

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, 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 Hololens	12
2.3.1 Περιγραφή Συσκευής	12
2.3.2 Τεχνικά Χαρακτηριστικά	12
2.3.3 Τρόποι Αλληλεπίδρασης	13
2.3.4 Spatial Mapping	13
2.3.5 Spatial Audio	13
2.4 Εργαλεία ανάπτυξης για Hololens	13
2.4.1 Unity	13
2.4.2 Microsoft Visual Studio	13

2.4.3 Mixed Reality Toolkit	13
3 Η υλοποίηση	15
3.1 Σενάριο Εφαρμογής	16
3.2 Σχεδιασμός και Περιορισμοί	16
3.3 Γλοποίηση	16
3.4 Λειτουργίες Εφαρμογής	16
4 Αξιολόγηση	17
4.1 Διαδικασία/Περιγραφή Πειράματος	18
4.2 Αποτελέσματα	18
5 Προεκτάσεις και Επίλογος	19
5.1 Μελλοντικές προεκτάσεις	20
5.2 Επίλογος	20
Βιβλιογραφία	21
Παράρτημα Α'	27
Παράρτημα Β'	57

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

ΚΑΤΑΛΟΓΟΣ ΠΙΝ'ΑΚΩΝ

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 ροή του ζωής 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

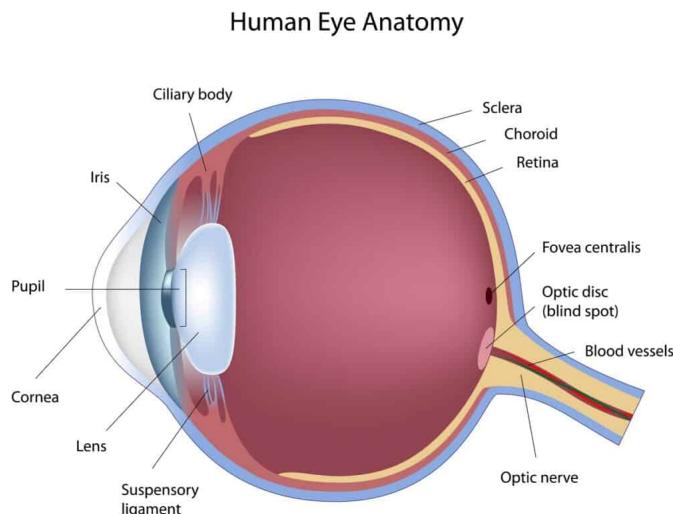
ΘΕΩΡΗΤΙΚΟ ΚΑΙ
ΤΕΧΝΟΛΟΓΙΚΟ ΥΠΟΒΑΘΡΟ

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), καθώς 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 προτροπή της απώλειας της ορασης 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 ορασής 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 πρβλήματα ορασης, 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.

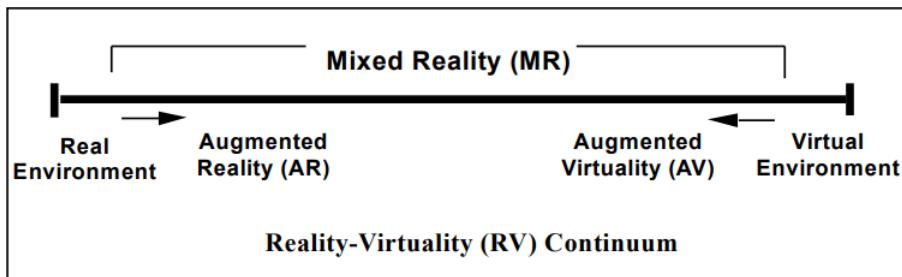
Στη σύγχρονή εποχή, όλο και περισσότερες συσκευές κατασκευάζονται λαμβάνοντας προκαταβολικά υπόψη τη δυνατότητα χρήσης αυτών από άτομα με περιορισμένη όραση. Οι κινητές συσκευές διαθέτουν λογισμικό screen reader (π.χ. TalkBack σε Android συσκεύες ή VoiceOver σε iOS συσκευές), διευκολύνοντας την πλοιήγηση 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, υπερσύνδεσμοι, 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. [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 κλίμακας, 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 ενισχημένη μορφή της τεχνολογίας επαυξημένης πραγματικότητας, όπου τα εικονικά αντικείμενα αλληλεπιδρούν με τα πραγματικά 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 Pro¹ 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 Hololens

