στη δεξιά πλευρά της προσωπίδας.
- **Power Button** (Κουμπί ενεργοποίησης): Το κουμπί ενεργοποίησης (και απενεργοποίησης) της συσκευής βρίσκεται στο δεξί μέρος του εξωτερικού καλύμματος στην πίσω πλευρά της συσκευής.
- **USB-C Port** (Θύρα τύπου USB-C): Η θύρα USB-C, η οποία βρίσκεται κάτω από το κουμπί ενεργοποίησης, χρησιμοποιείται για τη φόρτιση της συσκευής και για τη σύνδεση αυτής με άλλες συσκευές.

Επιπλέον, η συσκευή διαθέτει ένα πλήθος από αισθητήρες, οι οποίοι, όπως αναφέρθηκε, βρίσκονται στο visor (Σχήμα 2.7). Ειδικότερα, διαθέτει [65]:

- 4 κάμερες ορατού φωτός, κάθε μία από τις οποιες έχει πεδίο ορατότητας (Field of View, FOV) 96.1°. Οι κάμερες αυτές χρησιμοποιούνται για τον εντοπισμό της θέσης και του προσανατολισμού του κεφαλιού (**Head Tracking**).
- 2 κάμερες υπέρυθρου φωτός, οι οποίες χρησιμοποιούνται για τον εντοπισμό της κίνησης των ματιών και του βλέμματος (**Eye Tracking**).
- 1 Time-of-Flight αισθητήρα βάθους (**Depth**) του 1 megapixels, δηλαδή ένα αισθητήρα που μετρά αποστάσεις βάσει του χρόνου που χρειάζονται τα φωτόνια, που εκπέμπει, να χτυπήσουν ένα στόχο και να επιστρέψουν στο δέκτη του αισθητήρα. Σε συνδυασμό με τις κάμερες υπέρυθρου φωτός, είναι εφικτή η χωρική χαρτογράφηση (Κεφάλαιο 2.3.3) του περιβάλλοντος χώρου.
- 1 μονάδα μέτρησης αδράνειας (**Inertia Measurement Unit, IMU**), η οποία περιλαμβάνει επιταχυνσιόμετρο, γυροσκόπιο και μαγνητόμετρο.
- 1 κάμερα των 8 megapixels, η οποία καταγράφει και βίντεο σε ανάλυση 1080p και 30 FPS.



Σχήμα 2.7: Αισθητήρες στη προσωπίδα (visor) της συσκεύης Microsoft HoloLens 2 (Πηγή: learn.microsoft.com)

Τέλος, μερικές ακόμη σημαντικές προδιαγραφές της συσκευής αναφέρονται στον Πίνακα 2.1.

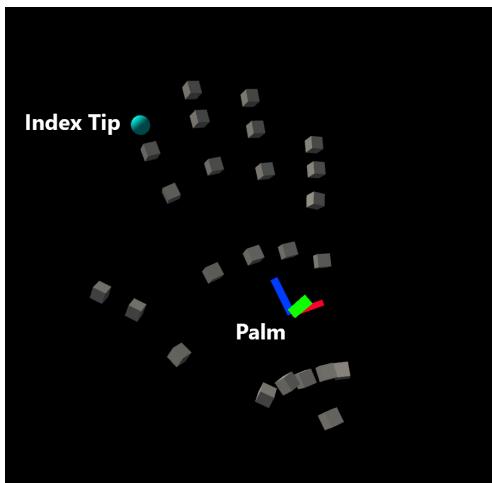
2.3.2 Τρόποι Αλληλεπίδρασης

Ο χοήστης έχει στη διάθεση τρεις βασικούς τρόπους με τους οποίους μπορεί να αλληλεπιδράσει με τη συσκευή και τα ολογράμματα που δημιουργεί [65]:

Μικρόφωνο	Συστοιχία 5 καναλιών
Ηχείο	Ενσωματώνει τεχνολογία χωρικού ήχου (Κεφάλαιο 2.3.4)
System-on-Chip	Qualcomm Snapdragon 850 Compute Platform
Holographic Processing Unit (Ολογραφική Μονάδα Επεξεργασίας)	Δεύτερης γενιάς, ειδικά κατασκευασμένη Holographic Processing Unit
Μνήμη	4GB
Αποθηκευτικός Χώρος	64GB
WiFi	802.11ac 2x2
Bluetooth	5.0
Βάρος	566 γραμμάρια

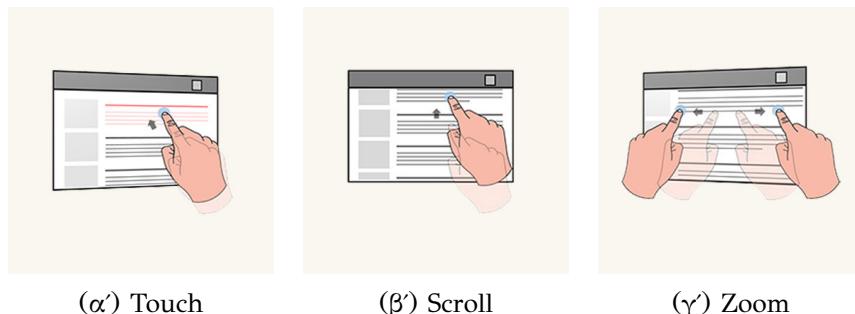
Πίνακας 2.1: Τεχνικές προδιαγραφές του headset Microsoft HoloLens 2

- Με εντοπισμό και χρήση των χεριών (Hand Tracking):** Είναι δυνατός ο εντοπισμός των κινήσεων και των δύο χεριών, καθώς και των δαχτύλων αυτών, όταν αυτά βρίσκονται εντός του πεδίου ορατότητας (FOV) της συσκευής. Τα εικονικά χέρια που δημιουργούνται για την αλληλεπίδραση αποτελούν ένα σύνολο σημείων (Σχήμα 2.8), όπου κάθε ένα αντιπροσωπεύει σύνδεσμο των χεριών ή σημεία αυτών [68].



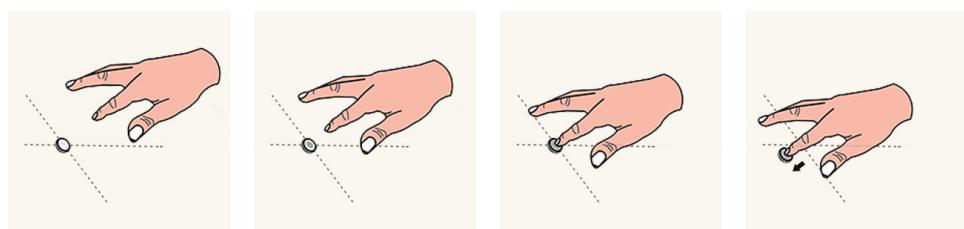
Σχήμα 2.8: Εικονική αναπαράσταση χεριού (Πηγή: learn.microsoft.com)

Ο χρήστης μπορεί να αληλεπιδράσει με τα εικονικά αντικείμενα, είτε από κοντινή [69], είτε από μακρινή απόσταση [70] και να πραγματοποιήσει συγκεκριμένες χειρονομίες (Gestures), όπως είναι το Air Tap, το Tap and Hold [71] και το Start Gesture (επιτρέπει την πρόσβαση στο κεντρικό μενού) [72], ώστε να υποδείξει την ενέργεια που επιθυμεί. Για παράδειγμα, με τους δείκτες των χέριων, ο χρήστης

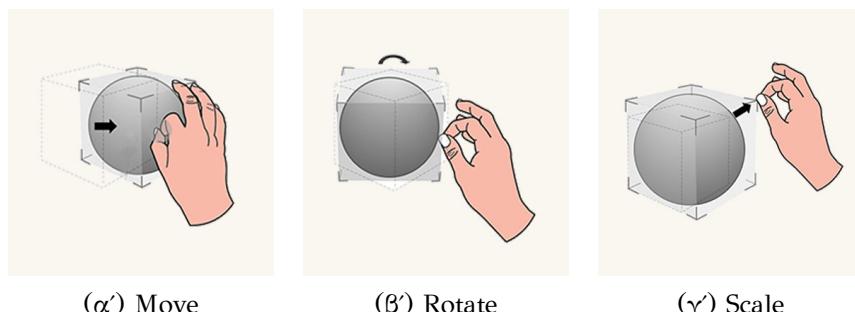


Σχήμα 2.9: Τρόποι αλληλεπίδρασης με τους δείκτες των χεριών με 2D αντικείμενα (Πηγή: learn.microsoft.com)

μπορεί να αγγίξει αντικείμενα (Touch) (Σχήμα 2.9α'), να πιέσει κουμπιά (Press) (Σχήμα 2.10), να κάνει ανακύλιση μιας σελίδας (Scroll) (Σχήμα 2.9β') ή να κάνει Zoom (Σχήμα 2.9γ') [69].



Σχήμα 2.10: Πίεση (Press) εικονικού κουμπιού (Πηγή: learn.microsoft.com)

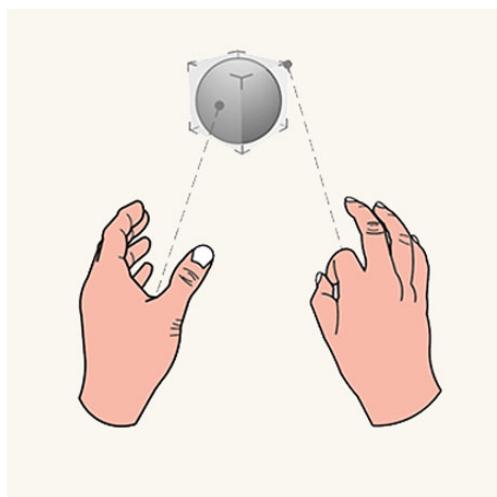


Σχήμα 2.11: Τρόποι αλληλεπίδρασης με το δείκτη και τον αντίχειρα με 3D αντικείμενα (Πηγή: learn.microsoft.com)

Αν ο χρήστης κρατήσει τον αντίχειρα και τον δείκτη ενωμένο, τότε μπορεί να μετακινήσει (Move) (Σχήμα 2.11α') και να αλλάξει την κλίμακα (Scale) (Σχήμα 2.11γ'), τόσο δισδιάστατων (2D), όσο και τρισδιάστατων (3D) αντικειμένων. Επιπλέον, για τα 3D εικονικά αντικείμενα, με την ίδια χειρονομία (gesture), μπορεί να τα περιστρέψει (Rotate) (Σχήμα 2.11β'). Το σημείο επαφής με το αντικείμενο είναι αυτό το οποίο καθορίζει ποια ένεργεια από τις τρεις ανωτέρω θα

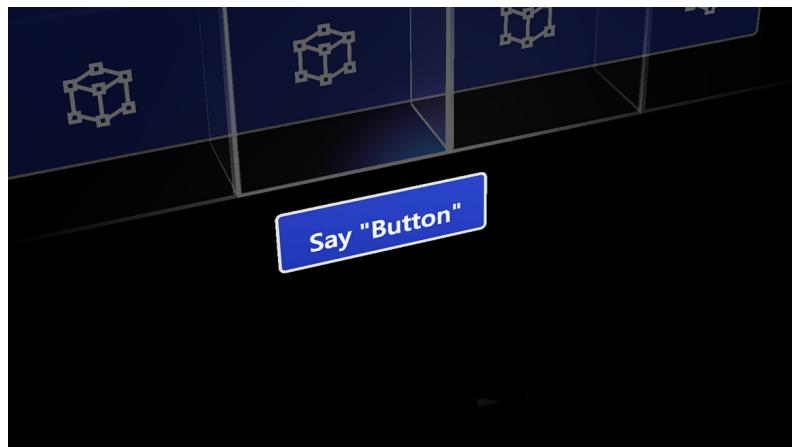
πραγματοποιήσει ο χρήστης [69].

Τέλος, οι ίδιες ενέργειες που αναφέρθηκαν ως τώρα, μπορούν να πραγματοποιηθούν και σε εικονικά αντικείμενα που βρίσκονται μακριά από το χρήστη (σε απόσταση μεγαλύτερη των 50 εκατοστών). Από το δάκτυλο του χρήστη εκτείνεται μια ακτίνα, στο τέλος του οποίου υπάρχει ένας κέρσορας, ώστε να διευκολύνει τον χρήστη να αντιληφθεί ποιο είναι το σημείο αλληλεπίδρασης με το αντικείμενο [70]. Ο ίδιος δείκτης εμφανίζεται και κατά την αλληλεπίδραση με αντικείμενα σε κοντινή απόσταση.



Σχήμα 2.12: Αλληλεπίδραση με εικονικό αντικείμενο σε μακρινή απόσταση
(Πηγή: learn.microsoft.com)

2. **Με το βλέμα και την κίνηση των ματιών ή του κεφαλιού (Gaze and Dwell) [73]:** Ο χρήστης ελέγχει ένα κέρσορα, τον οποίο μπορεί να μετακινήσει εντός του FOV, χρησιμοποιώντας είτε την κίνηση του κεφαλιού του [74], είτε την κίνηση των ματιών (Eye Tracking) [75]. Αυτός ο τρόπος αλληλεπίδρασης μπορεί να αποδειχθεί ιδιαίτερα χρήσιμος, σε περίπτωση που ο χρήστης αδυνατεί να χρησιμοποιήσει τα χέρια του, επειδή, για παράδειγμα, κρατά κάποιο εργαλείο ή αντιμετωπίζει κάποια δυσκολία ή πρόβλημα υγείας, ενώ έχει ακόμη μεγαλύτερη χρηστικότητα, αν συνδυαστεί με φωνητικές εντολές [71][76].
3. **Με φωνητικές εντολές (Voice Input):** Ο χρήστης έχει την δυνατότητα να χρησιμοποιήσει φωνητικές εντολές, ώστε να επιτελέσει ενέργειες. Η συσκευή ακολουθεί το μοντέλο ‘See It, Say It’, οπότε οι εντολές που μπορούν να χρησιμοποιηθούν υποδεικνύονται με ετικέτες (Σχήμα 2.13) και κουμπιά. Τέλος, αυτός ο τρόπος αλληλεπίδρασης μπορεί να συνδυάστει με οποιονδήποτε από τους δύο προηγούμενους για λόγους ευκολίας ή/και αύξησης απόδοσης [76].



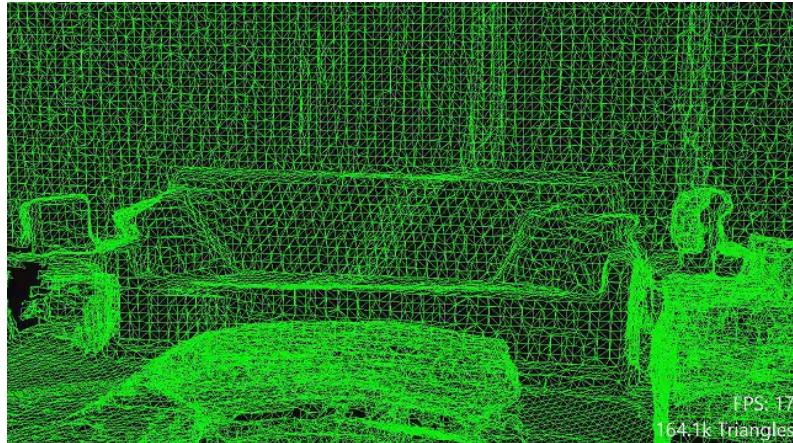
Σχήμα 2.13: Υπόδειξη διαθέσιμης φωνητικής εντολής μέσω ετικέτας (Πηγή: learn.microsoft.com)

2.3.3 Χωρική Χαρτογράφηση (Spatial Mapping)

Η χωρική χαρτογράφηση (spatial mapping) αποτελεί μία από τις κύριες δυνατότητες της συσκεύης Microsoft HoloLens και είναι το κύριο χαρακτηριστικό, στο οποίο στηρίζονται οι εφαρμογές που αναπτύσσονται για τη συσκευή για να υλοποιήσουν τις λειτουργίες μικτής πραγματικότητας. Χωρική χαρτογράφηση είναι η σάρωση και η δημιουργία μίας 3D αναπαράστασης του περιβάλλοντος χώρου του χρήστη (Σχήμα 2.14). Αυτό επιτρέπει στα εικονικά αντικείμενα να αλληλεπιδρούν με αντικείμενα του πραγματικού χώρου, κάθως στην πραγματικότητα αλληλεπιδρούν με το ‘πλέγμα’ (mesh), το οποίο είναι ένα σύνολο κορυφών και πολυγώνων (συνήθως τριγώνων) που αντιπροσωπεύουν το αντικείμενο [77], δρώντας ως το εικονικό του αντίγραφο, προσδίδοντας στο χρήστη την φευδαίσθηση ότι τα αντικείμενα βρίσκονται όντως στο περιβάλλοντα χώρο [78].

Σε αντίθεση με το HoloLens 1, το HoloLens 2 μπορεί να αξιοποιήσει τα δεδομένα που λαμβάνονται από το spatial mapping και, σε συνδυασμό με τεχνητή νοημοσύνη, να σχηματίζει περιοχές (Quads), ομαδοποιώντας πολύγωνα που μπορεί να ανήκουν στο ίδιο πραγματικό αντικείμενο ή/και επιφάνεια (π.χ. τοίχος, πάτωμα, τραπέζι κ.λ.π.) (Σχήμα 2.15). Με αυτό τον τρόπο, τα δεδομένα αποκτούν μια δομή, καθιστώντας τα πιο κατανοητά στον προγραμματιστή, που επιθυμεί να τα χρησιμοποιήσει [79]. Αυτό επιτυγχάνεται με το Scene Understanding SDK, το οποίο, πέρα των πλεονεκτημάτων που περιγράφηκαν προηγουμένων, φέρει και ορισμένα μειονεκτήματα σε σχέση με το spatial mapping, όπως είναι η καθυτσέρηση ανανέωσης των δεδομένων, καθώς τα δέδομενα που αξιοποιεί είναι στατικά [78].

Οι κυριότεροι λόγοι χρήσης της τεχνολογίας της χωρικής χαρτογράφησης είναι [78]:



Σχήμα 2.14: Τριασδιάστατη ανάπαρασταση ένος χώρου (Πηγή: learn.microsoft.com)

- Η τοποθέτηση εικονικών αντικειμένων σε πραγματικές επιφάνειες.
- Η απόκρυψη εικονικών αντικειμένων, όταν βρίσκονται πίσω από πραγματικές επιφάνειες.
- Η εφαρμογή φυσικής σε εικονικά αντικείμενα. Με αυτό τον τρόπο, τα ολογράμματα μπορούν να αλληλεπιδρούν με πιο ρεαλιστικό τρόπο με τα πραγματικά αντικείμενα.
- Η πλοήγηση εικονικών αντικειμένων, καθώς και χρηστών, στον πραγματικό χώρο.

Τέλος, πρέπει να υπογραμμιστεί ότι υπάρχει ένας αριθμός παραγόντων, οι οποίοι μπορεί να επηρέασουν την αποτελεσματικότητα και την ακρίβεια της χωρικής χαρτογράφησης. Τέτοιο παράγοντας είναι ο φωτισμός του χώρου, η έλλειψη ύπαρξης αναγνωρίσμων χαρακτηριστικών πάνω σε απλές επιφάνειες, η ύπαρξη wormholes (σημεία του χώρου που φέρουν τα ίδια ή παρόμοια χαρακτηριστικά, με αποτέλεσμα, να θεωρούνται, εσφαλμένα, από τη συσκευή ως ο ίδιος χώρος), οι συγχέσιμες και κινήσεις στο χώρο, οι ανακλαστικές επιφάνειες κ.α. [80].



Σχήμα 2.15: Περιοχές (Quads) που σχηματίστηκαν από το Scene Understanding (Πηγή: learn.microsoft.com)

2.3.4 Χωρικός Ήχος (Spatial Audio)

Η συσκευή Microsoft HoloLens 2 ενσωματώνει, επίσης, την τεχνολογία χωρικού ήχου, η οποία συνεισφέρει ακόμη περισσότερο στη μίξη του πραγματικού και του εικονικού κόσμου, προσφέροντας μια ρεαλιστική εμπειρία στο χρήστη. Η τεχνολογία βασίζεται στην ικανότητα του ανθρώπου να αντιληφθεί την θέση και την κατεύθυνση από την οποία προήλθε κάποιος ήχος. Με τη χρήση των δύο ηχείων, τα οποία βρίσκονται εκατέρωθεν του κεφαλιού του χρήστη, και τεχνολογίας βασισμένη στη Head-related συνάρτηση μεταφοράς (Head-related Transfer Function, HRTF), είναι εφικτό η συσκευή να παράγει ήχο, για τον όποιο, θα είναι εφικτό ο χρήστης να υποδείξει την κατεύθυνση προέλευσης αυτού [81]. Η HRTF αποτελεί το λόγο του μετασχηματισμού Fourier της πίεσης του ήχου στην είσοδου του καναλιού του αυτιού και της πίεσης αυτής στο μέσο του κεφαλιού [82]. Ανάλογα με το σχήμα και το μέγεθος του αυτιού του χρήστη, η συσκευή μπορεί να προσαρμόσει την HRTF με σκοπό να βελτιώσει την εμπειρία του χρήστη. Το spatial audio χρησιμοποιείται κυρίως για να τραβήξουμε την προσοχή του χρήστη προς κάποια συγκεκριμένη κατεύθυνση ή κάποιο συγκεκριμένο ολόγραμμα, το οποίο μπορεί να βρίσκεται εκτός του πεδίου ορατότητάς του [81].

2.4 Εργαλεία ανάπτυξης για HoloLens

Για την ανάπτυξη εφαρμογών για την συσκευή Microsoft HoloLens 2, είναι απαραίτητη η εγκατάσταση συγκεκριμένων προγραμμάτων και λογισμικών από μια λίστα εργαλείων που παρέχει η Microsoft [83][84]. Η επιλογή των εργαλείων γίνεται από τον προγραμματιστή ανάλογα με την εξικείωση του με τα εργαλεία αυτά και τις γλώσσες προγραμματισμού που χρησιμοποιούν, καθώς και το διαθέσιμο υλικό (documentation). Στο παρόν κεφάλαιο, θα δοθεί μια σύντομη περιγραφή των εργαλειών που θα αξιοποιηθούν για την ανάπτυξη της εφαρμογής, καθώς και των δυνατότητων που παρέχουν.

2.4.1 Unity

Η Microsoft δίνει την δυνατότητα στον προγραμματιστή να επιλέξει μεταξύ τεσσάρων μηχανών (software engines) και APIs στην οποία μπορεί να υλοποιήσει την εφαρμογή του [84], τα οποία παρατίθονται στον Πίνακα 2.2, μαζί με τη γλώσσα προγραμματισμού που χρησιμοποιεί κάθε μηχανή ή API. Στα πλαίσια ανάπτυξης της εφαρμογής μας, ως μηχανή επιλέχθηκε η μηχανή γραφικών Unity, λόγω του πλούσιου documentation που παρέχεται από την Microsoft [85] και από την Unity Technologies, καθώς και του υλικού και των οδηγών, που είναι διαθέσιμα στο διαδίκτυο (forums, videos κ.λ.π.) και παρεχόνται από μια ιδιαίτερα ενεργή κοινότητα.

Η Unity είναι μια μηχανή γραφικών, η οποία αναπτύχθηκε από

Engine και API	Γλώσσα Προγραμματισμού
Unity (Engine)	C#
Unreal Engine	C++
Custom Engine με χρήση του OpenXR API	C/C++
Web development με χρήση του WebXR API	JavaScript

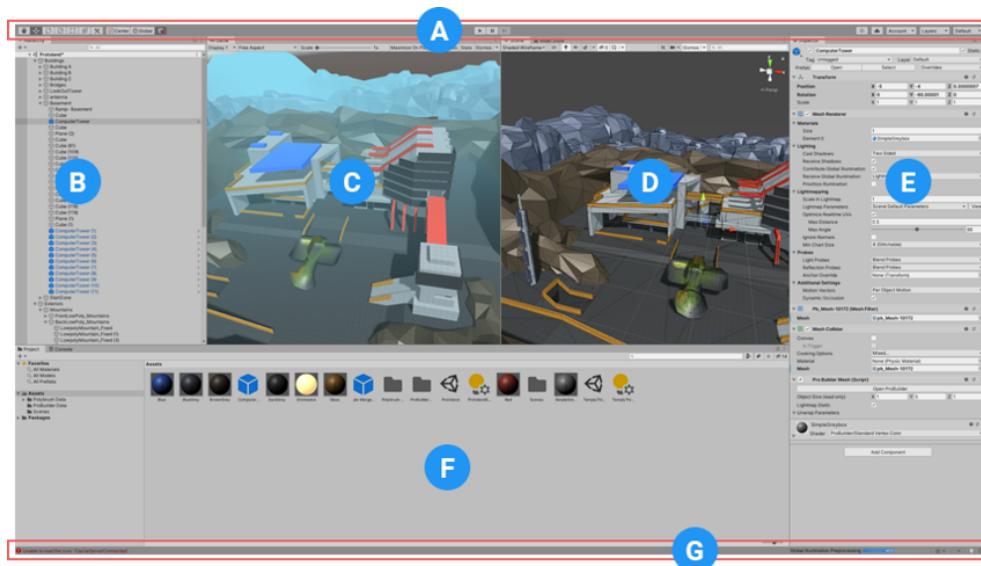
Πίνακας 2.2: Διαθέσιμα engines και APIs για την ανάπτυξη εφαρμογών για το Microsoft HoloLens 2

την Over the Edge Entertainment (η οποία μετανομάστηκε σε Unity Technologies το 2007) και κυκλοφόρησε στις 8 Ιουνίου 2005. Η μηχανή, αρχικά, αναπτύχθηκε για την υλοποίηση εφαρμογών στο λειτουργικό σύστημα Mac OS X, ενώ στη συνέχεια έγινε δυνατή η ανάπτυξη σε πληθώρα πλατφορμών, όπως οι κινητές συσκευές (Android, iOS), υπολογιστές (Mac, Windows, Linux), παιχνιδοκονσόλες, πλατφόρμες εικονικής/εκτεταμένης πραγματικότητας κ.α. Η Unity επιτρέπει την ανάπτυξη τριάσδιάστατων (3D), καθώς και δισδιάστατων (2D) εφαρμόγων και χρησιμοποιείται σε κλάδους και βιομηχανίες, όπως η βιομηχανία βιντεοπαιχνιδιών, κινηματογράφου, ο κλάδος της αρχιτεκτονικής κ.α. Η τρέχουσα LTS έκδοση της μηχανής είναι η Unity 2023.2.1. Ωστόσο, για λόγους συμβατότητας με άλλα εργαλεία που θα χρησιμοποιηθούν, κατά την ανάπτυξη της εφαρμογής προτιμήθηκε η έκδοση Unity 2020.3.48.

