Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών - Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

Ανάπτυξη Ηλεκτρονικού Παιγνίου Με Εφαρμογή Ηλεκτρονικού Εγκεφαλογράφου EEG

Θεοδωρακοπούλου Ανδριάνα (Α.Μ. 2620)

Επιβλέπων Καθηγητής:

Παπαδουράκης Γεώργιος, Δρ.

Ιούλιος 2015, Ηράκλειο Κρήτης

Ευχαριστίες

Abstract

In our times, there has been a tremendous evolution of technology that has, among others, extended the human capabilities. The notion of using a brain's electrical activity (electroencephalography) to interact with personal computers and other devices, is one of the greatest requests of our time and is receiving a widespread attention on a production level.

Until recently a purely lab based technology, brainwave (electroencephalograph or EEG) headsets are trickling into the marketplace in a number of different guises. But what exactly do these devices do, how do they differ from each other and - with potential applications ranging from medicine to gaming and market research - who will use them and for what purpose?

EEG headsets have broken out of the laboratory into the wild and while their numbers are still sparse, they can be expected to develop rapidly.

In the next few years we can expect some exciting applications to emerge, particularly in disability applications. In relation to gaming, they seem to be a solution looking for a problem at the moment, but they undeniably have novelty appeal.

This thesis aims to present the use of electroencephalography in order to improve players' gaming experience by approximating the underlying distributions, using simple statistical and probabilistic techniques, and use this to weed out the unrelated information, preserving the most relevant. With the brain-computer interface (BCI), the user is able to perform tasks and interact with the virtual world through his mind, where reliability is accomplished by adapting the produced information in real-time.

Σύνοψη

Η παρούσα πτυχιακή εργασία αναφέρεται στη μελέτη και την υλοποίηση ενός ηλεκτρονικού παιγνίου με εφαρμογή ηλεκτρονικού εγκεφαλογράφου (EEG). Συγκεκριμένα, η πτυχιακή εργασία σχετίζετε με τη δημιουργία ενός ολοκληρωμένου περιβάλλοντος παιχνιδιού όπου ο χρήστης μπορεί να χειρίζεται τον παίκτη σχεδόν όπως στα πιο πολλά παιχνίδια με τη διαφορά, ότι σε αυτή την υλοποίηση ο χρήστης έχει παραπάνω δυνατότητες χειρισμού, λόγω της χρήσης και της εφαρμογής του EEG.

Όπως είναι γνωστό ένα ηλεκτρονικό παιχνίδι στις μέρες μας αποτελείτε όχι μόνο από το κυρίως μέρος του παιχνιδιού αλλά και από επεξηγηματικά βίντεο, εικόνες, μηνύματα και λοιπά, έτσι χρειάστηκε να γίνει ένας συνδυασμός από προγράμματα ανάπτυξης για να φτάσουμε στο τελικό αποτέλεσμα. Το βασικότερο από αυτά είναι το Unity3D που δημιουργήθηκε εξ' ολοκλήρου το παιχνίδι. Τα υπόλοιπα χρησιμοποιήθηκαν για την επεξεργασία των μοντέλων, εικόνας και βίντεο τα οποία είναι το 3ds Max για την επεξεργασία των μοντέλων, Adobe Photoshop και After Effect για την επεξεργασία εικόνας και βίντεο.

Σκοπός της εργασίας είναι η εκμάθηση και η εφαρμογή νέων τεχνολογιών με στόχο τη σύνθεση και τη σωστή λειτουργία τους για το τελικό επιθυμητό αποτέλεσμα. Βέβαια ο απώτερος σκοπός πέραν της μάθησης που είναι το αρχικό βασικό στάδιο, είναι η δημιουργία αυτού του είδους παιχνιδιού με καλύτερο σενάριο, γραφικά και πιο πολλές χρήσης του ηλεκτροεγκεφαλογράφου. Παρακάτω θα παρουσιαστούν αναλυτικά τα βήματα υλοποίησης και η μελέτη διεξαγωγής του παιχνιδιού.

Περιεχόμενα

Πίνακας Εικόνων

Πίνακας Πηγαίου Κώδικα – Αλγορίθμων

1. Εισαγωγή

Ηλεκτροεγκεφαλογράφημα (EEG) λέγετε η μέτρηση της ηλεκτρικής δραστηριότητας του ανθρώπινου εγκεφάλου. Ο εγκέφαλός αποτελείται από νευρώνες, οι οποίοι την επικοινωνία τους χρησιμοποιούν ηλεκτρικό σήμα. Το ηλεκτρικό σήμα μπορεί να καταγραφεί από μια τέτοια συσκευή, εφαρμόζοντάς στο τριχωτό της κεφαλής τους κατάλληλους αισθητήρες (Electroencephalography, n.d.). Η πρώτη καταγραφή ηλεκτρικής εγκεφαλικής δραστηριότητας καταγράφηκε από τον Hans Berger το 1924, χρησιμοποιώντας ένα απλό γαλβανόμετρο, τοποθέτησε ένα μόνο ηλεκτρόδιο εντόπισε το κύμα Άλφα λεγόμενο και ως κύμα Berger.

Στην Ιατρική χρησιμοποιείτε ειδική συσκευή μεγάλης ακρίβειας η οποία αποτελείτε από 21 ηλεκτρόδια και χρησιμοποιούνται για τον εντοπισμό 5 βασικών κυμάτων, αυτό το είδος συσκευής κοστίζει χιλιάδες δολάρια. Τα τελευταία χρόνια, υπάρχουν φθηνές συσκευές ηλεκτροεγκεφαλογράφισης μερικές από αυτές είναι, της Avatar EEG Solutions, NeuroSky, OCZ technology, Interaxon, PLX Devices και Emotiv systems. Οι συσκευές αυτές δεν χρησιμοποιούνται για κλινική χρήση, αλλά είναι κατάλληλες για Brain-Computer interface (BCI). Η μέθοδος αυτή προσφέρει την άμεση επικοινωνία μεταξύ του εγκεφάλου με μια εξωτερική συσκευή. Για το λόγο αυτό η συσκευή που χρησιμοποιήθηκε στην υλοποίηση της πτυχιακής ανήκει σε μια από τις προαναφερθείς ομάδες (Mindwave της Neurosky) (Mindwave, n.d.).

Η ενσωμάτωση βιομετρικών στοιχείων στα βιντεοπαιχνίδια είναι θέμα που πραγματεύεται και η παρούσα πτυχιακή εργασία. Αυτός ο συνδυασμός παιχνίδι - βιομετρικές μετρήσεις και χρήση αυτών θα μπορούσε να ωθήσει τη βιομηχανία παιχνιδιών σε μια άλλη σφαίρα.

Από την έλευση των πρώτων ηλεκτρονικών παιχνιδιών και μέχρι τις μέρες μας τα ηλεκτρονικά παιχνίδια υιοθετούν ολοένα και πιο κεντρικό ρόλο στην καθημερινή πραγματικότητα τόσο των παιδιών όσο και των ενηλίκων. Στη σύγχρονη πραγματικότητα δεν αποτελούν πλέον απλώς ακόμη ένα επιτυχημένο τρόπο ψυχαγωγίας, αλλά ισχυρά μέσα που μπορούν να διαμορφώνουν απόψεις, να αναπτύσσουν τη δική τους αισθητική και να γεννούν νέους τρόπους κατανόησης του κόσμου. Για το λόγο αυτό, τα τελευταία χρόνια, το αντικείμενο των παιχνιδιών έχει εξελιχτεί σε διακριτό τομέα μελέτης κάτω από τον όρο «Game Studies», με σκοπό την προσπάθεια κατανόησης της φύσης των ηλεκτρονικών παιχνιδιών, των χρηστών τους και των περίπλοκων διαδράσεων μεταξύ παιχνιδιών και χρηστών. Ο συγκεκριμένος αυτός τομέας έρευνας λαμβάνει επίσης υπόψιν την ευρύτερη πολιτισμική επιρροή των παιχνιδιών, αναλύοντάς τα ως προς μια κοινωνική/ανθρωπιστική διάσταση.

Στο πλαίσιο αυτό κατανοούμε ότι πρέπει να συμβαδίζουμε με τις νέες τεχνολογίες και να επιχειρούμε πάνω σε αυτές. Η χρήση του Brain-Computer interface (BCI) έχει λάβει ευρεία προσοχή κατά τα τελευταία χρόνια, δεδομένου ότι επιτρέπει στο χρήστη να ελέγχει το περιβάλλον με την εγκεφαλική δραστηριότητα απευθείας, παρακάμπτοντας την ανάγκη για ομιλία, χειρονομίες, ή οποιαδήποτε άλλη μορφή μυϊκής δραστηριότητας. Γενικά το BCI

χρησιμοποιείτε επίσης σε άτομα που πάσχουν από σοβαρές εγκεφαλικές ή μυϊκές διαταραχές αλλά είναι παράλληλα τόσο ενδιαφέρουσα που μπορεί να χρησιμοποιηθεί σε διάφορους τομείς.

Ελέγχοντας ένα παιχνίδι σε πραγματικό χρόνο και με τη χρήση βιομετρικών μετρήσεων είμαστε σε θέση να βιώσουμε ένα άλλου είδους ψυχαγωγία. Ένα μεγάλο ποσοστό ανθρώπων έχει παίξει παιχνίδια που βρίσκονται απέναντι από την οθόνη του υπολογιστή ή της τηλεόρασης και με απλές κινήσεις του σώματος τους χειρίζονται ένα παιχνίδι. Αυτό μπορεί να προφέρει αρκετά οφέλη στο χρήστη, το πιο προφανές όφελος είναι ότι θα πρέπει να είναι ενεργός νους εν σώματι στο παιχνίδι, πράγμα που ωφελεί στη σωματική άσκηση. Συνεπώς, δίνετε η δυνατότητα βελτίωσης. Στη συγκεκριμένη υλοποίηση αν κάποιος έχει προβλήματα έλλειψης συγκέντρωσης θα δυσκολευτεί ιδιαίτερα να ανταπεξέλθει σε κάτι που απαιτεί την προσοχή, έτσι λοιπόν θα πρέπει να καταβάλει μεγαλύτερη προσπάθεια για να ανταπεξέλθει στις ανάγκες του παιχνιδιού.

Εύκολα λοιπόν οδηγούμαστε στο συμπέρασμα ότι μέσω της χρήσης τέτοιων συσκευών θα υπάρξει και η ανάγκη ολοένα και πιο έξυπνων παιχνιδιών μαθαίνοντας τον παίκτη να γνωρίσει τα βιομετρικά χαρακτηριστικά του και πώς το σώμα του αντιδρά όταν είναι υπό επίθεση κατά τη διάρκεια ενός παιχνιδιού.

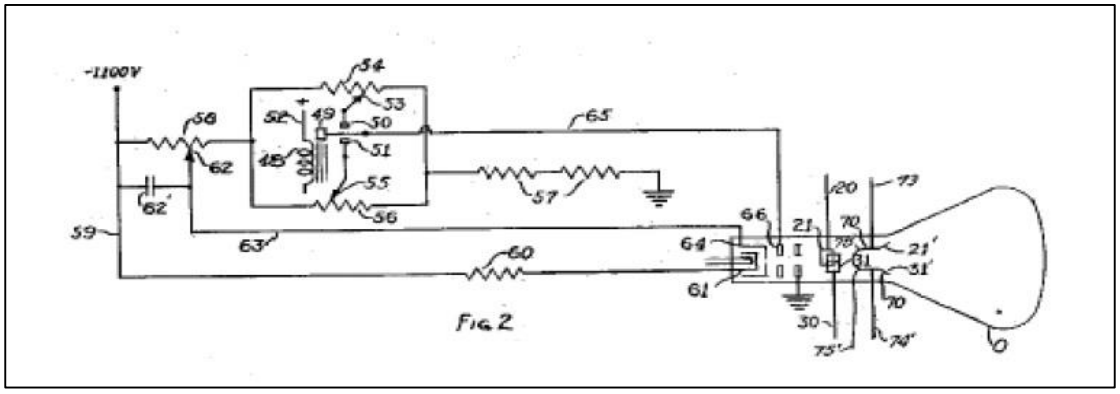
Εικόνα 1: Πρώτο full-brain ασύρματο EEG headset

1.1 Η Εξέλιξη Των Ηλεκτρονικών Παιχνιδιών

Τα ηλεκτρονικά παιχνίδια έχουν γίνει μέρος της καθημερινότητας των περισσότερων ανθρώπων. Οι κονσόλες, οι ηλεκτρονικοί υπολογιστές, τα κινητά τηλέφωνα, τα pda, είναι οι κύριες συσκευές που μπορεί κάποιος να απολαύσει ένα ηλεκτρονικό παιχνίδι. Τα καινούργια παιχνίδια πλέον έχουν εντυπωσιακά χρώματα, υπέροχους φωτισμούς και σκιάσεις σε πολύ υψηλές αναλύσεις και με δεκάδες περιφερειακά να προσφέροντα ειδικά για τους χρήστες παιχνιδιών. Τα ηλεκτρονικά παιχνίδια κάποτε δεν ήταν όπως σήμερα. Έχουν μια μακροχρόνια εξελικτική πορεία η οποία έχει περάσει αρκετά στάδια μέχρι να ωριμάσει.

Το πρώτο ηλεκτρονικό παιχνίδι (1947)

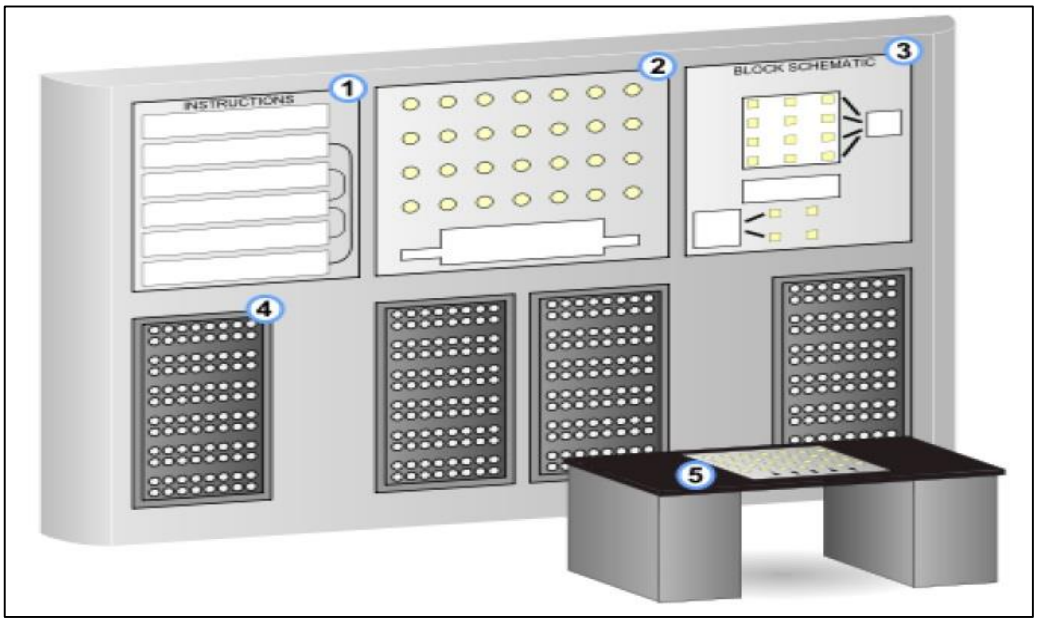
Το πρώτο γνωστό ηλεκτρονικό παιχνίδι που κατασκευάστηκε ποτέ, δημιουργήθηκε από τους Thomas T. Goldsmith Jr. και Estle Ray Mann το 1947. Το παιχνίδι ήταν ένας εξομοιωτής πυραύλων πάνω σε μια crt οθόνη της τότε εποχής. Ο πύραυλος κατευθυνόταν από τον χρήστη με αναλογικά και όχι ψηφιακά κυκλώματα. Μάλιστα ο στόχος δεν εμφανίζονταν στην οθόνη, αλλά ήταν κουκκίδες σε επικαλύμματα πάνω από την οθόνη. (First video game, n.d.)