2.3.1 Περιγραφή Συσκευής

Description of the device

2.3.2 Τεχνικά Χαρακτηριστικά

Hololens features

¹To headset μικτής πραγματικότητας Apple Vision Pro βρίσκεται σε διαδικάσια παραγωγής και η διάθεση του στην αγορά αναμένεται στις 02/02/2024. Την παρούσα χρονική στιγμή, μόνο μια παρουσίαση της εταιρείας είναι διαθέσιμη, η οποία επιδεικνύει τις δυνατότητες της συσκεύης.

2.3.3 Τρόποι Αλληλεπίδρασης

Τρόποι αλληλεπίδρασης

2.3.4 Spatial Mapping

2.3.5 Spatial Audio

2.4 Εργαλεία ανάπτυξης για Hololens

2.4.1 Unity

Περιγραφή περιβάλλοντος

2.4.2 Microsoft Visual Studio

Μικρή περιγραφή της χρήσης του

2.4.3 Mixed Reality Toolkit

ΚΕΦΑΛΑΙΟ

3

Η ΥΛΟΠΟΙΗΣΗ

Περιγραφή των ενοτήτων

3.1 Σενάριο Εφαρμογής

Στόχος εφαρμογής

Περιγραφή σεναρίου χρήσης εφαρμογής

3.2 Σχεδιασμός και Περιορισμοί

Αποφασείς για το τι ήταν ή δεν ήταν εφικτό να υλοποιηθεί

3.3 Γλοποίηση

Η υλοποίηση της εφαρμογής, τυηματοποιημένη

3.4 Λειτουργίες Εφαρμογής

Τρόπος χρήσης της εφαρμογής

ΚΕΦΑΛΑΙΟ

4

ΑΞΙΟΛΟΓΗΣΗ

Σκοπός της αξιολόγησης (γιατί κάνουμε αξιολόγηση)
Συμμετέχοντες
Προδιαγραφές (πως προετοιμαστήκαμε με τους εθελοντές για την αξιολόγηση)
Ερωτηματολόγια που χρησιμοποιήθηκαν

4.1 Διαδικασία/Περιγραφή Πειράματος

Τρόπος/συνθήκες διεξαγωγής
Πιθανοί περιορισμοί

4.2 Αποτελέσματα

Παρατηρήσεις
Απαντήσεις ερωτηματολογίου

ΚΕΦΑΛΑΙΟ

5

ΠΡΟΕΚΤΆΣΕΙΣ ΚΑΙ
ΕΠΙΛΟΓΟΣ

5.1 Μελλοντικές προεκτάσεις

5.2 Επίλογος

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. Arditi, 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-Movaghah, 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. Woldemariam, S. K. G. West, T. G. Y. Wondmeneh, T. Y. Wong, M. Yaseri, N. Yonemoto, C. S. Yu, M. S. Zastrozhanin, 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.blindsightsquare.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>

ПАР'АРТНМА А'

Κώδικας Εφαρμογής

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 }

```

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 }

```

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     /// castpxoxes</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
72     (

```

```

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     }
92
93     /// <summary>
94     /// In an array of RaycastHit (<paramref name="hitPointsList
95     /// "/>), it locates the index of the hitPoint, whose
96     /// distance is closer to the user.
97     /// </summary>
98     /// <param name="hitPointsList">An array of hitPoints</param>
99     /// <returns>It returns the index of the hitPoint closer to
100    /// the user</returns>
101    public int findClosestHitPointIndex(RaycastHit[]
102      hitPointsList)
103    {
104      int closestHitPointIndex = -1;
105      float minDistance = 999.0f;
106      List<Vector3> relPosList = new List<Vector3>();
107      for (int i = 0; i < hitPointsList.Length; i++)
108      {
109        if (hitPointsList[i].distance != 0.0f &&
110          hitPointsList[i].collider != null)
111        {
112          Vector3 relativePosition = variableAggInstance.
113            userPosition.transform.InverseTransformPoint(
114              hitPointsList[i].point);
115          relPosList.Add(relativePosition);
116        }
117      }
118      return closestHitPointIndex;
119    }
120  }

```