Η μηχανή γραφικών Unity αναπτύχθηκε σε γλώσσα προγραμματισμού C++ (runtime) και αξιοποιεί τη C# (Unity Scripting API), μια ανιτκειμενοστρεφής γλώσσα προγραμματισμού, η οποία αναπτύχθηκε από τη Microsoft. Αποτελεί εφαρμογή υψηλού επιπέδου, δίνοντας στον προγραμματιστή τη δυνατότητα να υλοποιήσει μια εφαρμογή με ελάχιστη γραφή κώδικα (scripts), διαθέτοντας έτοιμες προς χρήση λειτουργίες και αντικείμενα, των οποίων η συμπεριφορά μπορεί να παραμετροποιηθεί.

Εφόσον η γλώσσα προγραμματισμού, που μπορεί να χρησιμοποιήσει ο προγραμματιστής για να αναπτύξει λειτουργίες, που δεν παρέχονται από την μηχανή, είναι η ανιτκειμενοστρεφής C#, τότε βασικό στοιχείο της Unity είναι η χρήση των κλάσεων (Classes) και των αντικειμένων (Objects), με κυριότερη κλάση να είναι το [GameObject](#) [86]. Όλα τα στοιχεία που μπορούμε να εντοπίσουμε σε μια εφαρμογή, όπως οι χαρακτήρες, η κάμερα, ο φωτισμός, αποτελούν αντικείμενο της κλάσης [GameObject](#). Από μόνα τους, τα αντικείμενα αυτά δεν διαθέτουν κάποια ιδιαίτερη λειτουργία ή συμπεριφόρα, αλλά λειτουργούν ως ‘άδεια δοχεία’, στα οποία προστίθενται [Components](#) [87]. Τα [Components](#) αποτελούν τα λειτουργικά μέρη των [GameObjects](#), κάθως είναι αυτά τα οποία καθορίζουν τη συμπεριφορά τους. Κάθε [GameObject](#) μπορεί να διαθέτει άπειρα [Components](#), τα οποία διαθέτουν παραμέτρους, τις οποίες ο προγραμματιστής μπορεί να προσαρμό-

σει ανάλογα με την επιθυμητή συμπεριφορά. Ωστόσο όλα τα **GameObjects** διαθέτουν πάντα ένα και μόνο ένα **Transform** component, το οποίο καθορίζει τη θέση, περιστροφή και κλίμακα του αντικειμένου. Επιπλέον, είναι δυνατή η δημιουργία custom **Component** από τον προγραμματιστή με τη χρήση του Unity Scripting API, δηλαδή με τη συγγραφή αρχείων κώδικα σε C# (C# scripts) [88]. Ένα **GameObject**, επίσης, μπορεί να έχει πολλαπλά ‘children’ **GameObjects**, το καθένα με τα δικά του **Components** και παραμετροποιήσεις. Αν ο χρήστης θέλει να χρησιμοποίησει ένα τέτοιο σύνολο από **GameObjects** με τα **Components** που διαθέτουν, τότε, για λόγους ευκολίας, μπορεί να δημιουργήσει ένα **Prefab**, με βάση το οποίο δημιουργεί πολλαπλά αντίγραφα με κοινή συμπεριφορά.



Σχήμα 2.16: Το περιβάλλον ανάπτυξης (Editor) της Unity (Πηγή: docs.unity3d.com)

Η βασική διεπαφή της Unity, όπως μπορούμε να παρατηρήσουμε και στο Σχήμα 2.16, αποτελείται από 7 κύριες περιοχές/παράθυρα. Ειδικότερα, οι περιοχές αυτές είναι [89]:

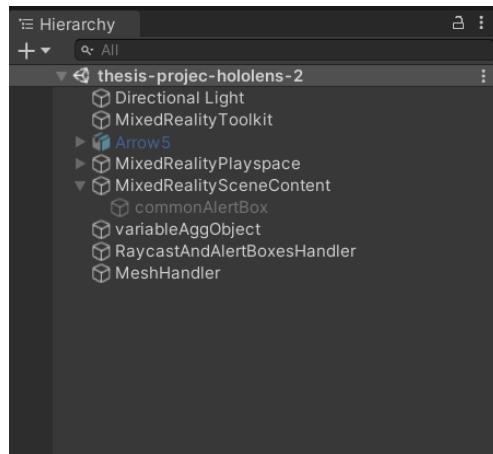
(A) Toolbar: Το toolbar σου δίνει πρόσβαση σε ένα πλήθος εργαλείων για [90]:

- την διαχείριση των **GameObjects** στο Scene view
- την διαχείριση της ροής στο Game view
- τον έλεγχο και την διαχείριση του λογαριασμού Unity του χρήστη
- την επιλογή του Layer, όπου τα αντικείμενα που ανήκουν σε αυτόν θα είναι ορατά στο Scene view
- την επιλογή της διάταξης των παραθύρων στη διεπαφή

(B) Hierarchy Window: Το παράθυρο αυτό (Σχήμα 2.17) παρουσιάζει, σε μορφή λίστας, τα **GameObjects** που βρίσκονται σε μια σκηνή (Scene)

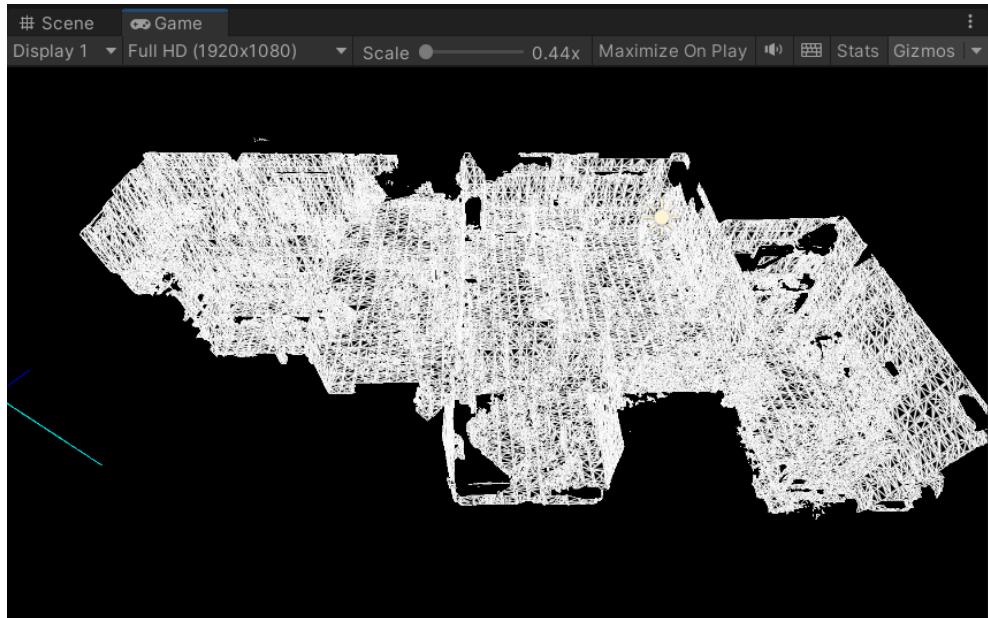
και τη δομή αυτών. Σκηνή (Scene) αποτελεί το αρχείο που περιέχει το περιβάλλον, το μενού και τα αντικείμενα της εφαρμογής.

Επιπλέον, έχουμε τη δυνατότητα να αλλάξουμε τη σειρά και το ‘parenting’ των **GameObjects**, καθώς και να δημιουργήσουμε, να αντιγράψουμε ή να διαγράψουμε ένα **GameObject**. Τέλος, μπορούμε και να επιλέξουμε ποια **GameObjects** θα είναι εμφανή στο Scene view [91].



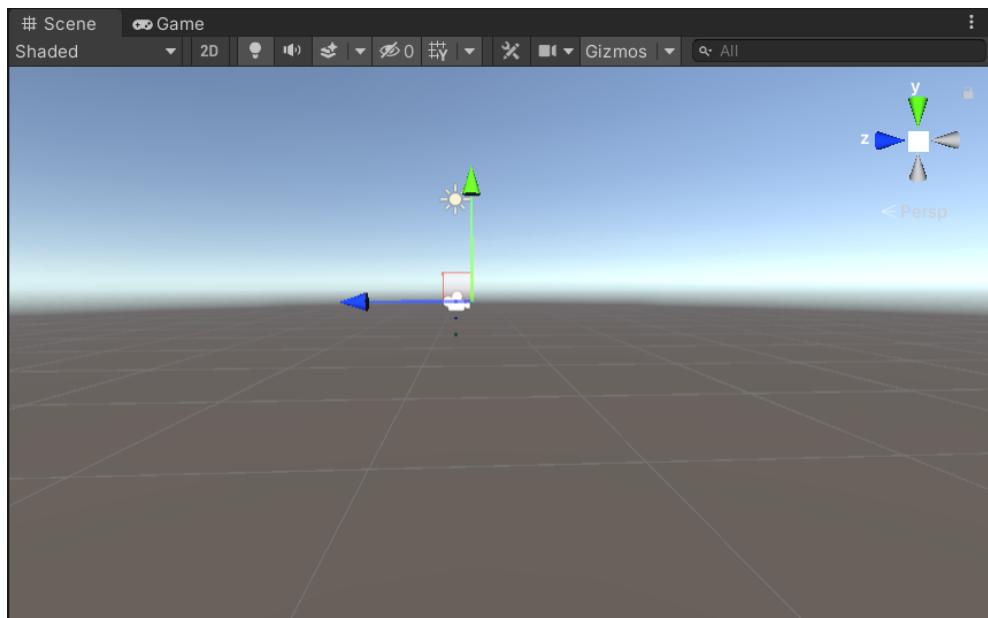
Σχήμα 2.17: Το Hierarchy window του Editor

- (C) Game View: Το Game view (Σχήμα 2.18) αποτελεί το παράθυρο στο οποίο παρουσιάζεται η τελική μορφή την οποία έχει η εφαρμογή μας, όπως αυτή γίνεται render από τις κάμερες (**Camera** component), που έχουμε στη σκηνή. Από τα κομπονέντρα που βρίσκονται στο toolbar μπορούμε να ορίσουμε τη ροή εκτέλεσης της εφαρμογής (έναρξη, παύση, τερματισμός κ.α.) στο Play mode. Play mode ονομάζεται η λειτουργία κατά την οποία εκτελείται η εφαρμογή και παρουσιάζεται στο Game view. Κατά την λειτουργία, μπορούν να πραγματοποιηθούν αλλαγές σε διάφορα στοιχεία, όπως **GameObjects**, παραμέτρους από **Components** κ.α., ωστόσο οι αλλαγές αυτές είναι προσωρινές και αναιρούνται, όταν ολοκληρωθεί η εκτέλεση της εφαρμογής. Τέλος, στην κορυφή του παραθύρου, υπάρχει μια εργαλειοθήκη ώστε να προσαρμόσει ο χρήστης το Game view παράθυρο και το περιεχόμενο αυτού [92].
- (D) Scene View: Αποτελεί το παράθυρο (Σχήμα 2.19), οπου ο χρήστης μπορεί να αλληλεπιδράσει με τα Game Objects τα οποία έχει εισάγει σε μια σκηνή. Στη Scene view, έχουμε μια τρισδιάστατη ή δισδιάστατη προοπτική της εφαρμογής, όπως αυτή εμφανίζεται πριν την εκτέλεση κάποιου κώδικα. Τέλος, όπως και στο Game view, στην κορυφή του παραθύρου, υπάρχει μια εργαλειοθήκη όπου ο χρήστης μπορεί να τροποποιήσει το Scene view με βάση τις ανάγκες, όπως είναι η απόκρυψη των Gizmos, η αφαίρεση του φωτισμού και η αλλαγή από 3D



Σχήμα 2.18: To Game view του Editor

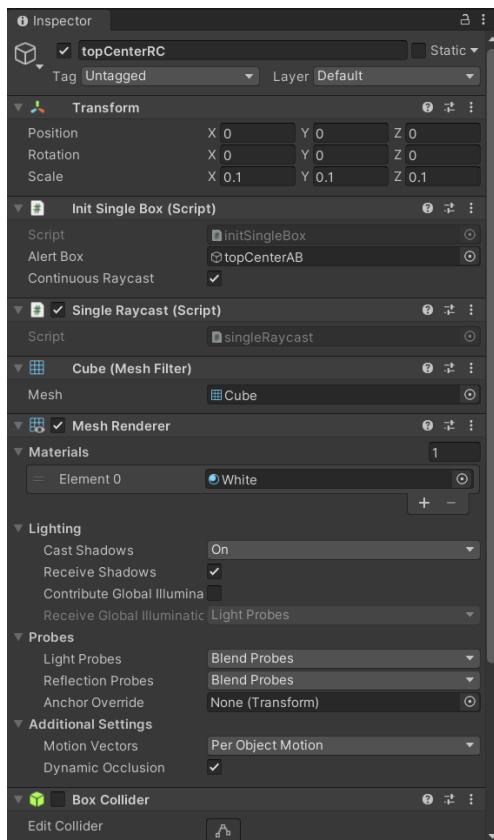
σε 2D προοπτική και αντίστροφα [93].



Σχήμα 2.19: To Scene view του Editor

- (E) Inspector Window: Από το παρόθυρο αυτό (Σχήμα 2.20), ο χρήστης μπορεί να δει και να τροποποιήσει παραμέτρους πολλών στοιχείων του Editor, όπως είναι τα [GameObjects](#), τα [Components](#), τα υλικά

(Materials), τα Assets, καθώς και ρυθμίσεις του Editor. Επιλέγοντας κάποιο από τα ανωτέρω στοιχεία, στον Inspector, εμφανίζονται όλοι οι παράμετροι, σχετικές με το στοιχείο αυτό, τις οποίες μπορεί να αλλάξει ο χρήστης. Για παράδειγμα, αν επιλέχθει ένα `GameObject`, τότε εμφανίζονται όλα τα `Components` που έχουν προστεθεί σε αυτόν. Αν κάποιο από τα `Components` αποτελεί Script που έχει δημιουργήσει ο χρήστης, τότε στον Inspector παρουσιάζονται οι public μεταβλητές του αρχείου αυτού [94].

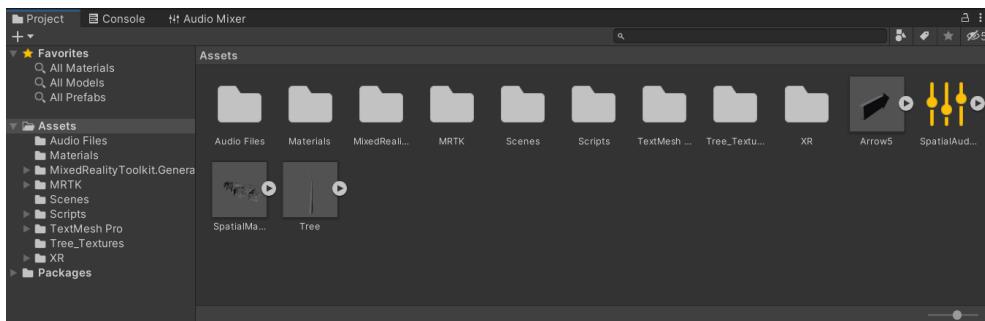


Σχήμα 2.20: Το Inspector window του Editor

(F) Project Window: Στο συγκεκριμένο παράθυρο εντοπίζονται όλα τα αρχεία, τα οποία σχετίζονται με την εφαρμογή, που υλοποιεί ο χρήστης. Ο χρήστης μπορεί να ανατρέξει στο παράθυρο αυτό σε περίπτωση που αναζητεί κάποιο Asset, το οποίο έχει ξαναχρησιμοποιήσει ή έχει αποθηκεύσει σε κάποιο φάκελο του project [95].

Το Asset είναι οποιοδήποτε αντικείμενο χρησιμοποιείται κατά την ανάπτυξη της εφαρμογής, όπως είναι ένα αρχείο ήχου, μια εικόνα, ένα 3D μοντέλο [96]. Αυτά τα assets μπορούν να έχουν δημιουργηθεί από το χρήστη ή από ένα τρίτο άτομο και έχουν αποκτήθει από το χρήστη από καταστήματα, όπως το Unity Asset Store.

Τέλος, όπως και σε άλλες περιοχές του Editor, στην κορυφή του παραθύρου υπάρχει μια εργαλειοθήκη, που επιτρέπει στο χρήστη να προσθέσει Assets στον τρέχοντα φάκελο ή να αναζητήσει κάποιο Asset με χρήση διάφορων φίλτρων [95].



Σχήμα 2.21: To Project window του Editor

(G) Status Bar: Στο κάτω μέρος του Editor εντοπίζεται η μπάρα κατάστασης (Status Bar) (Σχήμα 2.22), η οποία ενημερώνει σχετικά με την κατάσταση του Editor ή διαφόρων εργασιών που επιτελεί. Ειδικότερα, παρουσιάζεται μια προεπισκόπηση του τελευταίου μηνύματος, προειδοποιήσης ή σφάλματος που καταγράφηκε στην κονσόλα (Console). Επίσης, αν επιτελείται κάποια εργασία, εμφανίζεται μια μπάρα προόδου της εργασίας αυτής. Τέλος, στο δεξί μέρος του status bar, υπάρχουν ορισμένα κουμπιά και ενδείξεις, που επιτρέπουν στο χρήστη [97]:

- να αλλάξει ανάμεσα σε λειτουργία Debug και Release, το οποίο επηρεάζει την βελτιστοποίηση του κώδικα
- να ελέγξει την κατάσταση του cache server
- να ελέγξει την κατάστηση του Global illumination
- να ελέγξει την τρέχουσα κατάσταση του Editor, σε περίπτωση που αυτός πραγματοποιεί compilation αρχείων C# ή άλλες ασύγχρονες εργασίες



Σχήμα 2.22: To Status bar του Editor

2.4.2 Microsoft Visual Studio

Το Microsoft Visual Studio από το κύριο ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment, IDE) για την ανάπτυξη εφαρμογών στη Unity, όπως είναι οι εφαρμογές για τη συσκευή HoloLens [83]. Κατά τη δημιουργία ενός project στη Unity, δημιουργείται ένα Visual Studio Project. Ο συγκεκριμένος editor μπορεί να χρησιμοποιηθεί για τη συγγραφή κώδικα σε γλώσσα C#, για το compilation των αρ-

χείων κώδικα και το build του project, αλλά και για την αποσφαλμάτωση (debugging) του κώδικα. Τέλος, μέσω του Visual Studio, είναι δυνατό η φόρτωση του project στη συσκευή [98]. Ωστόσο, αυτή είναι μια ιδιαιτέρως χρονοβόρα διαδικασία, η οποία δημιουργεί επιπλέον καθυστερήσεις στη ανάπτυξη της εφαρμογής, όπου οι συχνές αλλαγές του κωδικά και δοκιμές αποτελούν συχνό φαινόμενο. Για το λόγο αυτό, η μεταφόρτωση της εφαρμογής πραγματοποιείται μέσω του εργαλείου Holographic Remoting που προσφέρει το MRTK (Κεφάλαιο 2.4.3) μέσω της Unity.

Στα πλαίσια της παρούσας διπλωματικής εργασίας, εγκαταστάθηκε η έκδοση του προγράμματος Visual Studio 2022, καθώς και τα απαιραίτητα workloads για την ανάπτυξη εφαρμογών μικτής πραγματικότητας στη Unity.

2.4.3 Mixed Reality Toolkit

Το Mixed Reality Toolkit (MRTK) αποτελεί μια εργαλειοθήκη, η οποία αναπτύχθηκε από την Microsoft, ενσωματώνεται σε project στη Unity ως package και έχει ως σκόπο να διευκολύνει τον προγραμματισμό στην ανάπτυξη εφαρμογών μικτής πραγματικότητας, παρέχοντας μια σειρά από εργαλεία. Πιο συγκεκριμένα, παρέχει σημαντικά [GameObjects](#) και [Components](#), τα οποία μπορούν να χρησιμοποιηθούν σε ένα εικονικό περιβάλλον ή σε αντικείμενα που υπάρχουν σε αυτόν.

Επίσης, δίνει τη δυνατότητα στο χρήστη να πραγματοποιήσει μια προσωμοίωση της χρήσης της εφαρμογής μέσω του editor της Unity, χωρίς της φόρτωση του project σε συσκευή HoloLens. Ωστόσο, στην περίπτωση αυτή, δεν είναι πάντα λειτουργικά όλα τα στοιχεία μικτής πραγματικότητας που έχουμε προσθέσει στο project και προσφέρει το MRTK [99].

Όπως αναφέρθηκε, το MRTK προστίθεται ως package σε ένα Unity project. Αυτό με τη χρήση του εργαλείου Mixed Reality Feature Tool, το οποίο αναπτύχθηκε επίσης από τη Microsoft. Μέσω του εργαλείου αυτού, μπορεί ο χρήστης να αναζητήσει και να προσθέσει στο Unity project του εργαλεία και components χρήσιμα για εφαρμογές μικτής πραγματικότητας [100].

Για την ανάπτυξη της εφαρμογής μας, εγκαταστήσαμε, μέσω του Mixed Reality Feature Tool, το MRTK 2.7.3, καθώς και τα packages Mixed Reality OpenXR plugin και Microsoft Spatializer.

ΚΕΦΑΛΑΙΟ

3

Η ΥΛΟΠΟΙΗΣΗ

Σκοπός του κεφαλαίου αποτελεί η παρουσίαση της διαδικασίας ανάπτυξης της εφαρμογής για τη συσκευή Microsoft HoloLens 2. Πιο συγκεκριμένα, θα παρουσιάσουμε αρχικά τον κύριο στόχο της εφαρμογής, ποιος είναι ο σκοπός που εξυπηρετεί και σε ποιο πρόβλημα προσπαθεί να προσφέρει λύση (Κεφάλαιο 3.1). Έπειτα, θα γίνει αναφορά στις αποφάσεις που λήφθηκαν κάτα την σχεδίαση και υλοποίηση της εφαρμογής με βάση και τους περιορισμούς που τέθηκαν (Κεφάλαιο 3.2). Το κεφάλαιο θα ολοκληρωθεί με μια αναλυτική περιγραφή της διαδικασίας υλοποίησης της εφαρμογής και επεξήγηση του κώδικα που αναπτύχθηκε, καθώς και με την παρουσίαση όλων των κύριων λειτουργιών (Κεφάλαιο 3.3).

3.1 Σενάριο Εφαρμογής

Τα άτομα με προβήματα όρασης έρχονται καθημερινά αντιμέτωπα με ένα πρόβλημα, το οποίο, για το μέσο πληθυσμό αποτελεί μια απλή καθημερινή ενέργεια: η απροβλημάτιστη μετακίνηση και η περιήγηση σε δημόσιους και ιδιωτικούς χώρους. Ως τώρα, παρά την τεχνολογική εξέλιξη και την ανάπτυξη σύγχρονων λύσεων, όπως αυτές περιγράφηκαν στο Κεφάλαιο 2.1.4, για την πλειονότητα αυτών των ατόμων, κύριος υποστηρικτής στην προσπάθεια αυτή αποτελεί το μπαστούνι, λόγω της απλότητας χρήσης του και του κόστους του.

Βασικός στόχος της εφαρμογής αποτελεί η παροχή βοήθειας σε τυφλούς και άτομα με προβλήματα όρασης, κατά την περιήγησή τους σε ένα αγνωστό προς αυτούς χώρο με ευκολία και ασφάλεια, χωρίς τη βοήθεια κάποιου επιπλέον οργάνου. Δηλαδή, η εφαρμογή πρέπει να προσφέρει ένα απλό και εύκολα κατανοητό τρόπο λειτουργίας, η οποία θα προσομοιάζει τον τρόπο χρήσης ενός μπαστονιού.

Αρχικά, ο χρήστης φορά το headset μικτής πραγματικότητας Microsoft Hololens 2 και επιλέγει την εφαρμογή που αναπτύξαμε. Κατά την εκκίνηση, καθώς και σε τακτά χρονικά διαστήματα κατά την εκτέλεση της εφαρμογής, πραγματοποιείται χωρική χαρτογράφηση του περιβάλλοντα χώρου. Με αυτό τον τρόπο δημιουργείται ένα mesh, ένα 3D μοντέλο του χώρου.