Εικόνα 2: Σχέδιο κυκλώματος του 1ου ηλεκτρονικού παιχνιδιού (1947)

Η πρώτη ψηφιακή παιχνιδομηχανή (1951)

Μετά από λίγα μόλις χρόνια έκανε την εμφάνισή του το πρώτο ψηφιακό παιχνίδι το 1951, ήταν ένα μαθηματικό παιχνίδι που λεγόταν NIM, παιζόταν στον υπολογιστή που τον έλεγαν NIMROD. Το παιχνίδι αυτό παρουσιάστηκε το 1951 στο φεστιβάλ της Βρετανίας. Χρησιμοποιούσε για οθόνη ένα πάνελ από φώτα και ήταν ουσιαστικά η πρώτη ψηφιακή παιχνιδομηχανή.



Εικόνα 3: Η πρώτη ψηφιακή παιχνιδομηχανή (NIMROD) το 1951

Στον παρακάτω πίνακα περιγράφονται τα βασικά μέρη της ψηφιακής παιχνιδομηχανής (NIMROD).

Ετικέτα	**Επεξήγηση**
1	Πίνακας οδηγιών
2	Κύριος πίνακας. (έδειχνε ότι φαινόταν στον πίνακα ελέγχου (5),για να παρακολουθούν και οι παρατηρητές)
3	Δείχνει τον τρέχον υπολογισμό σε περίπτωση που αργεί πολύ η επεξεργασία.
4	Τέσσερις πίνακες κρατούν τις λυχνίες που απαρτίζουν το λογικό κύκλωμα του υπολογιστή.
5	Πίνακας ελέγχου.

Το πρώτο οικιακό παιχνίδι (1961)

Η ανάγκη για διασκέδαση μέσω παιχνιδιών έφερε στο κόσμο της τεχνολογίας το πρώτο οικιακό παιχνίδι που φτιάχτηκε το 1961, από φοιτητές του (MIT Martin Graetz), Steve Russell, και Wayne Wiitanen, το παιχνίδι λεγόταν Spacewar. Το παιχνίδι παίζονταν στον υπολογιστή του 1960, DEC PDP1 και χρησιμοποιούσε για οθόνη έναν παλμογράφο. Σίγουρα ήταν ένα πολύ σημαντικό παιχνίδι στην ιστορία των ηλεκτρονικών παιχνιδιών μιας και κάνει το ένα βήμα παραπάνω δείχνοντας ένα

πρώτο δείγμα γραφικών και ενός πιο πετυχημένου εξομοιωτή από τα προηγούμενα. Μάλιστα το παιχνίδι είχε και παράγοντες να επηρεάζουν τα σκάφοι, με αρχικό αυτόν της βαρύτητας. Όπως προδίδει και το όνομα του, το παιχνίδι περιείχε δύο διαστημικούς πυραύλους, τα «Wedge» και «Needle», που πήραν τα ονόματα τους από το σχήμα τους. Ο σκοπός του παίκτη ήταν απλά να πυροβολεί το εχθρικό σκάφος μέχρι αυτό να καταστραφεί.

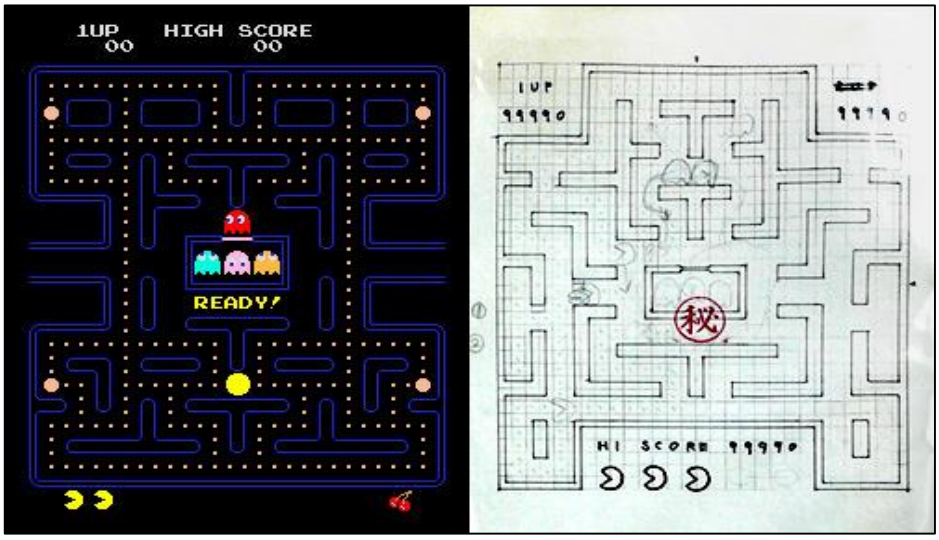
Η περίοδος των Arcade Game ‘70s

Τα Arcade Games γνώρισαν μεγάλη άνθηση στη δεκαετία του ’70 όταν δύο τελείως διαφορετικά παιχνίδια βρήκαν το φως της δημοσιότητας. Πρώτο, ήταν το περίφημο Space Invaders, από την ιαπωνική Taito, έθεσε νέα επίπεδα στο κοινωνικό gaming αφού ήταν το πρώτο που ενσωμάτωνε high score. Ο παίκτης, έχοντας διάφορα όπλα, προσπαθούσε να σκοτώσει τους εξωγήινους που έρχονταν κατά πάνω του σε σειρές. Ουσιαστικά ο παίκτης δεν μπορούσε να νικήσει, αλλά τα επίπεδα δυσκόλευαν μέχρι ο παίκτης να πεθάνει.

Αν και γενικά το παιχνίδι δε σημείωσε τρομακτική επιτυχία, κατόρθωσε να αναγκάσει την ιαπωνική κυβέρνηση να αυξήσει την παραγωγή κερμάτων, αφού στην Ιαπωνία για ένα δεκάλεπτο παιχνιδιού, κόστιζε 100 γιέν, ένα από τα πιο δυσεύρετα κέρματα στη χώρα του Ανατέλλοντος Ηλίου. Από την άλλη πλευρά, η Atari έδωσε στους gamers, τον πρώτο αξιέπαινο προσομοιωτή ποδοσφαίρου, το Atari Football. Αυτό ήταν το πρώτο ποδοσφαιράκι με κυλιόμενη οθόνη, ενώ χρησιμοποιούσε αναλογικό πλήκτρο για χειρισμό. Οι πωλήσεις του Atari Football ήταν παρόμοιες με αυτές του Space Invaders, έως ότου τη λήξη της σεζόν στην Αμερική το 1979. Ένα από τα διασημότερα παιχνίδια εκείνης της εποχης είναι το γνωστο σε ολοους μας Pac – Man της Namco και το δημοφιλέστερο arcade game όλων των εποχών. Εμφανίσθηκε πρώτη φορά στις 12 Μαΐου 1980 στην Ιαπωνία με το όνομα «Puck-Man». Το πρώτο αυτό όνομα προέρχεται από την ιαπωνική λέξη «*πάκου*», η οποία στα ιαπωνικά συμβολίζει το θόρυβο που κάνει κανείς ανοιγοκλείνοντας το στόμα του.

Το παιχνίδι δημιουργήθηκε από τον σχεδιαστή Τόρου Ιβατάνι. Το 1981 δημοσιεύτηκε στις Ηνωμένες Πολιτείες από την Midway με το όνομα Pac-Man, επειδή η επιγραφή «Puck-Man» με ευκολία μπορούσε να μετατραπεί σε υβριστική με απλή αντικατάσταση του γράμματος P σε F. Σαν παιχνίδι όμως ήταν το ίδιο όπως η ιαπωνική έκδοση με μόνη

διαφορά ότι τα φαντάσματα δεν λέγονταν άλλο Akabei, Pinky, Aosuke και Guzuta, αλλά Blinky, Pinky, Inky και Clyde.



Εικόνα 4: Pac – Man και το προσχέδιό

Καινοτομίες της Χρυσής Εποχής των Arcades

Κατά τη χρυσή εποχή των Arcades, όχι μόνο η αλλαγή του gameplay ήταν αυτή που προώθησε το χόμπι αυτό, αλλά και άλλες πολλές καινοτομίες. Η σημαντικότερη από αυτές ήταν η δημιουργία των λεγόμενων vector graphics. Τα γραφικά αυτά, δημιουργημένα από μια ακτίνα φωτός η οποία

παρήγαγε χρώματα σε μια μαύρη οθόνη, έδιναν τη δυνατότητα αναπαράστασης περισσοτέρων τρισδιάστατων αντικειμένων σε μακράν μεγαλύτερη ανάλυση από τα πρώτα παιχνίδια.

Η δεύτερη μεγαλύτερη καινοτομία, είναι η δημιουργία αποθηκευτικών μέσων για τα παιχνίδια. Αν και η SEGA ήταν η πρώτη που εισήγαγε μια τέτοια τεχνολογία, το arcade Dragon's Lair ήταν το πρώτο που ουσιαστικά τη χρησιμοποιούσε. Αποθηκεύοντας τα δεδομένα σε μια δισκέτα, υπήρχαν για πρώτη φορά σε παιχνίδι γενναιόδωρες στιγμές, κατά τις οποίες ένα κινηματογραφικό στιγμιότυπο παιζόταν. Παρόλο ,που εξαιτίας της τεχνολογίας αυτής, τα γραφικά ήταν άπιαστα για τότε, το παιχνίδι δεν προσέφερε κάποια replayability αφού μετά την πρώτη φορά η ιστορία ήταν γνωστή. Έτσι το παιχνίδι κατέληξε σε μια παροδική τάση. Η πάροδος του χρόνου, η τεχνογνωσία, η έμπνευση και η φαντασία συνέβαλαν στο αποτέλεσμα των παιχνιδιών που έχουμε σήμερα.

1.2 Επαυξημένη Πραγματικότητα

Στις μέρες μας η πρόοδος της τεχνολογίας μεγαλώνει και μαζί με αυτή και οι ανάγκες για καινούργιες εμπειρίες. Στη περίπτωση μας τα μέσα που χρησιμοποιούν τα ηλεκτρονικά παιχνίδια έχουν αλλάξει σε σύγκριση με τα παλιά όπου ήταν η καθιερωμένη οθόνη τηλεόρασης και η παιχνιδομηχανή ή τον υπολογιστή αντίστοιχα (Augmented reality, n.d.). Πλέον υπάρχουν ειδικά διαμορφωμένοι χώροι όπου κάποιος μπορεί να απολαύσει ένα παιχνίδι με όλες του τις αισθήσεις. Το επίπεδο των παιχνιδιών αυτών αγγίζει την Επαυξημένη Πραγματικότητα.

Με τον όρο Επαυξημένη Πραγματικότητα εννοούμε την άμεση ή έμμεση ζωντανή προβολή ενός φυσικού περιβάλλοντος, όπου κάποια στοιχεία του επαυξήθηκαν ή ενισχύθηκαν με την βοήθεια εικονικών αντικειμένων, προσώπων ή χώρων φτιαγμένων με την βοήθεια ηλεκτρονικού υπολογιστή. Αυτή η τεχνολογία θολώνει τα όρια μεταξύ του τι είναι πραγματικό και τι είναι φτιαγμένο από υπολογιστή, δημιουργημένο από την ενίσχυση της όρασης, ακοής, αίσθησης και όσφρηση.

Με άλλα λόγια ένας υπολογιστής δέχεται δεδομένα όπως ήχο, βίντεο, γραφικά ή δεδομένα GPS, και δημιουργεί γραφικά τα οποία προσθέτει στον πραγματικό κόσμο. Αυτή η ‘αρχή’ χρησιμοποιήθηκε από κάποιες εταιρείες παιχνιδιών και έχουν δημιουργήσει χώρους όπου ο παίκτης βλέπει το παιχνίδι μπροστά στα μάτια του σαν να είναι στη πραγματικότητα μέσα στο παιχνίδι, υπάρχει διάδρομος που τρέχει ή περπατάει αναλόγως με τις ανάγκες του παιχνιδιού, έχει “όπλο” που λειτουργεί και ως χειριστήριο και επίσης φοράει μια ειδική μάσκα που λέγεται Oculus και βλέπει το παιχνίδι μπρος στα μάτια του και νιώθει σαν να έχει μπει στη πραγματικότητα στο χώρο του παιχνιδιού. Είναι λοιπόν σαν να λέμε ότι ζει το παιχνίδι. Ο χώρος αυτός υπάρχει στη πραγματικότητα και έχει δημιουργηθεί από την Virtuix Omni, φυσικά η εγκατάσταση αυτή απαιτεί αρκετές χιλιάδες ευρώ.

Από τη άλλη υπάρχει και ένας άλλος τομέας της Πληροφορικής που ονομάζεται, Βιομετρικά Χαρακτηριστικά, με τον όρο αυτό εννοούμε τις μετρήσεις που σχετίζονται με τα ανθρώπινα χαρακτηριστικά. Είναι χαρακτηριστικά, μετρήσιμα και χρησιμοποιούνται για να περιγράψουν τα άτομα. Τα βιομετρικά στοιχεία συνήθως είναι τα δακτυλικά αποτυπώματα, η αναγνώριση προσώπου, DNA, η γεωμετρία του χεριού, η αναγνώριση της ίριδας, ο καρδιακός παλμός, όπως επίσης και τα εγκεφαλικά κύματα. Η χρήση των βιομετρικών χαρακτηριστικών χρησιμοποιείτε για την ταυτοποίηση ανθρώπων όπως για παράδειγμα η ίριδα του ματιού ανιχνεύεται από ειδική κάμερα και τακτοποιεί κάποιο πρόσωπο, σε ηλεκτρονικά διαβατήρια, σε εφαρμογές ακόμη και σε παιχνίδια.

Στη παρούσα υλοποίηση της εργασίας θα χρησιμοποιήσουμε μόνο το δεύτερο κομμάτι, το κομμάτι των βιομετρικών μετρήσεων που γίνονται με το Mindwave συσκευή που έχει δημιουργηθεί από τη Neurosky και συγκαταλέγετε στη κατηγορία των βίο-αισθητήρων. Σε μελλοντική εξέλιξη του ηλεκτρονικού παίγνιου θα ενσωματωθεί και το πρώτο που είναι της επαυξημένης πραγματικότητας με τη χρήση του Kinect.

1.3 Περίληψη Εργασίας

Η πτυχιακή εργασία αναφέρεται στη μελέτη και την υλοποίηση ενός ηλεκτρονικού παιγνίου με εφαρμογή ηλεκτροεγκεφαλογράφου (EEG).

Πιο συγκεκριμένα, σχετίζετε με τη δημιουργία ενός ολοκληρωμένου περιβάλλοντος παιχνιδιού όπου ο χρήστης μπορεί να χειρίζεται τον παίκτη σχεδόν όπως στα περισσότερα παιχνίδια με τη διαφορά, ότι έχουμε παραπάνω δυνατότητες που προσφέρονται στον χρήστη, λόγω της χρήσης του EEG. Οι τιμές που λαμβάνονται από τον ηλεκτροεγκεφαλογράφου προς άμεση χρήση είναι το Meditation (=Χαλάρωση) και το Attention (=Προσοχή).