```

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.
162         GetComponent< AudioSource >().isPlaying)

```

```

152         castBox.GetComponent<initSingleBox>().alertBox.
153             GetComponent< AudioSource >().Play();
154         castBox.GetComponent<initSingleBox>().alertBox.transform.
155             position = hitPoint.point;
156         castBox.GetComponent<initSingleBox>().alertBox.transform.
157             rotation = castBox.transform.rotation;
158     }
159
160     /// <summary>
161     /// Places an <paramref name="alertBox"/> in the position of
162     /// a <paramref name="hitPoint"/>.
163     /// </summary>
164     /// <param name="alertBox">The alert box which will be
165     /// transformed</param>
166     /// <param name="castBox">The castBox whose rotation will be
167     /// used</param>
168     /// <param name="hitPoint">The hitPoint to which the alert
169     /// box will be placed</param>
170     public void placeAlertBox(GameObject alertBox, GameObject
171         castBox, RaycastHit hitPoint, bool
172         placeAlertBoxToEarLevel)
173     {
174         alertBox.SetActive(true);
175         var alertTrans = alertBox.transform;
176         if (!alertBox.GetComponent< AudioSource >().isPlaying)
177             alertBox.GetComponent< AudioSource >().Play();
178
179         /*
180         var hitPointVec = hitPoint.point;
181         var userPositionTrans = variableAggInstance.userPosition.
182             transform;
183         if (variableAggInstance.placeAlertBoxAtEyeLevel)
184         {
185             alertTrans.position = new Vector3(hitPointVec.x,
186                 userPositionTrans.position.y, hitPointVec.z);
187         }
188         else
189         {
190             alertTrans.position = hitPointVec;
191         }
192         alertBox.transform.rotation = castBox.transform.rotation;
193         */
194     }
195
196     public void placePeripheralAlertBox(GameObject alertBox,
197         RaycastHit hitPoint)
198     {
199         alertBox.SetActive(true);
200         if (!alertBox.GetComponent< AudioSource >().isPlaying)
201         {
202             alertBox.GetComponent< AudioSource >().Play();
203         }
204         Vector3 relativePosition = variableAggInstance.
205             userPosition.transform.InverseTransformPoint(hitPoint
206             .point);
207         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     /// <summary>
203     /// Deactivates all the alert boxes, except the one, whose
204     /// castBox casted and hit the point closest to the user
205     /// </summary>
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     public void deactivateAlertBoxesExceptTheClosest(GameObject[]
210         castBoxesArray, int castHitPointIndex)
211     {
212         for (int i = 0; i < castBoxesArray.Length; i++)
213         {
214             if (i != castHitPointIndex)
215             {
216                 castBoxesArray[i].GetComponent<initSingleBox>().
217                     alertBox.SetActive(false);
218             }
219         }
220     }
221
222     /// <summary>
223     /// Deactivates all the alert boxes, based on an array of
224     /// castBoxes
225     /// </summary>
226     /// <param name="castBoxesArray">The array of castBoxes,
227     /// whose alert boxes will be deactivated</param>
228     public void deactivateAlertBox(GameObject[] castBoxesArray)
229     {
230         for (int i = 0; i < castBoxesArray.Length; i++)
231         {
232             if (castBoxesArray[i].GetComponent<initSingleBox>().
233                 alertBox.GetComponent< AudioSource >().isPlaying)
234                 castBoxesArray[i].GetComponent<initSingleBox>().
235                     alertBox.GetComponent< AudioSource >().Pause();
236         }
237     }
238
239     /// <summary>
240     /// Deactivates an alert box
241     /// </summary>
242     /// <param name="alertBox">The alert box which will be
243     /// deactivated</param>
244     public void deactivateAlertBox(GameObject alertBox)
245     {
246         if (alertBox.GetComponent< AudioSource >().isPlaying)
247             alertBox.GetComponent< AudioSource >().Pause();
248     }

```

238 }

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 set to 'True', rays will be casted in every
11        frame.")]
12    public bool continuousRaycast = false;
13 }
```