Στη συνέχεια, καθώς ο χρήστης περπατά στο χώρο, προσπαθούμε να εντοπίσουμε, σε περιοχή γύρω από αυτόν, πιθανά εμπόδια. Σε περίπτωση που η συσκευή ανιχνεύσει κάποιο εμπόδιο, προειδοποιεί τον χρήστη με τρεις διαφορετικούς τρόπους. Αναπαράγει, αρχικά, μια σύντομη ηχητική ειδοποίηση, η οποία διαθέτει ιδιότητες χωρικού ύχου. Έτσι, ο χρήστης μπορεί να αναγνωρίσει την ‘εικονική’ θέση προέλευσης του ύχου, που αποτελεί και θέση του εμποδίου, καθώς και την απόστασή του από αυτήν. Παράλληλα, προβάλλονται οπτικά προειδοποιητικά σήματα στο πεδίο ορατότητας του ατόμου που επιδεικνύουν την θέση του εμποδίου. Τα σήματα αυτά μπορούν να αξιοποιηθούν από άτομα που αντιμετωπίζουν μερική απώλεια όρασης. Τέλος, προαιρετικά, προσφέρεται απτική ανάδραση προς το χρή-

στη, το οποίο του επιτρέπει να αντιληφθεί μόνο την θέση του εμποδίου στον οριζόντιο άξονα, αν θεωρήσουμε σύστημα αξόνων με αρχή τον χρήστη, δηλαδή πόσο αριστερά ή δεξιά του βρίσκεται.

Τέλος, αναπτύχθηκε ένας πρόσθετος τρόπο εντοπισμό εμποδίων. Ο εντοπισμός δε γίνεται σε συγκεκριμένη περιοχή γύρω από το χρήστη, όπως περιογράφηκε προηγουμένως. Αντιθέτως, ο ίδιος προτάσσει τα χεριά του προς την κατεύθυνση που επιθυμεί να ελέγξει.

Τέλος, πρέπει να τονιστεί ότι, λόγω της ιδιαιτερότητας των χρηστών, όλες οι λειτουργίες εντός της εφαρμογής πραγματοποιούνται με χρήση φωνητικών εντολών.

3.2 Σχεδιασμός και Περιορισμοί

Κατά τον σχεδιασμό της εφαρμογής, καθώς και κατά τη διαρκεία υλοποίησης αυτής, υπήρξαν πληθώρα περιπτώσεων, όπου κληθήκαμε να λάβαμε ιδιαιτερα σημαντικές αποφάσεις, που επηρέασαν σημαντικά την πορεία ανάπτυξης, λόγων των συνθηκών και προδιαφραφών που τέθηκαν ή των περιορισμών και δυσκολιών που αντιμετωπίσαμε.

Αρχικά, όσον αφορά την προετοιμασία και την εγκατάσταση των εργαλείων, ήρθαμε αντιμέτωποι με ένα ιδιαιτερα λειψό εγχειρίδιο (documentation) [101], το οποίο παρέχει η Microsoft. Ειδικότερα, το documentation είναι πλούσιο σε πληροφοριές, που βοηθούν προγραμματιστές, οι οποίοι έρχονται πρώτη φορά σε επαφή με το headset, να κατανοήσουν πλήρως έννοιες και τεχνολογίες που αξιοποοούνται από τη συσκευή, καθώς και τον τρόπο λειτουργίας της. Ωστόσο, σε προγραμματιστικό επίπεδο, οι πληροφορίες είναι ελάχιστες, ενώ οι επεξηγήσεις αντικειμένων, κλάσεων και συναρτήσεων είναι ιδιαιτερα ελλιπείς, καθώς περιορίζονται σε επιδερμικές περιογραφές αυτών και την παρουσιασή παραδειγμάτων και σε ένα πολύ μικρό δείγμα αυτών. Παράλληλα, η Microsoft παρέχει μαθήματα (courses) για την πρακτική εκμάθηση των προγραμματιστών στην ανάπτυξη εφαρμογών για τη συσκεύη με τη χρήση του περιβάλλοντος Unity. Όμως, και σε αυτή την περίπτωση, τα courses περιορίζονται στη χρήση έτοιμων projects, όπου η μόνη συμβολή του χρήστη στην ανάπτυξη αυτών είναι οι αλλαγή ορισμένων απλών ρυθμίσεων. Επομένως, κατά τη διάρκεια της ανάπτυξης (development), αναγκαστήκαμε να καταφύγουμε σε οδηγούς (tutorials) και forums, τα οποία αναπτύχθηκαν από τρίτους, από κοινότητες προγραμματιστών που διέθεταν εμπειρία πάνω στη συγκεκριμένη τεχνολογία. Τέλος, ιδιαιτερο πρόβλημα αποτέλεσε η ασυμβατότητα (incompatibility) μεταξύ των διαφορετικών εκδόσεων των εργαλείων που χρησιμοποιήθηκαν (Unity, MRTK, Visual Studio), το οποίο καθυστέρησε την έναρξη της φάσης ανάπτυξης (development phase).

Επιπλέον, λόγω της ιδιαιτερότητας των χρηστών προς τους οποίους απευθύνεται η εφαρμογή, εκ πρώτης όψεως, φαίνεται αρκέτα οξύμωρη η επιλογή μια συσκευής που ενσωματώνει την τεχνολογία Μικτής Πραγματι-

κότητας, όπου, επί τον πλέιστον, οι εφαρμογές απαιτούν από τους χρήστες να αλληλεπιδράσουν με εικονικά αντικείμενα και η όραση αποδεικνύεται εξαιρετικά σημαντική. Παρ' όλα αυτά, η συσκευή επιλέχθηκε λόγω compact σχεδιασμού, ενσωματώναντας πληθώρα αισθητήρων σε μια μικρή, φορητή συσκευή. Επίσης, ο προγραμματιστής έχει τη δυνατότητα να ενσωματώσει φωνητικές εντολές, όπου οι χρήστες μπορούν χρησιμοποιήσουν για να αλληλεπιδράσουν με τις λειτουργίες της εφαρμογής.

Πέρα από τις δυνατότητες που προσφέρει η συσκευή, καθίστοντας την ιδανική επιλογή, θέτει, παράλληλα, και ορισμένους περιορισμούς. Συγκεκριμένα, οι περιορισμοί αφορούν τις συνθήκες του περιβάλλοντος, όπου χρησιμοποιείται η συσκευή [80]. Για την αποδοτική λειτουργία της συσκευής, πρέπει ο χώρος να είναι καλά φωτιζόμενος, να μην είναι ιδιαίτερα, αλλά, αντιθέτως, να διαθέτει πολλά, μοναδικά αντικείμενα, τα οποία θα εξυπηρετήσουν στην καλύτερη αναγνώριση των αντικειμένων και των εμποδίων κατά τη χωρική χαρτογράφηση. Τέλος, οι συχνές αλλαγές και μετακινήσεις στο χώρο, καθώς και η ύπαρξη ανακλαστικών μπορούν να επηρεάσουν αρνητικά το tracking και τη χαρτογράφηση της συσκευής. Για τους ανωτέρω λόγου, αποφασίστηκε η χρήση της εφαρμογής να περιοριστεί σε εσωτερικούς χώρους.

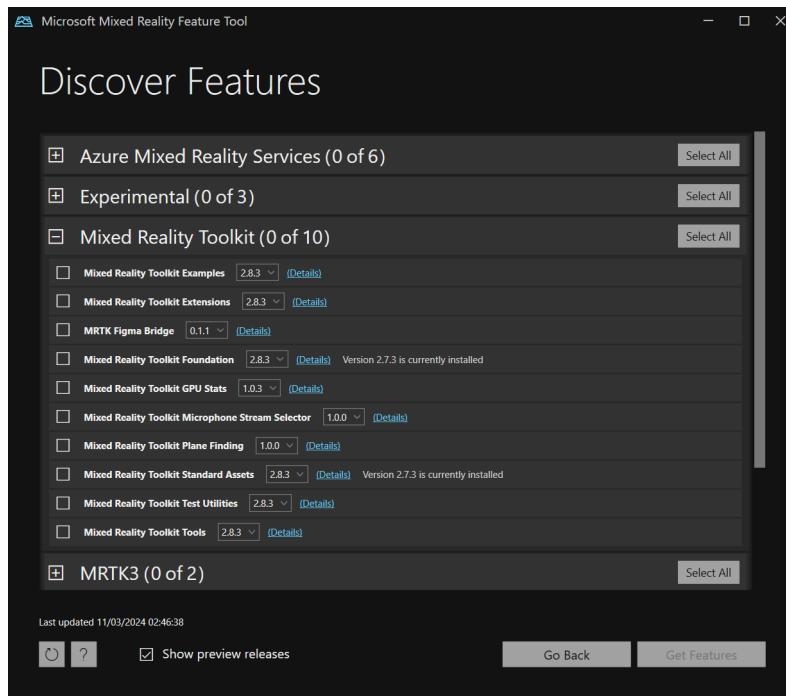
3.3 Υλοποίηση

Στο παρόν κεφάλαιο, θα πραγματοποιηθεί μια αναλυτική περιγραφή όλων των σταδίων της φάσης ανάπτυξης (development phase) της εφαρμογής. Παράλληλα, θα εξηγήσουμε όλες τις λειτουργίες της εφαρμογής μαζί με τη λογική αυτών, παρουσίαζοντας τμήματα του κώδικα που αναπτύχθηκε. Ολόκληρος ο κώδικας της εργασίας βρίσκεται στο Παράρτημα A' μαζί με σχόλια σχετικά με τη λειτουργία κάθε μεθόδου.

3.3.1 Προετοιμασία και Εγκατάσταση

Αρχικά εγκαταστήσαμε όλα τα απαραίτητα εργαλεία, τα οποία συνιστά η Microsoft για την ανάπτυξη εφαρμογών για το headset HoloLens 2. Αφού η συνιστώμενη γλώσσα προγραμματισμού, την οποία και επιλέξαμε, είναι η C#, έπρεπε να εγκαταστήσουμε το IDE Visual Studio, τη μηχανή γραφικών Unity, καθώς και το εργαλείο Mixed Reality Feature Tool (Σχήμα 3.1) για την προσθήκη του MRTK στο project.

Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο (Κεφάλαιο 3.2), ένα σημαντικό πρόβλημα, που δημιούργησε το δύσχρηστο documentation, ήταν η έλλειψη καθοδήγησης όσον αφορά τη συμβατότητα των διάφορων εκδόσεων των προγραμμάτων. Για το λόγο αυτό, δεν επιλέχθηκαν οι πιο πρόσφατες εκδόσεις, αλλά εκδόσεις, οι οποίες χρησιμοποιούνταν κατά κόρον στους διάφορους διαθέσιμους οδηγούς. Ειδικότερα, επιλέχθηκε η Unity 2020.3.48f1 και εγκαταστήθηκε το Visual Studio Community 2019 μαζί με



Σχήμα 3.1: Το Mixed Reality Feature Tool

τα απαραίτητα components για την ανάπτυξη εφαρμογών μικτής πραγματικότητας, καθώς και την τελευταία έκδοση του Mixed Reality Feature Tool (version 1.0.2209.0). Επίσης, για τη διαχείριση των εκδόσεων του κώδικα, που αναπτύξαμε, χρησιμοποιήθηκε το λογισμικό Git, ενώ, για την αποθήκευση αυτού, έγινε χρήση της πλατφόρμας GitHub.

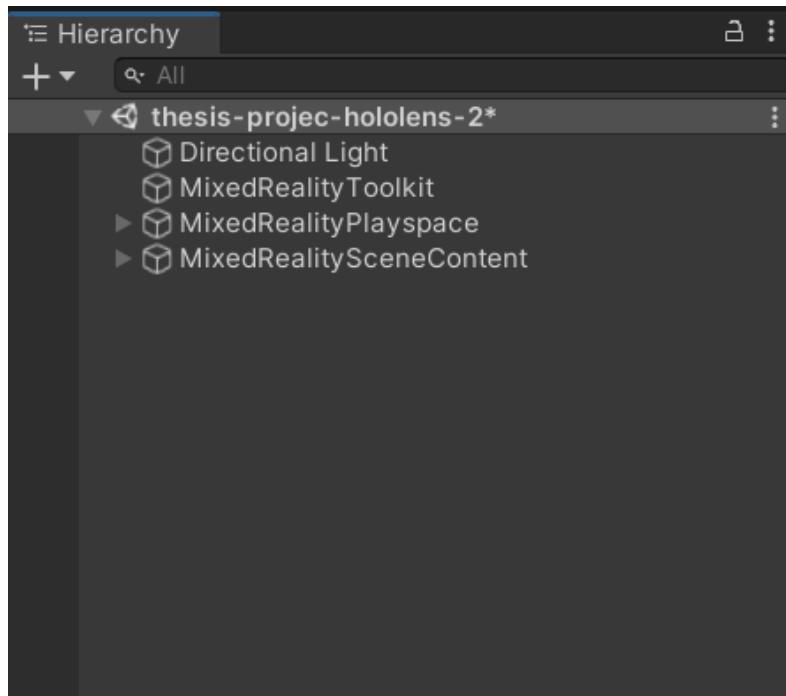
Μετά την ολοκλήρωση της εγκατάστασης, πρώτο βήμα στην ανάπτυξη της εφαρμογής ήταν η δημιουργία ενός project στη Unity, το οποίο διαθέτει μια σκηνή με τα αρχικώς απαραίτητα [GameObjects](#), όπως είναι η Camera, ενώ παρέχει πληθώρα από [Components](#) και Assets. Ωστόσο, αυτές οι ‘πρώτες ύλες’ δεν επαρκούν για την ανάπτυξη εφαρμογών μικτής πραγματικότητας. Για το λόγο αυτό, θα αξιοποιήσουμε το Mixed Reality Feature Tool για την προσθήκη των αναγκαίων packages, όπως αυτά αναφέρονται στον Πίνακα 3.1, με σημαντικότερο από όλα το MRTK.

Mixed Reality Toolkit Foundation	version 2.7.3
Mixed Reality Toolkit Standard Assets	version 2.7.3
Mixed Reality OpenXR Plugin	version 1.4.0
Microsoft Spatializer	version 2.0.37

Πίνακας 3.1: Packages που ενσωματώθηκαν στο project με τη χρήση του Mixed Reality Feature Tool

Με την ενσωμάτωση του MRTK, προστίθενται στη σκηνή ορισμένα βασικά **GameObjects**, ενώ στο project συμπεριλαμβάνονται απαραίτητα components για την βελτίωση της συμπεριφοράς των εικονικών αντικειμένων και την καλύτερη αλληλεπίδραση του χρήστη με αυτά, assets για τη διαμόρφωση του UI (User Interface) της εφαρμογής, καθώς και λειτουργίες, οι οποίες θα διευκολύνουν σημαντικά την αναπτύξη και δοκιμή της εφαρμογής μας [99].

Πλέον, η ιεραρχία των **GameObjects** στη σκηνή διαμορφώνεται με τον τρόπο που φαίνεται στο Σχήμα 3.2.



Σχήμα 3.2: Το hierarchy window μετά την προσθήκη του MRTK

Τα νέα **GameObjects**, που προστέθηκαν, είναι:

- **MixedRealityToolkit**: Αποτελεί το κύριο **GameObject** της εργαλειοθήκης, καθώς διαθέτει όλες τις ρυθμίσεις της, όπως για το σύστημα εισόδου (Input System), την κάμερα, την χωρική χαρτογράφηση και άλλα.
- **MixedRealityPlayspace**: Είναι το **GameObject** που αντιπροσωπεύει το headset στη σκηνή. Εντός αυτού του αντικειμένου, υπάρχει και το αντικείμενο της κάμερας.
- **MixedRealitySceneContent**: Αποτελεί τον χωρό των εικονικών αντικειμένων. Τα αντικειμένα τοποθετούνται εντός αυτού τους **GameObject**, με σκοπό να εξασφαλιστεί η σωστή κλίμακα αυτών, όταν προβάλλονται στον πραγματικό χώρο.

Επιπλέον, το MRTK μας δίνει τη δυνατότητα να κάνουμε προ-

σομοίωση χρήσης της εφαρμογής μέσα από τον Editor της Unity, το οποίο ήταν καθοριστικό στα πρώτα στάδια ανάπτυξης της εφαρμογής για το debugging και τις δοκιμές αυτής. Τέλος, η λειτουργία Holographic Remoting του MRTK επιτρέπει στη Unity να συνδεθεί με το Hololens 2 μέσω του Holographic Remoting Player (Σχήμα 3.3), ώστε το headset να ‘τρέξει’ την εφαρμογή μας και να δοκιμαστεί υπό πραγματικές συνθήκες [102]. Η λειτουργία αυτή αξιοποιήθηκε στα τελευταία στάδια ανάπτυξης, με σκοπό να ελεγχθεί ταχύτατα η λειτουργικότητα της εφαρμογής, αλλά και για να γίνουν οι απαραίτητες αλλαγές σε προβλήματα που δεν θα μπορούσαμε να εντοπίσουμε κατά την προσομοίωση.



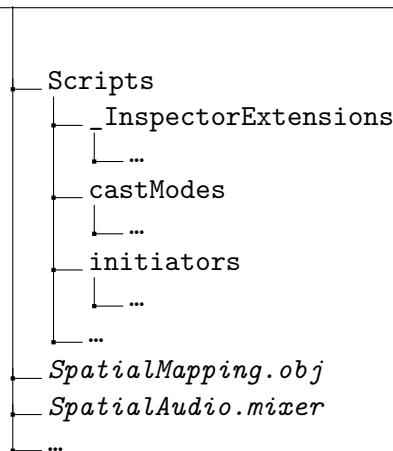
Σχήμα 3.3: Το Holographic Remoting Player σε αναμονή σύνδεσης

3.3.2 Οργάνωση του Project

Σε αυτή το κεφάλαιο, θα πραγματοποιήσουμε μια μικρή περιήγηση στη σκηνή της εφαρμογής μας. Θα παρουσιάσουμε τα **GameObjects** που δημιουργήθηκαν κατά την ανάπτυξη, τον τροπό οργάνωσης αυτών και των **Components**, τα οποία συντάξαμε και θα περιγράψουμε συνοπτικά τη λειτουργία τους. Μια αναλύτική περιγραφή όλων των παραπάνω θα δοθεί στα ακόλουθα κεφάλαια.

Αρχικά, οι φάκελοι του project έχουν την ακόλουθη δομή:

```
Thesis-project-2
└ Assets
    └ Audio Files
        └ ...
    └ Materials
        └ ...
```



Ειδικότερα, στο φάκελο **Audio Files**, τοποθετήθηκαν αρχεία ήχου σε μορφή **.mp3**, ενώ στο φάκελο **Materials**, τοποθετήθηκαν τα υλικά, τα οποία εφαρμόστηκαν στα GameObjects που δημιοργήσαμε και τους προσδίδουν υφή και χρώμα. Ο φάκελος **Scripts**, καθώς και οι υποφακέλοι αυτού, περιέχουν αρχεία κώδικα σε μορφή **.cs** και αποτελούν ουσιαστικά τα **Components** που αναπτύξαμε για τα **GameObjects** μας, αλλά και μερικά αρχεία για την καλύτερη απεικόνιση πληροφοριών στο Inspector window. Τέλος, εντός του φακέλου **Assets** υπάρχει το αρχείο **SpatialMapping**, το οποίο είναι ένα 3D μοντέλο κλειστού χώρου, που χρησιμοποιήθηκε στις προσομοιώσεις και ένα mixer για τον έλεγχο του ήχου.

Σχετικά με τα **GameObjects** και τη θέση αυτών (Σχήμα 3.4), παρατηρούμε ότι ορισμένα από αυτά είναι ‘εμφωλευμένα’, εντός των **MixedRealityPlaySpace** και **MixedRealitySceneContent**. Αυτά αποτελούν τα εικονικά αντικείμενα, τα οποία θα τοποθετούνται και απεικονίζονται στο πραγματικό χώρο. Το ίδιο συμβαίνει και με τα **GameObjects**, τα οποία

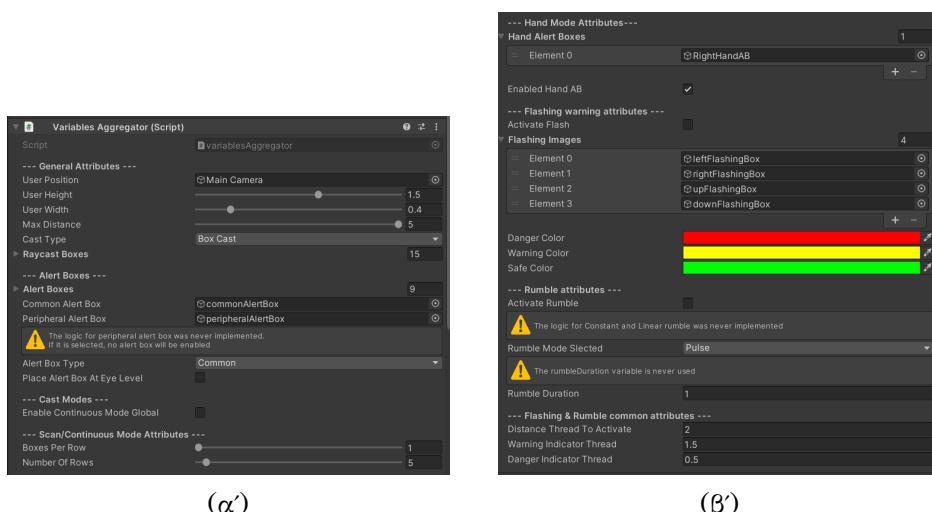


Σχήμα 3.4: Η τελική μορφή του Hierarchy Window

είναι ‘children’ (‘παιδιά’) της **Main Camera** με μόνη διαφορά ότι η θέση τους μεταβάλλεται σύνεχως σύμφωνα με την κίνηση του headset, δηλαδή την κίνηση του κεφαλιού του χρήστη. Τα **GameObjects**, που δεν βρίσκονται εντός των ειδικά διαμορφωμένων **GameObjects** από το MRTK, επιτελούν βοηθητικό ρόλο, καθώς ενσωματώνουν scripts που βοηθούν στη διαχείριση των λειτουργιών της εφαρμογής και των λοιπών **GameObjects** στη σκηνή.

Όσον αφορά την ονοματοδοσία των **GameObjects**, αυτή καθορίζεται από το σκοπό τους. Συγκεριμένα, αντικείμενα, τα οποία χρησιμοποιούνται για την αναπαραγωγή προειδοποιητικών ήχων, περιέχουν τα γράμματα ‘AB’ στο τέλος του ονόματος, ενώ στην αρχή αυτού αναφέρεται η θέση των αντικειμένων αυτών κατά την έναρξη εκτέλεσης της εφαρμογής, σε CamelCase μορφή. Κάτι αντίστοιχο ισχύει και στην περίπτωση των **GameObjects**, τα οποία ενσωματώνουν components για τον εντοπισμό εμποδίων στη διαδρομή του χρήστη. Για την διαφοροποίηση των αντικειμένων αυτών, η κατάληξη της ονομασίας τους είναι ‘RC’. Αντικείμενα, τα οποία επιτελούν βοηθητικό ρόλο, όπως αναφέρθηκε και προηγουμένως, διαθέτουν την λέξη ‘Handler’ στην ονομασία τους (από εδώ και στο εξής τα αντικείμενα αυτά μαζί με τα components τους θα αποκαλούνται ‘handlers’).

Τέλος, ειδική αναφορά πρέπει να πραγματοποιηθεί στο **GameObject variableAggObject**, στο όποιο έχει προστεθεί το component **variablesAggregator.cs** (*Assets/Scripts*). Σκοπός του αρχείου είναι να συγκεντρώνει την πλειοψηφία των κοινών μεταβλητών που χρειάζονται περισσότερα του ενός scripts, ορίζοντας αυτές ως public, δηλαδή να είναι προσβάσιμες από άλλες κλάσεις πέρα αυτής στην οποία ορίστηκαν. Με τον τρόπο αυτό, είναι ευκολότερο για εμάς να διαχειριστούμε ένα μεγάλο αριθμό μεταβλητών, τις οποίες συχνά καλούμαστε να αλλάξουμε πριν ή κατά την εκτέλεση της εφαρμογής για λόγους δοκιμής ή διορθώσεων.



Σχήμα 3.5: To Inspector window για το variableAggObject

3.3.3 Εντοπισμός Εμποδίων

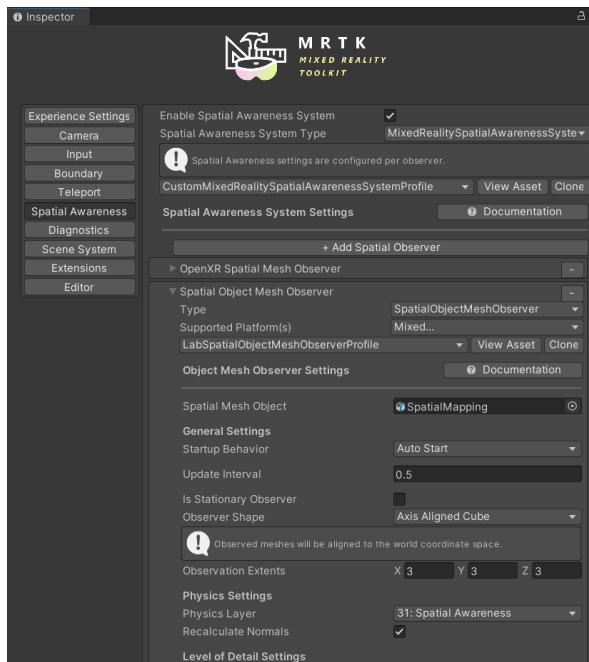
Η σημαντικότερη λειτουργία της εφαρμογής αποτελεί ο εντοπισμός των εμποδίων στη διαδρομή ενός χρήστη. Ο χρήστης μπορεί να περιηγηθεί σε ένα κλειστό χώρο, γνωστό ή άγνωστο προς αυτόν. Η εφαρμογή οφείλει να καταγράψει το χώρο και να αναζητήσει προς την κατεύθυνση πορείας του χρήστη πιθανά εμπόδια. Σε περίπτωση που εντοπίσει κάποιο εμπόδιο, πρέπει να προειδοποιήσει το χρήστη με διάφορους τρόπους (Κεφάλαιο 3.3.4), ώστε αυτός, αναλογά με τη θέση και την απόσταση του εμποδίου, να επιλέξει αν επιθυμεί να αλλάξει πορεία κατεύθυνσης ή να προσπεράσει το εμπόδιο.