Η ανάπτυξη του παιχνιδιού πραγματοποιήθηκε στο Unity3d το οποίο είναι εξ ολοκλήρου ολοκληρωμένο πρόγραμμα για ανάπτυξη παιχνιδιών, δίνει τη δυνατότητα προγραμματισμού των εισερχόμενων μοντέλων σε C# πράγμα το οποίο δίνει μεγάλη ελευθέρια σε οποιεσδήποτε πρακτικές υλοποίησης. Εκτός από αυτό ένα παιχνίδι χεριάζετε επεξηγηματικά βίντεο, Textures, επεξεργασία, δημιουργία ή μορφοποίηση μοντέλων πριν την εισαγωγή τους στο πρόγραμμα έτσι λοιπόν χρειάστηκε να γίνει ένας συνδυασμός από προγράμματα ανάπτυξης για να φτάσουμε στο τελικό αποτέλεσμα. Μερικά από αυτά είναι το 3ds Max για την επεξεργασία των μοντέλων, Adobe Photoshop και After Effect για την επεξεργασία εικόνας, βίντεο. Τέλος το απαραίτητο συστατικό για την υλοποίηση είναι ο ηλεκτροεγκεφαλογράφος Mindwave που λαμβάνονται τα εγκεφαλικά κύματα ως δεδομένα προς το πρόγραμμα ανάπτυξης.

1.4 Σκοπός Και Στόχος Εργασίας

Πρωταρχικός στόχος είναι η εκμετάλλευση του είδη υπάρχοντος εξοπλισμού Mindwave (EEG), ο οποίος προέρχεται από το εργαστήριο Ευφυών Συστημάτων του ΤΕΙ Κρήτης, αμέσως επόμενος και προσωπικός είναι η εφαρμογή νέων τεχνολογιών και η βέβαιος η μάθηση ενός καινούργιου αντικειμένου.

Οι στόχοι που της εργασίας είναι η επίτευξη της σωστής λειτουργίας του παιχνιδιού, όπως επίσης η δημιουργία ενός όμορφου περιβάλλοντος που ο χρήστης θα περάσει ευχάριστα το χρόνο του. Επίσης όπως έχει αναφερθεί και παραπάνω είναι η επίτευξη περαιτέρω ιδιοτήτων του παίκτη πέραν των καθιερωμένων όπως για παράδειγμα, να πάρει ένα αντικείμενο από τη σκηνή αλλά να γίνετε μόνο του με τη δύναμη της σκέψης του.

Ένας ακόμη στόχος μελλοντικής επέκτασης του παιχνιδιού είναι ο χρήστης να είναι σε θέση να ελέγχει το παιχνίδι εξολοκλήρου χωρίς τη χρήση πληκτρολογίου ή joystick αλλά με τη δραστηριότητα του εγκεφάλου και κινήσεων.

1.5 Δομή Εργασίας

Κεφάλαιο 2 : Μεθοδολογία

Στο κεφάλαιο αυτό περιγράφετε η μέθοδος, η έρευνα υλικού για την έναρξη της εργασίας και τα βήματα που ακολουθήθηκαν για την εκπόνηση της, όπως επίσης γίνετε αναφορά και για τον απαραίτητο εξοπλισμό. Σε αυτό το σημείο θα δείξουμε αναλυτικά τη μεθοδολογία και τον κατακερματισμό του προβλήματος σε μικρότερα μέρη έτσι ώστε να γίνει πιο ομαλή και πιο εύκολη η υλοποίηση του παιχνιδιού.

[σελίδες 11 - 12]

Κεφάλαιο 3 : Ανάλυση Μελέτη ηλεκτρονικού Παιγνίου

Στο κεφάλαιο αυτό θα ασχοληθούμε με τις κατηγορίες παιχνιδιών που υπάρχουν και την επιλογή του κατάλληλου προς εμάς για υλοποίηση παιχνίδι (η επιλογή του είδους είναι εντελώς προσωπική). Επιπλέον, θα αναφέρουμε το σενάριο το οποίο είναι απαραίτητο να υπάρχει εξαρχής για να δημιουργηθούν τα κατάλληλα μοντέλα όπως οι χαρακτήρες, τα κτήρια, η εποχή που δίνουν τον χαρακτήρα ενός παιχνιδιού.

[σελίδες 14 - 17]

Κεφάλαιο 4 : Εργαλεία Υλοποίησης

Για τη δημιουργία ενός ηλεκτρονικού παιχνιδιού χρειάζεται η χρήση διάφορων προγραμμάτων για το τελικό αποτέλεσμα. Σε αυτό το κεφάλαιο θα δείξουμε τα προγράμματα αυτά και την χρήση τους. Τα προγράμματα αυτά που χρειαζόμαστε είναι το Unity3D στο οποίο γίνετε εξολοκλήρου το στήσιμο του παιχνιδιού, το 3ds Max για την επεξεργασία των μοντέλων, το Adobe Photoshop για την επεξεργασία διάφορων textures και εικόνων, το After Effect για την επεξεργασία των βίντεο. Χρειαζόμαστε επίσης μία βιβλιοθήκη (Thinkgear) για τον ηλεκτροεγκεφαλογράφο που είναι απαραίτητη για την ενσωμάτωση του στο Unity3D. Τέλος, είναι απαραίτητη η γνώση αντικειμενοστραφή προγραμματισμού συγκεκριμένα C#.

[σελίδες 20 - 36]

Κεφάλαιο 5 : Υλοποίηση

Στο κεφάλαιο αυτό θα δείξουμε πως έγινε η υλοποίηση και πως έγινε ο κατάλληλος συνδυασμός για το τελικό αποτέλεσμα. Θα μπορούσαμε να πούμε ότι το κεφάλαιο αυτό διακρίνετε σε δύο επιμέρους μέρη. Την υλοποίηση του παιχνιδιού στο περιβάλλον του

Unity3D και το δεύτερο μέρος είναι η προσαρμογή του EEG και οι εφαρμογή του μέσα στο παιχνίδι.

[σελίδες 42 - 71]

Κεφάλαιο 6: Αποτελέσματα

Στο κεφάλαιο αυτό παραθέτουμε τα αποτελέσματα και κάποιες παρατηρήσεις που προέκυψαν από τη συνολική εργασία, όπως επίσης θα αναφερθούν και μελλοντικές βελτιώσεις.

[σελίδες 73 - 79]

2. Μεθοδολογία

Η μέθοδος που ακολουθήθηκε είναι γνωστή ως μέθοδος ‘Ανάλυσης Περιεχομένου’, η μέθοδος έρευνας αυτή, αντλεί στοιχεία για την έρευνα του θέματος από πηγές όπως για παράδειγμα έντυπο υλικό. Βασικές απαιτήσεις αυτής της μεθόδου είναι η αντικειμενικότητα, η εγκυρότητα και η αξιοπιστία.

Ο παρακάτω πίνακας παρουσιάζει τα βασικά στάδια διεξαγωγής :

Βήματα Μελέτης και Υλοποίησης																	
Εργασία	**Σειρά Ενεργειών**																
Προσδιορισμός Θέματος	■																
Ανασκόπηση Σχετικής Βιβλιογραφίας		■													■		■
Επιλογή Μεθόδου – Ανάπτυξης			■		■			■			■						
Μελέτη Συλλογής Υλικού				■													
Υλοποίηση Γραφικού Μέρους						■											
Έλεγχος Γραφικού Μέρους						■	■										
Εφαρμογή EEG									■								
Έλεγχος Λειτουργίας EEG									■	■							
Υλοποίηση Λογισμικού												■					
Αποσφαλμάτωση Λογισμικού												■	■		■	■	■
Αποσφαλμάτωση Συνόλου														■		■	■

Πίνακας 1: Βήματα Μελέτης και Υλοποίησης

Όπως έγινε αναφορά το πρώτο στάδιο είναι ο προσδιορισμός του θέματος, το αμέσως επόμενο βήμα είναι η αναζήτηση πηγών και ο έλεγχος της εγκυρότητας τους. Αμέσως μετά από αυτό το στάδιο επιλέγουμε τον τρόπο υλοποίησης και σε ποσά τμήματα θα πρέπει να κατακερματίσουμε το πρόβλημα για τη διευκόλυνσή μας αλλά και να μειώσουμε το πρόβλημα σε μικρότερα και ευκολότερα μέρη για να φτάσουμε στο επιθυμητό αποτέλεσμα «Διαίρει και Βασίλευε».

2.1 Ανάλυση Προβλήματος

Για την μελέτη και την ανάλυση του συνόλου της πτυχιακής εργασίας προαπαιτήθηκε ο χωρισμός της σε τμήματα έτσι ώστε ο καθορισμός των εργασιών που έπρεπε να γίνουν και η αναζήτηση πληροφοριών για κάθε σημείο της εργασίας να επιτευχθεί χωρίς προβλήματα. Έτσι λοιπόν μπορούμε να διακρίνουμε μερικές ενότητες.

Σενάριο

Η δημιουργία του σεναρίου κατέχει βασικό ρόλο στην υλοποίηση ενός ηλεκτρονικού παιχνιδιού, καταρχάς είναι απαραίτητο γιατί με βάση το σενάριο κατευθύνονται τα γραφικά, η εποχή, το ύφος και συνεπώς το είδος του παιχνιδιού. Το σενάριο είναι βασισμένο σε αληθινά ιστορικά γεγονότα την εποχή της λήξης του δευτέρου παγκοσμίου πολέμου το 1942 συνεπώς το πρώτο βήμα είναι να δημιουργήσουμε και να συλλέξουμε 3D μοντέλα που αρμόζουν σε εκείνη την εποχή.

Λογική

Με το όρο Λ*ογική* εννοούμε τη συνοχή, η λογική ενός παιχνιδιού είναι και αυτό απαραίτητο κομμάτι για την πραγματοποίηση του. Σε αυτό το σημείο ορίζετε το είδος του, αν θα είναι παραδείγματος χάριν FPS Game (First Person Shooter), Mini Game, Racing Game ή Adventure Game και ούτω καθεξής. Στη περίπτωση μας η επιλογή που έγινε είναι να δημιουργήσουμε ένα Puzzle Game αυτό το είδος παιχνιδιού απαιτεί την επίλυση λογικών γρίφων συνεπώς θα πρέπει να δημιουργήσουμε γρίφους, διαλόγους και στοιχεία για να οδηγηθεί ο χρήστης σε κάποιο αποτέλεσμα και να τερματίσει το παιχνίδι. Αυτού του είδους ηλεκτρονικά παιχνίδια τις περισσότερες φορές συνδυάζουν τις κατηγορίες Adventure και Educational Game.

Σχεδιασμός πίστας

Σε αυτό το σημείο θα ασχοληθούμε με τη σύνθεση και τη μελέτη του γραφικού περιβάλλοντος.

Προγραμματισμός μοντέλων

Αυτή η ενότητα συνδέεται παράλληλα και με το σχεδιασμό της πίστας διότι δε χρησιμοποιούμε μόνο στατικά μοντέλα αλλά και κινούμενα όπως οι άνθρωποι, ένα παράδειγμα είναι τα φώτα ενός περιπολικού που εναλλάσσονται μεταξύ τους και το βλέπουμε όταν παίζουμε το παιχνίδι.

Προγραμματισμός λογικής

Ένα ακόμη κομμάτι που πρέπει να διαχωριστεί από τα υπόλοιπα είναι και αυτό του προγραμματισμού της λογικής. Σε αυτό το κομμάτι γίνετε ο προγραμματισμός του menu, του Inventory, του Dialogue System και του EEG.

3. Ανάλυση Μελέτη Ηλεκτρονικού Παιγνίου

Λέγοντας ηλεκτρονικό παιχνίδι εννοούμε την αλληλεπίδραση με μια διεπαφή χρήστη για την παραγωγή οπτικής ανάδρασης σε μια συσκευή βίντεο. Η λέξη *βίντεο* στο *βιντεοπαιχνίδι* παραδοσιακά αναφερόταν σε μια συσκευή εμφάνισης. Ωστόσο, η δημοφιλής χρήση του όρου «βιντεοπαιχνίδι», τώρα υπονοεί κάθε τύπο συσκευής εμφάνισης, που μπορεί να απεικονίσει δισδιάστατα ή τρισδιάστατα γραφικά. Τα ηλεκτρονικά συστήματα που χρησιμοποιούνται για να παιχτούν βιντεοπαιχνίδια είναι γνωστά ως πλατφόρμες. Παραδείγματα αυτών είναι οι προσωπικοί υπολογιστές και οι παιχνιδομηχανές (Eck, 2006). Αυτές οι πλατφόρμες εκτείνονται από μεγάλα συστήματα υπολογιστή μέχρι συσκευές χειρός. Ειδικευμένα βιντεοπαιχνίδια όπως τα παιχνίδια arcade, ενώ προηγουμένως ήταν κοινά, έχουν σταδιακά μειωθεί σε χρήση.

Πρόσθετες δυνατότητες

Τα βιντεοπαιχνίδια τυπικά χρησιμοποιούν επιπρόσθετα μέσα, με τα οποία παρέχουν αλληλεπίδραση και πληροφορίες στον παίκτη. Η χρήση ήχου είναι καθολική, με περιφερειακά αναπαραγωγής ήχου, όπως τα ηχεία και τα ακουστικά. Επιπρόσθετα, ανάδραση πληροφορίας προέρχεται από οπτικά περιφερειακά, με ανάδραση δόνησης ή/και δύναμης.

Τα ηλεκτρονικά παιχνίδια δημιουργούνται ως δημιουργική διέξοδος, από ερασιτέχνες, και για την παραγωγή κέρδους από mainstream αλλά και από ανεξάρτητες εταιρείες. Η ανάπτυξη του παιχνιδιού λαμβάνει χρηματοδότηση, συνήθως, από έναν εκδότη (publisher). Ηλεκτρονικά παιχνίδια υψηλής ποιότητας θα αποφέρουν στην εταιρεία κέρδη άμεσα. Παρ' όλα αυτά, είναι σημαντικό να ληφθούν υπόψη οι οικονομικές απαιτήσεις που θα χρειαστεί το παιχνίδι, όπως τα κόστη ανάπτυξης των επιμέρους χαρακτηριστικών του. (Gibson, 2014)

Η βιομηχανία παιχνιδιών απαιτεί καινοτομίες καθώς οι εκδότες δεν μπορούν να επωφεληθούν από την ανάπτυξη επαναλαμβανόμενων παιχνιδιών. Κάθε χρόνο εμφανίζονται νέες ανεξάρτητες εταιρείες ανάπτυξης ηλεκτρονικών παιχνιδιών, οι οποίες καταφέρνουν να παράγουν έναν τουλάχιστον επιτυχημένο τίτλο. Ομοίως, πολλοί developers κλείνουν επειδή δεν μπορούν να βρουν μια εκδοτική σύμβαση ή γιατί οι παραγωγές τους δεν είναι επικερδής. Είναι δύσκολο να ξεκινήσει μια νέα εταιρεία λόγω της υψηλής αρχικής επένδυσης που απαιτείται. Παρ 'όλα αυτά, η ανάπτυξη casual παιχνιδιών (ιδιαίτερα για κινητά τηλέφωνα), επέτρεψε σε μικρότερες ομάδες να εισέλθουν στην αγορά. Μόλις οι επιχειρήσεις σταθεροποιηθούν οικονομικά, μπορεί να επεκταθούν και να πάρουν το ρίσκο να αναπτύξουν μεγαλύτερης κλίμακας παιχνίδια. (Gibson, 2014)

Η ανάπτυξη ηλεκτρονικών παιχνιδιών (Game Development) είναι μια διαδικασία που ξεκινά από μια ιδέα ή έννοια. Συχνά, η ιδέα βασίζεται σε μια τροποποίηση κάποιου ήδη

υπάρχοντος concept παιχνιδιού. Η ιδέα του παιχνιδιού μπορεί να εμπίπτει ανάμεσα σε ένα ή περισσότερα είδη. Οι σχεδιαστές συχνά πειραματίζονται με διαφορετικούς συνδυασμούς παιχνιδιών. Στην συνέχεια ο game designer παράγει ένα πρώτο αρχείο προτάσεων, το οποίο περιέχει το concept, το gameplay, την λίστα των χαρακτηριστικών του παιχνιδιού, την τοποθεσία και την ιστορία, το κοινό που στοχεύει, τις απαιτήσεις και το budget και πολλά άλλα. Η κάθε εταιρεία έχει την δική της φιλοσοφία πάνω στο στάδιο αυτό αλλά υπάρχουν πολλές κοινές πρακτικές. Τα βιντεοπαιχνίδια έχουν πλέον εξελιχθεί σε μία μορφή τέχνης και σε βιομηχανία.