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         /// <summary>
19         /// Disables the flash effect of all the gameObjects
20         /// </summary>
21         public void StopFlashes()
22     {
23         GameObject[] flashImagesArray = variableAggInstance.
24             flashingImages;
25
26         for (int i = 0; i < flashImagesArray.Length; i++)
27         {
28             flashImagesArray[i].GetComponent<flashingLight>().
29                 StopFlash();
30         }
31
32         /// <summary>
33         /// Disables the flash effect of specific gameObjects
34         /// </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     }
43     /// <summary>
44     /// Enables the flash effect for specific gameObjects, based
45     // on the position of the hitPoint
46     /// </summary>
47     /// <param name="lightPositions">The position of the hitPoint
48     // relative to the user</param>
49     /// <param name="flashColor">The color of the flash</param>
50     public void StartMultipleFlashes(variablesAggregator.
51         lightPositionEnum[] lightPositions, Color flashColor)
52     {
53         GameObject[] flashImagesArray = variableAggInstance.
54             flashingImages;
55         for (int i = 0; i < flashImagesArray.Length; i++)
56         {
57             bool shouldBeEnabled = false;
58             for (int j = 0; j < lightPositions.Length; j++)
59             {
60                 if (flashImagesArray[i].GetComponent<
61                     flashingLight>().lightPosition ==
62                     lightPositions[j]) shouldBeEnabled = true;
63             }
64             if (shouldBeEnabled)
65             {
66                 flashImagesArray[i].GetComponent<flashingLight>()
67                     .StartFlash(1, 0.75f, flashColor);
68             } else
69             {
70                 flashImagesArray[i].GetComponent<flashingLight>()
71                     .StopFlash();
72             }
73         }
74     }
75     /// <summary>
76     /// Detects the position of the hitPoint relative to the user
77     /// </summary>
78     /// <param name="hitPoint">The closest point of the mesh to
79     // the user</param>
80     /// <returns>The position of the hitPoint (<see cref="
81     // variablesAggregator.lightPositionEnum"/>)</returns>
82     public variablesAggregator.lightPositionEnum[]
83         FindHitPointPosition(RaycastHit hitPoint)

```

```

76     {
77         List<variablesAggregator.lightPositionEnum>
78             lightPositionToreturn = new List<variablesAggregator.
79                 lightPositionEnum>();
80         Vector3 relativePosition = variableAggInstance.
81             userPosition.transform.InverseTransformPoint(hitPoint
82                 .point);
83
84         if (relativePosition.x > 0.3f)
85         {
86             lightPositionToreturn.Add(variablesAggregator.
87                 lightPositionEnum.Right);
88         }
89         else if (relativePosition.x < -0.3f)
90         {
91             lightPositionToreturn.Add(variablesAggregator.
92                 lightPositionEnum.Left);
93         }
94
95         if (relativePosition.y > 0)
96         {
97             lightPositionToreturn.Add(variablesAggregator.
98                 lightPositionEnum.Up);
99         }
100
101        /// <summary>
102        /// Based on the distance, it returns the color of the flash
103        /// which will be applied to the gameObjects.
104        /// <list type="bullet">
105        ///     <item>
106        ///         <description>Red: If the distance of the hitPoint
107        ///             is less than <see cref="variablesAggregator.
108        ///             dangerIndicatorThread">dangerIndicatorThread</see></
109        ///             description>
110        ///         </item>
111        ///         <item>
112        ///             <description>
113        ///                 Yellow: If the distance of the hitPoint is
114        ///                     more than <see cref="variablesAggregator.
115        ///                     dangerIndicatorThread">dangerIndicatorThread</see>,
116        ///                     but less than <see cref="variablesAggregator.

```