Για να επιτευχθεί αυτή η λειτουργία, αρχικά, θα πρέπει να πραγματοποιηθεί μια καταγραφή ο περιβάλλοντος χώρου του χρήστη. Για αυτό θα αξιοποιήσουμε την τεχνολογία της χωρικής χαρτογράφησης (spatial mapping) που προσφέρει το headset. Ειδικότερα, καθώς η συγκεκριμένη λειτουργία είναι ενργοποιημένοι, ο χρήστης περιστρέφει το κεφάλι του μαζί με το headset προς διάφορες κατεύθυνσεις. Με τον τρόπο συλλέγονται πλήθος σημείων, δημιουργόντας ένα νέφος (point cloud) και με βάση τα σημεία αυτά δημιουργείται ένα σύνολο τριγώνων, που αποτελούν το mesh. Διάφοροι παράμετροι σχετικά με τον τρόπο λειτουργίας της χαρτογράφησης μπορούν να τροποποιηθούν.

Η ενεργοποίηση της χωρικής χαρτογράφησης γίνεται μέσα από τις ρυθμίσεις που διαθέτει το **GameObject MixedRealityToolkit**, στην καρτέλα ‘Spatial Awareness’ (Σχήμα 3.6). Αφού ενεργοποιήσουμε τη λειτουργία και δημιουργήσουμε τα custom profiles μας για την αποθήκευση των τροποποιήσεών μας, πρέπει να δημιουργήσουμε ένα ‘Spatial Surface Observer’, το οποίο αποτελεί ουσιαστικά το αντικείμενο που διαχειρίζεται τον τρόπο δημιουργίας και απεικόνισης των meshes [78]. Στην δική μας περίπτωση, χρησιμοποιήσαμε δύο Observers, ένας ο οποίος χρησιμοποιήθηκε κάτα την προσομοίωση της εφαρμογής (Spatial Object Mesh Observer [103]) και ένας ο οποίος χρησιμοποιήθηκε κατά τις δοκιμές της εφαρμογής με χρήση του headset (OpenXR Spatial Mesh Observer). Ωστόσο η πλειονότητα των αλλαγών, που πραγματοποιήθηκαν στο setup αυτών, ήταν ίδιες και στις δύο περιπτώσεις.

Θα περιγράψουμε περιληπτικά τις κύριες παραμέτρους που αλλάζαμε, καθώς και τις τιμές που θέσαμε [104]:

- **Startup Behavior:** Καθορίζει αν η καταγραφή του χώρου θα ξεκινήσει αμέσως μετά την αρχικοποίηση της λειτουργίας ή αν θα καθοριστεί από τον προγραμματιστή. Θέσαμε την τιμή **Auto Start**, ώστε η καταγραφή να αρχίσει με την έναρξη χρήσης της εφαρμογής.
- **Update Interval:** Είναι ο ρυθμός ανανέωσης σε δευτερόλεπταν δεδομένων του mesh. Για την προσομοίωση, επιλέξαμε το mesh να ανανεώνεται κάθε 0.5 δευτερόλεπτα, ενώ στις πραγματικές δοκιμές κάθε 0.1, αφού οι αλλαγές στο χώρο είναι πιο συχνές.



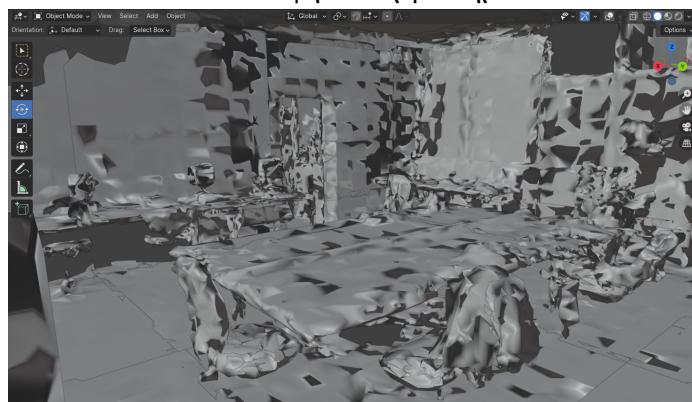
Σχήμα 3.6: Οι ρυθμίσεις της λειτουργίας ‘Spatial Awareness’ και των ‘Spatial Surface Observers’

- **Is Stationary Observer:** Καθορίζει αν ο Observer, που πραγματοποιεί την καταγραφή, θα βρίσκεται σταθερά στο χώρο ή θα κινείται μαζί με τον χρήστη. Επιλέξαμε την τιμή **False**, καθώς ο χρήστης μπορεί να κινείται σε πολλά δωμάτια, τα οποία δεν έχουν καταγραφεί.
- **Observation Extents:** Ορίζουμε τις τιμές **x**, **y**, **z**, οι οποίες είναι οι διαστάσεις του όγκου που αποτελεί την περιοχή καταγραφής. Για την προσομοίωση, θέσαμε την τιμή **5** για όλες τις διαστάσεις, ενώ για τις πραγματικές δοκιμές την τιμή **8**, ώστε να διαθέτουμε περισσότερα δεδομένα του χώρου εκ των προτέρων, δηλαδή πριν προλάβει ο χρήστης να φθάσει στο σημείο.
- **Level of Detail:** Αποτελεί το βαθμό λεπτομέρειας του mesh, δηλαδή την πυκνότητα των σημείων και των τριγώνων. Για την προσομοίωση, θέσαμε την τιμή **Coarse**, ενώ για τις πραγματικές δοκιμές την τιμή **Unlimited**, ώστε να έχουμε όσο το δυνατόν λεπτομέρες πλέγμα.
- **Display Settings:** Η κατηγορία αυτή διαθέτει τρεις μεταβλητές (**Display Option**, **Visible Material**, **Occlusion Material**), οι οποίες καθορίζουν τον τρόπο απεικόνισης του mesh. Και στις δύο περιπτώσεις, το mesh ήταν εμφανές προς τον χρήστη.
- **Runtime Spatial Mesh Prefab:** Πρόκειται για ένα prefab, το οποίο θα χρησιμοποιηθεί ως mesh κατά την εκτέλεση της εφαρμογής. Το prefab αυτό ορίστικε μόνο στην περίπτωση της προσομοίωσης, όπου θέσαμε το αρχείο **SpatialMapping.obj** (**Assets**), που αποτελεί ένα 3D μοντέλο

του χώρου του εργαστηρίου (Σχήμα 3.7), το οποίο καταγράφηκε από το headset HoloLens 2.



(α') Κάτοψη του εργαστηρίου



(β') Εσωτερική απεικόνιση του κύριου χώρου του εργαστηρίου

Σχήμα 3.7: Απεικόνιση του εργαστηρίου με 3D μοντέλο

Τέλος, δημιουργήθηκε ο handler **meshHandler**, στον οποίο προσθέσαμε το Component *hideMesh.cs* (*Assets/Scripts*), όπου μπορούμε με χορήση φωνητικών εντολών να κρύψουμε ή να εμφανίσουμε το mesh.

Αφού ολοκληρώσαμε τη διαδικασία χωρικής χαρτογράφησης του χώρου (Σχήμα 3.8), πρέπει να υλοποιήσουμε τη λογική για την ανίχνευση των εμποδίων με βάση το mesh που διαθέτουμε. Θα αξιοποιήσουμε την κλάση **Physics** του Physics module της Unity, το οποίο χρησιμοποιείται για την εφαρμογή φυσικής σε 3D αντικείμενα. Η κλάση αυτή διαθέτει ποικίλες μεθόδους, οι οποίες μας επιτρέπουν να ανιχνεύσουμε τη σύγκρουση δύο αντικείμενων [105]. Συγκεκριμένα, θα προσπαθήσουμε να εντοπίσουμε το σημείο τομής μιας ακτίνας (ray), που θα αποστέλλουμε από τη θέση του χρήστη, με κάποιο σημείο του mesh.

Οι μέθοδοι αυτοί είναι τα **RayCast**, **BoxCast**, **CapsuleCast**, **SphereCast**, των οποίων η διαφορά τους έγκειται στο σχήμα της ακτίνας, η οποία ‘εκπέμπεται’. Στα πλαίσια της εφαρμογής μας, χρησιμοποιήθηκαν δύο μέ-



Σχήμα 3.8: Χωρική χαρτογράφηση του χώρου του εργαστηρίου και απεικόνιση του mesh

θοδοι, το [RayCast](#), που χρησιμοποιήθηκε στην αρχή του development, και το [BoxCast](#), που χρησιμοποιείται στην τελική έκδοση της εφαρμογής. Η δεύτερη μέθοδος επιλέχθηκε λόγω του σχήματος της ακτίνας και του μεγέθους, το οποίο μπορούμε να το ορίσουμε έτσι ώστε να καλύπτει μια ικανοποιητική επιφάνεια μπροστά από το χρήστη. Η μέθοδος δέχεται ως παραμέτρους την αρχική θέση από όπου θα εκπεμθεί η ακτίνα, την κλίμακα αυτής, καθώς έχει το σχήμα κύβου, την περιστροφή της, την κατεύθυνση εκπομπής και τη μέγιστη απόσταση, όπου μέχρι αυτή θα γίνεται έλεγχος επαφής της ακτίνας με κάποια επιφάνειας που διαθέτει collider. Οι ίδιες παραμέτροι χρειάζεται και η κλήση της μεθόδου [RayCast](#), με μόνη εξαίρεση ότι δεν είναι απαραίτητη η κλίμακα και η περιστροφή της ακτίνας, αφού αυτή είναι μονοδιάστατη. Τέλος, η μέθοδος επιστρέφει ένα flag σχετικά με το αν υπήρξε επαφή με αντικείμενο και πληροφορίες σχετικά με αυτή τη σύγκρουση. Στην περίπτωση που υπήρξε επαφή με περισσότερα του ενός αντικείμενα, τότε επιστρέφονται πληροφορίες σχετικά με την πρώτη επαφή, καθώς αυτή θα είναι και η πλησιέστερη από το σημείο έναρξης (Origin Point).

Όπως αντιλαμβανόμαστε από τις παραμέτρους που απαιτούνται από τις δύο μεθόδους, πρέπει να όρισουμε ένα σημείο έναρξης, από όπου θα ξεκινήσει η εκπομπή των ακτινών. Ως σημείο έναρξης ορίζουμε τη θέση του χρήστη, η οποία πρέπει να αλλάζει όταν αυτός θα μετακινείται. Η κατεύθυνση, που επιλέχθηκε, είναι αυτή προς την οποία κοιτά ο χρήστης, καθώς, συνήθως, αυτή είναι και η κατεύθυνση προς την οποία κινείται. Δεδομένου των ανωτέρω, τοποθετήσαμε ορισμένα [GameObjects](#) ως children του [GameObject](#) Main Camera, διότι, σε αυτή την περίπτωση, μαζί με τη θέση και την περιστροφή της κάμερας, που ακολουθεί την κίνηση του κεφαλιού, ανανεώνονται και η θέση και η περιστροφή των [GameObjects](#). Η ονομασία των αντικειμένων αυτών διαθέτει την κατάληξη ‘RC’ και όλα τους ενσω-

ματώνουν δύο components, το *initSingleBox.cs* (*Assets/Scripts/initiators*) και το *singleRaycast.cs* (*Assets/Scripts*). Το πλήθος των origin points, επομένως και των ακτινών, το μέγεθος και η διάταξη αυτών μπορεί να παραμετροποιηθεί από τις public μεταβλητές του **variableAggObject** (Κώδικας 3.1), ενώ η διαδικασία τοποθέτησης και διαμόρφωσης των διαστάσεων πραγματοποιείται από τη συνάρτηση *positionAndScaleBoxes* (Κώδικας 3.2), η οποία καλείται κατά την εκτέλεση της εφαρμογής και ανήκει στο component **Init Boxes** (*Assets/Scripts/initiators*), το οποίο και ενσωματώνεται στο **GameObject RaycastAndAlertBoxesHandler**. Έπειτα από δοκιμές, επιλέχθηκε η χρήση πέντε σημείων έναρξης εκπομπής ακτινών, τοποθετημένα κάθετα, με τέτοιο τρόπο και διαστάσεις, όπου το πρώτο origin point βρίσκεται στο ύψος των ματιών του χρήστη και το τελευταίο σε ύψος λίγο χαμηλότερο του γονάτου.

```

1   // Variables for debugging purposes
2   [Header("---- General Attributes ---")]
3   public GameObject userPosition;
4   [Tooltip("User's height in meters (m)"), Range(0.0f, 2.5f)]
5   public float userHeight = 0.0f;
6   [Tooltip("User's width in meters (m)"), Range(0.0f, 2.5f)]
7   public float userWidth = 0.0f;
8   [Header("---- Scan/Continuous Mode Attributes ---")]
9   [Tooltip("The number of cast boxes placed in each row"),
10  Range(1.0f, 100.0f)]
11  public int boxesPerRow=3;
12  [Tooltip("The number of cast boxes placed in each column"),
13  Range(1.0f, 100.0f)]
14  public int numberOfRows=3;
```

Κώδικας 3.1: Μεταβλητές για τη διάταξη των origin points και το μέγεθος των ακτινών

```

1   /// <summary>
2   /// A method which places the castBoxes and their alert boxes
3   /// in the correct position based on the parameters provided
4   ///
5   /// </summary>
6   /// <param name="raycstBoxesToPosition">An array of the
7   /// castboxes</param>
8   /// <param name="boxesPerRow">The number of castBoxes which
9   /// will be placed in each row</param>
10  /// <param name="numberOfRows">The number of rows, in which
11  /// the cast boxes will be placed</param>
12  /// <param name="userHeight">The height of the user in meters
13  /// </param>
14  public void positionAndScaleBoxes
15  {
```

```

16     float boxLength = userWidth / boxesPerRow;
17     float boxHeight = userHeight / numberOfRows;
18
19     float boxPlacementX = -boxLength * (boxesPerRow - 1);
20     float boxPlacementY = 0.0f;
21     for (int i = 0; i < raycstBoxesToPosition.Length; i++)
22     {
23         if (i % boxesPerRow == 0)
24         {
25             boxPlacementX = -boxLength * (boxesPerRow - 1);
26
27             if (i != 0) boxPlacementY -= boxHeight;
28         }
29
30         raycstBoxesToPosition[i].transform.position = new
31             Vector3(boxPlacementX, boxPlacementY, -0.1f);
32         raycstBoxesToPosition[i].transform.localScale = new
33             Vector3(boxLength, boxHeight, boxLength);
34
35         raycstBoxesToPosition[i].GetComponent<initSingleBox
36             >().alertBox.transform.localScale = new Vector3(
37                 boxLength, boxHeight, 0.1f);
38
39         boxPlacementX += boxLength * (boxesPerRow - 1);
40     }
41 }
```

Κώδικας 3.2: Η συνάρτηση **positionAndScaleBoxes** για την τοποθέτηση και αλλαγή διαστάσεων αντικειμένων

Σχετικά με τα components, το **Init Single Box** διαθέτει ορισμένες public μεταβλητές που αφορούν κάθε αντικείμενο και χρησιμοποιούνται αποκλειστικά από άλλα components. Αντιθέτως, το script **Single Raycast** περιέχει τη συνάρτηση, η οποία ξεκινά την εκπομπή της ακτίνας και επιστρέφει τις πληροφορίες για το σημείο επαφής αυτής με το mesh (Κώδικας ζ 3.3). Είναι δυνατό επίσης να επιλέξουμε αν θα γίνει χρήση της μεθόδου **RayCast** ή **BoxCast** θέτοντας την αντίστοιχη τιμή στη μεταβλητή **castType** του **variableAggObject**.

```

1   /// <summary>
2   ///     Casts a raycast or boxcast from the position of a
3   ///     castBox and detects a hitPoint.
4   /// </summary>
5   /// <param name="variableAggInstance">The variable aggregator
6   ///     GameObjetc</param>
7   /// <returns>
8   ///     The hitPoint of the cast
9   /// </returns>
10  public RaycastHit SingleRaycastFunc(variablesAggregator
11      variableAggInstance)
12  {
13      Vector3 fwd = transform.TransformDirection(Vector3.
14          forward); // forward direction
15 }
```

```

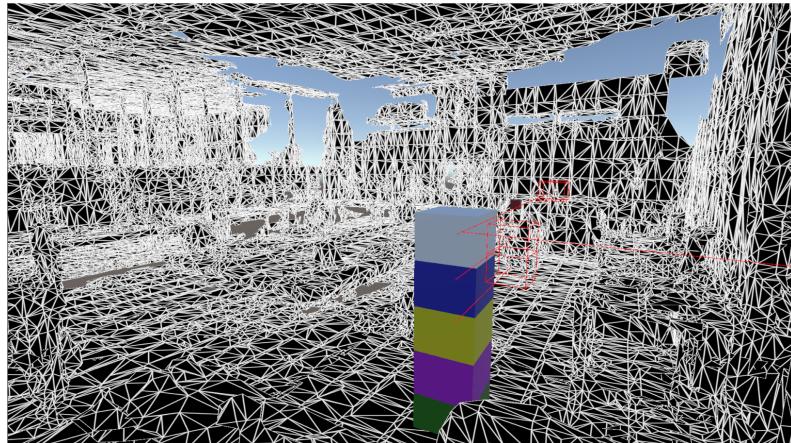
12     float maxDistance = variableAggInstance.maxDistance;
13
14     variablesAggregator.CastTypeEnum castType =
15         variableAggInstance.castType;
16
17     switch (castType) {
18         case variablesAggregator.CastTypeEnum.Raycast:
19             m_HitDetect = Physics.Raycast(transform.position,
20                 fwd, out hitInfo, 5.0f);
21             break;
22
23         case variablesAggregator.CastTypeEnum.BoxCast:
24             m_HitDetect = Physics.BoxCast(transform.position,
25                 transform.localScale, fwd, out hitInfo,
26                 transform.rotation, maxDistance);
27             break;
28     }
29
30     return hitInfo;
31 }
```

Κώδικας 3.3: Η συνάρτηση `SingleRaycastFunc` για τον εντοπισμό εμποδίων

Επομένως, από πέντε σημεία έναρξης, των οποίων η θέση και η περιστροφή επηρεάζεται από την κίνηση του κεφαλιού του χρήστη, εκπέμπονται ισάριθμες ακτίνες σε σχήμα κύβου (μέθοδος `BoxCast`). Αν οι ακτίνες αυτές έρθουν σε επαφή με κάποιο σημείο του πλέγματος, το οποίο ισοδυναμεί με την ύπαρξη εμποδίου προς την κατεύθυνση κίνησης του χρήστη, τότε καταγράφουμε τις πληροφορίες για το σημείο επαφής. Για παράδειγμα, στο Σχήμα 3.9, μπορούμε να παρατηρήσουμε πέντε κύβους, των οποίων τα κέντρα αποτελούν τα σημεία έναρξης. Οι ακτίνες εκπέμπονται από τα κέντρα και οι πορείες τους αναπαρίστανται με κόκκινες γραμμές, οι οποίες καταλήγουν σε έναν κύβο με κόκκινες ακμές σε περίπτωση που υπήρξε επαφή με το πλέγμα της χωρικής χαρτογράφησης.

Τα τελευταία ερωτήματα, τα οποία καλούμαστε να απαντήσουμε, με σκοπό να ολοκληρώσουμε τη λειτουργία εντοπισμού εμποδίων αφορούν τη συχνότητα κλήσης της συνάρτησης `SingleRaycastFunc` και τον τρόπο διαχείρισης πολλαπλών σημείων επαφής. Για τα ερωτήματα αυτά, αναπτύχθηκαν τρεις λύσεις, κάθε μία από τις οποίες έχει τα πλεονεκτήματα και τα μειονεκτήματά της. Κάθε λύση αναπτύχθηκε σε ξεχωριστό component και όλα προστέθηκαν στο `GameObject RaycastAndAlertBoxesHandler`

Η πρώτη, χρονικά, λύση, που αναπτύχθηκε, ονομάστηκε **Scan Mode**. Η εκπομπή των ακτινών ενεργοποιείται ‘χειροκίνητα’, δηλαδή κάθε φορά που ο χρήστης χρησιμοποιεί μια συγκεκριμένη φωνητική εντολή. Τότε, από όλα τα origin points εκπέμπονται τα rays (με μέθοδο `BoxCast`). Συγκεντρώνονται σε ένα `List` όλα τα σημεία επαφής και επιλέγεται το πλησιέστερο (με χρήση της συνάρτησης `findClosestHitPointIndex` του component **Init Boxes**). Η λογική αυτή περιλαμβάνεται στο component



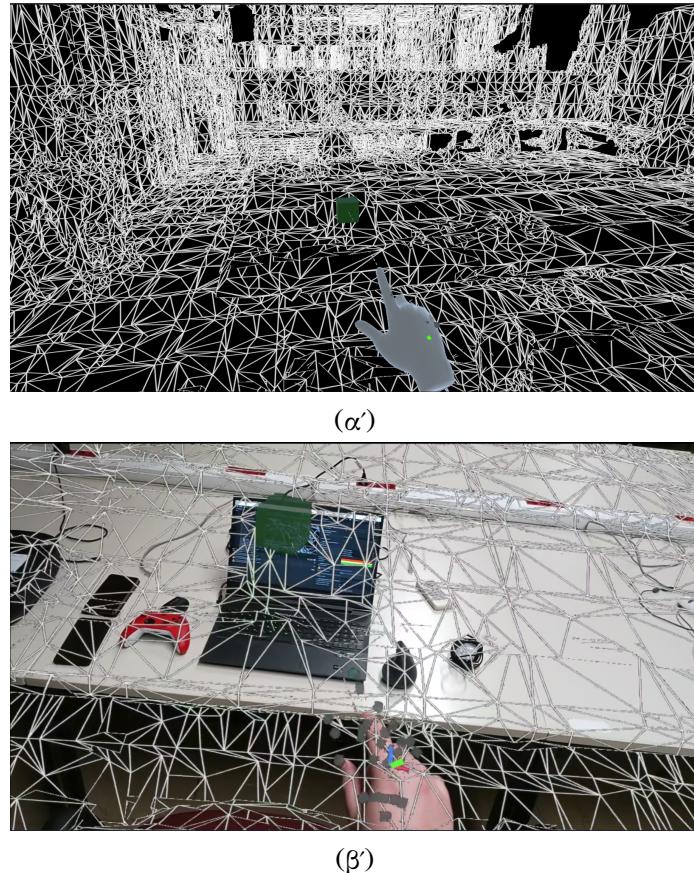
Σχήμα 3.9: Παράδειγμα εντοπισμού εμποδίων με τη μέθοδο BoxCast

Scan Cast (Assets/Scripts/castModes). Με αυτή τη λύση, ο χρήστης μπορεί να διαχειριστεί τη συχνότητα λήψης ειδοποιήσεων για εμπόδια και να τις λαμβάνει, οπότε αυτός κρίνει απαραίτητο. Ωστόσο, αυτός ο τρόπος ενεργοποίησης μπορεί να αποδειχθεί κουραστικός, ενώ, σε περίπτωση αλλαγής του περιβάλλοντα χώρου, μπορεί να είναι και εσφαλμένος.

Η δεύτερη λύση είναι η **Continuous Mode** και το component της είναι το **Continuous Cast (Assets/Scripts/castModes)**. Η λογική σχετικά με την εκπομπή των rays και διαχείριση των πολλαπλών σημείων επαφής είναι με αυτή της προηγούμενης λύσης. Ωστόσο, στην περίπτωση αυτή, ο εντοπισμός εμποδίων πραγματοποιείται σε κάθε frame. Έτσι, ο χρήστης παραμένει πάντα ενήμερος για πιθανά εμπόδια στο χώρο του, ακόμη και πραγματοποιεί κάποια αλλαγή σε αυτόν.

Τέλος, η τρίτη λύση αποτελεί και την πιο ιδιάζουσα. Γλοποιείται εντός του component **Hand Raycast (Assets/Scripts/initiators)** και ονομάζεται **Hands Mode**, διότι το σημείο έναρξης των rays αποτελούν το χέρι του χρήστη. Ειδικότερα, εκμεταλλεύμενοι το γεγονός ότι τα χέρια του χρήστη είναι Pointers [106], που τους επιτρέπουν να αλληλεπιδρούν με εικονικά αντικείμενα σε κοντινή ή μακρινή απόσταση. Στη περίπτωση της αλληλεπίδρασης από μακρινή απόσταση, ο τρόπος εντοπισμού των εικονικών αντικειμένων είναι παρόμοιος με αυτόν που περιγράφηκε στις προηγούμενες λύσεις. Με σημείο έναρξης το χέρι, εκπέμπεται ένα **RayCast**, το οποίο αναζητά σημείο επαφής με άλλα εικονικά αντικείμενα, ώστε να αλληλεπιδράση με αυτά. Επομένως, στο αντικείμενο Pointer αποθήκευονται πληροφορίες για το σημείο επαφής, όπως είναι η απόσταση από το χέρι και οι συντεταγμένες του στο χώρο. Επίσης, πρέπει να τονίσουμε ότι τα αντικείμενα Pointers δημιουργούνται και διατηρούνται κάθε φορά που τα χεριά του χρήστη βρίσκονται εντός του πεδίου ορατότητας (FOV) του headset. Με βάση τα ανωτέρω, κάθε φορά που τα χέρια του χρήστη βρίσκονται εντός του FOV, μπορούμε να αναζητήσουμε το σημείο επαφής των raycasts που

εκπέμπονται από τα χεριά με το mesh που έχει δημιουργηθεί από τη χωρική χαρτογράφηση και, αν το σημείο αυτό βρίσκεται εντός μια συγκεκριμένης απόστασης, να ειδοποιήσουμε το χρήστη για εμπόδιο (Σχήμα 3.10). Ουσιαστικά, τα raycasts λειτουργούν σαν ένα εικονικό μπαστούνι, το οποίο επιστρέφει feedback στο χειριστή του, όταν ‘έρθει σε επαφή’ με εικονικά αντικείμενα, δηλαδή το mesh.