3.1 Είδη Ηλεκτρονικών Παιγνίων

Τα ηλεκτρονικά παιχνίδια μπορούν να χωριστούν σε είδη:

- **MMORPG'S (Massive multiplayer online role playing game):** Τα παιχνίδια αυτά διαφέρουν από τα υπόλοιπα παιχνίδια σε σύνδεση για δύο λόγους: 1) Για το μεγάλο αριθμό ταυτόχρονων παικτών οι οποίοι συμμετέχουν σε ένα μεμονωμένο παιχνίδι, και 2) για την επίμονη φύση των παιχνιδιών. Τα παιχνίδια ρόλων όπου οι συμμετέχοντες αναλαμβάνουν τους ρόλους φανταστικών χαρακτήρων και δημιουργούν ή ακολουθούν ιστορίες σε συνεργασία, κυριαρχούν σε αυτή τη κατηγορία. Σε ένα τέτοιο παιχνίδι ο παίκτης φτιάχνει έναν χαρακτήρα με συγκεκριμένα χαρακτηριστικά, που προσδιορίζονται κατά βάση από την φυλή και την ιδιότητα που επιλέγει αλλά και από τα επιτεύγματα του μέσα στον αχανή κόσμο του παιχνιδιού. Η απήχησή τους οφείλεται στο ότι μπορούν να καλύψουν πολλές διαφορετικές κατηγορίες των Gamers.

- **Stand-Alone:** Τα παιχνίδια που υπάγονται στην κατηγορία αυτή περιλαμβάνουν έναν παίκτη μόνο στον υπολογιστή, δίνουν ωστόσο την δυνατότητα σε όποιον το επιθυμεί να συνδεθεί στο διαδίκτυο και να βρει κάποιον αντίπαλο. Τέτοια παιχνίδια είναι το The Simpsons Arcade Game και το Lost Odyssey.

- **Local and Wide Networks Games:** Τα παιχνίδια αυτά δημιουργήθηκαν από την επιθυμία να ενωθούν μεταξύ τους αρκετοί παίκτες ώστε να υποστηρίζονται παιχνίδια τουρνουά. Το κοινό όλων αυτών των παιχνιδιών είναι μια σχετικά περιορισμένη αφήγηση, με έμφαση στην στρατηγική. Η ανάπτυξη των χαρακτήρων είναι μηδαμινή, αν όχι ανύπαρκτη. Αυτό το στυλ παιχνιδιού ενθαρρύνει τους παίκτες να δημιουργήσουν ομάδες ακόμα και ομάδες.("Clans"). Αυτό το είδος παιχνιδιού έχει γίνει τόσο δημοφιλές που διοργανώνονται ακόμα και πάρτυ LAN, όπου εκατοντάδες άτομα συνδέονται για να διαγωνιστούν.

- **Mini Games:** Αυτά είναι δικτυακές εκδοχές κλασσικών arcade, επιτραπέζιων ή ψηφιακών παιχνιδιών. Είναι συνήθως δωρεάν και είναι συχνά διαθέσιμα σε

δικτυακούς τόπους και κόμβους παιχνιδιών που υποστηρίζονται από τη διαφήμιση. Αυτά τα παιχνίδια είναι κυρίως για έναν παίκτη, και δεν έχουν σχέση με κάποιο εικονικό, αφηγηματικό κόσμο.

- **FPS (first person shooters):** Όπως φανερώνουν οι δυο πρώτες λέξεις, πρόκειται για τα παιχνίδια οπτικής πρώτου προσώπου. Το 'shooters' σημαίνει ότι ο παίκτης φέρει στη συντριπτική πλειοψηφία των περιπτώσεων , κάποιο όπλο για να αντιμετωπίσει τις ορδές εχθρών που απειλούν τη ζωή του χαρακτήρα του παιχνιδιού. Χρησιμοποιούνται ειδικές μηχανές τρισδιάστατης απεικόνισης ώστε να δίνεται η ψευδαίσθηση του πραγματικού χώρου (όπως και στις (κυρίως πειραματικές ακόμα) εφαρμογές εικονικής πραγματικότητας). Το συγκεκριμένο είδος των παιχνιδιών είναι ιδιαίτερα δημοφιλές για lan parties και διαγωνισμούς , τα ίντερνετ καφέ , όπου οι παίκτες χωρίζονται σε δύο συνήθως ομάδες και διαγωνίζονται σε διάφορα 'σενάρια'. Χαρακτηριστικά παιχνίδια αυτής της τεχνικής είναι το Counter-Strike, το Call of Duty και άλλα.

- **Adventure Games:** Θεωρούν τον παίκτη ως πρωταγωνιστή μιας ιστορίας στην οποία συμμετέχει ο ίδιος. Συνήθως τα παιχνίδια αυτής της κατηγορίας προϋποθέτουν ότι ο παίκτης θα λύσει διάφορα παζλ και θα βρει διάφορα τεχνουργήματα. Η πρώτη γενιά αυτής της κατηγορίας ηλεκτρονικών παιχνιδιών βασίζονταν σε κείμενο, στη συνέχεια σε μίξη εικονικού περιβάλλοντος με κείμενο και στην παρούσα φάση βασίζονται στο "point-n-click".Σήμερα τα περισσότερα βασίζονται σε παιχνίδια 'επιστημονικής φαντασίας'. Ένας πιο ακριβής όρος για αυτή τη κατηγορία είναι ο όρος 'action adventure'. Το πρώτο ηλεκτρονικό παιχνίδι αυτής της κατηγορίας δημιουργήθηκε το 1970 με όνομα 'Colossal Cave Adventure'.Στην ορολογία των console video games τα παιχνίδια αυτής της κατηγορίας εμπεριέχουν την εξερεύνηση και την αλληλεπίδραση με το περιβάλλον του παιχνιδιού. Ο όρος adventure εδώ αναφέρεται στο παιχνίδι 'Adventure' (Atari 1978).

- **Fighting:** Είναι παιχνίδια που γίνονται μάχες μεταξύ δύο παικτών από τους οποίους ο ένας μπορεί να ελέγχεται από τον υπολογιστή. Αυτή η κατηγορία ηλεκτρονικών παιχνιδιών εμφανίστηκε στα μέσα του 1980 και σήμερα είναι πολύ δημοφιλής.

- **Puzzle:** Απαιτούν από το χρήστη να επιλύσει διάφορα λογικά παζλ ή να κατευθυνθεί σε περίπλοκες περιοχές όπως είναι οι λαβύρινθοι. Αυτά τα ηλεκτρονικά παιχνίδια τις περισσότερες φορές συνδυάζουν τις κατηγορίες Adventure και Educational.

- **RTS (Real Time Strategy):** Σε ένα παιχνίδι RTS ο παίκτης αναλαμβάνει την διαχείριση μονάδων και υλικών μιας ομάδας με σκοπό την ανάπτυξή της. Η ανάπτυξη αυτή επιτυγχάνεται με την εκμετάλλευση των πόρων του περιβάλλοντος ενώ ο παίκτης θα πρέπει να φροντίζει η αναπτυσσόμενη κοινότητά του να έχει πάντοτε επάρκεια πόρων και μονάδων. Ένας γύρος σε ένα RTS τελειώνει με την

τελική μάχη ανάμεσα στους παίκτες , οι οποίοι έχουν φτιάξει τους στρατούς τους και οργανώσει τις άμυνες τους. Γνωστά παιχνίδια στρατηγικής είναι το Travian, Grepolis, The West, War2, Ikariam .

3.2 Επιλογή Και Ανάλυση

Τα Adventure Games θεωρούν τον παίκτη πρωταγωνιστή μιας ιστορίας στην οποία συμμετέχει ο ίδιος. Συνήθως τα παιχνίδια αυτής της κατηγορίας προϋποθέτουν ότι ο παίκτης θα λύσει διάφορα παζλ και θα βρει διάφορα τεχνουργήματα. Ένας πιο ακριβής όρος για αυτή τη κατηγορία είναι ο όρος 'action adventure' στην ορολογία των console video games τα παιχνίδια αυτής της κατηγορίας εμπεριέχουν την εξερεύνηση και την αλληλεπίδραση με το περιβάλλον του παιχνιδιού.

Η υλοποίηση του παιχνιδιού έγινε βασιζομένη πάνω σε αυτό το είδος παίγνιου. Μέσω αυτής της επιλογής η υλοποίηση και οι εφαρμογές του ηλεκτροεγκεφαλογράφου (EEG) ανταποκρίνονται καλυτέρα στην ανάδειξη του μέσα στο παιχνίδι.

Σύμφωνα με τη πρόταση του Lars Konzack (Konzack, 2002) για την ανάλυση των παιχνιδιών αποτελεί μια από κάτω προς τα πάνω (bottom-up) προσέγγιση, ένα μεθοδολογικό πλαίσιο που συνίσταται σε επίπεδα, όπως και αυτή του Lindley, ταξινομημένα όμως σε κάποια ιεραρχία. Το μοντέλο του Konzack αναγνωρίζει επτά διαφορετικά στρώματα ανάλυσης, καθένα από τα οποία είναι δυνατόν να αναλυθεί ξεχωριστά. Επιπλέον, η ανάλυση ενός παιχνιδιού είναι εφικτή χωρίς να ληφθούν υπόψη όλα τα στρώματα.

Ωστόσο, μια ολοκληρωμένη ανάλυση θα πρέπει, σύμφωνα με τον Konzack, να περιλαμβάνει όλες τις διαφορετικές οπτικές γωνίες, δηλαδή ένα παιχνίδι θα ήταν θεμιτό να αναλυθεί από τεχνική, αισθητική αλλά και κοινωνικοπολιτιστική άποψη. Ο Espen Aarseth, σχολιάζει σχετικά ότι η καλύτερη χρήση της μεθόδου του Konzack θα ήταν αυτή που θα την αναγνώριζε ως ένα ανοιχτό πλαίσιο, στο οποίο ο αναλυτής έχει τη δυνατότητα να επιλέξει οποιαδήποτε 2-4 από τα επτά στρώματα και να εργαστεί με αυτά, αναλύοντάς τα μαζί και αγνοώντας τα υπόλοιπα. (Konzack, 2002)

Τα στρώματα του μοντέλου του Konzack είναι τα εξής:

- Υλικό (Hardware)

Ο υπολογιστής στον οποίο εκτελείται το παιχνίδι μπορεί να είναι διαφόρων τύπων, όπως για παράδειγμα, προσωπικός υπολογιστής συνδεδεμένος στο διαδίκτυο, κινητό τηλέφωνο, κονσόλα παιχνιδιών κλπ. Τα χαρακτηριστικά και οι δυνατότητες του

παιχνιδιού εξαρτώνται σε μεγάλο βαθμό από τις δυνατότητες του φυσικού μέσου, το υλικό ωστόσο δε μπορεί να προσδιορίσει τον τύπο του προς εξέταση παιχνιδιού.

- Κώδικας προγράμματος (Program code)

Ο κώδικας μπορεί να μας δώσει μια καλή εικόνα σχετικά με τις ενέργειες του υπολογιστή κατά την εκτέλεση του παιχνιδιού. Ωστόσο, όπως σχολιάζει ο Konzack, ακόμη και χωρίς να έχουμε πρόσβαση στον κώδικα του παιχνιδιού είμαστε σε θέση να περιγράψουμε και να αναλύσουμε όλα τα υπόλοιπα στρώματα του παιχνιδιού, ενώ μεγάλο ποσοστό του προγράμματος μπορεί να αναλυθεί έμμεσα, μέσω του στρώματος της λειτουργικότητας, το οποίο εξετάζεται παρακάτω.

- Λειτουργικότητα (Functionality)

Η λειτουργικότητα εξαρτάται από τον κώδικα και από τα φυσικά χαρακτηριστικά του υπολογιστή και αφορά στη συμπεριφορά του υπολογιστή και στις αποκρίσεις της διεπαφής του στην είσοδο του χρήστη, δηλαδή στη διάδραση που προσφέρει. Όπως σχολιάζει ο Konzack, στο στρώμα αυτό δεν διευκρινίζεται ακόμη ο παιγνιώδης χαρακτήρας του προς εξέταση παιχνιδιού, δηλαδή αυτό θα μμπορούσε να αποτελεί μια οποιαδήποτε πολυμεσική εφαρμογή.

3.3 Σενάριο - Ιστορικά Γεγονότα

“The Little Boy”

*Το όνομα του παιχνιδιού είναι εμπνευσμένο από το όνομα της βόμβας που ρήφθηκε από τους Αμερικανούς στις 6 Αυγούστου το 1945 λίγο πριν τη λήξη του Β' Παγκοσμίου πολέμου.

Τον Αύγουστο του 1942 ξεκίνησε το μυστικό στρατιωτικό πρόγραμμα για την κατασκευή της ατομικής βόμβας με την κωδική ονομασία « Σχέδιο Μανχάταν» (The Manhattan Project). Επικεφαλής του προγράμματος ήταν ο στρατηγός Leslie Groves. Το εργαστήριο, στο οποίο γινόταν ο σχεδιασμός της ατομικής βόμβας, βρισκόταν στο Λος Αλάμος της πολιτείας του Νέου Μεξικού και επιστημονικός επικεφαλής του εργαστηρίου είχε διορισθεί ο Robert Oppenheimer. Στην πολιτεία Τενεσί κατασκευάστηκαν τεράστια εργοστάσια για το διαχωρισμό του ουρανίου -235, ενώ στην πολιτεία Ουάσιγκτον κατασκευάστηκαν πυρηνικοί αντιδραστήρες για την παραγωγή του πλουτωνίου -239.

Οι επιστήμονες που συμμετείχαν στο πρόγραμμα είχαν εγκριθεί και από τον στρατό και από τις μυστικές υπηρεσίες. Κάποιοι από αυτούς ήταν κάτοχοι του βραβείου Νόμπελ

Φυσικής ή Χημείας και κάποιοι άλλοι βραβεύτηκαν μετά τον πόλεμο με το βραβείο Νόμπελ. Αυτό δείχνει ότι η κατασκευή της βόμβας απαιτούσε τεράστια θεωρητική και πειραματική έρευνα, η οποία γινόταν ομαδικά από μια πλειάδα ικανών επιστημόνων.

Αποτέλεσμα του Σχεδίου Μανχάταν ήταν η κατασκευή τριών ατομικών βομβών. Η πρώτη ρίχθηκε για σκοπούς δοκιμής στις 16 Ιουλίου 1945 σε μια ερημική περιοχή του Νέου Μεξικού. Ήταν μια βόμβα πλουτωνίου. Η δεύτερη, με την κωδική ονομασία «Little Boy», ήταν η βόμβα ουρανίου που έπεσε στη Χιροσίμα στις 6 Αυγούστου 1945. Η Τρίτη με το όνομα «Fat Man» ήταν η βόμβα Πλουτωνίου που κατέστρεψε τη Χιροσίμα στις 9 Αυγούστου.

Η απόφαση για το βομβαρδισμό της Ιαπωνίας με τις ατομικές βόμβες ήταν μια καθαρά στρατιωτικοπολιτική απόφαση και η άποψη των επιστημόνων που κατασκεύασαν τη βόμβα δεν είχε βαρύνουσα σημασία. Αρκετοί από αυτούς ανάμεσα τους και ο

Robert Oppenheimer εναντιώθηκαν στη χρησιμοποίηση της βόμβας , αφού ήταν φανερό ότι ο πόλεμος έβαινε προς το τέλος του. Αργότερα ο Robert Oppenheimer εναντιώθηκε στην κατασκευή της βόμβας υδρογόνου, μιας βόμβας πολύ πιο ισχυρής από την ατομική. Τάχθηκε επίσης υπέρ της χρησιμοποίησης της ατομικής ενέργειας για ειρηνικούς σκοπούς. Για τη στάση του αυτή διώχθηκε από τις αμερικανικές αρχές και κατηγορήθηκε αβάσιμα ότι ήταν κατάσκοπος της Σοβιετικής Ένωσης.