```

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

```

Aρχείο 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         /// <summary>
41         /// Detects and returns the current gamepad
42         /// </summary>
43         /// <returns>The current gamepad</returns>
44         public Gamepad GetGamepad()
45     {
46         return Gamepad.current;
47         /*
48         Gamepad gamepadTemp = null;
49         Gamepad[] allGamepads = Gamepad.all.ToArray();
50         InputDevice[] allInputDevices = _playerInput.devices.
51             ToArray();
52         bool gamepadFound = false;
53
54         for (int i = 0; i < allGamepads.Length; i++)
55         {
56             for (int j = 0; j < allInputDevices.Length; j++)
57             {
58                 if (allGamepads[i].deviceId == allInputDevices[j].
59                     deviceId)
60                 {
61                     gamepadTemp = allGamepads[i];
62                     break;
63                 }
64             }
65             if (gamepadFound) break;
66         }
67     }

```

```
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);
117         }
118         // rumbleDuration = Time.time + duration
119     }
120 }
```

```

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
166     /// </summary>
167     /// <param name="hitPoint">The closest point of an obstacle
168     /// relative to the user</param>

```

```

157     /// <param name="motorSpeed">The speed, which will be applied
158     /// to one of the motors</param>
159     /// <returns>A dictionary which dictates what the speed of
160     /// each motor should be</returns>
161     public Dictionary<string, float> GetSpeedOfEachMotor(
162         RaycastHit hitPoint, float motorSpeed)
163     {
164         Vector3 relativePosition = variableAggInstance.
165             userPosition.transform.InverseTransformPoint(hitPoint
166             .point);
167         Dictionary<string, float> motorToEnable = new Dictionary<
168             string, float>()
169         {
170             {"low", 0.0f},
171             {"high", 0.0f}
172         };
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         }
214     }

```

```

199         }
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 }
```

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))]
```

```

8 public class continuousCast : MonoBehaviour
9 {
10     private variablesAggregator variableAggInstance;
11     private bool enableContinuousModeGlobal;
12     private GameObject[] castBoxesWithContinuous;
13     private initFlashHandler flashHandler;
14
15     // Start is called before the first frame update
16     void Start()
17     {
18         variableAggInstance = this.GetComponent<
19             initCastAndBoxesHandler>().variableAggregatorObject.
20             GetComponent<variablesAggregator>();
21         enableContinuousModeGlobal = variableAggInstance.
22             enableContinuousModeGlobal;
23         castBoxesWithContinuous = this.GetComponent<initBoxes>().
24             findCastBoxesWithContinuousEnabled(
25                 variableAggInstance.raycastBoxes);
26         flashHandler = this.GetComponent<initFlashHandler>();
27     }
28
29     // Update is called once per frame
30     void Update()
31     {
32         if (enableContinuousModeGlobal)
33         {
34             List<RaycastHit> hitPointsList = new List<RaycastHit>();
35
36             for (int i = 0; i < castBoxesWithContinuous.Length; i++)
37             {
38                 RaycastHit hitPoint = castBoxesWithContinuous[i].
39                     GetComponent<singleRaycast>().
40                     SingleRaycastFunc(variableAggInstance);
41                 hitPointsList.Add(hitPoint);
42             }
43             int hitPointIndex = this.GetComponent<initBoxes>().
44                 findClosestHitPointIndex(hitPointsList.ToArray());
45             ;
46             if (hitPointIndex != -1)
47             {
48                 switch (variableAggInstance.alertBoxType)
49                 {
50                     case variablesAggregator.AlertBoxTypeEnum.
51                         BoxSpecific:
52                         // Use the alert box of each castBox
53                         this.GetComponent<initBoxes>().
54                             placeAlertBox(castBoxesWithContinuous
55                             [hitPointIndex], hitPointsList[
56                             hitPointIndex]);
57                         break;
58
59                     case variablesAggregator.AlertBoxTypeEnum.
60                         Common:
61                         // Use a common alert for all castBoxes
62
63                 }
64             }
65         }
66     }
67 }
```