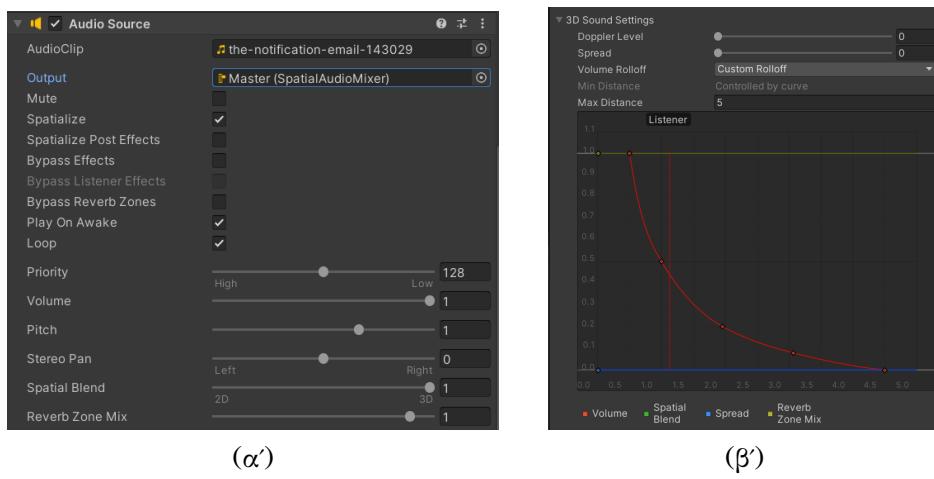
Σχήμα 3.10: Εντοπισμός εμποδίων με τη λειτουργία **Hands Mode**

3.3.4 Προειδοποίηση Χρήστη

Μετά τον εντοπισμό εμποδίων, εξίσου ουσιαστική είναι και η έγκαιρη προειδοποίηση του χρήστη για την ύπαρξη αυτών στη διαδρομή του. Σημαντικό ρόλο στην ανάπτυξη αυτής της λειτουργίας θα παίξει η τεχνολογία χωρικού ήχου (spatial sound), που διαθέτει το headset. Η συγκεκριμένη λειτουργία επιτρέπει τη δημιουργία ενός 3D ήχου, ο οποίος θα βοήθησε το χρήστη να αντιληφθεί τη θέση του στο χώρο και ως προς αυτόν.

Αρχικά, θα πρέπει να τοποθετήσουμε την πηγή ήχου στο σημείο που βρίσκεται το εμπόδιο, έτσι ώστε να μπορέσει ο χρήστης να αντιληφθεί που βρίσκεται αυτό. Βασιζόμενοι στη λογική που αναπτύχθηκε και περιγρά-

φηκε στο Κεφάλαιο 3.3.3, γνωρίζουμε ήδη τη θέση του πλησιέστερου προς το χρήστη σημίου επαφής με το πλέγμα σημείων, το οποίο αντιπροσωπεύει το εμπόδιο. Επομένως, τοποθετούμε στις συντεταγμένες του σημείου αυτού ένα **GameObject**, στο οποίο προσθέτουμε το component **Audio Source**. Με το συγκεκριμένο component, μπορούμε να προσθέσουμε ένα αρχείο, ώστε, από το σημείο όπου τοποθετήσαμε το **GameObject** και το οποίο ονομάσαμε **CommonAlertBox**, να αναπαράγεται ένας προειδοποιητικός ήχος.



Σχήμα 3.11: Το component **Audio Source**

Για να προσδώσουμε στον ήχο τις ιδιότητες ενός χωρικού ήχου, προσθέτουμε ένα **Audio Mixer**, στο οποίο έχουμε ενσωματώσει το Microsoft Spatializer Mixer effect. Επίσης, ενεργοποιούμε στο **Audio Source** το spatialization, θετόντας το flag **Spatialize** σε **True**, ενώ, παράλληλα, θέτουμε την τιμή **1** για τη μεταβλητή **Spatial Blend**. Ορισμένες ακόμη αλλαγές που πραγματοποιήσαμε στο component είναι να ενεργοποιήσαμε τις λειτουργίες **Play On Awake** και **Loop**, ώστε ο ήχος να ενεργοποιηθεί με την έναρξη της εφαρμογής και να μην σταματήσει η αναπαραγώγει μέχρι αυτό να υποδειχθεί από τον κώδικα, αντίστοιχα. Τέλος, διαμορφώσαμε την καμπύλη έντασης του ήχου ως προς την απόσταση του χρήστη από την πηγή, έτσι ώστε ο ήχος να γίνεται αισθητός σε απόσταση **5¹** από την πηγή, να αυξάνεται εκθετικά και να φθάνει το μέγιστο σε απόσταση **1**.

Όσον αφορά την τοποθέτηση της πηγής ήχου και τον έλεγχο αναπαραγωγής αυτού στις διάφορες λειτουργίες εντοπισμού εμποδίων, ισχύουν τα κατώθι:

- Στη λειτουργία **Continuous Mode**, ο εντοπισμός πλησιέστερου εμποδίου πραγματοποιείται σε κάθε ανανέωση του frame. Αν εντοπιστεί κάποιο εμπόδιο, τότε η πηγή τοποθετείται στις συντεταγμένες του σημείου και ξεκινά η αναπαραγωγή του προειδοποιητικού ήχου. Αν

¹Θεωρούμε ότι η αναλογία απόστασης σε μονάδα του περιβάλλοντος Unity προς μέτρα στον πραγματικό κόσμο είναι κατά προσέγγιση 1:1

σε κάποιο update, βρέθει άλλο εμπόδιο, πλησιέστερο προς το χρήστη, τότε η πηγή του ήχου μετακινείται. Αν δεν εντοπιστεί κάποιο εμπόδιο, η αναπαραγωγή του ήχου σταματά.

- Στην περίπτωση του **Scan Mode**, η πηγή ήχου τοποθετείται στο πλησιέστερο εμπόδιο και παραμένει στο σημείο, ακόμη και αν ο χρήστης έχει μετακινηθεί ή αλλάξει κατεύθυνση. Η πηγή ήχου μετακινείται μόνο αν ο χρήστης ενεργοποιήσει ξανά τη λειτουργία με φωνητική εντολή και εντοπιστεί κάποιο νέο εμπόδιο, ενώ, αν δεν εντοπιστεί, τότε ο ήχος απενεργοποιείται.
- Στη λειτουργία **Hands Mode**, η λογική μετακίνησης πηγής και αναπαραγωγής/παύσης του ήχου είναι ίδια με αυτή της λειτουργίας **Continuous Mode**. Επιπλέον, ο χρήστης έχει τη δυνατότητα, προσωρινά, να παύσει την αναπαραγωγή του ήχου για όσο διάστημα πραγματοποιεί το Pinch gesture. Τέλος, ο ήχος απαενεργοποιείται και στην περίπτωση που τα χέρια βρεθούν εκτός του field-of-view του headset.

Αξίζει να αναφερθούμε στο γεγονός ότι υλοποιήθηκε και η ιδέα χρήσης πολλαπλών πηγών και, πιο συγκεκριμένα, μία πηγή για κάθε ray που θα χρησιμοποιούταν για τον εντοπισμό εμποδίων. Αυτή η τεχνική αξιοποίηθηκε στην περίπτωση του **Scan Mode**, ώστε ο χρήστης να είναι ενήμερος για περισσότερα του ενός εμποδίων και να περιορίσουμε τη συχνότητα χρήσης της φωνητικής εντολής για την ενεργοποίηση της λειτουργίας. Ωστόσο, κατά τις δοκιμές, η ταυτόχρονη ενεργοποίηση πολλαπλών πηγών ήχου δυσκόλεψε στη διάκριση της κάθε πηγής ήχου και επομένως της θέσης του εμποδίου, λόγω της ιδιαιτερα κοντινής απόστασης στην οποία τοποθετούνταν.

Τέλος, εκτός από την ηχητική προειδοποίηση του χρήστη, αναπτύχθηκαν δύο ακόμη μέθοδοι: μία οπτική και μία με απτική ανάδραση. Η οπτική προειδοποίηση απευθύνεται σε άτομα που δεν αντιμετωπίζουν πλήρη απώλεια όρασης και μπορούν να αντιληφθούν χρώματα και σχήματα. Σε αυτό τον τρόπο προειδοποίησης, τοποθετούμε τέσσερα σχήματα μπροστά στο οπτικό πεδίο του χρήστη, το καθένα σε μια διαφορετική σχέση: στο πάνω, δεξί, κάτω και αριστερό μέρος του πεδίου του. Όταν εντοπιστεί κάποιο εμπόδιο, ανάλογα με τη θέση του εμποδίου ως προς τη κατεύθυνση που κοιτά ο χρήστης, τα αντίστοιχα σχήματα αρχίζουν να αναβοσβήνουν. Το χρώμα των σχημάτων αλλάζει ανάλογα με την απόσταση του χρήστη από το εμπόδιο. Ξεκινώντας από το πράσινο και όσο πλησιάζει προς το εμπόδιο, το χρώμα αλλάζει σε κίτρινο και τέλος σε κόκκινο. Κάθε σχήμα αποτελεί ένα **GameObject**, το οποίο διαθέτει το component **Flashing Light** (*Assets/Scripts*), που περιέχει τη λογική για το flashing effect και τη θέση του αντικειμένου στο πεδίο ορατότητας του χρήστη. Ο κώδικας για την επιλογή ενεργοποίησης των κατάλληλων αντικειμένων ανάλογα με την θέση των εμποδίων είναι στο component **Init Flash Handler** (*Assets/Scripts/initiators*) του **RaycastAndAlertBoxesHandler**.

Τέλος, αναπτύχθηκε και μέθοδος της απτικής ανάδρασης για την προειδοποίηση του χρήστη. Για τη μέθοδο αυτή, χρησιμοποιήθηκε η συσκευή Microsoft XBOX Series controller (Σχήμα 3.12). Πρόκειται για ένα χειριστήριο, το οποίο επιλέχθηκε λόγω της απλότητας της ασύρματης σύνδεσης αυτού με τον υπολογιστή, για την ανάπτυξη του κώδικα, και του headset, για την πρακτική χρήση αυτού. Επιπλέον, η Unity παρέχει έτοιμες βιβλιοθήκες με κλάσεις, οι οποίες μας επιτρέπουν να αναγνωρίσουμε την σύνδεση/αποσύνδεση του χειριστηρίου, το input του χρήστη μέσω αυτού, καθώς και να ελέγξουμε τμήματα αυτού. Η απτική ανάδραση επιτυγχάνε-



Σχήμα 3.12: Το χειριστήριο Microsoft XBOX Series (Πηγή: microsoft.com)

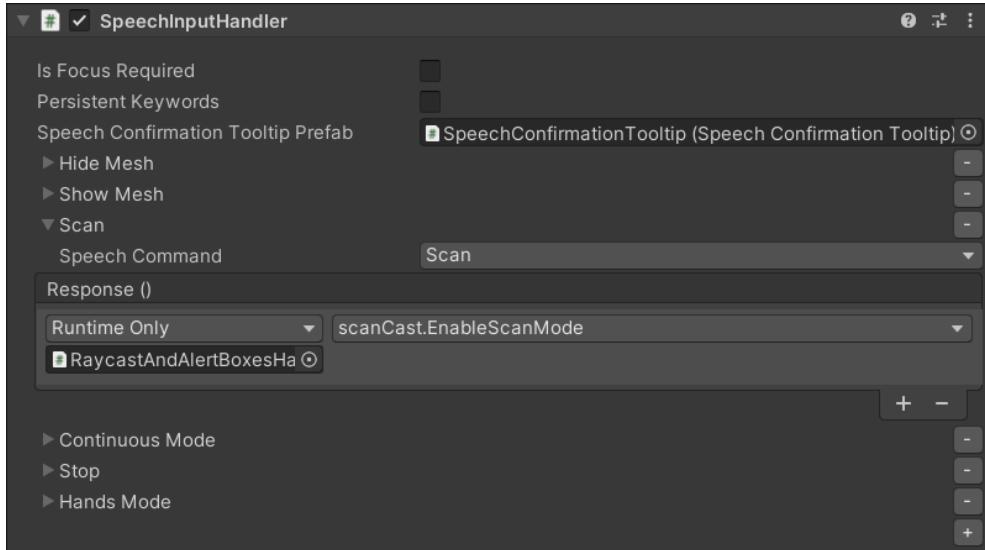
ται με την ενεργοποίηση ενός εκ των δύο μοτέρ που διαθέτει το controller, ένα σε κάθε πλευρά αυτού. Ανάλογα με τη θέση του εμποδίου ως προς την κατεύθυνση που κοιτά ο χρήστης, ενεργοποιείται το αντίστοιχο μοτέρ παράγοντας μια παλμική δόνηση. Όσο πιο πολύ πλησιάζει ο χρήστης στο εμπόδιο, τόσο η ένταση της δόνησης και η συχνότητα των παλμών αυξάνονται. Τέλος, ο χρήστης μπορεί να ενεργοποίηση και να απενεργοποιήση αυτόν τον τρόπο προειδοποίησης πιέζοντας τα κουμπιά ‘A’ και ‘B’ αντίστοιχα. Η διαχείριση του χειριστηρίου πραγματοποιείται από το component **Init Rumble Handler** (*Assets/Scripts/initiators*) του αντικειμένου **RaycastAndAlertBoxesHandler**.

3.3.5 Φωνητικές Εντολές

Αφού ολοκληρώσαμε την ανάπτυξη της λογικής για τον εντοπισμό εμποδίων και την προειδοποίηση του χρήστη, καλούμαστε τώρα να δώσουμε στο χρήστη τη δυνατότητα ελέγχου αυτών των λειτουργιών. Αυτό καθίσταται εφικτό με τη χρήση φωνητικών εντολών.

Η δυνατότητα χρήσης φωνητικών εντολών μπορεί να εναργοποιηθεί από το **MixedRealityToolkit**, στην καρτέλα ‘Input’. Εκεί μπορούμε να προσθέσουμε τις λέξεις-κλειδιά, εκφρασμένες στην Αγγλική γλώσσα, οι οποίες θα αποτελέσουν τις εντολές της εφαρμογής. Στη συνέχεια δημιουργούμε το **GameObject SpeechInputHandler_Global**, στο οποίο προσθέτουμε το component **SpeechInputHandler** (Σχήμα 3.13), το οποίο προσφέρει το MRTK [107] για τη διαχείριση των φωνητικών εντολών. Συγκεκριμένα,

όταν προσθέτουμε μια νέα φωνητική εντολή, επιλέγουμε την λέξη-κλειδί, η οποία θα την ενεργοποιεί, και ένα **GameObject**. Το **GameObject** διαθέτει components, από τα οποία επιλέγουμε μία συνάρτησή τους, η οποία θα εκτελείται, όταν θα χρησιμοποιείται η εντολή.



Σχήμα 3.13: Το component **SpeechInputHandler** και οι φωνητικές εντολές της εφαρμογής

Οι διαθέσιμες φωνητικές εντολές, καθώς και οι λειτουργίες τους, είναι οι εξής:

- **Hide Mesh:**
 - **Συνάρτηση:** hideMeshFunc (**Component:** *hideMesh.cs*)
 - **Λειτουργία:** Απόκρυψη απεικόνισης του mesh
- **Show Mesh:**
 - **Συνάρτηση:** showMeshFunc (**Component:** *hideMesh.cs*)
 - **Λειτουργία:** Απεικόνιση του mesh
- **Scan:**
 - **Συνάρτηση:** EnableScanMode (**Component:** *scanCast.cs*)
 - **Λειτουργία:** Ενεργοποίηση του **Scan Mode** για τον εντοπισμό εμποδίων
- **Continuous Mode:**
 - **Συνάρτηση:** continuousModeGlobalToggle (**Component:** *continuousCast.cs*)
 - **Λειτουργία:** Ενεργοποίηση/απενεργοποίηση του Continuous Mode για τον εντοπισμό εμποδίων
- **Stop:**
 - **Συνάρτηση:** EnableStopMode (**Component:** *stopCast.cs*)

- Λειτουργία: Απενεργοποίηση του **Continuous Mode** και παύση όλων των προειδοποιητικών ήχων
- **Hands Mode:**
 - Συνάρτηση: enabledHandABToggle (**Component:** *handRaycast.cs*)
 - Λειτουργία: Ενεργοποίηση/απενεργοποίηση της λειτουργίας **Hands Mode**

ΚΕΦΑΛΑΙΟ

4

ΑΞΙΟΛΟΓΗΣΗ

Σκοπός της αξιολόγησης (γιατί κάνουμε αξιολόγηση)

4.1 Διαδικασία αξιολόγησης

4.1.1 Περιγραφή Πειράματος

Αναλυτική περιγραφή των πειραμάτων που θα διεξαχθούν
Ανάλυση περιορισμών > Ως προς το χώρο, τους συμμετέχοντες κλπ

4.1.2 Προδιαγραφές Αξιολόγησης

Αριθμό συμμετεχόντων / Περιγραφή αυτών
Μετρικές (Χρόνοι, Πλήθος συγκρούσεων), ερωτηματολόγια που θα χρησιμοποιηθούν

4.1.3 Διεξαγωγή Πειράματος

Χώρος διεξαγωγής

4.2 Αποτελέσματα

Πίνακας με τους καταγεγραμμένους χρόνους
Στατιστική ανάλυση των αποτελεσμάτων και για τα δύο πειράματα
Παρατηρήσεις
Απαντήσεις ερωτηματολογίου
Απαντήσεις ανοιχτών ερωτήσεων

ΚΕΦΑΛΑΙΟ

5

ΠΡΟΕΚΤΆΣΕΙΣ ΚΑΙ
ΕΠΙΛΟΓΟΣ

5.1 Μελλοντικές προεκτάσεις

Ο κύριος στόχος της διπλωματικής εργασίας ήταν η υλοποίησης μιας εφαρμογής, η οποία θα προσέφερε ουσιαστική βοήθεια σε άτομα με προβλήματα στην περιήγηση τους σε έναν άγνωστο χώρο. Η ανάπτυξη της βασίζεται στην αξιοποίηση σύγχρονων τεχνολογιών, όπως είναι η Μικτή Πραγματικότητα, η οποία, μέχρι και σήμερα, δεν έχει καταφέρει να εισέλθει σε μεγάλο βαθμό στην καθημερινότητα του μέσου ανθρώπου. Η χρήση της συσκευής Microsoft HoloLens 2 αποτέλεσε ιδανική επιλογή, διότι διαθέτει πληθώρα αισθητήρων απαραίτητων για τους σκοπούς της εφαρμογής μας σε μια φορητή συσκευή, μικρού σχετικά σχετικά μεγέθους. Έτσι μπορέσαμε να επικεντρωθούμε στην ανάπτυξη λογισμικού χωρίς να υπάρχει η ανάγκη δημιουργίας εξειδικευμένου hardware. Επιπλέον, αποτελεί ευκαιρία ώστε άτομα με προβλήματα όρασης να βιώσουν - εν μέρει - την εμπειρία της Μικτής Πραγματικότητας, η οποία γίνεται περισσότερο προσβάσιμη προς αυτούς.

Η εφαρμογή, η οποία δημιουργήθηκε, διαθέτει τις απαραίτητες λειτουργίες, ώστε να αποτελέσει ένα πρωτότυπο και να παρουσιαστεί η χρηστικότητά της, ωστόσο βρίσκεται σε ένα αρχικό στάδιο ανάπτυξης. Επιδέχεται πλήθος αναβαθμίσεων, οι οποίες θα μπορούσαν να βελτιώσουν την απόδοσή της, καθώς και την εμπειρία του χρήστη. Μεταξύ των άλλων, μεγαλύτερο ενδιαφέρον παρουσιάζει η χρήση τεχνητής νοημοσύνης σε συνδυασμό με υπολογιστική όραση με σκοπό, όχι μόνο τον καλύτερο εντοπισμό εμποδίων, αλλά και για την πρόβλεψη πιθανών συγκερούσεων όπως και για την αναγνώριση αντικειμένων και εμποδίων. Προσφέροντας στο χρήστη τη γνώση για το εμπόδιο που βρίσκεται στο δρόμο του, δηλαδή αν πρόκειται για ένα τοίχο, μια καρέκλα, έναν άνρθρωπο ή μια πόρτα, του δίνεται η δυνατότητα να προσαρμόσει ανάλογα τον τρόπο αντιμετώπισης αυτής της δυσκολίας, καθώς και να κινείται με μεγαλύτερη ελευθερία στο χώρο, γνωρίζοντας τι υπάρχει σε αυτόν ή από που να μεταβεί σε κάποιο άλλο χώρο. Επίσης, χρήσιμη θα ήταν η δυνατότητα σύνδεσης του κινητού τηλεφώνου του χρήστη με την εφαρμογή μέσω ενός companion app, καθιστώντας τη διαχείριση αυτής και των λειτουργιών της ευκολότερη λόγω των επιλογών προσβασιμότητας που προσφέρει, όπως είναι οι screen readers. Με αυτό τον τρόπο, ο χρήστης δε θα χρειάζεται να βασίζεται αποκλειστικά στη χρήση φωνητικών εντολών.

5.2 Επίλογος

Στην παρούσα διπλωματική εργασία, παρουσιάσαμε, άρχικα, το θεωρητικό και τεχνολογικό υπόβαθροστο και, έπειτα, τον τρόπο υλοποίησης μιας εφαρμογής αξιοποιώντας τις δυνατότητες της συσκευής μικτής πραγματικότητας Microsoft HoloLens 2. Με την ανάπτυξη της, προσπαθήσαμε να ευαισθητοποιήσουμε σχετικά με ένα καθημερινό πρόβλημα που αντιμετωπίζει μια ιδιαίτερα ευάλωτη, προσφέροντας, παράλληλα, μία λύση σε

αυτό. Η λύση αυτή εκμεταλλεύεται νέες τεχνολογίες, οι οποίες δεν είχαν αξιοποιηθεί ευρύτατα για παρόμοια ζητήματα, με σκοπό να εκσυγχρονίσει ή να χρησιμοποιηθεί συνδυαστικά με ήδη υπάρχουσες λύσεις, ώστε να προσφέρει στο χρήστη περισσότερη ελευθερία και ανεξαρτησία.