Τα παραπάνω ιστορικά γεγονότα στάθηκαν η αφορμή για τη δημιουργία σεναρίου του παιχνιδιού. Έχουμε δυο επιστήμονες που έχουν βασικές αντιπαραθέσεις για τη χρήση της ατομικής ενέργειας, τον επιστημονικό επικεφαλή Robert Oppenheimer της βόμβας ουρανίου και τον Eduard Teller, ο οποίος τέθηκε επικεφαλής του σχεδίου για την κατασκευή της βόμβας υδρογόνου μιας βόμβας πολύ πιο ισχυρής από την ατομική ο οποίος συμφώνησε υπέρ για το βομβαρδισμό της Ιαπωνίας.

Λόγω αυτής της διαφωνίας και των αντικρουόμενων συμφερόντων ο Eduard Teller κατηγόρησε αβάσιμα τον Robert Oppenheimer ότι ήταν κατάσκοπος της Σοβιετικής Ένωσης με αποτέλεσμα λίγο πριν τη σύλληψη του από τις αμερικανικές αρχές να σχεδιάσει τη δολοφονία του Eduard Teller.

4. Εργαλεία Υλοποίησης

Για την υλοποίηση του παιγνίου χρειάστηκε μόνο η πλατφόρμα λογισμικού Unity3D που είναι διαθέσιμο δωρεάν, εκεί χτίστηκαν τα γραφικά και η λογική του παιχνιδιού. Επίσης υπάρχει και η εμπορική έκδοση που προφέρει περισσότερες δυνατότητες. Στην εργασία χρησιμοποιήθηκε η δωρεάν έκδοση. Για την καταγραφή και επεξεργασία των σημάτων που λαμβάνουμε από τον ηλεκτροεγκεφαλογράφο, δημιουργήσαμε μια σύνδεση από τη συσκευή του EEG προς την πλατφόρμα, θα παρουσιαστεί παρακάτω πως έγινε η σύνδεση.

4.1 Ηλεκτροεγκεφαλογράφος (EEG)

Ο ανθρώπινος εγκέφαλος αποτελείται από δισεκατομμύρια διασυνδεδεμένους νευρώνες, τα πρότυπα αλληλεπίδρασης μεταξύ αυτών εκπροσωπούνε ως σκέψεις και συναισθηματικές καταστάσεις. Κάθε αλληλεπίδραση μεταξύ των νευρώνων δημιουργεί ηλεκτρικές εκκενώσεις. Η δραστηριότητα αυτή δημιουργεί κάποιες συχνότητες τα λεγόμενα εγκεφαλικά κύματα τα οποία μπορούν να μετρηθούν από μια τέτοια συσκευή. (Electroencephalography, n.d.)

Το EEG μέτρα τις διακυμάνσεις της τάσης που προκύπτουν από ιοντικό ρεύμα που ρέει εντός των νευρώνων του εγκεφάλου. Σε κλινικά πλαίσια, το EEG αναφέρεται στην καταγραφή της αυθόρμητης ηλεκτρικής δραστηριότητας του εγκεφάλου σε σύντομο χρονικό διάστημα, συνήθως 20-40 λεπτά, όπως καταγράφεται από τα πολλαπλά ηλεκτρόδια που τοποθετούνται στο τριχωτό της κεφαλής.

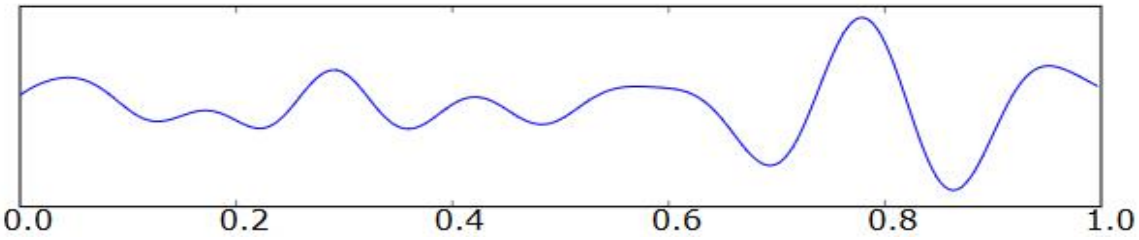
Ωστόσο, περιγράφει τη ρυθμική δραστηριότητα. Η ρυθμική δραστηριότητα χωρίζεται σε ζώνες ανάλογα με τη συχνότητα. Ορίζει τα πρότυπα των ζωνών συχνοτήτων (δηλαδή, κάθε ρυθμική δραστηριότητα μεταξύ 6-12 Hz μπορεί να περιγραφεί ως ‘άλφα’). Οι διάφορες καταστάσεις του εγκεφάλου προκαλούν το αποτέλεσμα των διαφόρων μορφών νευρικών αλληλεπιδράσεων. Αυτά τα πρότυπα οδηγούν σε κύματα που χαρακτηρίζονται από διαφορετικά εύρη και συχνότητες, για παράδειγμα τα κύματα μεταξύ 12 και 30 hertz, λέγοντε Κύματα Βήτα και συνδέονται με τη συγκέντρωση, ενώ τα κύματα μεταξύ 8 και 12 hertz, λέγοντε Κύματα Άλφα τα οποία συνδέονται με την χαλάρωση και την κατάσταση της ψυχικής ηρεμίας. Ακόμα η συσκευή αυτή μπορεί να ανιχνεύσει το άνοιγμα και το κλείσιμο των βλεφάρων. (Salabun, 2014)

Στην εργασία χρησιμοποιήθηκε η συσκευή Mindwave της NeuroSky, Inc. Είναι μια σχετικά προσιτή οικονομικά συσκευή που μπορεί να χρησιμοποιηθεί εκτός εργαστηριακού χώρου διότι έχει τη δυνατότητα μεταφερσιμότητας λόγο του ιδανικού μεγέθούς της. Είναι μια εξαιρετική επιλογή για gaming, έρευνα, ή δημόσιες εγκαταστάσεις. Μια τέτοια

συσκευή μπορεί να χρησιμοποιηθεί σε διάφορους τομείς όπως είναι η ψυχαγωγία, την εκπαίδευση, την υγεία ακόμα και την αυτοκινητοβιομηχανία για τη δημιουργία εφαρμογών πρόληψης ατυχημάτων.

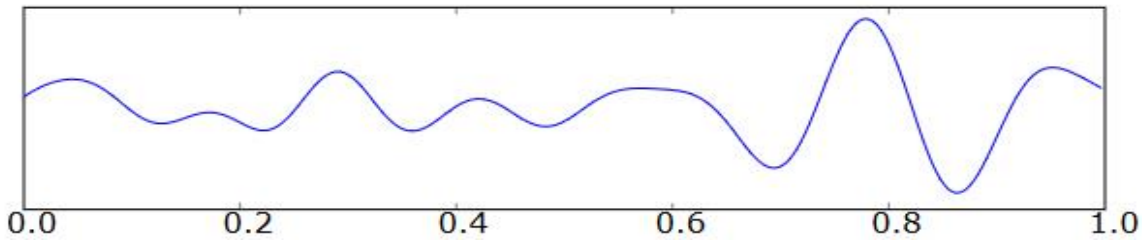
Εγκεφαλικά κύματα

- Τα **κύματα *δ*,** η περιοχή συχνοτήτων κυμαίνεται από 0 έως και 4 Hz. Τείνει να έχουν το υψηλότερο πλάτος και τη βραδύτερη κυματομορφή. Βιώνουμε αυτή την κατάσταση στα βαθύτερα στάδια του ύπνου.



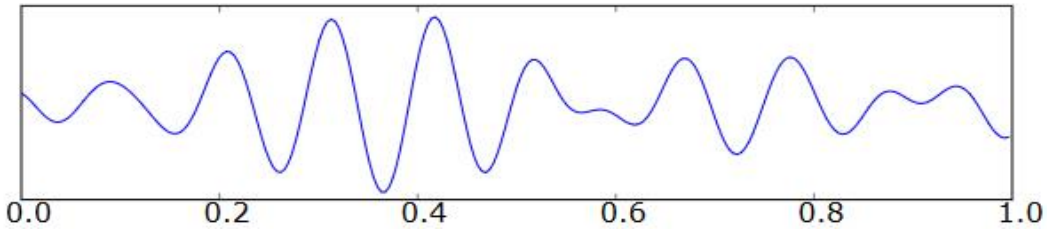
Εικόνα 5: Τα κύματα δ (0-4 Hz)

- Τα **κύματα *θ*,** η περιοχή συχνοτήτων κυμαίνεται από 4 Hz έως 7 Hz. Παρατηρείται κανονικά σε μικρά παιδιά. Επίσης παρατηρείται κάποιες φορές όταν μεγαλύτερα παιδιά και ενήλικες βρίσκονται σε κατάσταση υπνηλίας ή διέγερσης, σημαίνει επίσης ότι το άτομό βρίσκετε σε κατάσταση βαθιάς χαλάρωσης.



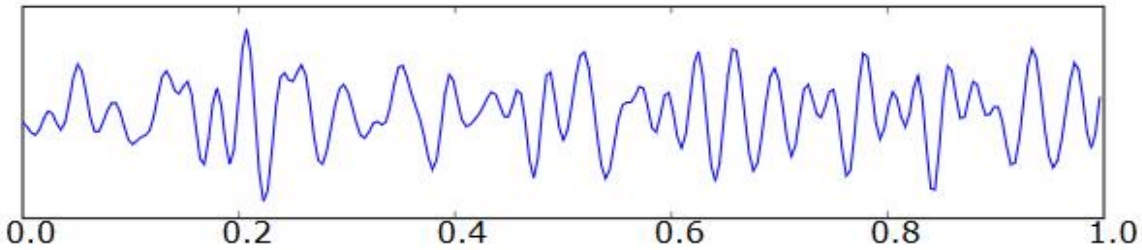
Εικόνα 6: Τα κύματα θ (4-7 Hz)

- Τα **κύματα *α*,** η περιοχή συχνοτήτων κυμαίνεται από 8 Hz έως 12 Hz. Προκύπτουν με το κλείσιμο των ματιών και με τη χαλάρωση και εξασθενεί με το άνοιγμα των ματιών ή με διανοητική άσκηση.



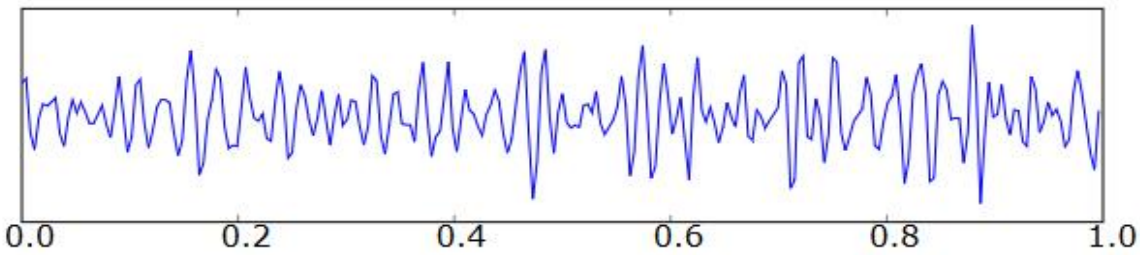
Εικόνα 7: Τα κύματα α (8-12 Hz)

- Τα **κύματα *β,*** η περιοχή συχνοτήτων κυμαίνεται από 12 Hz έως 30 Hz. Παρατηρούνται στην κανονική κατάσταση εγρήγορσης του εγκεφάλου όπου έχουμε πλήρη συνείδηση, ετοιμότητα για σωματική δράση



Εικόνα 8: Τα κύματα β (12-30 Hz)

- Τα **κύματα *γ,*** η περιοχή συχνοτήτων κυμαίνεται από 30 Hz έως 100 Hz. Παρατηρούνται όταν μαθαίνουμε κι επεξεργαζόμαστε πληροφορίες, επίσης εμπλέκονται στην ανώτερη πνευματική δραστηριότητα, συμπεριλαμβανομένης της αντίληψης και της συνείδησης.



Εικόνα 9: Τα κύματα γ (30-100 Hz)

Τεχνικά Χαρακτηριστικά EEG

Οι συσκευές BCI χρησιμοποιούνται για κλινική χρήση. Η μέθοδος αυτή προσφέρει άμεση επικοινωνία μεταξύ του εγκεφάλου με μια εξωτερική συσκευή. Στην αγορά υπάρχουν οικονομικά προσιτές συσκευές που παράγουν ηλεκτροεγκεφαλογραφήματα Electroencephalography Headset (EEG) που είναι εφικτό να χρησιμοποιηθούν και εκτός εργαστηρίου. (Mindwave, n.d.)

Η συγκεκριμένη συσκευή περιλαμβάνει έναν αισθητήρα που αγγίζει το μέτωπο και ένα clip που εφαρμόζετε στο αυτί του χρήστη. Αφού γίνει η λήψη των σημάτων υπάρχει ένας ενσωματωμένος Bio-Sensor που επεξεργάζεται, ψηφιοποιεί και ενισχύει τα αναλογικά σήματα. Τα δεδομένα που λαμβάνονται είναι ακατέργαστα εγκεφαλικά σήματα (raw brainwaves) και οι eSense μετρήσεις όπου είναι το Attention και το Meditation του χρήστη.

Measures:

- Raw-Brainwaves
- EEG power spectrums (Alpha, Beta, etc.)
- eSense meter for Attention and Meditation
- Quality of data being collected (can be used to detect poor contact and whether the device is off the head)

Physical:

- 190g weight
- Length 21cm X width 17.5cm X height 5.0cm

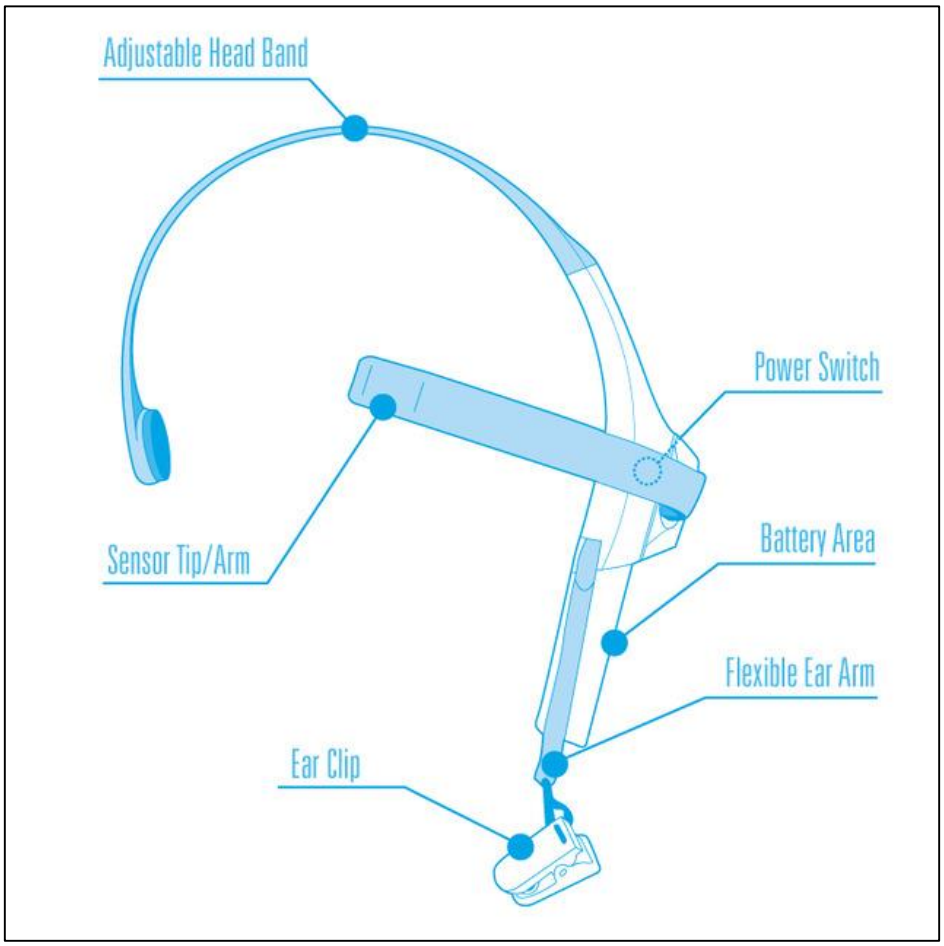
Signal and EEG:

- Maximum signal input range: 1mV pk-pk
- Hardware filter range: 3Hz to 100Hz
- MS001: includes 60Hz environmental AC noise filtering
- MS002: includes 50Hz environmental AC noise filtering
- Amplification gain: 2000x • ADC resolution: 12 bits (-2048 to 2047)
- Sampling rate: 512Hz • eSense interpretation update rate: 1Hz
- 0% byte loss (ie packet loss) • 1Hz eSense interpretation rate

- UART Baudrate: 57,600 Baud
- SPP through put: 9600 Baud
- S/N ratio: >70dB
- Class 2 Bluetooth Radio Range: 10 m 6dBm RF max power
- 250kbit/s RF data rate

Compatible/Recommended Bluetooth Receivers:

- Windows: Mavin Bluetooth Dongle
- Mac OSX: use the built-in Bluetooth receivers



Εικόνα 10: Mindwave Headset (EEG)

4.1.1 Χρήση Και Εφαρμογές

Η εφαρμογή του EEG στην πραγματοποίηση του παίγνιου αποτέλεσε το δεύτερο κομμάτι της υλοποίησης. Το πρώτο αναφέρεται στην εγκατάσταση των μοντέλων, την δημιουργία των σκηνών και των προγραμματισμό τους.