```

48         this.GetComponent<initBoxes>().
        placeAlertBox(variableAggInstance.
        commonAlertBox,
        castBoxesWithContinuous[hitPointIndex],
        hitPointsList[hitPointIndex],
        variableAggInstance.
        placeAlertBoxAtEyeLevel);
        break;
49
50
51     case variablesAggregator.AlertBoxTypeEnum.
52         Peripheral:
53             this.GetComponent<initBoxes>().
54                 placePeripheralAlertBox(
55                     variableAggInstance.
56                     peripheralAlertBox, hitPointsList[
57                         hitPointIndex]);
58             break;
59         }
60
61     // Debug.Log(hitPointsList[hitPointIndex].
62     //             distance);
63     print(hitPointIndex + ": " + hitPointsList[
64         hitPointIndex].point);
65
66     if (variableAggInstance.activateRumble)
67     {
68         float motorSpeed = this.GetComponent<
69             initRumbleHandler>().FindMotorsLowAndHigh(
70             hitPointsList[hitPointIndex].distance);
71         float stopFreq = this.GetComponent<
72             initRumbleHandler>().GetStopFrequency(
73             hitPointsList[hitPointIndex].distance);
74         Dictionary<string, float> motorSpeedDict =
75             this.GetComponent<initRumbleHandler>().
76             GetSpeedOfEachMotor(hitPointsList[
77                 hitPointIndex], motorSpeed);
78
79         print("Motor speed: " + motorSpeed);
80         if (motorSpeed > 0.2f)
81         {
82             this.GetComponent<initRumbleHandler>().
83                 RumblePulse(motorSpeedDict["low"],
84                     motorSpeedDict["high"], stopFreq, 0.5
85                     f);
86         }
87         else
88         {
89             this.GetComponent<initRumbleHandler>().
90                 StopRumble();
91         }
92     }
93
94
95
96
97     /*
98     if (variableAggInstance.activateFlash)
99     {

```

```

80             if (hitPointsList[hitPointIndex].distance <=
81                 variableAggInstance.
82                 distanceThreadToActivate)
83             {
84                 variablesAggregator.lightPositionEnum[]
85                     lightPositionsToStart = flashHandler.
86                     FindHitPointPosition(hitPointsList[
87                         hitPointIndex]);
88                 Color colorToUse = flashHandler.
89                     FindColorBasedOnDistance(
90                         hitPointsList[hitPointIndex].distance
91                     );
92                 flashHandler.StartMultipleFlashes(
93                     lightPositionsToStart, colorToUse);
94             } else
95             {
96                 flashHandler.StopFlashes();
97             }
98         }
99     /**
100    */
101    } else
102    {
103        this.GetComponent<initRumbleHandler>().StopRumble
104            ();
105        switch (variableAggInstance.alertBoxType)
106        {
107            case variablesAggregator.AlertBoxTypeEnum.
108                BoxSpecific:
109                    // Use the alert box of each castBox
110                    this.GetComponent<initBoxes>().
111                        deactivateAlertBox(
112                            castBoxesWithContinuous);
113                    break;
114
115            case variablesAggregator.AlertBoxTypeEnum.
116                Common:
117                    // Use a common alert for all castBoxes
118                    this.GetComponent<initBoxes>().
119                        deactivateAlertBox(
120                            variableAggInstance.commonAlertDialog);
121                    break;
122
123            case variablesAggregator.AlertBoxTypeEnum.
124                Peripheral:
125                this.GetComponent<initBoxes>().
126                    deactivateAlertBox(
127                        variableAggInstance.
128                        peripheralAlertDialog);
129                    break;
130
131        }
132    }
133}
134
135    /**
136     * <summary>
137     *
138     * </summary>
139     */

```

```

116     /// Toggles the continuous mode on or off
117     /// </summary>
118     /// <remarks>
119     /// The function is called with the voice input <c>"Continuous Mode"</c>
120     /// </remarks>
121     public void continuousModeGlobalToggle()
122     {
123         enableContinuousModeGlobal = !enableContinuousModeGlobal;
124
125         if (enableContinuousModeGlobal == true)
126         {
127             // Position of Continuous Mode
128             castBoxesWithContinuous = this.GetComponent<initBoxes>()
129                 .findCastBoxesWithContinuousEnabled(
130                     variableAggInstance.raycastBoxes);
131             this.GetComponent<initBoxes>().positionAndScaleBoxes(
132                 castBoxesWithContinuous,
133                 variableAggInstance.boxesPerRow,
134                 variableAggInstance.numberOfWorks,
135                 this.GetComponent<initBoxes>().userHeight
136             );
137         } else
138         {
139             // Disable alert boxes of cast boxes in continuous
140             // mode
141             this.GetComponent<initBoxes>().deactivateAlertBox(
142                 castBoxesWithContinuous);
143             this.GetComponent<initBoxes>().deactivateAlertBox(
144                 variableAggInstance.commonAlertBox);
145             /*for (int i = 0; i < castBoxesWithContinuous.Length;
146                 i++)
147             {
148                 if (castBoxesWithContinuous[i].GetComponent<
149                     initSingleBox>().alertView.activeSelf == true)
150                 {
151                     castBoxesWithContinuous[i].GetComponent<
152                         initSingleBox>().alertView.SetActive(false
153                     );
154                 }
155             }*/
156
157         }
158     }

```