ΒΙΒΛΙΟΓΡΑΦΙΑ

- [1] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino, “Augmented reality: A class of displays on the reality-virtuality continuum,” *Telemanipulator and Telepresence Technologies*, vol. 2351, 01 1994.
- [2] A. Bradford and A. Harvey, “The five (and more) senses,” Live Science, 2017. [Online]. Available: <https://www.livescience.com/60752-human-senses.html>
- [3] W. H. Organization, “Blindness and vision impairment,” World Health Organization, 08 2023. [Online]. Available: <https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment>
- [4] M. Langelaan, Quality of Life of Visually Impaired Working Age Adults, 2007. [Online]. Available: https://www.researchgate.net/publication/239845719_Quality_of_life_of_visually_impaired_working_age_adults
- [5] K. K. Morgan, “The role of augmented reality in medicine,” WebMD, 05 2021. [Online]. Available: <https://www.webmd.com/a-to-z-guides/features/augmented-reality-medicine>
- [6] W. Contributors, “Mixed reality,” Wikipedia, 04 2019. [Online]. Available: https://en.wikipedia.org/wiki/Mixed_reality
- [7] N. E. Institute, “How the eyes work | national eye institute,” Nih.gov, 04 2022. [Online]. Available: <https://www.nei.nih.gov/learn-about-eye-health/healthy-vision/how-eyes-work>
- [8] K. Anspaugh, S. Goncalves, E. Jackson-Osagie, and S. Q. Smith, “Vision,” *louis.pressbooks.pub*, 08 2022. [Online]. Available: <https://louis.pressbooks.pub/medicalterminology/chapter/vision/>
- [9] W. H. Organization, “World report on vision,” www.who.int, 10

2019. [Online]. Available: <https://www.who.int/publications/item/9789241516570>

- [10] J. D. Adelson, R. R. A. Bourne, P. S. Briant, S. R. Flaxman, H. R. B. Taylor, J. B. Jonas, A. A. Abdoli, W. A. Abrha, A. Abualhasan, E. G. Abu-Gharbieh, T. G. Adal, A. Afshin, H. Ahmadieh, W. Alemayehu, S. A. S. Alemzadeh, A. S. Alfaar, V. Alipour, S. Androudi, J. Arabloo, A. B. Ardit, B. B. Aregawi, A. Arrigo, C. Ashbaugh, E. D. Ashrafi, D. D. Atnafu, E. A. Bagli, A. A. W. Baig, T. W. Bärnighausen, M. B. Parodi, M. S. Beheshti, A. S. Bhagavathula, N. Bhardwaj, P. Bhardwaj, K. Bhattacharyya, A. Bijani, M. Bikbov, M. Bottone, T. M. Braithwaite, A. M. Bron, S. A. B. Nagaraja, Z. A. Butt, F. L. L. C. d. Santos, V. L. J. Carneiro, R. J. Casson, C.-Y. J. Cheng, J.-Y. J. Choi, D.-T. Chu, M. V. M. Cicinelli, J. M. G. Coelho, N. G. A. Congdon, R. A. A. Couto, E. A. M. Cromwell, S. M. Dahlawi, X. Dai, R. Dana, L. Dandona, R. A. Dandona, M. A. D. Monte, M. D. Molla, N. A. Dervenis, A. A. P. Desta, J. P. Deva, D. Diaz, S. E. Djalalinia, J. R. Ehrlich, R. R. Elayedath, H. R. B. Elhabashy, L. B. Ellwein, M. H. Emamian, S. Eskandarieh, F. G. Farzadfar, A. G. Fernandes, F. S. Fischer, D. S. M. Friedman, J. M. Furtado, S. Gaidhane, G. Gazzard, B. Gebremichael, R. George, A. Ghashghaei, S. A. Gilani, M. Golechha, S. R. Hamidi, B. R. R. Hammond, M. E. R. K. Hartnett, R. K. Hartono, A. I. Hashi, S. I. Hay, K. Hayat, G. Heidari, H. C. Ho, R. Holla, M. J. Househ, J. J. E. Huang, S. E. M. Ibitoye, I. M. D. Ilic, M. D. D. Ilic, A. D. N. Ingram, S. S. N. Irvani, S. M. S. Islam, R. Itumalla, S. P. Jayaram, R. P. Jha, R. Kahloun, R. Kalhor, H. Kandel, A. S. Kasa, T. A. Kavetskyy, G. A. H. Kayode, J. H. Kempen, M. Khairallah, R. A. Khalilov, E. A. C. Khan, R. C. Khanna, M. N. A. Khatib, T. A. E. Khoja, J. E. Kim, Y. J. Kim, G. R. Kim, S. Kisa, A. Kisa, S. Kosen, A. Koyanagi, B. K. Bicer, V. P. Kulkarni, O. P. Kurmi, I. C. Landires, V. C. L. Lansingh, J. L. E. Leasher, K. E. LeGrand, N. Leveziel, H. Limburg, X. Liu, S. M. Kunjathur, S. Maleki, N. Manafi, K. Mansouri, C. G. McAlinden, G. G. M. Meles, A. M. Mersha, I. M. R. Michalek, T. R. Miller, S. Misra, Y. Mohammad, S. F. A. Mohammadi, J. A. H. Mohammed, A. H. Mokdad, M. A. A. Moni, A. A. R. Montasir, A. R. F. Morse, G. F. C. Mulaw, M. Naderi, H. S. Naderifar, K. S. Naidoo, M. D. Naimzada, V. Nangia, S. M. N. Swamy, D. M. Naveed, H. L. Negash, H. L. Nguyen, V. A. Nunez-Samudio, F. A. Ogbo, K. T. Ogundimu, A. T. E. Olagunju, O. E. Onwujekwe, N. O. Otstavnov, M. O. Owolabi, K. Pakshir, S. Panda-Jonas, U. Parekh, E.-C. Park, M. Pasovic, S. Pawar, K. Pesudovs, T. Q. Peto, H. Q. Pham, M. Pinheiro, V. Podder, V. Rahimi-Movagh, M. H. U. Y. Rahman, P. Y. Ramulu, P. Rathi, S. L. Rawaf, D. L. Rawaf, L. Rawal, N. M. Reinig, A. M. Renzaho, A. L. Rezapour, A. L. Robin, L. Rossetti, S. Sabour, S. Safi, A. Sahebkar, M. A. M. Sahraian, A. M. Samy,

- B. Sathian, G. K. Saya, M. A. Saylan, A. A. A. Shaheen, M. A. T. Shaikh, T. T. Shen, K. S. Shibuya, W. S. Shiferaw, M. Shigematsu, J. I. Shin, J. C. Silva, A. A. Silvester, J. A. Singh, D. S. Singhal, R. S. Sitorus, E. Y. Skiadaresi, V. Y. A. Skryabin, A. A. Skryabina, A. B. Soheili, M. B. A. R. C. Sorrie, R. A. R. C. T. Sousa, C. T. Sreeramareddy, D. G. Stambolian, E. G. Tadesse, N. I. Tahhan, M. I. Tareque, F. X. Topouzis, B. X. Tran, G. K. Tsegaye, M. K. Tsilimbaris, R. Varma, G. Virgili, A. T. Vongpradith, G. T. Vu, Y. X. Wang, N. H. Wang, A. H. K. Weldemariam, S. K. G. West, T. G. Y. Wondmeneh, T. Y. Wong, M. Yaseri, N. Yonemoto, C. S. Yu, M. S. Zastrozhin, Z.-J. R. Zhang, S. R. Zimsen, S. Resnikoff, and T. Vos, "Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness in relation to vision 2020: the right to sight: an analysis for the global burden of disease study," *The Lancet Global Health*, vol. 9, p. e144–e160, 02 2021. [Online]. Available: [https://www.thelancet.com/journals/langlo/article/PIIS2214-109X\(20\)30489-7/fulltext](https://www.thelancet.com/journals/langlo/article/PIIS2214-109X(20)30489-7/fulltext)
- [11] S. K. West, G. S. Rubin, A. T. Broman, B. Muñoz, K. Bandeen-Roche, K. Turano, and S. Project, "How does visual impairment affect performance on tasks of everyday life?: the see project," *Archives of Ophthalmology*, vol. 120, no. 6, pp. 774–780, 06 2002. [Online]. Available: <https://doi.org/10.1001/archopht.120.6.774>
- [12] M. KHORRAMI-NEJAD, A. SARABANDI, M.-R. AKBARI, and F. ASKARIZADEH, "The impact of visual impairment on quality of life," *Medical Hypothesis, Discovery and Innovation in Ophthalmology*, vol. 5, p. 96–103, 2016. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5347211/>
- [13] M. Langelaan, M. R. de Boer, R. M. A. van Nispen, B. Wouters, A. C. Moll, and G. H. M. B. van Rens, "Impact of visual impairment on quality of life: A comparison with quality of life in the general population and with other chronic conditions," *Ophthalmic Epidemiology*, vol. 14, pp. 119–126, 01 2007.
- [14] M. Mashiata, T. Ali, P. Das, Z. Tasneem, F. R. Badal, S. K. Sarker, M. Hasan, S. H. Abhi, M. R. Islam, F. Ali, H. Ahamed, M. Islam, and S. K. Das, "Towards assisting visually impaired individuals: A review on current status and future prospects," *Biosensors and Bioelectronics: X*, vol. 12, p. 100265, 10 2022.
- [15] I. U. Library, "Libguides: Blind/visual impairment: Common assistive technologies," Illinois.edu, 2013. [Online]. Available: <https://guides.library.illinois.edu/c.php?g=526852&p=3602299>
- [16] A. F. for the Blind, "Screen readers | american foundation for the blind," Afb.org, 2019. [Online]. Available: <https://www.afb.org/blindness-and-low-vision/using-technology/assistive-technology-products/screen-readers>

- [17] W3Schools, “Html accessibility,” W3schools.com, 2020. [Online]. Available: https://www.w3schools.com/html/html_accessibility.asp
- [18] “biped | smart copilot for blind and visually impaired people,” biped. [Online]. Available: <https://www.biped.ai>
- [19] A. Ghebali, “Orcam myeye - the ultimate visual aid device for the people with visual impairment,” OrCam Technologies, 02 2023. [Online]. Available: <https://www.orcam.com/en-us/orcam-myeye>
- [20] “Blindsight,” BlindSquare. [Online]. Available: <https://www.blindsight.com>
- [21] “Be my eyes - bringing sight to blind and low-vision people,” Bemyeyes.com, 2019. [Online]. Available: <https://www.bemyeyes.com>
- [22] “WeWalk smart cane,” WeWALK Smart Cane. [Online]. Available: <https://wewalk.io/en/>
- [23] “Seeing ai,” Microsoft Garage. [Online]. Available: <https://www.microsoft.com/en-us/garage/wall-of-fame/seeing-ai/>
- [24] S. Mann, V. Phillip, T. A. Furness, Y. C. Yuan, J. Iorio, and Z. Wang, “Fundamentals of all the realities: Virtual, augmented, mediated, multimediated, and beyond,” *Springer eBooks*, pp. 3–34, 01 2023.
- [25] S. Mann and C. Wyckoff, “Extended reality,” Wearcam.org, 1991. [Online]. Available: <http://wearcam.org/xr.txt>
- [26] J. Weiler, “Phantom touch: Vr reveals new insights in human perception,” Neuroscience News, 11 2023. [Online]. Available: <https://neurosciencenews.com/vr-tactile-perception-phantom-touch-25208/>
- [27] Touching Virtual Reality: A Review of Haptic Gloves, ACTUATOR 2018; 16th International Conference on New Actuators. VDE, 2018.
- [28] H. E. Lowood, Virtual Reality | Computer Science, 11 2018. [Online]. Available: <https://www.britannica.com/technology/virtual-reality>
- [29] C. E. Loeffler, “Distributed virtual reality: Applications for education, entertainment and industry,” *Telektronikk*, vol. 89, p. 83–88, 1993.
- [30] Meta, “Xtadium on meta quest: Get closer to sports you love in vr,” Meta, 11 2022. [Online]. Available: <https://about.fb.com/news/2022/11/xtadium-quest-sports-in-vr/>
- [31] A. Ansari, V. K. Shukla, K. Saxena, and B. Filomeno, “Implementing virtual reality in entertainment industry,” *Springer eBooks*, pp. 561–570, 09 2021.
- [32] A. Hamad and B. Jia, “How virtual reality technology has changed our lives: An overview of the current and potential applications and limitations,” *International Journal of Environmental Research and Public Health*, vol. 19, p. 11278, 09 2022. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9517547/>
- [33] S. Kavanagh, A. Luxton-Reilly, B. Wuensche, and B. Plimmer, “A systematic review of virtual reality in education,” *Themes in Science*

- and Technology Education, vol. 10, p. 85–119, 12 2017.
- [34] A literature review on immersive virtual reality in education: state of the art and perspectives, vol. 1, no. 133, 2015.
- [35] A. Chirico, F. Lucidi, M. De Laurentiis, C. Milanese, A. Napoli, and A. Giordano, “Virtual reality in health system: Beyond entertainment. a mini-review on the efficacy of vr during cancer treatment,” Journal of Cellular Physiology, vol. 231, pp. 275–287, 10 2015.
- [36] A. J. Snoswell and C. L. Snoswell, “Immersive virtual reality in health care: Systematic review of technology and disease states,” JMIR Biomedical Engineering, vol. 4, p. e15025, 09 2019.
- [37] J. Gerup, C. B. Soerensen, and P. Dieckmann, “Augmented reality and mixed reality for healthcare education beyond surgery: an integrative review,” International Journal of Medical Education, vol. 11, pp. 1–18, 01 2020.
- [38] W. L. Hosch, Augmented Reality | Computer Science, 2020. [Online]. Available: <https://www.britannica.com/technology/augmented-reality>
- [39] J. Carmigniani and B. Furht, “Augmented reality: An overview,” Handbook of Augmented Reality, pp. 3–46, 2011.
- [40] S. M. Ko, W. S. Chang, and Y. G. Ji, “Usability principles for augmented reality applications in a smartphone environment,” International Journal of Human-Computer Interaction, vol. 29, pp. 501–515, 08 2013.
- [41] H.-K. Wu, S. W.-Y. Lee, H.-Y. Chang, and J.-C. Liang, “Current status, opportunities and challenges of augmented reality in education,” Computers and Education, vol. 62, pp. 41–49, 03 2013.
- [42] K. Lee, “Augmented reality in education and training.” TechTrends, vol. 56, pp. 13–21, 02 2012.
- [43] S. Kim, M. A. Nussbaum, and J. L. Gabbard, “Augmented reality “smart glasses” in the workplace: Industry perspectives and challenges for worker safety and health,” IIE Transactions on Occupational Ergonomics and Human Factors, vol. 4, pp. 253–258, 07 2016.
- [44] M. Funk, A. Bächler, L. Bächler, T. Kosch, T. Heidenreich, and A. Schmidt, “Working with augmented reality?” Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments, 06 2017.
- [45] A. C. Pereira, A. C. Alves, and P. Arezes, “Augmented reality in a lean workplace at smart factories: A case study,” Applied Sciences, vol. 13, p. 9120, 01 2023. [Online]. Available: <https://www.mdpi.com/2076-3417/13/16/9120>
- [46] K. Klinker, M. Wiesche, and H. Krcmar, “Digital transformation in health care: Augmented reality for hands-free service innovation,” Information Systems Frontiers, 06 2019.

- [47] E. Zhu, A. Lilienthal, L. A. Shluzas, I. Masiello, and N. Zary, “Design of mobile augmented reality in health care education: A theory-driven framework,” *JMIR Medical Education*, vol. 1, p. e10, 09 2015.
- [48] L. Solbiati, N. Gennaro, and R. Muglia, “Augmented reality: From video games to medical clinical practice,” *CardioVascular and Interventional Radiology*, 07 2020.
- [49] S.-W. Hung, C.-W. Chang, and Y.-C. Ma, “A new reality: Exploring continuance intention to use mobile augmented reality for entertainment purposes,” *Technology in Society*, vol. 67, p. 101757, 11 2021.
- [50] Z. Yovcheva, D. Buhalis, and C. Gatzidis, “Smartphone augmented reality applications for tourism,” *e-Review of Tourism Research (eRTR)*, vol. 10, p. 63–66, 2012. [Online]. Available: <http://eprints.bournemouth.ac.uk/20219/>
- [51] C. D. Kounavis, A. E. Kasimati, and E. D. Zamani, “Enhancing the tourism experience through mobile augmented reality: Challenges and prospects,” *International Journal of Engineering Business Management*, vol. 4, p. 10, 01 2012.
- [52] M. Speicher, B. D. Hall, and M. Nebeling, “What is mixed reality?” *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 05 2019.
- [53] P. Knierim, T. Kosch, M. Hoppe, and A. Schmidt, “Challenges and opportunities of mixed reality systems in education,” *dl.gi.de*, 2018.
- [54] *Mixed reality classroom: Learning from entertainment*. Association for Computing Machinery, 2007. [Online]. Available: <https://doi.org/10.1145/1306813.1306833>
- [55] L. Chen, T. W. Day, W. Tang, and N. W. John, “Recent developments and future challenges in medical mixed reality,” *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 10 2017. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/8115411>
- [56] O. M. Tepper, H. L. Rudy, A. Lefkowitz, K. A. Weimer, S. M. Marks, C. S. Stern, and E. S. Garfein, “Mixed reality with hololens: Where virtual reality meets augmented reality in the operating room,” www.ingentaconnect.com, 11 2017. [Online]. Available: <https://www.ingentaconnect.com/content/wk/prs/2017/00000140/00000005/art00063>
- [57] X. Wang and M. A. Schnabel, *Mixed Reality in Architecture, Design, and Construction*. Springer Science & Business Media, 12 2008.
- [58] P. S. Dunston and X. Wang, “Mixed reality-based visualization interfaces for architecture, engineering, and construction industry,” *Journal of Construction Engineering and Management*, vol. 131, no. 12, pp. 1301–1309,

2005. [Online]. Available: <https://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9364%282005%29131%3A12%281301%29>
- [59] “Hololens (1st gen) hardware,” Microsoft Learn, 11 2021. [Online]. Available: <https://learn.microsoft.com/en-us/hololens/hololens1-hardware>
- [60] A. Kipman, “Announcing microsoft hololens development edition open for pre-order, shipping march 30,” Microsoft Devices Blog, 02 2016. [Online]. Available: <https://blogs.windows.com/devices/2016/02/29/announcing-microsoft-hololens-development-edition-open-for-pre-order-shipping-march-30/>
- [61] “Hololens 1st (gen) release notes,” Microsoft Learn, 10 2023. [Online]. Available: <https://learn.microsoft.com/en-us/hololens/hololens1-release-notes>
- [62] Z. Bowden, “The original hololens will no longer receive major os updates,” Windows Central, 07 2019. [Online]. Available: <https://www.windowscentral.com/original-hololens-will-no-longer-receive-major-os-updates>
- [63] J. White, “Microsoft at mwc barcelona: Introducing microsoft hololens 2,” The Official Microsoft Blog, 02 2019. [Online]. Available: <https://blogs.microsoft.com/blog/2019/02/24/microsoft-at-mwc-barcelona-introducing-microsoft-hololens-2/>
- [64] Microsoft, “Hololens 2—pricing and options | microsoft hololens,” Microsoft.com, 2019. [Online]. Available: <https://www.microsoft.com/en-us/hololens/buy>
- [65] scooley, “Hololens 2 hardware,” Microsoft Learn, 03 2023. [Online]. Available: <https://learn.microsoft.com/en-us/hololens/hololens2-hardware>
- [66] R. Seiler, “Microsoft brings windows 11 to hololens 2,” Windows Experience Blog, 04 2023. [Online]. Available: <https://blogs.windows.com/windowsexperience/2023/04/13/microsoft-brings-windows-11-to-hololens-2/>
- [67] Microsoft, “Hololens 2—overview, features, and specs | microsoft hololens,” www.microsoft.com. [Online]. Available: <https://www.microsoft.com/en-us/hololens/hardware#document-experiences>
- [68] keveleigh, “Hand tracking - mrkt 2,” Microsoft Learn, 02 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/mrkt-unity/mrkt2/features/input/hand-tracking?view=mrktunity-2022-05>
- [69] caseymeechhof, “Direct manipulation with hands - mixed reality,” Microsoft Learn, 09 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/direct-manipulation>
- [70] —, “Point and commit with hands - mixed reality,” Microsoft Learn, 08 2022. [Online]. Available: <https://learn.microsoft.com/>

- en-us/windows/mixed-reality/design/point-and-commit
- [71] sostel, “Gaze and commit - mixed reality,” Microsoft Learn, 03 2023. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/gaze-and-commit>
 - [72] shengkait, “Start gesture - mixed reality,” Microsoft Learn, 02 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/system-gesture>
 - [73] sostel, “Gaze and dwell - mixed reality,” Microsoft Learn, 07 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/gaze-and-dwell>
 - [74] Sean-Kerawala, “Head-gaze and dwell - mixed reality,” Microsoft Learn, 07 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/gaze-and-dwell-head>
 - [75] sostel, “Eye-gaze and dwell - mixed reality,” Microsoft Learn, 03 2023. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/gaze-and-dwell-eyes>
 - [76] HakOn, “Voice input - mixed reality,” Microsoft Learn, 03 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/voice-input>
 - [77] B. Furht, Ed., Mesh, 3D, ser. Encyclopedia of Multimedia. Springer US, 2006, p. 406–407. [Online]. Available: https://doi.org/10.1007/0-387-30038-4_26
 - [78] mattzmsft, “Spatial mapping - mixed reality,” Microsoft Learn, 02 2023. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/spatial-mapping>
 - [79] SzymonS, “Scene understanding - mixed reality,” Microsoft Learn, 09 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/scene-understanding>
 - [80] dorreneb, “Hololens environment considerations,” Microsoft Learn, 03 2022. [Online]. Available: <https://learn.microsoft.com/en-us/hololens/hololens-environment-considerations>
 - [81] kegodin, “Spatial sound overview - mixed reality,” Microsoft Learn, 02 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/design/spatial-sound>
 - [82] H. Møller, “Fundamentals of binaural technology,” Applied Acoustics, vol. 36, no. 3, pp. 171–218, 1992. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0003682X9290046U>
 - [83] thetuvix, “Install the tools - mixed reality,” Microsoft Learn, 01 2023. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/develop/install-the-tools>
 - [84] qianw211, “Choosing your engine - mixed reality,” Microsoft Learn, 06 2022. [Online]. Available: <https://learn.microsoft.com/>

- en-us/windows/mixed-reality/develop/choosing-an-engine
- [85] ——, “Unity development for hololens - mixed reality,” Microsoft Learn, 09 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/develop/unity-development-overview>
- [86] U. Technologies, “Gameobjects (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/GameObjects.html>
- [87] ——, “Introduction to components (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/Components.html>
- [88] ——, “Creating and using scripts (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/CreatingAndUsingScripts.html>
- [89] ——, “Unity’s interface (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/UsingTheEditor.html>
- [90] ——, “Toolbar (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/Toolbar.html>
- [91] ——, “Hierarchy window (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/Hierarchy.html>
- [92] ——, “Game view (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/GameView.html>
- [93] ——, “Scene view (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/UsingTheSceneView.html>
- [94] ——, “Inspector window (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/UsingTheInspector.html>
- [95] ——, “Project window (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/ProjectView.html>
- [96] ——, “Asset workflow (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/AssetWorkflow.html>
- [97] ——, “Status bar (unity - manual),” docs.unity3d.com, 04 2023. [Online]. Available: <https://docs.unity3d.com/2020.3/Documentation/Manual/StatusBar.html>
- [98] vtieto, “Using visual studio to deploy and debug

- mixed reality,” Microsoft Learn, 09 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/using-visual-studio?tabs=hl2>
- [99] polar kev, “Mrtk2-unity developer documentation - mrtk 2,” Microsoft Learn, 12 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/?view=mrtkunity-2022-05>
- [100] Sean-Kerawala, “Welcome to the mixed reality feature tool - mixed reality,” Microsoft Learn, 12 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/develop/unity/welcome-to-mr-feature-tool>
- [101] hferrone, “Mixed reality documentation - mixed reality,” learn.microsoft.com. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/>
- [102] FlorianBagarMicrosoft, “Holographic remoting player - mixed reality,” learn.microsoft.com, 02 2023. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/develop/native/holographic-remoting-player>
- [103] davidkline ms, “Spatial object mesh observer - mrtk 2,” learn.microsoft.com, 08 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/features/spatial-awareness/spatial-object-mesh-observer?view=mrtkunity-2022-05>
- [104] ——, “Configuring mesh observers for device - mrtk 2,” learn.microsoft.com, 08 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/features/spatial-awareness/configuring-spatial-awareness-mesh-observer?view=mrtkunity-2022-05>
- [105] U. Technologies, “Physics (unity - scripting api),” docs.unity3d.com. [Online]. Available: <https://docs.unity3d.com/ScriptReference/Physics.html>
- [106] keveleigh, “Pointers - mrtk 2,” learn.microsoft.com, 08 2022. [Online]. Available: <https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/features/input/pointers?view=mrtkunity-2022-05>
- [107] “Speechinpushandler class (microsoft.mixedreality.toolkit.input),” learn.microsoft.com. [Online]. Available: <https://learn.microsoft.com/en-us/dotnet/api/microsoft.mixedreality.toolkit.input.speechinpushandler?view=mixed-reality-toolkit-unity-2020-dotnet-2.8.0>

ПАР'АРТНМА А'

Κώδικας Εφαρμογής

Κώδικας A.1: Αρχείο variablesAggregator.cs

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class variablesAggregator : MonoBehaviour
6  {
7      // Variables for debugging purposes
8      [Header("--- General Attributes ---")]
9      public GameObject userPosition;
10     [Tooltip("User's height in meters (m)"), Range(0.0f, 2.5f)]
11     public float userHeight = 0.0f;
12     [Tooltip("User's width in meters (m)"), Range(0.0f, 2.5f)]
13     public float userWidth = 0.0f;
14     [Tooltip("The max distance of the raycasts"), Range(0.0f, 5.0
15         f)]
16     public float maxDistance = 5.0f;
17     public enum CastTypeEnum
18     {
19         RayCast,
20         BoxCast
21     };
22     [Tooltip("The cast type which will be used")]
23     public CastTypeEnum castType;
24     [Tooltip("The list of the raycast boxes"), ]
25     public GameObject[] raycastBoxes;
26     [Header("--- Alert Boxes ---")]
27     [Tooltip("The list of the alert boxes")]
28     public GameObject[] alertBoxes;
29     [Tooltip("The common alert box")]
30     public GameObject commonAlertBox;
31     [Tooltip("The peripheral alert box"), ]
32     public GameObject peripheralAlertBox;
33     public enum AlertBoxTypeEnum
34     {
35         BoxSpecific,
36         Common,
37         Peripheral
38     }
39     #if UNITY_EDITOR
40         [Help("The logic for peripheral alert box was never
41             implemented.\nIf it is selected, no alert box will be
42             enabled",
43             UnityEditor.MessageType.Warning)]
44     #endif
45     public AlertBoxTypeEnum alertBoxType;
46     [Tooltip("The alert box is placed at the height level of the
47         eyes instead of the height level of the hitPoint")]
48     public bool placeAlertBoxAtEyeLevel = true;
49     public enum CastModeEnum
50     {

```

```

47         ContinuousMode ,
48         ScanMode
49     }
50     // [Tooltip("The cast mode which will be used")]
51     // public CastModeEnum enabledModeGlobal;
52     [Header("---- Cast Modes ---")]
53     public bool enableContinuousModeGlobal = false;
54
55     [Header("---- Scan/Continuous Mode Attributes ---")]
56     [Tooltip("The number of cast boxes placed in each row"),
57      Range(1.0f, 100.0f)]
57     public int boxesPerRow=3;
58     [Tooltip("The number of cast boxes placed in each column"),
59      Range(1.0f, 100.0f)]
59     public int numberOfRows=3;
60
61     [Header("---- Hand Mode Attributes---")]
62     [Tooltip("The list of the hand alert boxes")]
63     public GameObject[] handAlertBoxes;
64     [Tooltip("Enables the 'Hands mode'")]
65     public bool enabledHandAB = true;
66     public enum lightPositionEnum
67     {
68         Up,
69         Right,
70         Down,
71         Left
72     }
73     [Header("---- Flashing warning attributes ---")]
74     [Tooltip("Enables the flash effect")]
75     public bool activateFlash = false;
76     [Tooltip("The gameObjects which have the flashing effect")]
77     public GameObject[] flashingImages;
78     [Tooltip("The color of the flash for dangerous distance"),
79      ColorUsage(false)]
79     public Color dangerColor;
80     [Tooltip("The color of the flash for warning"), ColorUsage(
81      false)]
81     public Color warningColor;
82     [Tooltip("The color of the flash for safe distance"),
83      ColorUsage(false)]
83     public Color safeColor;
84
85     [Header("---- Rumble attributes ---")]
86     [Tooltip("Enables the rumble effect")]
87     public bool activateRumble;
88     public enum RumbleModes
89     {
90         Constant ,
91         Linear ,
92         Pulse
93     }
94     #if UNITY_EDITOR
95         [Help("The logic for Constant and Linear rumble was never
96             implemented", UnityEditor.MessageType.Warning)]
96     #endif

```