Ο στόχος της υλοποίησης είναι η λήψη των μετρήσεων που προέρχονται από τη συσκευή και η εφαρμογή τους στο παιχνίδι ως επιπλέον εργαλεία του παίκτη για τη λύση ενός προβλήματος ή γρίφου και με αποτέλεσμα να φτάσει στον τερματισμό του παιχνιδιού.

Η συσκευή EEG παρέχει τιμές από τις συχνότητες που εκπέμπει ο εγκέφαλος όπου μετά από επεξεργασία αυτές μετρήσεις μετατρέπονται, σε attention, meditation αλλά και το Blink eye που αναφέρετε στους μύες του ματιού όταν ανοίγει και κλείνει. Πιο συγκεκριμένα στη παρούσα υλοποίηση χρησιμοποιήθηκαν οι παρακάτω μέτρησης. (Mindwave, n.d.)

<u>eSense(tm) Meters</u>

- **Attention eSense**

  Οι τιμές του Attention προσδιορίζουν την τρέχουσα eSense μέτρηση δηλαδή την προσοχή του χρήστη, γεγονός που δείχνει τα επίπεδα ‘Focus‘ ή ‘προσοχή’, όπως αυτή που συμβαίνει κατά τη διάρκεια έντονης συγκέντρωσης και πνευματικής δραστηριότητας. Η κλίμακα αυτής της μέτρησης κυμαίνεται από 0 έως 100.

- **Meditation eSense**

  Οι τιμές του Meditation προσδιορίζουν την τρέχουσα eSense μέτρηση δηλαδή τον ‘Διαλογισμό’ του χρήστη, το οποίο δείχνει το επίπεδο της ψυχικής ‘ηρεμίας’ ενός χρήστη ή τη ‘χαλάρωση’. Η κλίμακα αυτής της μέτρησης κυμαίνεται από 0 έως 100.

Λαμβάνοντας αυτές τις δυο βασικές χαρακτηρίστηκες τιμές, δοκιμάστηκαν διάφοροι τρόποι χρήσης τους.

Η πρώτη προσέγγιση χρήσης του EEG ήταν η δοκιμή του Πειράματος Χρωμάτων που αναλύεται στο υποκεφάλαιο 5.3 όπου σε αυτό το σημείο οι τιμές αξιοποιήθηκαν με επιστημονικό τρόπο χρήσης τους. Μετά από μερικές σημαντικές παρατηρήσεις το παιχνίδι αναπτύχθηκε με ένα συγκεκριμένο τρόπο τοποθέτησης των χρωμάτων που επηρεάζουν θετικά ή αρνητικά τις eSense τιμές του.

Η επόμενη προσέγγιση έχει να κάνει ως επιτοπλείστον με τις τιμές του attention και meditation που προέρχονται από τη συσκευή και τη χρήση τους ώστε να επηρεάσουν κάποιο αντικείμενο στο χώρο της σκηνής. Έχοντας αυτή τη δυνατότητα κατασκευάστηκαν εργαλεία με τα οποία ο χρήστης μπορεί να ανιχνεύσει πιο ευκολά κρυμμένα αντικείμενα που βρίσκονται στο χώρο της πίστας και είναι μείζονος σημασίας για τη λύση μιας υπόθεσης. Τα αντικείμενα αυτά είναι άμεσα συνδεδεμένα με το attention που έχει ο χρήστης την εκάστοτε στιγμή. Όσο πιο μεγάλες τιμές έχει τόσο πιο εμφανή είναι τα αντικείμενα εάν προφανώς βρίσκετε κοντά σε αυτά. Δε μπορεί παραδείγματος χάριν να τα εντοπίσει εάν βρίσκετε σε άλλο σημείο της πίστας.

Μια ακόμη λειτουργεία που έχει ο χρήστης μέσω αυτών των τιμών και συγκεκριμένα του meditation είναι να επιλεγεί τη σωστή απάντηση σε μια σειρά από ερωτήσεις και απαντήσεις. Πιο συγκεκριμένα ο χρήστης σε κάποιο συγκεκριμένο σημείο του παιχνιδιού θα έχει μπροστά του μια ερώτηση και θα καλείται να δώσει μια απάντηση μεταξύ δυο. Η σωστή απάντηση θα δίνετε αυτόματα όταν ο χρήστης βρίσκετε ένα συγκεκριμένο εύρος τιμών που αφορούν το meditation. Αν οι τιμές του είναι χαμηλές θα λαμβάνετε η λανθασμένη απάντηση. Όλα τα παραπάνω θα γίνονται με τη δύναμη του μυαλού, χωρίς τη χρήση του πληκτρολογίου ή του ποντικιού του υπολογιστή, αλλά μόνο με τα επίπεδα ψυχικής ηρεμίας.

Μια ακόμη χρήση της συσκευής είναι η δυνατότητα της εκκίνησης ενός χρονομέτρου (Timer) το οποίο θα τρέχει πιο γρήγορα ή πιο αργά ανάλογα με το meditation του χρήστη. Έτσι εάν τα επίπεδα είναι χαμηλά το χρονόμετρο θα τρέχει γρηγορότερα σε αντίθετη περίπτωση θα κυλάει πιο αργά. Τέλος, μια ακόμη λειτουργία με τη χρήση των μετρήσεων που αναφέρονται στο meditation είναι ο επηρεασμός των ιδιοτήτων αντικειμένων θα επηρεάζεται σε μερικά αντικείμενα το Opacity τους με αυτό τον τρόπο θα γίνονται ορατά και μη.

Στο κεφάλαιο 5 ‘Υλοποίηση’ περιγράφετε αναλυτικά, ο τρόπος χρήσης και ο κώδικας του ηλεκτροεγκεφαλογράφου που χρησιμοποιήθηκε για την επίτευξη της ενσωμάτωσης του.

4.1.2 Δημιουργία Σύνδεσης EEG Με Το Unity3d

Για τη δημιουργία της σύνδεσης του EEG με το περιβάλλον του Unity3d χρειάστηκε η εκτενής γνώση που αναφέραμε στο παραπάνω κεφάλαιο. Πιο απλά η συσκευή συλλεγεί τις συχνότητες που εκπέμπονται από το κρανίο μέσω του biosensor που είναι ενσωματωμένος πάνω στη συσκευή και στη συνέχεια στέλνει τις ακατέργαστες τιμές στον Bluetooth adaptor. Το επόμενο βήμα είναι να δημιουργηθεί η βιβλιοθήκη που θα πραγματοποιεί την επικοινωνία μεταξύ του game engine και του EEG. (Thinkgear Communications Protocol [NeuroSky Developer - Docs], n.d.) Η δημιουργία της βασίστηκε στο πρωτόκολλο και τρόπο λήψης που η συσκευή δέχεται τα πακέτα. Πιο συγκεκριμένα χρειάστηκε η δημιουργία τεσσάρων βασικών συναρτήσεων:

T_SPReceiver()

Η συνάρτηση αυτή απευθύνετε στη ροή δεδομένων που δεχόμαστε από τη συσκευή και τον έλεγχο ότι δεν έχει χαθεί κάποιο πακέτο στη μεταφορά.

```
45  public static void T_SPReceiver()
46  {
47
48      while (SPRStatus) {
49          try
50          {
51              while (SPRStatus)
52              {
53                  byte recByte = (byte)serialPort.ReadByte();
54
55                  if (recByte == Messages.recSync[0] && isSync1 == false) { isSync1 = true; continue; }
56                  if (recByte == Messages.recSync[0] && isSync2 == false) { isSync2 = true; continue; }
57                  if (isSync1 == true && isSync2 == true && ispLength == false)
58                  {
59                      if (recByte == 170) continue;
60                      if (recByte > 169) { isSync1 = false; isSync2 = false; continue; }
61                      pLength = recByte; currPoint = 0; checksum = 0; ispLength = true;
62                      continue;
63                  }
64                  if (isSync1 == true && isSync2 == true && ispLength == true && currPoint < pLength)
65                  {
66                      payload[currPoint++] = recByte; checksum += recByte; continue;
67                  }
68                  if (isSync1 == true && isSync2 == true && ispLength == true && currPoint >= pLength)
69                  {
70                      checksum &= 255; checksum = ~checksum & 255;
71                      if (recByte == checksum) ParsingPacket();
72
73                      isSync1 = false; isSync2 = false; ispLength = false; checksum = 0; currPoint = 0; continue;
74                  }
75              }
76          }
77          catch (Exception ex)
78          {
79          }
80      }
81  }
```

Πίνακας 2: Συνάρτηση T_SPReceiver()

ParsingPackets()

Η ParsingPackets απομονώνει και μετατρέπει το ωφέλιμο φορτίο του πακέτου και το εκχωρεί στην αντίστοιχη μεταβλητή που είναι τύπου String, αυτές είναι το σήμα του EEG, οι eSenses τιμές: meditation, attention, blink και συνεπώς τις τιμές των κυμάτων που είναι οι δέλτα, θήτα, χαμηλής-άλφα , υψηλής-άλφα, χαμηλής-βήτα, με υψηλό βήτα, χαμηλής-γάμμα και μέσα γάμμα.

```
129     private static void ParsingPacket()
130     {
131         int bytesParsed = 0;
132         while (bytesParsed < pLength)
133         {
134             byte code = payload[bytesParsed++];
135 
136             switch (code)
137             {
138             case 128:
139             {
140                 bytesParsed++;
141                 bytesParsed += 2;
142                 break;
143             }
144             case 2:
145             {
146                 signal = payload[bytesParsed++].ToString();
147                 break;
148             }
149             case 131:
150             {
151                 bytesParsed++;
152                 byte[] deltab={0,payload[bytesParsed++], payload[bytesParsed++],payload[bytesParsed++]};
153                 byte[] thetab = { 0, payload[bytesParsed++], payload[bytesParsed++], payload[bytesParsed++] };
154                 byte[] lalphab = { 0, payload[bytesParsed++], payload[bytesParsed++], payload[bytesParsed++] };
155                 byte[] halphab = { 0, payload[bytesParsed++], payload[bytesParsed++], payload[bytesParsed++] };
156                 byte[] lbetab = { 0, payload[bytesParsed++], payload[bytesParsed++], payload[bytesParsed++] };
157                 byte[] hbetab = { 0, payload[bytesParsed++], payload[bytesParsed++], payload[bytesParsed++] };
158                 byte[] lgammab = { 0, payload[bytesParsed++], payload[bytesParsed++], payload[bytesParsed++] };
159                 byte[] mgammab = { 0, payload[bytesParsed++], payload[bytesParsed++], payload[bytesParsed++] };
160                 delta   =   BitConverter.ToInt32(thetab,0).ToString() ;
161                 theta   =   BitConverter.ToInt32(deltab,0).ToString() ;
162                 lalpha  =   BitConverter.ToInt32(lalphab,0).ToString();
163                 halpha  =   BitConverter.ToInt32(halphab,0).ToString();
164                 lbeta   =   BitConverter.ToInt32(lbetab,0).ToString() ;
165                 hbeta   =   BitConverter.ToInt32(hbetab,0).ToString() ;
166                 lgamma  =   BitConverter.ToInt32(lgammab,0).ToString();
167                 mgamma  =   BitConverter.ToInt32(mgammab,0).ToString();
168                 break;
169             }


170             case 4:
171             {
172                 nattention =  payload[bytesParsed++].ToString();
173                 break;
174             }
175             case 5:
176             {
177                 nmeditation =  payload[bytesParsed++].ToString();
178                 break;
179             }
180             case 22:
181             {
182                 blink =  payload[bytesParsed++].ToString();
183                 break;
184             }
185             }
186         }
187 
188     }
189 }
```

Πίνακας 3: Συνάρτηση ParsingPackets()

Connect()

Η συνάρτηση Connect ανοίγει την αντίστοιχη Serial Port επικοινωνίας που επιλέγει ο χρήστης από το Option Menu του παιχνιδιού και στη συνέχεια ενεργοποιεί τη συνάρτηση T_SPReceiver() και ακολουθεί η ροή των δεδομένων.

Disconnect()

Η συνάρτηση Disconnect λειτουργεί αντίστροφα με τη Connect κλείνει δηλαδή τη θύρα επικοινωνίας με το EEG και συνεπώς τη ροή.

```
100     public static void Connect(SerialPortName commPort)
101     {
102         SPReceiver = new Thread(T_SPReceiver);
103 
104         serialPort = new SerialPort(commPort.ToString(), 115200, Parity.None, 8, StopBits.One);
105         if (serialPort.IsOpen == true) { Disconnect(); return; }
106             serialPort.Open();
107         serialPort.Write(Messages.reqAutoConnect, 0, Messages.reqAutoConnect.Length);
108         SPRStatus = true;
109         SPReceiver.Start ();
110     }
111     public static void Disconnect()
112     {
113         try{
114         SPReceiver.Abort ();
115         SPRStatus = false;
116         serialPort.Write(Messages.reqDisconnect, 0, Messages.reqDisconnect.Length);
117         serialPort.Close();
118     }
119     catch (Exception ex)
120     {
121     }
122 
123     }
```

Πίνακας 4: Συναρτήσεις Connect() / Disconnect()

4.2 Unity (Περιβάλλον Υλοποίησης)

Το Unity3d είναι πλατφόρμα cross-platform λέγοντας στην πληροφορική ανεξάρτητο πλατφόρμας (Αγγλικά: cross-platform ή multi-platform) αναφερόμαστε στο λογισμικό το οποίο τρέχει σε διαφορετικά λειτουργικά συστήματα ή πλατφόρμες υλικού και χρησιμοποιείται για την ανάπτυξη ηλεκτρονικών παιχνιδιών για PC, κονσόλες, φορητές συσκευές και ιστοσελίδες. (Unity - Game Engine, n.d.) Πρώτα ανακοινώθηκε μόνο για Mac OS, στο Παγκόσμιο Συνέδριο Apple's Worldwide Developers της Apple το 2005, από τότε έχει επεκταθεί σε περισσότερες από δεκαπέντε πλατφόρμες. Το Unity 3d παρέχει ένα καθαρό και πολύ εύχρηστο γραφικό περιβάλλον το οποίο ο χρήστης μπορεί να αναπροσαρμόσει όπως επιθυμεί.