```

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
32        // Update is called once per frame
33        void Update()
34        {
35            enabledHandAB = variableAggInstance.enabledHandAB;
36            if (enabledHandAB == true)
37            {
38                var isHandPresent = false;
39
40                foreach (var source in CoreServices.InputSystem.
41                    DetectedInputSources)
42                {
43                    // Ignore anything that is not a hand because we
44                    // want articulated hands
45                    if (source.SourceType == InputSourceType.Hand &&
46                        source.SourceName != "None Hand")
47                    {
48                        isHandPresent = true;
49                        foreach (var p in source.Pointers)
50                        {
51                            if (p is IMixedRealityNearPointer)
52                            {
53                                // Ignore near pointers, we only want
54                                // the rays
55                                continue;
56                            }
57                        }
58                    }
59                }
60            }
61        }
62    }
63}
```

```

81                                     AudioSource>().Pause();
82                                 }
83             }
84         {
85             if (handAlertBoxesDict.ContainsKey(
86                 source.SourceName))
87             {
88                 if (handAlertBoxesDict[source.
89                     SourceName].GetComponent<
90                         AudioSource>().isPlaying ==
91                         false
92                         && handAlertBoxesDict[source.
93                             SourceName].activeSelf)
94                 {
95                     handAlertBoxesDict[source.
96                         SourceName].GetComponent<
97                             AudioSource>().Play();
98                 }
99             }
100        }
101    }
102    if (!isHandPresent)
103    {
104        foreach (var handAlertBox in handAlertBoxes)
105        {
106            handAlertBox.SetActive(false);
107        }
108    }
109    public void enabledHandABToggle()
110    {
111        enabledHandAB = !enabledHandAB;
112        if (enabledHandAB == false)
113        {
114            foreach (var handAlertBox in handAlertBoxes)
115            {
116                handAlertBox.SetActive(false);
117            }
118        }
119    }
120 }
```

Aρχείο castModes/handRaycast.cs

```

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

```

    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)
35            {
36                allCastBoxes[i].GetComponent<initSingleBox>().
37                    alertBox.SetActive(false);
38            }
39        }*/
40    }
41 }
42 }
```

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     /// <summary>
21     ///     Castes a raycast or boxcast from the position of a
22     ///     castBox and detects a hitPoint.
23     /// </summary>
24     /// <param name="variableAggInstance">The variable aggregator
25     ///     GameObjetc</param>
26     /// <returns>
27     ///     The hitPoint of the cast
28     /// </returns>
29     public RaycastHit SingleRaycastFunc(variablesAggregator
30     variableAggInstance)
31     {
32         Vector3 fwd = transform.TransformDirection(Vector3.
33             forward); // forward direction
34
35         float maxDistance = variableAggInstance.maxDistance;
36
37         variablesAggregator.CastTypeEnum castType =
38             variableAggInstance.castType;
39
40         switch (castType) {
41             case variablesAggregator.CastTypeEnum.Raycast:
42                 m_HitDetect = Physics.Raycast(transform.position,
43                     fwd, out hitInfo, 5.0f);
44                 break;
45
46             case variablesAggregator.CastTypeEnum.BoxCast:
47                 m_HitDetect = Physics.BoxCast(transform.position,
48                     transform.localScale, fwd, out hitInfo,
49                     transform.rotation, maxDistance);
50                 break;
51         }
52
53         return hitInfo;
54     }
55
56     /*
57     void OnDrawGizmos()
58     {
59         Gizmos.color = Color.red;
60
61         //Check if there has been a hit yet
62         if (m_HitDetect)
63         {
64             //Draw a Ray forward from GameObject toward the hit
65             Gizmos.DrawRay(transform.position, transform.forward
66                 * hitInfo.distance);
67             //Draw a cube that extends to where the hit exists
68             Gizmos.DrawWireCube(transform.position + transform.
69                 forward * hitInfo.distance, transform.localScale)
70                 ;
71         }

```