```

97     public RumbleModes rumbleModeSelected;
98     #if UNITY_EDITOR
99         [Help("The rumbleDuration variable is never used",
100             UnityEditor.MessageType.Warning)]
101    #endif
102    public float rumbleDuration = 1.0f;
103    /*public Dictionary<string, float> stopTimes = new Dictionary
104        <string, float>()
105    {
106        {"safeStopTime", 2.0f},
107        {"warningStopTime", 1.0f},
108        {"dangerousStopTime", 0.25f}
109    };*/
110    [Header("---- Flashing & Rumble common attributes ---")]
111    [Tooltip("The maximum distance considered safe"), Min(0.0f)]
112    public float distanceThreadToActivate = 2.0f;
113    [Tooltip("The maximum distance for which a user should be
114        cautious"), Min(0.0f)]
115    public float warningIndicatorThread = 1.5f;
116    [Tooltip("The maximum distance considered dangerous"), Min
117        (0.0f)]
118    public float dangerIndicatorThread = 0.5f;
119
120 }

```

Κώδικας A.2: Αρχείο initiators/initCastAndBoxesHandler.cs

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class initCastAndBoxesHandler : MonoBehaviour
6 {
7     public GameObject variableAggregatorObject;
8 }

```

Κώδικας A.3: Αρχείο initiators/initBoxes.cs

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.InputSystem;
5 using UnityEngine.Serialization;
6
7 [RequireComponent(typeof(initCastAndBoxesHandler))]
8 public class initBoxes : MonoBehaviour
9 {
10     [HideInInspector]
11     public float userHeight = 0.0f;
12     private float userWidth = 0.0f;
13     private variablesAggregator variableAggInstance;
14     private bool enabledContinuousMode;
15     private GameObject[] raycastBoxes;
16     private int boxesPerRow;
17     private int numberOfRows;

```

```

18
19     // Start is called before the first frame update
20     void Start()
21     {
22         variableAggInstance = this.GetComponent<
23             initCastAndBoxesHandler>().variableAggregatorObject.
24             GetComponent<variablesAggregator>();
25         userHeight = variableAggInstance.userHeight;
26         userWidth = variableAggInstance.userWidth;
27         raycastBoxes = variableAggInstance.raycastBoxes;
28         enabledContinuousMode = variableAggInstance.
29             enableContinuousModeGlobal;
30         boxesPerRow = variableAggInstance.boxesPerRow;
31         numberOfRows = variableAggInstance.numberOfRows;
32
33         if (userHeight <= 0)
34         {
35             userHeight = 1.8f;
36         }
37
38         // For some reason, 1 unit = 0.7cm
39         // userHeight -= 0.30f;
40
41         switch (enabledContinuousMode)
42         {
43             // Scan Mode
44             case false:
45                 this.positionAndScaleBoxes(raycastBoxes, 3, 3,
46                     userHeight);
47                 break;
48             // Continuous Mode
49             case true:
50                 GameObject[] castBoxesToUse = this.
51                     findCastBoxesWithContinuousEnabled(
52                         raycastBoxes);
53                 this.positionAndScaleBoxes(castBoxesToUse,
54                     boxesPerRow, numberOfRows, userHeight);
55                 break;
56         }
57
58     /// <summary>
59     /// A method which places the castBoxes and their alert boxes
60     /// in the correct position based on the parameters provided
61     ///
62     /// </summary>
63     /// <param name="raycstBoxesToPosition">An array of the
64     /// castpxoses</param>
65     /// <param name="boxesPerRow">The number of castBoxes which
66     /// will be placed in each row</param>
67     /// <param name="numberOfRows">The number of rows, in which
68     /// the cast boxes will be placed</param>
69     /// <param name="userHeight">The height of the user in meters
70     /// </param>
71     public void positionAndScaleBoxes

```

```

61      (
62          GameObject[] raycstBoxesToPosition,
63          int boxesPerRow,
64          int numberOfRows,
65          float userHeight
66      )
67  {
68      float boxLength = userWidth / boxesPerRow;
69      float boxHeight = userHeight / numberOfRows;
70
71      float boxPlacementX = -boxLength * (boxesPerRow - 1);
72      float boxPlacementY = 0.0f;
73      for (int i = 0; i < raycstBoxesToPosition.Length; i++)
74  {
75          if (i % boxesPerRow == 0)
76          {
77              boxPlacementX = -boxLength * (boxesPerRow - 1);
78
79              if (i != 0) boxPlacementY -= boxHeight;
80          }
81
82          raycstBoxesToPosition[i].transform.position = new
83              Vector3(boxPlacementX, boxPlacementY, -0.1f);
84          raycstBoxesToPosition[i].transform.localScale = new
85              Vector3(boxLength, boxHeight, boxLength);
86
87          raycstBoxesToPosition[i].GetComponent<initSingleBox
88              >().alertBox.transform.localScale = new Vector3(
89                  boxLength, boxHeight, 0.1f);
90
91      /// <summary>
92      /// In an array of RaycastHit (<paramref name="hitPointsList
93      /// "/>), it locates the index of the hitPoint, whose
94      /// distance is closer to the user.
95      /// </summary>
96      /// <param name="hitPointsList">An array of hitPoints</param>
97      /// <returns>It returns the index of the hitPoint closer to
98      /// the user</returns>
99      public int findClosestHitPointIndex(RaycastHit[]
100          hitPointsList)
101  {
102      int closestHitPointIndex = -1;
103      float minDistance = 999.0f;
104      List<Vector3> relPosList = new List<Vector3>();
105      for (int i = 0; i < hitPointsList.Length; i++)
106  {
107          if (hitPointsList[i].distance != 0.0f &&
108              hitPointsList[i].collider != null)
109          {
110              Vector3 relativePosition = variableAggInstance.
111                  userPosition.transform.InverseTransformPoint(
112                      hitPointsList[i].point);

```

```

106         relPosList.Add(relativePosition);
107         if (hitPointsList[i].distance < minDistance)
108         {
109             minDistance = hitPointsList[i].distance;
110             closestHitPointIndex = i;
111         }
112     }
113 }
114
115     return closestHitPointIndex;
116 }
117
118 public void CalculateDeadZones()
119 {
120 }
121
122
123 /// <summary>
124 ///     Finds the castBoxes for which the continuous mode has
125 ///     been enabled..
126 /// </summary>
127 /// <param name="allCastBoxes">An array of all the castBoxes
128 ///     in the scene</param>
129 /// <returns>The castBoxes that meet the criteria</returns>
130 public GameObject[] findCastBoxesWithContinuousEnabled(
131     GameObject[] allCastBoxes)
132 {
133     List<GameObject> castBoxesWithContinuous = new List<
134         GameObject>();
135
136     for (int i = 0; i < allCastBoxes.Length; i++)
137     {
138         if (allCastBoxes[i].GetComponent<initSingleBox>().
139             continuousRaycast == true)
140         {
141             castBoxesWithContinuous.Add(allCastBoxes[i]);
142         }
143     }
144
145     return castBoxesWithContinuous.ToArray();
146 }
147
148 /// <summary>
149 /// Places the alert box of a <paramref name="castBox"/> in
150 ///     the position of the <paramref name="hitPoint"/>.
151 /// </summary>
152 /// <param name="castBox">The castBox whose alert box will be
153 ///     transformed</param>
154 /// <param name="hitPoint">The hitPoint to which the alert
155 ///     box will be placed</param>
156 public void placeAlertBox(GameObject castBox, RaycastHit
157     hitPoint)
158 {
159     castBox.GetComponent<initSingleBox>().alertBox.SetActive(
160         true);
161     if (!castBox.GetComponent<initSingleBox>().alertBox.

```

```

152     GetComponent< AudioSource >().isPlaying)
153     castBox.GetComponent< initSingleBox >().alertBox.
154         GetComponent< AudioSource >().Play();
155     castBox.GetComponent< initSingleBox >().alertBox.transform.
156         position = hitPoint.point;
157     castBox.GetComponent< initSingleBox >().alertBox.transform.
158         rotation = castBox.transform.rotation;
159 }
160
161 /**
162  * Places an <paramref name="alertBox"/> in the position of
163  * a <paramref name="hitPoint"/>.
164  */
165 /**
166  * <param name="alertBox">The alert box which will be
167  * transformed</param>
168  * <param name="castBox">The castBox whose rotation will be
169  * used</param>
170  * <param name="hitPoint">The hitPoint to which the alert
171  * box will be placed</param>
172  */
173 public void placeAlertBox(GameObject alertBox, GameObject
174     castBox, RaycastHit hitPoint, bool
175     placeAlertBoxToEarLevel)
176 {
177     alertBox.SetActive(true);
178     var alertTrans = alertBox.transform;
179     if (!alertBox.GetComponent< AudioSource >().isPlaying)
180         alertBox.GetComponent< AudioSource >().Play();
181
182     /*
183     var hitPointVec = hitPoint.point;
184     var userPositionTrans = variableAggInstance.userPosition.
185         transform;
186     if (variableAggInstance.placeAlertBoxAtEyeLevel)
187     {
188         alertTrans.position = new Vector3(hitPointVec.x,
189             userPositionTrans.position.y, hitPointVec.z);
190     }
191     else
192     {
193         alertTrans.position = hitPointVec;
194     }
195     alertBox.transform.rotation = castBox.transform.rotation;
196     */
197 }
198
199 public void placePeripheralAlertBox(GameObject alertBox,
200     RaycastHit hitPoint)
201 {
202     alertBox.SetActive(true);
203     if (!alertBox.GetComponent< AudioSource >().isPlaying)
204     {
205         alertBox.GetComponent< AudioSource >().Play();
206     }
207     Vector3 relativePosition = variableAggInstance.
208         userPosition.transform.InverseTransformPoint(hitPoint
209         .point);

```

```

193     alertBox.transform.position = new Vector3(
194         variableAggInstance.userPosition.transform.position.x
195             + /*Normalize x value*/ (relativePosition.x),
196         variableAggInstance.userPosition.transform.position.y
197             + relativePosition.y,
198         variableAggInstance.userPosition.transform.position.z
199     );
200 }
201 /**
202  * Deactivates all the alert boxes, except the one, whose
203  * castBox casted and hit the point closest to the user
204  */
205 /**
206  * <param name="castBoxesArray">The array of castBoxes</param>
207  * <param name="castHitPointIndex">The index of the castBox,
208  * whoe cast hit the point closest to the user</param>
209  */
210 public void deactivateAlertBoxesExceptTheClosest(GameObject []
211     castBoxesArray, int castHitPointIndex)
212 {
213     for (int i = 0; i < castBoxesArray.Length; i++)
214     {
215         if (i != castHitPointIndex)
216         {
217             castBoxesArray[i].GetComponent<initSingleBox>().
218                 alertBox.SetActive(false);
219         }
220     }
221 }
222 /**
223  * Deactivates all the alert boxes, based on an array of
224  * castBoxes
225  */
226 /**
227  * <param name="castBoxesArray">The array of castBoxes,
228  * whose alert boxes will be deactivated</param>
229  */
230 public void deactivateAlertBox(GameObject [] castBoxesArray)
231 {
232     for (int i = 0; i < castBoxesArray.Length; i++)
233     {
234         if (castBoxesArray[i].GetComponent<initSingleBox>().
235             alertBox.GetComponent< AudioSource >().isPlaying)
236             castBoxesArray[i].GetComponent<initSingleBox>().
237                 alertBox.GetComponent< AudioSource >().Pause();
238     }
239 }
240 /**
241  * Deactivates an alert box
242  */
243 /**
244  * <param name="alertBox">The alert box which will be
245  * deactivated</param>
246  */
247 public void deactivateAlertBox(GameObject alertBox)
248 {
249     if (alertBox.GetComponent< AudioSource >().isPlaying)
250         alertBox.GetComponent< AudioSource >().Pause();

```

```
237     }
238 }
```

Κώδικας A.4: Αρχείο initiators/initSingleBox.cs

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class initSingleBox : MonoBehaviour
6 {
7     [Tooltip("The object which will operate as an alert box for
8         the specific raycast.")]
9     public GameObject alertBox;
10    [Tooltip("If it is set to 'True', rays will be casted in
11        every frame.")]
12    public bool continuousRaycast = false;
13 }
```

Κώδικας A.5: Αρχείο initiators/initFlashHandler.cs

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5
6 [RequireComponent(typeof(initCastAndBoxesHandler))]
7 public class initFlashHandler : MonoBehaviour
8 {
9     private variablesAggregator variableAggInstance;
10
11     // Start is called before the first frame update
12     void Start()
13     {
14         variableAggInstance = this.GetComponent<
15             initCastAndBoxesHandler>().variableAggregatorObject.
16             GetComponent<variablesAggregator>();
17     }
18
19     /// <summary>
20     /// Disables the flash effect of all the gameObjects
21     /// </summary>
22     public void StopFlashes()
23     {
24         GameObject[] flashImagesArray = variableAggInstance.
25             flashingImages;
26
27         for (int i = 0; i < flashImagesArray.Length; i++)
28         {
29             flashImagesArray[i].GetComponent<flashingLight>().
30                 StopFlash();
31         }
32     }
33
34     /// <summary>
```

```

31     /// Disables the flash effect of specific gameObjects
32     /// </summary>
33     /// <param name="flashImagesArray">The array of gameObjects
34     /// for which the flash effect will be disabled</param>
35     public void StopFlashes(GameObject[] flashImagesArray)
36     {
37         for (int i = 0; i < flashImagesArray.Length; i++)
38         {
39             flashImagesArray[i].GetComponent<flashingLight>().
40                 StopFlash();
41         }
42     /// <summary>
43     /// Enables the flash effect for specific gameObjects, based
44     /// on the position of the hitPoint
45     /// </summary>
46     /// <param name="lightPositions">The position of the hitPoint
47     /// relative to the user</param>
48     /// <param name="flashColor">The color of the flash</param>
49     public void StartMultipleFlashes(variablesAggregator.
50                                     lightPositionEnum[] lightPositions, Color flashColor)
51     {
52         GameObject[] flashImagesArray = variableAggInstance.
53                         flashingImages;
54
55         for (int i = 0; i < flashImagesArray.Length; i++)
56         {
57             bool shouldBeEnabled = false;
58
59             for (int j = 0; j < lightPositions.Length; j++)
60             {
61                 if (flashImagesArray[i].GetComponent<
62                     flashingLight>().lightPosition ==
63                     lightPositions[j]) shouldBeEnabled = true;
64             }
65
66             if (shouldBeEnabled)
67             {
68                 flashImagesArray[i].GetComponent<flashingLight>()
69                     .StartFlash(1, 0.75f, flashColor);
70             } else
71             {
72                 flashImagesArray[i].GetComponent<flashingLight>()
73                     .StopFlash();
74             }
75         }
76     /// <summary>
77     /// Detects the position of the hitPoint relative to the user
78     /// </summary>
79     /// <param name="hitPoint">The closest point of the mesh to
80     /// the user</param>
81     /// <returns>The position of the hitPoint (<see cref="
82     /// variablesAggregator.lightPositionEnum"/>)</returns>

```

```

75     public variablesAggregator.lightPositionEnum[]
76         FindHitPointPosition(RaycastHit hitPoint)
77     {
78         List<variablesAggregator.lightPositionEnum>
79             lightPositionToreturn = new List<variablesAggregator.
80                 lightPositionEnum>();
81         Vector3 relativePosition = variableAggInstance.
82             userPosition.transform.InverseTransformPoint(hitPoint
83                 .point);
84
85         if (relativePosition.x > 0.3f)
86         {
87             lightPositionToreturn.Add(variablesAggregator.
88                 lightPositionEnum.Right);
89         }
90         else if (relativePosition.x < -0.3f)
91         {
92             lightPositionToreturn.Add(variablesAggregator.
93                 lightPositionEnum.Left);
94         }
95
96         if (relativePosition.y > 0)
97         {
98             lightPositionToreturn.Add(variablesAggregator.
99                 lightPositionEnum.Up);
100        }
101    else
102    {
103        lightPositionToreturn.Add(variablesAggregator.
104            lightPositionEnum.Down);
105    }
106
107    return lightPositionToreturn.ToArray();
108 }
109
110 /// <summary>
111 /// Based on the distance, it returns the color of the flash
112 /// which will be applied to the gameObjects.
113 /// There are three possible colors to be applied:
114 /// <list type="bullet">
115 ///     <item>
116 ///         <description>Red: If the distance of the hitPoint
117 ///         is less than <see cref="variablesAggregator.
118 /// dangerIndicatorThread">dangerIndicatorThread</see></
119 ///         description>
120 ///     </item>
121 ///     <item>
122 ///         <description>
123 ///             Yellow: If the distance of the hitPoint is
124 ///             more than <see cref="variablesAggregator.
125 /// dangerIndicatorThread">dangerIndicatorThread</see>,
126 ///             but less than <see cref="variablesAggregator.
127 /// warningIndicatorThread">warningIndicatorThread</see>
128 ///         </description>
129 ///     </item>
130 ///     <item>

```

```

115     ///          <description>
116     ///          Yellow: If the distance of the hitPoint is
117     ///          more than <see cref="variablesAggregator.
118     ///          warningIndicatorThread">warningIndicatorThread</see>,
119     ///          but less than <see cref="variablesAggregator.
120     ///          distanceThreadToActivate">distanceThreadToActivate</see>
121     ///          </description>
122     ///          </item>
123     ///      </list>
124     ///  </summary>
125     ///  <param name="distance"></param>
126     ///  <returns></returns>
127     public Color FindColorBasedOnDistance(float distance)
128     {
129         Color dangerColor = variableAggInstance.dangerColor;
130         Color warningColor = Color.yellow;
131         Color safeColor = Color.green;
132
133         if (distance >= 0 && distance <= variableAggInstance.
134             dangerIndicatorThread)
135         {
136             return dangerColor;
137         }
138         else if (distance > variableAggInstance.
139             dangerIndicatorThread && distance <=
140             variableAggInstance.warningIndicatorThread)
141         {
142             return warningColor;
143         }
144         else if (distance > variableAggInstance.
145             warningIndicatorThread && distance <
146             variableAggInstance.distanceThreadToActivate)
147         {
148             return safeColor;
149         }

```

Κώδικας A.6: Αρχείο initiators/initRumbleHandler.cs

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.InputSystem;
5
6 [RequireComponent(typeof(initCastAndBoxesHandler))]
7 public class initRumbleHandler : MonoBehaviour
8 {
9

```

```

10     private variablesAggregator variableAggInstance;
11     private variablesAggregator.RumbleModes rumbleModeSelected;
12     // private PlayerInput _playerInput;
13     private Gamepad gamepad = null;
14     // private float rumbleDuration;
15     private float pulseDuration = 0.0f;
16     private float stopDuration = 0.0f;
17     private float lowA = 0.0f;
18     private float highA = 0.0f;
19     private float rumbleStep = 0.0f;
20     private float stopStep = 0.0f;
21     private bool isMotorActive = false;
22     private bool isRumbleCurrentlyActive = false;
23
24     private void Awake()
25     {
26         // _playerInput = GetComponent<PlayerInput>();
27     }
28
29     // Start is called before the first frame update
30     void Start()
31     {
32         variableAggInstance = this.GetComponent<
33             initCastAndBoxesHandler>().variableAggregatorObject.
34             GetComponent<variablesAggregator>();
35         rumbleModeSelected = variableAggInstance.
36             rumbleModeSlected;
37         // rumbleDuration = variableAggInstance.rumbleDuration;
38         gamepad = GetGamepad();
39     }
40
41     /// <summary>
42     /// Detects and returns the current gamepad
43     /// </summary>
44     /// <returns>The current gamepad</returns>
45     public Gamepad GetGamepad()
46     {
47         return Gamepad.current;
48         /*
49         Gamepad gamepadTemp = null;
50         Gamepad[] allGamepads = Gamepad.all.ToArray();
51         InputDevice[] allInputDevices = _playerInput.devices.
52             ToArray();
53         bool gamepadFound = false;
54
55         for (int i = 0; i < allGamepads.Length; i++)
56         {
57             for (int j = 0; j < allInputDevices.Length; j++)
58             {
59                 if (allGamepads[i].deviceId == allInputDevices[j].
60                     deviceId)
61                 {
62                     gamepadTemp = allGamepads[i];
63                     break;
64                 }
65             }
66         }
67     }

```

```

61             if (gamepadFound) break;
62         }
63
64     return gamepadTemp;
65     /**
66  */
67 }
68
69     /// <summary>
70     /// Disables the rumble effect in the current gamepad
71     /// </summary>
72     public void StopRumble()
73 {
74     Gamepad gamepad = GetGamepad();
75
76     if (gamepad != null && isRumbleCurrentlyActive)
77     {
78         isRumbleCurrentlyActive = false;
79         gamepad.SetMotorSpeeds(0, 0);
80         lowA = 0;
81         highA = 0;
82         rumbleStep = 0;
83         stopStep = 0;
84         pulseDuration = 0;
85         stopDuration = 0;
86     }
87 }
88
89     /// <summary>
90     /// Activates the rumble with a pulse effect in the current
91     /// gamepad
92     /// </summary>
93     /// <param name="low">Speed of low-frequency motor</param>
94     /// <param name="high">Speed of high-frequency motor</param>
95     /// <param name="stopTime">The time between each rumble (in
96     /// seconds)</param>
97     /// <param name="burstTime">The duration of the rumble (in
98     /// seconds)</param>
99     public void RumblePulse(float low, float high, /*float
100     duration,*/ float stopTime = 0.5f, float burstTime = 0.5f
101     )
102 {
103     if (rumbleModeSelected == variablesAggregator.RumbleModes
104         .Pulse)
105     {
106         isRumbleCurrentlyActive = true;
107         lowA = low;
108         highA = high;
109         rumbleStep = burstTime;
110         stopStep = stopTime;
111         if (pulseDuration == 0.0f)
112         {
113             pulseDuration = Time.time + burstTime;
114             isMotorActive = true;
115             if (gamepad != null) gamepad.SetMotorSpeeds(lowA,
116                 highA);

```

```

110         }
111         // rumbleDuration = Time.time + duration
112
113         // Invoke(nameof(StopRumble), duration);
114     }
115 }
116
117 /// <summary>
118 /// Based on the distance of the hitPoint, the speed of the
119 /// motors are calculated.
120 /// The speed is inversely proportional of the distance of the
121 /// user from the hitPoint
122 /// </summary>
123 /// <param name="distance"></param>
124 /// <returns></returns>
125 public float FindMotorsLowAndHigh (float distance)
126 {
127     return 1.0f - Mathf.Floor((distance / variableAggInstance
128     .maxDistance) * 100.0f) / 100.0f;
129 }
130
131 /// <summary>
132 /// Based on the distance of the hitPoint, it calculates the
133 /// duration between each rumble
134 /// </summary>
135 /// <param name="hitPointDistance">The distance of the
136 /// hitPoint from the user</param>
137 /// <returns>The stop time</returns>
138 public float GetStopFrequency (float hitPointDistance)
139 {
140     float stopStep = 0.0f;
141
142     if (0 <= hitPointDistance && hitPointDistance <=
143         variableAggInstance.dangerIndicatorThread)
144     {
145         stopStep = 0.25f;
146     }
147     else if (variableAggInstance.dangerIndicatorThread <
148             hitPointDistance && hitPointDistance <=
149             variableAggInstance.warningIndicatorThread)
150     {
151         stopStep = 1.0f;
152     }
153     else if (variableAggInstance.warningIndicatorThread <
154             hitPointDistance && hitPointDistance <=
155             variableAggInstance.distanceThreadToActivate)
156     {
157         stopStep = 2.0f;
158     }
159
160     return stopStep;
161 }
162
163 /// <summary>
164 /// Applies the speed to the correct motor based on the
165 /// position of the hitPoint relative to the user

```

```

155     /// </summary>
156     /// <param name="hitPoint">The closest point of an obstacle
157     // relative to the user</param>
158     /// <param name="motorSpeed">The speed, which will be applied
159     // to one of the motors</param>
160     /// <returns>A dictionary which dictates what the speed of
161     // each motor should be</returns>
162     public Dictionary<string, float> GetSpeedOfEachMotor(
163         RaycastHit hitPoint, float motorSpeed)
164     {
165         Vector3 relativePosition = variableAggInstance.
166             userPosition.transform.InverseTransformPoint(hitPoint
167             .point);
168         Dictionary<string, float> motorToEnable = new Dictionary<
169             string, float>()
170         {
171             {"low", 0.0f},
172             {"high", 0.0f}
173         };
174         // float floorX = Mathf.Floor(relativePosition.x * 100.0f
175         // ) / 100.0f;
176
177         if (relativePosition.x >= 0.0f)
178         {
179             motorToEnable["high"] = motorSpeed;
180         }
181         if (relativePosition.x <= 0.0f)
182         {
183             motorToEnable["low"] = motorSpeed;
184         }
185
186         return motorToEnable;
187     }
188
189     /// <summary>
190     /// Toggles the activation of the rumble effect based on the
191     // button pressed by the user on the gamepad
192     /// </summary>
193     /// <remarks>
194     /// <para>If the user presses A on the gamepad, the rumble
195     // effect will be activated (if it was deactivated before)
196     // .</para>
197     /// <para>If the user presses B on the gamepad, the rumble
198     // effect will be deactivated (if it was activated before)
199     // .</para>
200     /// </remarks>
201     private void rumbleToggleGamepad()
202     {
203         if (gamepad.bButton.wasPressedThisFrame &&
204             variableAggInstance.activateRumble)
205         {
206             variableAggInstance.activateRumble = false;
207             StopRumble();
208         }
209         else if (gamepad.aButton.wasPressedThisFrame && !
210             variableAggInstance.activateRumble)
211         {
212             variableAggInstance.activateRumble = true;
213             StartRumble();
214         }
215     }