Εικόνα 11: Λογότυπο προγράμματος Unity3D

4.2.1 Game Engines

Game Engine (μηχανή παιχνιδιού) στη πληροφορική λέγετε το σύστημα λογισμικού που είναι σχεδιασμένο για τη δημιουργία και την ανάπτυξη βιντεοπαιχνιδιών. Υπάρχουν πολλές μηχανές που είναι σχεδιασμένες για να λειτουργούν σε κονσόλες βιντεοπαιχνιδιών και λειτουργικά συστήματα επιτραπέζιων υπολογιστών όπως τα Microsoft Windows, το Linux, και το Mac OS X. Οι λειτουργίες που παρέχοντες από Game Engines περιλαμβάνουν ένα σύστημα φωτοαπόδοσης ("Renderer") για 2D ή 3D γραφικά, μηχανή φυσικής ή εντοπισμού συγκρούσεων collision detection, καθώς και collision response, ήχο, scripting, animation. τεχνητή νοημοσύνη, δικτύωση, streaming, διαχείριση μνήμης, νήματα (threading), και μια αναπαράσταση σκηνής (Scene Graph).

Οι μηχανές παιχνιδιών παρέχουν μια σουίτα οπτικών εργαλείων ανάπτυξης. Αυτά τα εργαλεία γενικά παρέχονται σε ένα ολοκληρωμένο περιβάλλον ανάπτυξης ώστε να καθιστούν ικανή την απλή, γρήγορη ανάπτυξη παιχνιδιών.

Οι μηχανές παιχνιδιών συχνά καλούνται "game middleware" επειδή, όπως με την εμπορική έννοια του όρου, παρέχουν μια ευέλικτη και επαναχρησιμοποιήσιμη πλατφόρμα λογισμικού η οποία παρέχει όλη την κεντρική λειτουργικότητα που απαιτείται, αμέσως έξω από το κουτί, για την ανάπτυξη μια εφαρμογής παιχνιδιού ενώ ταυτόχρονα μειώνει τα κόστη, τις πολυπλοκότητες, και το χρόνο μέχρι την αγορά (time-to-market) — όλοι κρίσιμοι παράγοντες στην υψηλά ανταγωνιστική βιομηχανία βιντεοπαιχνιδιών. Όπως άλλες λύσεις middleware, οι μηχανές παιχνιδιών συνήθως παρέχουν μια αφαίρεση της πλατφόρμας, επιτρέποντας στο ίδιο παιχνίδι να τρέχει σε διάφορες πλατφόρμες συμπεριλαμβανομένων των κονσόλων παιχνιδιών και των προσωπικών υπολογιστών με λίγες, αν όχι καθόλου, αλλαγές στον πηγαίο κώδικα του παιχνιδιού. Συχνά, το middleware παιχνιδιού σχεδιάζεται με μια βασισμένη σε στοιχεία αρχιτεκτονική η οποία επιτρέπει σε συγκεκριμένα συστήματα στη μηχανή να αντικατασταθούν ή να επεκταθούν με πιο εξειδικευμένα (και συχνά πιο ακριβά) στοιχεία middleware όπως το Havok για φυσική, το FMOD για ήχο, ή το Scaleform για UI και Βίντεο.

Μερικές μηχανές παιχνιδιών όπως η RenderWare είναι και ακόμη σχεδιασμένες ως μια σειρά χαλαρά συνδεδεμένων στοιχείων middleware τα οποία μπορούν να συνδυαστούν κατά βούληση για τη δημιουργία μια προσαρμοσμένης μηχανής, αντί για την πιο κοινή προσέγγιση της επέκτασης ή της προσαρμογής μια ευέλικτης ενσωματωμένης λύσης. Με οποιοδήποτε τρόπο και αν η επεκτασιμότητα επιτυγχάνεται, παραμένει μια υψηλή προτεραιότητα στις μηχανές παιχνιδιών λόγω της ευρείας ποικιλίας χρήσεων για την οποίες αυτές εφαρμόζονται. Παρά την συγκεκριμένη έννοια του ονόματος, οι μηχανές παιχνιδιών συχνά χρησιμοποιούνται για άλλα είδη διαδραστικών εφαρμογών με πραγματικού χρόνου γραφικές απαιτήσεις όπως επιδείξεις μάρκετινγκ, αρχιτεκτονικές οπτικοποιήσεις, εκπαιδευτικές εξομοιώσεις, και περιβάλλοντα μοντελοποίησης. Μερικές μηχανές παιχνιδιών παρέχουν μόνο ικανότητες φωτοαπόδοσης 3D πραγματικού χρόνου (real time 3D rendering) αντί της ευρείας γκάμας λειτουργικότητας που απαιτείται από τα παιχνίδια. Αυτές οι μηχανές βασίζονται στον δημιουργό του παιχνιδιού να εφαρμόσει το

υπόλοιπο αυτής της λειτουργικότητας ή να το συνθέσει από άλλα στοιχεία middleware παιχνιδιού.

Αυτού του τύπου οι μηχανές γενικά αναφέρονται ως «μηχανή γραφικών», «μηχανή rendering», ή «μηχανή 3D» αντί για τον πιο περιληπτικό όρο «μηχανή παιχνιδιού». Ωστόσο, αυτή η ορολογία χρησιμοποιείται ανακόλουθα καθώς πολλές πλήρεις σε χαρακτηριστικά μηχανές 3D παιχνιδιών αναφέρονται απλά ως «μηχανές 3D». Μερικά παραδείγματα μηχανών γραφικών είναι οι εξής: RealmForge, Truevision3D, OGRE, Crystal Space, Genesis3D, Irrlicht και JMonkey Engine. Οι μοντέρνες μηχανές παιχνιδιών ή γραφικών γενικά παρέχουν ένα scene graph, ο οποίος είναι μια αντικειμενοστραφής αναπαράσταση του 3D κόσμου του παιχνιδιού η οποία συχνά απλοποιεί τον σχεδιασμό του παιχνιδιού και μπορεί να χρησιμοποιηθεί για πιο αποδοτικό rendering τεράστιων εικονικών κόσμων. Πιο συχνά, οι 3D μηχανές ή τα συστήματα rendering στις μηχανές παιχνιδιών κατασκευάζονται πάνω σε ένα API γραφικών, όπως τα Direct3D ή OpenGL, το οποίο παρέχει μια αφαιρετικότητα λογισμικού της GPU ή της κάρτας βίντεο. Βιβλιοθήκες χαμηλού επιπέδου όπως τα DirectX, SDL και OpenAL επίσης χρησιμοποιούνται συχνά σε παιχνίδια καθώς παρέχουν πρόσβαση ανεξάρτητη από το υλικό σε άλλο υλικό του υπολογιστή όπως συσκευές εισόδου (ποντίκι (υπολογιστές), πληκτρολόγιο και joystick), κάρτες δικτύου και κάρτες ήχου. Πριν από τα επιταχυνόμενα από το υλικό 3D γραφικά, είχαν χρησιμοποιηθεί renderers λογισμικού. Το rendering λογισμικού ακόμα χρησιμοποιείται σε κάποια εργαλεία μοντελοποίησης ή για εικόνες ακίνητα rendered όταν η οπτική ακρίβεια αποτιμάται πάνω από την επίδοση πραγματικού χρόνου (καρέ ανά δευτερόλεπτο) ή όταν το υλικό του υπολογιστή δεν συναντά απαιτήσεις όπως υποστήριξη shader ή, στην περίπτωση των Windows Vista, υποστήριξη για το Direct3D 10.

Με την ανατολή της επιταχυνόμενης από το υλικό επεξεργασίας φυσικής, διάφορα API φυσικής όπως το PAL και οι επεκτάσεις φυσικής του COLLADA (μιας ανταλλακτικής μορφής για 3D στοιχεία) έγιναν διαθέσιμα για να παρέχουν μια αφαιρετικότητα λογισμικού της μονάδας επεξεργασίας φυσικής διαφορετικών παρόχων middleware και πλατφορμών κονσόλων. Η πρώτη γενιά μηχανών γραφικών τρίτων μερών ή renderers (και προγόνων αυτών που γνωρίζουμε ως μηχανές) κυριαρχούταν από τρεις παίκτες: την BRender από την Argonaut Software, την Renderware από την Criterion Software Limited και την Reality Lab της RenderMorphics.

Η Reality Lab ήταν η ταχύτερη από τις τρεις και ήταν η πρώτη που αναλήφθηκε σε μια επιθετική κίνηση της Microsoft. Η ομάδα της RenderMorphics, οι Servan Keondjian, Kate Seekings και Doug Rabson, ακολούθως προσχώρησαν στο εγχείρημα της Microsoft το οποίο μετέτρεψε την Reality Lab στο Direct3D, πριν οι Keondjian και Rabson φύγουν για να ξεκινήσουν μια άλλη εταιρεία middleware Qube Software. Η Renderware τελικά αγοράστηκε από την EA αλλά τέθηκε στην άκρη από τον γίγαντα των παιχνιδιών.

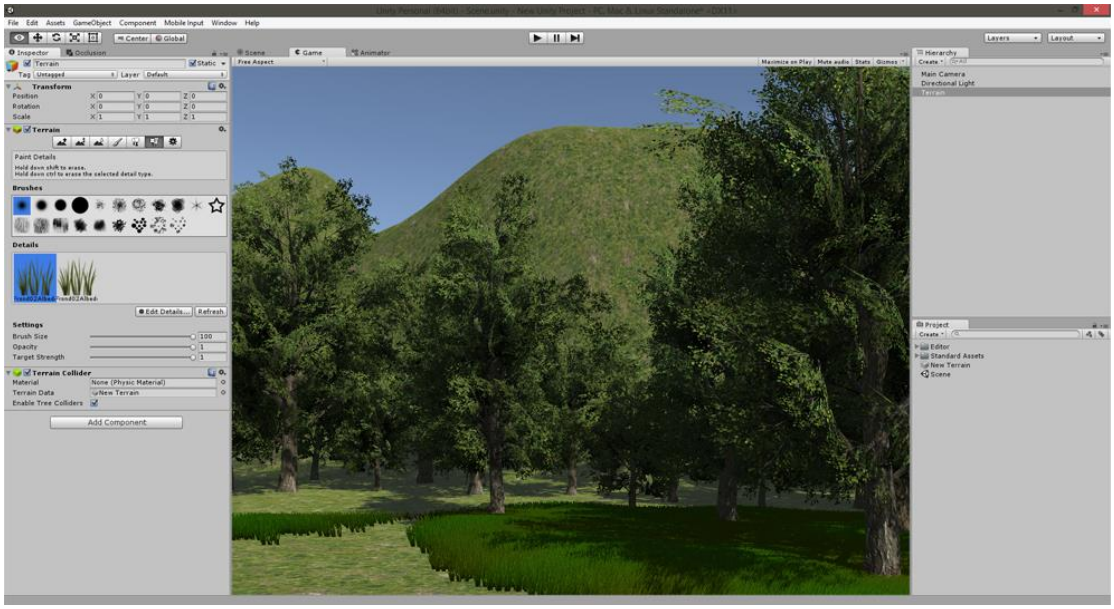
Ο όρος «μηχανή παιχνιδιού» ήρθε στην επιφάνεια στα μέσα της δεκαετίας του 1990, ειδικά σε σχέση με 3D παιχνίδια όπως τα παιχνίδια βολών πρώτου προσώπου (FPS). Τόση ήταν η δημοτικότητα των παιχνιδιών Doom και Quake που, αντί να δουλέψουν από μηδενική βάση, άλλοι δημιουργοί πήραν άδεια για τα κεντρικά κομμάτια του λογισμικού και σχεδίασαν τα δικά τους γραφικά, χαρακτήρες, όπλα και επίπεδα – το «περιεχόμενο του

παιχνιδιού» ή τα «στοιχεία του παιχνιδιού». Ο διαχωρισμός των ειδικών για το παιχνίδι κανόνων και δεδομένων από βασικές έννοιες όπως ο έλεγχος σύγκρουσης και οι οντότητες του παιχνιδιού σήμαινε ότι οι ομάδες μπορούσαν να μεγαλώσουν και να ειδικευτούν. (Game Engine, n.d.)

4.2.2 Παρουσίαση Χρήσης

Το Unity 3d παρέχει ένα καθαρό και πολύ εύχρηστο γραφικό περιβάλλον στο οποίο ο χρήστης μπορεί να αναπροσαρμόσει όπως επιθυμεί. Αποτελείται από πέντε παράθυρα. Το πρώτο, Scene/Game, στήνουμε τις σκηνές του παιχνιδιού, όταν πατάμε Play εμφανίζεται η προεπισκόπηση του παιχνιδιού. Το δεύτερο παράθυρο, Console, είναι αυτό στο οποίο εμφανίζονται όλα τα μηνύματα του συστήματος και τυχών errors, τα οποία μπορούμε εύκολα να διαχειριστούμε. Στο παράθυρο Hierarchy βλέπουμε όλα τα αντικείμενα που είναι τοποθετημένα στην σκηνή. Όλοι οι φάκελοι και τα αρχεία του παιχνιδιού απεικονίζονται στο παράθυρο Project. Τέλος στο παράθυρο Inspector βλέπουμε λεπτομέρειες για κάθε επιλεγμένο στοιχείο. Το Unity 3d παρέχει όλα τα εργαλεία που θα περίμενε κανείς από μια game engine. Παρέχει την δυνατότητα δημιουργίας primitives, κατασκευή terrain για τη δημιουργία πιστών, δυνατότητες φυσικής, εισαγωγή δυναμικού φωτισμού στη σκηνή και πολλά άλλα χρήσιμα εργαλεία. Επίσης παρέχεται ο editor Mono Develop για συγγραφή και διαχείριση των script. (Unity Manual, n.d.)

Ξεκινώντας ένα νέο project στο Unity πρέπει να έχουμε στο μυαλό μας την οργάνωση των φακέλων και των αρχείων. Μια καλή πρακτική είναι να δημιουργούνται φάκελοι με ένα περιγραφικό όνομα όπου τοποθετούνται όλα τα αρχεία για την εκάστοτε σκηνή του παιχνιδιού. Γενικά οτιδήποτε μπορεί να ταξινομηθεί σε μια ομάδα είναι καλό να τοποθετείται σε ένα ξεχωριστό φάκελο. Ξεκινώντας το project πρέπει να δημιουργηθεί ενα terrain στο οποίο επάνω στήνετε όλη η πίστα. Ο terrain editor του Unity 3d είναι ένα πολύ χρήσιμο και εύχρηστο εργαλείο το οποίο παρέχει πολλές δυνατότητες. Ο χρήστης μπορεί να δημιουργήσει υψώματα και λακκούβες, να του βάλει textures και να προσθέσει πολλές λεπτομέρειες όπως δέντρα, θάμνους και γρασίδι.



Εικόνα 12: Unity Environment

Κάνοντας κλικ στο terrain στο άνω toolbar και επιλέγοντας new terrain δημιουργούμε ένα καινούργιο terrain. Πριν ξεκινήσουμε όμως πρέπει να ορίσουμε κάποιες σημαντικές λεπτομέρειες όπως, ανάλυση του terrain, μέγεθος κτλ. Πηγαίνοντας ξανά στο toolbar βλέπουμε ότι εμφανίζονται όλες οι επιλογές που πριν ήταν γκριζαρισμένες. Set Resolution: Στο νέο παράθυρο που ανοίγει ρυθμίζουμε την ανάλυση του terrain και των λεπτομερειών του. Flatten Heightmap: Εδώ μπορούμε να δώσουμε μια τιμή έτσι ώστε να μπορούμε να σκάψουμε το terrain. Πλέον μπορούμε να ξεκινήσουμε να δίνουμε μορφή στο terrain. Έχοντας το επιλεγμένο στο hierarchy μπορούμε να δούμε στον inspector όλα τα εργαλεία που έχουμε στην διάθεσή μας. Μπορούμε να υψώσουμε το terrain (κρατώντας πατημένο το shift μπορούμε να το σκάψουμε), να το εξομαλύνουμε κτλ. Επίσης μπορούμε να επιλέξουμε τα textures που θα χρησιμοποιήσουμε και με τα οποία θα βάψουμε το terrain

4.3 Motion Builder

Το MotionBuilder της Autodesk είναι ένα επαγγελματικό πρόγραμμα για 3d animation. Χρησιμοποιείται για παραγωγή ψηφιακού περιεχομένου, επαγγελματικό motion capture αλλά και για παραδοσιακό key frame animation. Χρησιμοποιείται στην παραγωγή παιχνιδιών, κινηματογραφικών ταινιών, τηλεοπτικών εκπομπών. Κάποιοι mainstream τίτλοι που έχουν επωφεληθεί από αυτό είναι τα ηλεκτρονικά παιχνίδια Assassins creed, Killzone 2 και στην ταινία Avatar. Επίσης αξίζει να αναφέρουμε ότι το πολυχρησιμοποιημένο format fbx για ανταλλαγή δεδομένων τρισδιάστατου περιεχομένου ξεκίνησε και πρωτοχρησιμοποιήθηκε από το MotionBuilder.