```

61         //If there hasn't been a hit yet, draw the ray at the
62         //maximum distance
63     else
64     {
65         //Draw a Ray forward from GameObject toward the
66         //maximum distance
67         // Gizmos.DrawRay(transform.position, transform.
68         //forward * m_MaxDistance);
69         //Draw a cube at the maximum distance
70         // Gizmos.DrawWireCube(transform.position + transform
71         .forward * m_MaxDistance, transform.localScale);
72     }
73 }
74 */
75 }
```

Aρχείο 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    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    /// <summary>
23    /// Starts the flash effect for the specific gameObject
24    /// </summary>
25    /// <param name="flashInterval">The duration of the flash
26    // effect</param>
27    /// <param name="maxAlpha">The max value of the alpha (max
28    // opacity)</param>
29    /// <param name="colorOfTheFlash">The color of the flash</
30    // param>
31    public void StartFlash(float flashInterval, float maxAlpha,
32        Color colorOfTheFlash)
33    {
34
35        newColorSetColor("_Color", colorOfTheFlash);
36        this.GetComponent<Renderer>().SetPropertyBlock(newColor);
37    }
38}
```

```

34 // 0 <= maxAlpha <= 1
35 maxAlpha = Mathf.Clamp(maxAlpha, 0, 1);
36
37 if (currentFlashRoutine == null)
38 {
39     // StopCoroutine(currentFlashRoutine);
40     currentFlashRoutine = Flash(flashInterval, maxAlpha);
41     shouldFlashStop = false;
42     StartCoroutine(currentFlashRoutine);
43 }
44
45 }
46
47 /// <summary>
48 /// Stops the flash effect for the specific gameObject
49 /// </summary>
50 public void StopFlash()
51 {
52     shouldFlashStop = true;
53     if (currentFlashRoutine != null)
54     {
55         StopCoroutine(currentFlashRoutine);
56         currentFlashRoutine = null;
57
58         newColor.SetColor("_Color", new Color(0, 0, 0, 0));
59         this.GetComponent<Renderer>().SetPropertyBlock(
60             newColor);
61     }
62 }
63
64 /// <summary>
65 /// Applies the flash effect to the gameObject. The effect is
66 /// looped.
67 /// </summary>
68 /// <param name="flashInterval">The duration of one flash (in
69 /// seconds)</param>
70 /// <param name="maxAlpha">The max value of the alpha</param>
71 /// <returns></returns>
72 IEnumerator Flash(float flashInterval, float maxAlpha)
73 {
74     // Flash In
75     float flashInDuration = flashInterval / 2;
76
77     while(!shouldFlashStop)
78     {
79         for (float t = 0; t <= flashInDuration; t += Time.
80             deltaTime)
81         {
82             Color colorThisFrame = newColor.GetColor("_Color"
83                 );
84             colorThisFrame.a = Mathf.Lerp(0, maxAlpha, t /
85                 flashInDuration);
86             newColor.SetColor("_Color", colorThisFrame);
87             this.GetComponent<Renderer>().SetPropertyBlock(
88                 newColor);
89         }
90     }
91 }

```

```
83             // wait until next frame
84             yield return null;
85         }
86
87         // Flash Out
88         float flashOutDuration = flashInterval / 2;
89         for (float t = 0; t <= flashOutDuration; t += Time.
90             deltaTime)
91         {
92             Color colorThisFrame = newColor.GetColor("_Color"
93                 );
94             colorThisFrame.a = Mathf.Lerp(maxAlpha, 0, t /
95                 flashOutDuration);
96             newColor.SetColor("_Color", colorThisFrame);
97             this.GetComponent<Renderer>().SetPropertyBlock(
98                 newColor);
99             yield return null;
100        }
101    }
102 }
103 }
```

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.