```

```

197         variableAggInstance.activateRumble)
198     {
199     variableAggInstance.activateRumble = true;
200   }
201
202   private void Update()
203   {
204     gamepad = GetGamepad();
205     if (gamepad == null) return;
206
207     rumbleToggleGamepad();
208
209     if (!variableAggInstance.activateRumble) return;
210
211     // if (Time.time > rumbleDuration) return;
212
213     if (!isRumbleCurrentlyActive) return;
214
215     switch (rumbleModeSelected)
216     {
217       /*case variablesAggregator.RumbleModes.Constant:
218       break;*/
219
220       case variablesAggregator.RumbleModes.Pulse:
221         if (isMotorActive && Time.time > pulseDuration)
222         {
223           isMotorActive = !isMotorActive;
224           stopDuration = Time.time + stopStep;
225           gamepad.SetMotorSpeeds(0, 0);
226         } else if (!isMotorActive && Time.time >
227                     stopDuration)
228         {
229           isMotorActive = !isMotorActive;
230           pulseDuration = Time.time + rumbleStep; //(
231             Update pulseDuration
232             gamepad.SetMotorSpeeds(lowA, highA);
233         }
234         break;
235
236       /*case variablesAggregator.RumbleModes.Linear:
237       break;*/
238     }
239   }
240 }
```

Kώδικας A.7: Αρχείο castModes/scanCast.cs

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 [RequireComponent(typeof(initCastAndBoxesHandler)),
```

```

    RequireComponent(typeof(initBoxes))]
6 public class scanCast : MonoBehaviour
7 {
8     private variablesAggregator variableAggInstance;
9
10    void Start()
11    {
12        variableAggInstance = this.GetComponent<
13            initCastAndBoxesHandler>().variableAggregatorObject.
14            GetComponent<variablesAggregator>();
15    }
16
17    public void EnableScanMode()
18    {
19        GameObject[] allCastBoxes = variableAggInstance.
20            raycastBoxes;
21        List<RaycastHit> castHitPointsList = new List<RaycastHit
22            >();
23
24        for (int i = 0; i < allCastBoxes.Length; i++)
25        {
26            RaycastHit hitPoint = allCastBoxes[i].GetComponent<
27                singleRaycast>().SingleRaycastFunc(
28                variableAggInstance);
29            castHitPointsList.Add(hitPoint);
30        }
31
32        //ToArray doesn't change the order of the elements
33        int hitPointIndex = this.GetComponent<initBoxes>().
34            findClosestHitPointIndex(castHitPointsList.ToArray())
35        ;
36        if (hitPointIndex != -1)
37        {
38            this.GetComponent<initBoxes>().placeAlertBox(
39                variableAggInstance.commonAlertBox, allCastBoxes[
40                    hitPointIndex], castHitPointsList[hitPointIndex],
41                    variableAggInstance.placeAlertBoxAtEyeLevel);
42        }
43        else
44        {
45            switch (variableAggInstance.alertBoxType)
46            {
47                case variablesAggregator.AlertBoxTypeEnum.
48                    BoxSpecific:
49                    // Use the alert box of each castBox
50                    this.GetComponent<initBoxes>().
51                        deactivateAlertBox(variableAggInstance.
52                            alertBoxes);
53                    break;
54
55                case variablesAggregator.AlertBoxTypeEnum.Common:
56                    // Use a common alert for all castBoxes
57                    this.GetComponent<initBoxes>().
58                        deactivateAlertBox(variableAggInstance.
59                            commonAlertBox);
60                    break;
61
62            }
63        }
64    }
65
66    void Update()
67    {
68        if (Input.GetKeyDown(KeyCode.Space))
69        {
70            EnableScanMode();
71        }
72    }
73}

```

```

45
46     case variablesAggregator.AlertBoxTypeEnum.
47         Peripheral:
48             this.GetComponent<initBoxes>().
49                 deactivateAlertBox(variableAggInstance.
50                     peripheralAlertBox);
51         break;
52     }

```

Κώδικας A.8: Αρχείο castModes/continuousCast.cs

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5
6 [RequireComponent(typeof(initCastAndBoxesHandler)),
7     RequireComponent(typeof(initBoxes)),
8     RequireComponent(typeof(initFlashHandler)), RequireComponent(
9         typeof(initRumbleHandler))]
10 public class continuousCast : MonoBehaviour
11 {
12     private variablesAggregator variableAggInstance;
13     private bool enableContinuousModeGlobal;
14     private GameObject[] castBoxesWithContinuous;
15     private initFlashHandler flashHandler;
16
17     // Start is called before the first frame update
18     void Start()
19     {
20         variableAggInstance = this.GetComponent<
21             initCastAndBoxesHandler>().variableAggregatorObject.
22             GetComponent<variablesAggregator>();
23         enableContinuousModeGlobal = variableAggInstance.
24             enableContinuousModeGlobal;
25         castBoxesWithContinuous = this.GetComponent<initBoxes>().
26             findCastBoxesWithContinuousEnabled(
27                 variableAggInstance.raycastBoxes);
28         flashHandler = this.GetComponent<initFlashHandler>();
29     }
30
31     // Update is called once per frame
32     void Update()
33     {
34         if (enableContinuousModeGlobal)
35         {
36             List<RaycastHit> hitPointsList = new List<RaycastHit>();
37
38             for (int i = 0; i < castBoxesWithContinuous.Length; i
39                ++)
40             {
41                 RaycastHit hit;
42
43                 if (Physics.Raycast(castBoxesWithContinuous[i].transform.
44                     position, castBoxesWithContinuous[i].transform.
45                     forward, out hit))
46                 {
47                     hitPointsList.Add(hit);
48                 }
49             }
50
51             foreach (RaycastHit hit in hitPointsList)
52             {
53                 Vector3 raycastHitPosition = hit.point;
54
55                 if (hit.distance < 0.5f)
56                 {
57                     Vector3 raycastHitNormal = hit.normal;
58
59                     if (raycastHitNormal.y > 0.5f)
60                     {
61                         flashHandler.flash();
62                     }
63                 }
64             }
65         }
66     }
67 }

```

```

33         RaycastHit hitPoint = castBoxesWithContinuous[i].
34             GetComponent<singleRaycast>().
35             SingleRaycastFunc(variableAggInstance);
36         hitPointsList.Add(hitPoint);
37     }
38     int hitPointIndex = this.GetComponent<initBoxes>().
39         findClosestHitPointIndex(hitPointsList.ToArray());
40     ;
41     if (hitPointIndex != -1)
42     {
43         switch (variableAggInstance.alertBoxType)
44         {
45             case variablesAggregator.AlertBoxTypeEnum.
46                 BoxSpecific:
47                 // Use the alert box of each castBox
48                 this.GetComponent<initBoxes>().
49                     placeAlertBox(castBoxesWithContinuous
50                     [hitPointIndex], hitPointsList[
51                     hitPointIndex]);
52                     break;
53
54             case variablesAggregator.AlertBoxTypeEnum.
55                 Common:
56                 // Use a common alert for all castBoxes
57                 this.GetComponent<initBoxes>().
58                     placeAlertBox(variableAggInstance.
59                     commonAlertBox,
60                     castBoxesWithContinuous[hitPointIndex]
61                     , hitPointsList[hitPointIndex],
62                     variableAggInstance.
63                     placeAlertBoxAtEyeLevel);
64                     break;
65
66             case variablesAggregator.AlertBoxTypeEnum.
67                 Peripheral:
68                 this.GetComponent<initBoxes>().
69                     placePeripheralAlertBox(
70                     variableAggInstance.
71                     peripheralAlertBox, hitPointsList [
72                     hitPointIndex]);
73                     break;
74     }
75
76     // Debug.Log(hitPointsList[hitPointIndex].
77     // distance);
78     print(hitPointIndex + ": " + hitPointsList [
79         hitPointIndex].point);
80
81     if (variableAggInstance.activateRumble)
82     {
83         float motorSpeed = this.GetComponent<
84             initRumbleHandler>().FindMotorsLowAndHigh
85             (hitPointsList[hitPointIndex].distance);
86         float stopFreq = this.GetComponent<
87             initRumbleHandler>().GetStopFrequency(
88             hitPointsList[hitPointIndex].distance);

```

```

63             Dictionary<string, float> motorSpeedDict =
64                 this.GetComponent<initRumbleHandler>().
65                     GetSpeedOfEachMotor(hitPointsList[
66                         hitPointIndex], motorSpeed);
67
68             print("Motor speed: " + motorSpeed);
69             if (motorSpeed > 0.2f)
70             {
71                 this.GetComponent<initRumbleHandler>().
72                     RumblePulse(motorSpeedDict["low"],
73                         motorSpeedDict["high"], stopFreq, 0.5
74                         f);
75             }
76             else
77             {
78                 this.GetComponent<initRumbleHandler>().
79                     StopRumble();
80             }
81         }
82
83
84         /**
85         if (variableAggInstance.activateFlash)
86         {
87             if (hitPointsList[hitPointIndex].distance <=
88                 variableAggInstance.
89                 distanceThreadToActivate)
90             {
91                 variablesAggregator.lightPositionEnum[]
92                     lightPositionsToStart = flashHandler.
93                         FindHitPointPosition(hitPointsList[
94                             hitPointIndex]);
95                 Color colorToUse = flashHandler.
96                     FindColorBasedOnDistance(
97                         hitPointsList[hitPointIndex].distance
98                         );
99                 flashHandler.StartMultipleFlashes(
99                     lightPositionsToStart, colorToUse);
100             } else
101             {
102                 flashHandler.StopFlashes();
103             }
104         }
105         /**
106     } else
107     {
108         this.GetComponent<initRumbleHandler>().StopRumble
109             ();
110         switch (variableAggInstance.alertBoxType)
111         {
112             case variablesAggregator.AlertBoxTypeEnum.
113                 BoxSpecific:
114                 // Use the alert box of each castBox
115                 this.GetComponent<initBoxes>().
116                     deactivateAlertBox(
117

```

```

100           castBoxesWithContinuous);
101           break;
102
103       case variablesAggregator.AlertBoxTypeEnum.
104           Common:
105           // Use a common alert for all castBoxes
106           this.GetComponent<initBoxes>().
107               deactivateAlertBox(
108                   variableAggInstance.commonAlertBox);
109           break;
110
111       case variablesAggregator.AlertBoxTypeEnum.
112           Peripheral:
113           this.GetComponent<initBoxes>().
114               deactivateAlertBox(
115                   variableAggInstance.
116                       peripheralAlertBox);
117           break;
118
119       }
120   }
121
122   /**
123    * Toggles the continuous mode on or off
124    */
125   public void continuousModeGlobalToggle()
126   {
127       enableContinuousModeGlobal = !enableContinuousModeGlobal;
128
129       if (enableContinuousModeGlobal == true)
130       {
131           // Position of Continuous Mode
132           castBoxesWithContinuous = this.GetComponent<initBoxes
133               >().findCastBoxesWithContinuousEnabled(
134                   variableAggInstance.raycastBoxes);
135
136           this.GetComponent<initBoxes>().positionAndScaleBoxes(
137               castBoxesWithContinuous,
138               variableAggInstance.boxesPerRow,
139               variableAggInstance.numberOfWorks,
140               this.GetComponent<initBoxes>().userHeight
141           );
142       } else
143       {
144           // Disable alert boxes of cast boxes in continuous
145           mode
146           this.GetComponent<initBoxes>().deactivateAlertBox(
147               castBoxesWithContinuous);
148           this.GetComponent<initBoxes>().deactivateAlertBox(
149               variableAggInstance.commonAlertBox);
150
151           /*for (int i = 0; i < castBoxesWithContinuous.Length;
152               i++)
153
154           */
155
156       }
157   }

```

```

141         {
142             if (castBoxesWithContinuous[i].GetComponent<
143                 initSingleBox>().alertBox.activeSelf == true)
144             {
145                 castBoxesWithContinuous[i].GetComponent<
146                     initSingleBox>().alertBox.SetActive(false
147                     );
148             }
149         }/*
150
151         // Position of Scan Mode
152         this.GetComponent<initBoxes>().positionAndScaleBoxes(
153             variableAggInstance.raycastBoxes,
154             3,
155             3,
156             this.GetComponent<initBoxes>().userHeight
157         );
158     }

```

Κώδικας A.9: Αρχείο castModes/handRaycast.cs

```

1 using Microsoft.MixedReality.Toolkit;
2 using Microsoft.MixedReality.Toolkit.Input;
3 using System.Collections.Generic;
4 using UnityEngine;
5
6 [RequireComponent(typeof(initCastAndBoxesHandler))]
7 public class handRaycast : MonoBehaviour
8 {
9     private variablesAggregator variableAggInstance;
10    private GameObject[] handAlertBoxes;
11    private bool enabledHandAB;
12    private Dictionary<string, GameObject> handAlertBoxesDict =
13        new Dictionary<string, GameObject>();
14
15    public void Start()
16    {
17        variableAggInstance = this.GetComponent<
18            initCastAndBoxesHandler>().variableAggregatorObject.
19            GetComponent<variablesAggregator>();
20        handAlertBoxes = variableAggInstance.handAlertBoxes;
21        enabledHandAB = variableAggInstance.enabledHandAB;
22
23        foreach (var handAlertBox in handAlertBoxes)
24        {
25            if (!handAlertBox.CompareTag("Untagged"))
26            {
27                handAlertBoxesDict.Add(handAlertBox.tag,
28                    handAlertBox);
29            }
30        }
31    }

```

```

28
29     // Update is called once per frame
30     void Update()
31     {
32         enabledHandAB = variableAggInstance.enabledHandAB;
33         if (enabledHandAB == true)
34         {
35             var isHandPresent = false;
36
37             foreach (var source in CoreServices.InputSystem.
38                     DetectedInputSources)
39             {
40                 // Ignore anything that is not a hand because we
41                 // want articulated hands
42                 if (source.SourceType == InputSourceType.Hand &&
43                     source.SourceName != "None Hand")
44                 {
45                     isHandPresent = true;
46                     foreach (var p in source.Pointers)
47                     {
48                         if (p is IMixedRealityNearPointer)
49                         {
50                             // Ignore near pointers, we only want
51                             // the rays
52                             continue;
53                         }
54                         if (p.Result != null)
55                         {
56                             //var startPoint = p.Position;
57                             var endPoint = p.Result.Details.Point
58                             ;
59                             var rayDist = p.Result.Details.
60                             RayDistance;
61                             //var hitObject = p.Result.Details.
62                             Object;
63                             if (rayDist <= variableAggInstance.
64                             maxDistance)
65                             {
66                                 if (handAlertBoxesDict.
67                                     ContainsKey(source.SourceName
68                                     )) {
69                                     handAlertBoxesDict[source.
70                                     SourceName].SetActive(
71                                     true);
72                                     handAlertBoxesDict[source.
73                                     SourceName].transform.
74                                     position = endPoint;
75                                 }
76                                 /*var sphere = GameObject.
77                                     CreatePrimitive(PrimitiveType
78                                     .Sphere);
79                                     sphere.transform.localScale =
80                                     Vector3.one * 0.01f;
81                                     sphere.transform.position =
82                                     endPoint;*/
83                             } else
84                         }
85                     }
86                 }
87             }
88         }
89     }
90 }
```

```

66
67          {
68              if (handAlertBoxesDict.
69                  ContainsKey(source.SourceName
70                      ))
71          {
72              handAlertBoxesDict[source.
73                  SourceName].SetActive(
74                      false);
75          }
76      }
77
78      if (p.Controller.Interactions[2].BoolData
79          ==
80              true)
81      {
82          if (handAlertBoxesDict.ContainsKey(
83              source.SourceName))
84          {
85              if (handAlertBoxesDict[source.
86                  SourceName].GetComponent<
87                      AudioSource>().isPlaying ==
88                      true)
89              {
90                  handAlertBoxesDict[source.
91                      SourceName].GetComponent<
92                      AudioSource>().Pause();
93              }
94          }
95      }
96
97      if (!isHandPresent)
98      {
99          foreach (var handAlertBox in handAlertBoxes)
100          {
101              handAlertBox.SetActive(false);
102          }
103      }

```

```

104         }
105     }
106 }
107
108 public void enabledHandABToggle()
109 {
110     enabledHandAB = !enabledHandAB;
111
112     if (enabledHandAB == false)
113     {
114         foreach (var handAlertBox in handAlertBoxes)
115         {
116             handAlertBox.SetActive(false);
117         }
118     }
119 }
120 }
```

Kώδικας A.10: Αρχείο castModes/stopCast.cs

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 [RequireComponent(typeof(initCastAndBoxesHandler)),
6  RequireComponent(typeof(continuousCast))]
6 public class stopCast : MonoBehaviour
7 {
8     private variablesAggregator variableAggInstance;
9
10    void Start()
11    {
12        variableAggInstance = this.GetComponent<
13            initCastAndBoxesHandler>().variableAggregatorObject.
14            GetComponent<variablesAggregator>();
15    }
16    public void EnableStopMode()
17    {
18        if (variableAggInstance.enableContinuousModeGlobal ==
19            true)
20        {
21            this.GetComponent<continuousCast>().
22                continuousModeGlobalToggle();
23        }
24
25        GameObject[] allCastBoxes = variableAggInstance.
26            raycastBoxes;
27        this.GetComponent<initBoxes>().deactivateAlertBox(
28            allCastBoxes);
29        this.GetComponent<initBoxes>().deactivateAlertBox(
30            variableAggInstance.commonAlertBox);
31        /*for (int i = 0; i < allCastBoxes.Length; i++)
32        {
33            if (allCastBoxes[i].GetComponent<initSingleBox>().
34                alertBox.activeSelf)
```

```

27         {
28             allCastBoxes[i].GetComponent<initSingleBox>().
29                 alertBox.SetActive(false);
30         }
31     }*/
32 }

```

Κώδικας A.11: Αρχείο singleRaycast.cs

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 [RequireComponent(typeof(initSingleBox))]
6 public class singleRaycast : MonoBehaviour
7 {
8     // public GameObject variableAggregatorObject;
9     /*public GameObject handlerObject;*/
10    // private variablesAggregator variableAggInstance;
11
12    private bool m_HitDetect;
13    private RaycastHit hitInfo;
14
15    public void Start()
16    {
17        // variableAggInstance = variableAggregatorObject.
18        // GetComponent<variablesAggregator>();
19    }
20
21    /// <summary>
22    ///     Casts a raycast or boxcast from the position of a
23    ///     castBox and detects a hitPoint.
24    /// </summary>
25    /// <param name="variableAggInstance">The variable aggregator
26    /// GameObjetc</param>
27    /// <returns>
28    ///     The hitPoint of the cast
29    /// </returns>
30    public RaycastHit SingleRaycastFunc(variablesAggregator
31                                         variableAggInstance)
32    {
33        Vector3 fwd = transform.TransformDirection(Vector3.
34                                                 forward); // forward direction
35
36        float maxDistance = variableAggInstance.maxDistance;
37
38        variablesAggregator.CastTypeEnum castType =
39            variableAggInstance.castType;
40
41        switch (castType) {
42            case variablesAggregator.CastTypeEnum.Raycast:
43                m_HitDetect = Physics.Raycast(transform.position,
44                                              fwd, out hitInfo, 5.0f);
45                break;

```

```

39
40         case variablesAggregator.CastTypeEnum.BoxCast:
41             m_HitDetect = Physics.BoxCast(transform.position,
42                                           transform.localScale, fwd, out hitInfo,
43                                           transform.rotation, maxDistance);
44             break;
45     }
46     return hitInfo;
47 }
48 /*
49 void OnDrawGizmos()
50 {
51     Gizmos.color = Color.red;
52
53     //Check if there has been a hit yet
54     if (m_HitDetect)
55     {
56         //Draw a Ray forward from GameObject toward the hit
57         Gizmos.DrawRay(transform.position, transform.forward
58                         * hitInfo.distance);
59         //Draw a cube that extends to where the hit exists
60         Gizmos.DrawWireCube(transform.position + transform.
61                             forward * hitInfo.distance, transform.localScale)
62                         ;
63     }
64     //If there hasn't been a hit yet, draw the ray at the
65     // maximum distance
66     else
67     {
68         //Draw a Ray forward from GameObject toward the
69         // maximum distance
70         // Gizmos.DrawRay(transform.position, transform.
71                         forward * m_MaxDistance);
72         //Draw a cube at the maximum distance
73         // Gizmos.DrawWireCube(transform.position + transform
74                         .forward * m_MaxDistance, transform.localScale);
75     }
76 }
77 */
78 }
```

Kώδικας A.12: Αρχείο **flashingLight.cs**

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5
6 public class flashingLight : MonoBehaviour
7 {
8     public GameObject variableAggregatorObject;
9     private variablesAggregator variableAggInstance;
10    [Tooltip("Position of the GameObject relative to the user")]
11 }
```

```

11     public variablesAggregator.lightPositionEnum lightPosition;
12     private IEnumerator currentFlashRoutine = null;
13     private bool shouldFlashStop = false;
14     private MaterialPropertyBlock newColor;
15
16     void Start()
17     {
18         variableAggInstance = variableAggregatorObject.
19             GetComponent<variablesAggregator>();
20         newColor = new MaterialPropertyBlock();
21     }
22
23     /// <summary>
24     /// Starts the flash effect for the specific gameObject
25     /// </summary>
26     /// <param name="flashInterval">The duration of the flash
27     /// effect</param>
28     /// <param name="maxAlpha">The max value of the alpha (max
29     /// opacity)</param>
30     /// <param name="colorOfTheFlash">The color of the flash</
31     /// param>
32     public void StartFlash(float flashInterval, float maxAlpha,
33                             Color colorOfTheFlash)
34     {
35
36         newColorSetColor("_Color", colorOfTheFlash);
37         this.GetComponent<Renderer>().SetPropertyBlock(newColor);
38
39         // 0 <= maxAlpha <= 1
40         maxAlpha = Mathf.Clamp(maxAlpha, 0, 1);
41
42         if (currentFlashRoutine == null)
43         {
44             // StopCoroutine(currentFlashRoutine);
45             currentFlashRoutine = Flash(flashInterval, maxAlpha);
46             shouldFlashStop = false;
47             StartCoroutine(currentFlashRoutine);
48         }
49     }
50
51     /// <summary>
52     /// Stops the flash effect for the sepcific gameObject
53     /// </summary>
54     public void StopFlash()
55     {
56         shouldFlashStop = true;
57         if (currentFlashRoutine != null)
58         {
59             StopCoroutine(currentFlashRoutine);
60             currentFlashRoutine = null;
61
62             newColorSetColor("_Color", new Color(0, 0, 0, 0));
63             this.GetComponent<Renderer>().SetPropertyBlock(
64                 newColor);
65         }
66     }

```

```

61     }
62
63     /// <summary>
64     /// Applies the flash effect to the gameObject. The effect is
65     /// looped.
66     /// </summary>
67     /// <param name="flashInterval">The duration of one flash (in
68     /// seconds)</param>
69     /// <param name="maxAlpha">The max value of the alpha</param>
70     /// <returns></returns>
71     IEnumerator Flash(float flashInterval, float maxAlpha)
72     {
73         // Flash In
74         float flashInDuration = flashInterval / 2;
75
76         while(!shouldFlashStop)
77         {
78             for (float t = 0; t <= flashInDuration; t += Time.
79                 deltaTime)
80             {
81                 Color colorThisFrame = newColor.GetColor("_Color"
82                     );
83                 colorThisFrame.a = Mathf.Lerp(0, maxAlpha, t /
84                     flashInDuration);
85                 newColor.SetColor("_Color", colorThisFrame);
86                 this.GetComponent<Renderer>().SetPropertyBlock(
87                     newColor);
88
89                 // wait until next frame
90                 yield return null;
91             }
92
93             // Flash Out
94             float flashOutDuration = flashInterval / 2;
95             for (float t = 0; t <= flashOutDuration; t += Time.
96                 deltaTime)
97             {
98                 Color colorThisFrame = newColor.GetColor("_Color"
99                     );
100                colorThisFrame.a = Mathf.Lerp(maxAlpha, 0, t /
101                    flashOutDuration);
102                newColor.SetColor("_Color", colorThisFrame);
103                this.GetComponent<Renderer>().SetPropertyBlock(
104                    newColor);
105            }
106        }
107
108        // ensure alpha is 0
109        newColor.SetColor("_Color", new Color(0, 0, 0, 0));
110        this.GetComponent<Renderer>().SetPropertyBlock(newColor);
111    }
112 }

```

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using Microsoft.MixedReality.Toolkit;
4 using Microsoft.MixedReality.Toolkit.SpatialAwareness;
5 using UnityEngine;
6
7 public class hideMesh : MonoBehaviour
8 {
9     public bool hideMeshAtStart = false;
10
11    private void Start()
12    {
13        if (hideMeshAtStart == true)
14        {
15            hideMeshFunc();
16        }
17    }
18
19    // Start is called before the first frame update
20    public void hideMeshFunc()
21    {
22        var observer = CoreServices.
23            GetSpatialAwarenessSystemDataProvider<
24                IMixedRealitySpatialAwarenessMeshObserver>();
25
26        observer.DisplayOption =
27            SpatialAwarenessMeshDisplayOptions.None;
28    }
29
30    public void showMeshFunc()
31    {
32        var observer = CoreServices.
33            GetSpatialAwarenessSystemDataProvider<
34                IMixedRealitySpatialAwarenessMeshObserver>();
35
36        observer.DisplayOption =
37            SpatialAwarenessMeshDisplayOptions.Visible;
38    }
39 }
```

ПАР'АРТНМА В'

Ερωτηματολόγιο

Πανεπιστήμιο Πατρών, Πολυτεχνική Σχολή
Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών
Άγρελος Καρδούτσος του Απόστολου
© Φεβρουάριος 2024 – Με την επιφύλαξη παντός δικαιώματος.