Σε αυτό το υποκεφάλαιο θα δούμε πως γίνεται σε ένα rigged τρισδιάστατο μοντέλο να εφαρμόσουμε ένα αρχείο κίνησης τύπου .bvh. Για να εφαρμόσουμε κάποιο animation στο μοντέλο που θέλουμε πρέπει να ακολουθήσουμε την προβλεπόμενη διαδικασία που ορίζει το MotionBuilder. Σε αντίθεση με ότι έχουμε δει μέχρι τώρα το MotionBuilder έχει μια πολύ συγκεκριμένη διαδικασία που πρέπει να ακολουθηθεί. Το πρώτο πράμα που πρέπει να κάνουμε είναι να φορτώσουμε το .bvh αρχείο που θέλουμε να εφαρμόσουμε στον χαρακτήρα μας στο πρόγραμμα. Με το που το κάνουμε αυτό θα δούμε τον σκελετό του animation να εμφανίζεται στο interface. Εάν πατήσουμε το κουμπί play στην μπάρα ελέγχου θα δούμε το animation εν κινήσει.

Το αρχείο που μόλις εισαγάγαμε, δεν μπορούμε να το εφαρμόσουμε απευθείας πάνω στον σκελετό του rigged μοντέλου μας. Πρέπει πρώτα να δημιουργήσουμε ένα characterized σκελετό, πάνω στον οποίο θα περάσουμε όλα τα δεδομένα του animation και στην συνέχεια θα συνδέσουμε αυτόν τον σκελετό με το μοντέλο μας ώστε να δεχτεί το animation. Άρα το πρώτο πράγμα που θα κάνουμε είναι να πούμε στο MotionBuilder πως θέλουμε να προβούμε στην δημιουργία του ειδικού σκελετού. Επιλέγοντας από το μενού κάτω δεξιά την επιλογή characters, θα εμφανιστεί ένα νέο μενού.

Στο δεξί μέλος έχουμε όλα τα μέλη του σκελετού που δεν έχει χαρακτηριστεί και επομένως, τα μέλη αυτά, δεν έχουν όνομα. Στο αριστερό μέρος κάνοντας collapse το bvh αρχείο εμφανίζονται όλα τα μέλη (χαρακτηρισμένα) του animation. Αυτό που θα πρέπει να κάνουμε για να χαρακτηριστεί ο σκελετός (control rig), είναι με drag and drop να κάνουμε τις σωστές αντιστοιχίες όπως φαίνεται στην εικόνα 21. Εδώ το βασικό που πρέπει να έχουμε στο νου μας είναι ότι πρέπει να αντιστοιχιστεί οπωσδήποτε το base. Με το που τελειώσει η αντιστοίχιση τσεκάρουμε το κουτάκι που γράφει characterize και στο νέο pop up window ενημερώνουμε το MotionBuilder ότι το control rig θα εφαρμοστεί σε biped.

Το τελικό βήμα είναι να πατήσουμε create και η διαδικασία αυτή θα λάβει τέλος. Στην οθόνη τώρα βλέπουμε το characterized control rig. Αν πατήσουμε play βέβαια θα παρατηρήσουμε πως δεν κινείται. Αυτό συμβαίνει διότι το animation πρέπει να γίνει bake στον νέο μας σκελετό. Από το menu character controls πάνω δεξιά θα πρέπει να επιλέξουμε none στο δεύτερο drop down menu. Στην συνέχεια κάνοντας κλικ στο μπλε κουμπί επιλέγουμε Bake to control rig. Με το που γίνει bake αν πατήσουμε play θα δούμε πωw το control rig έλαβε επιτυχώς τα δεδομένα του .bvh αρχείου.

4.4 Βιβλιοθήκες

Στην πληροφορική καλούμε βιβλιοθήκη (Library) μια συλλογή από έτοιμα υποπρογράμματα που χρησιμοποιούνται για την ανάπτυξη λογισμικού. Οι βιβλιοθήκες περιέχουν υποβοηθητικό κώδικα και δεδομένα, παρέχοντας, με αυτόν τον τρόπο, υπηρεσίες σε προγράμματα. Αυτό επιτρέπει τον διαμοιρασμό και τη χρήση του κώδικα και

των δεδομένων με αρθρωτό τρόπο. Η έννοια της βιβλιοθήκης είναι αναπόσπαστο τμήμα του δομημένου προγραμματισμού και αναπτύχθηκε παράλληλα με αυτόν.

Οι βιβλιοθήκες μπορούν να είναι *στατικές* ή *δυναμικές*, καθώς και *κοινόχρηστες* ή μη. Όταν ένα πρόγραμμα καλεί υποπρογράμματα από στατικές βιβλιοθήκες, ο linker (ο οποίος συμβουλεύεται κατάλληλα αρχεία ρυθμίσεων ή το ΛΣ για να βρει τις βιβλιοθήκες) ενσωματώνει πραγματικά τον αντικειμενικό τους κώδικα στο εκτελέσιμο που παράγει και επανατοποθετεί κατάλληλα τις διευθύνσεις μνήμης σε χρόνο σύνδεσης. Αντίθετα, οι δυναμικές βιβλιοθήκες ενσωματώνονται και επανατοποθετούνται στον τελικό κώδικα απευθείας στη μνήμη και ενώ το πρόγραμμα εκτελείται, με αποτέλεσμα τα εκτελέσιμα αρχεία να έχουν σαφώς μικρότερο μέγεθος (αφού περιέχουν απλώς οδηγίες προς έναν linker χρόνου εκτέλεσης, παρεχόμενου από το ΛΣ, αντί για τον ίδιο τον αντικειμενικό κώδικα των καλούμενων υποπρογραμμάτων βιβλιοθήκης), αλλά να απαιτείται η μόνιμη παρουσία του αντικειμενικού αρχείου της βιβλιοθήκης στο σύστημα αρχείων προκειμένου να είναι δυνατή η εκτέλεση του προγράμματος.

4.4.1 iTween

Εικόνα 13:Λογότυπο βιβλιοθήκης iTween

Σε αυτό το κεφάλαιο θα εξηγήσουμε τον τρόπο χρήσης της βιβλιοθήκης iTween. Η βιβλιοθήκη αυτή είναι ένα αρχείο γραμμένο σε C#. Μπορεί να χρησιμοποιηθεί με οποιαδήποτε από τις γλώσσες προγραμματισμού που υποστηρίζει το Unity, καθώς και όλες τις εκδόσεις του Unity. Αν ο προγραμματισμός γίνετε σε JavaScript ή Boo θα χρειαστεί να δημιουργήσετε ένα φάκελο "Plugins" (αν δεν έχετε ήδη) στη λίστα των φακέλων ‘Project’, αν ο προγραμματισμός γίνετε σε C#, μπορεί να το τοποθετηθεί οπουδήποτε θέλετε μέσα στη λίστα των φακέλων ‘Project’. (Berkebile, n.d.)

Η εγκατάσταση του είναι μια πολύ απλή διαδικασία, χρειάζεται μόνο να κατεβάσουμε το αρχείο από την official σελίδα του iTween ή να το κατεβάσουμε από το Asset Store του Unity. Αφού γίνει αυτό μετά κάνουμε Import μέσα στο πρόγραμμα και το χρησιμοποιούμε.

Εφαρμογή και χρήση του iTween

Το iTween μας προσφέρει τη δυνατότητα να μετακινήσουμε, να περιστρέψουμε, να μεγεθύνουμε αντικείμενα επίσης να χρησιμοποιηθεί για fade in/out σε κάμερες αλλά και να κάνουμε, control audio. Όπως είπαμε παραπάνω οι βιβλιοθήκες διακρίνονται σε Στατικές και Δυναμικές η iTween είναι μια Στατική Κλάση η οποία επιτρέπει να εκτελέσουμε τις μεθόδους ή να καλέσουμε τις ιδιότητές του, αλλά δεν πρέπει να γράψουμε μέσα στη κλάση. Στη σελίδα του iTween υπάρχει ένα τεράστιο Documentation με πάνω από 80 functions που μπορούμε να χρησιμοποιήσουμε μερικές από αυτές είναι: DrawPath, FadeTo, MoveTo, RotateBy, PutOnPath, ScaleTo. Στο κεφάλαιο Υλοποίηση θα δείξουμε πως γίνεται η χρήση και η εφαρμογή της μέσα στο Project.

4.4.2 Thinkgear

Εικόνα 14: Λογότυπο ThinkGear

Η βιβλιοθήκη ThinkGear χρησιμοποιήθηκε για τη σύνδεση και τη λήψη των δεδομένων από τον ηλεκτροεγκεφαλογράφο στο Unity3d. Πιο συγκεκριμένα το ThinkGear Socket Protocol (TGSP) βασίζετε στη λειτουργία του JSON Protocol (JavaScript Object Notation) για τη μετάδοση και λήψη εγκεφαλικών δεδομένων μεταξύ ενός χρήστη και ενός εξυπηρετητή. (Thinkgear Communications Protocol [NeuroSky Developer - Docs], n.d.) Ο διακομιστής είναι η συσκευή ή εφαρμογή που υλοποιεί TGSP, και είναι υπεύθυνη, μεταξύ άλλων, για να ανταποκρίνεται στα αιτήματα αδειοδότησης και να μεταδίδει δεδομένα στο Headset. Ειδικότερα περιγράφει:

- Την **ανάλυση** των σειριακών δεδομένων για την κατασκευή των διάφορων εγκεφαλικών τύπων που αποστέλλονται από το ThinkGear.

- Την **ερμηνεία** και τη **χρήση** των κυμάτων που αποστέλλονται από το ThinkGear (Συμπεριλαμβανομένων Attention, Meditation και Signal Quality Data) σε μια BCI Brain-Computer Interface εφαρμογή.

- Την **αποστολή acknowledgment** στη συσκευή για απευθείας προσαρμογή συμπεριφοράς του module behavior και του output.

Επιμέρους στοιχεία του ThinkGear

- *ThinkGear Data Values:* Ορίζει τους τύπους των Data Values τις τιμές των δεδομένων που προέρχονται από το ThinkGear.

- *ThinkGear Packets:* Περιγράφονται τα πακέτα που στέλνονται από τα ThinkGear Data Values.

- *ThinkGear Command Bytes:* Το κομμάτι αυτό αναφέρεται στους προχωρημένους χρήστες το οποίο επεξηγεί το πώς στέλνονται τα Raw Data στη συσκευή προκειμένου να κάνει επιμέρους αλλαγές όπως για παράδειγμα να αλλάξει το baud rate να ενεργοποιήσει ή να απενεργοποιήσει συγκεκριμένες τιμές που προέρχονται από τη συσκευή.

ThinkGear Data Values

Ορίζει τους τύπους των Data Values τις τιμές των δεδομένων που προέρχονται από το ThinkGear.

Poor_Signal Quality

Το Poor_Signal Quality περιγράφεται από ένα ακέραιο αριθμό και κυμαίνεται από 0 έως 255. Αυτές οι τιμές εξόδου λαμβάνονται κάθε δευτερόλεπτο, και δείχνουν πάντα τις πιο πρόσφατες μετρήσεις. Οι μη μηδενικές τιμές υποδεικνύουν ότι έχουμε κάποιο είδος θορύβου. Όσο υψηλότερες είναι οι τιμές, τόσο περισσότερος θόρυβος ανιχνεύεται. Η τιμή 200 έχει ιδιαίτερη σημασία, γιατί σημαίνει ότι τα ηλεκτρόδια δεν είναι σε επαφή με το δέρμα του χρήστη. Κάποιοι επίσης λόγοι που μπορούν να προκαλέσουν μερική απώλεια σήματος είναι η κακή επαφή των ηλεκτροδίων με το κεφάλι του χρήστη, η υπερβολική κίνηση του κεφαλιού ή του σώματος του χρήστη , ο ηλεκτροστατικός θόρυβος του περιβάλλοντος και ο non-EEG biometric noise για παράδειγμα EMG, EKG/ECG, EOG, και λοιπά.

Μια ορισμένη ποσότητα του θορύβου είναι αναπόφευκτη κατά τη χρήση του ThinkGear όμως έχουν δημιουργηθεί αλγόριθμοι φιλτραρίσματος για τον εντοπισμό, τη διόρθωση και την αντιστάθμιση του θορύβου. Η τιμή της ποιότητας POOR_SIGNAL είναι χρήσιμη σε ορισμένες εφαρμογές που πρέπει να είναι πιο ευαίσθητες στο θόρυβο (όπως ορισμένα ιατρικά ή ερευνητικές εφαρμογές.

eSense(tm) Meters

Οι τιμές για τους τύπους των eSenses τιμών δηλαδή Attention και Meditation κυμαίνονται στη κλίμακα από 1 έως 100. Οι τιμές μεταξύ 40 έως 60 σε κάποια δεδομένη χρονική στιγμή θεωρούνται "ουδέτερες" , αποτελούν την έννοια της βάσης ‘γραμμές βάσης’. Οι τιμές από 60 έως 80 θεωρούνται ‘ελαφρώς αυξημένες’ και τέλος οι τιμές 80-100 θεωρούνται ‘υψηλές’ που σημαίνει ότι ο χρήστης έχει αυξημένης επίπεδα Attention και Meditation αντίστοιχα. Παρομοίως, στο άλλο άκρο της κλίμακας, οι τιμές μεταξύ 20 και 40 υποδεικνύει ‘μειωμένα’ επίπεδα, ενώ οι τιμές μεταξύ 1 έως 20 δείχνει ‘έντονη μείωση’. Τα επίπεδα αυτά μπορεί να δείχνουν καταστάσεις απόσπασης προσοχής, διέγερση ή ανωμαλία. Τέλος, αν η τιμή που θα λάβουμε είναι 0 σημαίνει ότι δείχνει το ThinkGear δεν λαμβάνει κανένα σήμα. Αυτό μπορεί να συμβαίνει (και είναι συνήθως) λόγω υπερβολικού θορύβου, όπως περιγράφεται στην ενότητα POOR_SIGNAL Quality που αναφέραμε παραπάνω.

Ο λόγος του ευρύ φάσματος για κάθε ερμηνεία γίνετε γιατί ορισμένα μέρη του αλγορίθμου eSense πραγματοποιούνται δυναμικά και αν κάποιες χρονικές περιόδους πραγματοποιούνται ‘slow-adaptive’ αλγόριθμοι για να προσαρμόσουν τις φυσικές διακυμάνσεις και τάσεις του κάθε χρήστη. Αυτός είναι ένας λόγος όπου οι ThinkGear αισθητήρες είναι σε θέση να λειτουργούν σε ευρύ φάσμα ατόμων κάτω από ένα εξαιρετικά ευρύ φάσμα των προσωπικών και περιβαλλοντικών συνθηκών, όπως επίσης δίνει ακρίβεια και αξιοπιστία.

Attention eSense

Οι τιμές του Attention προσδιορίζουν την τρέχουσα eSense μέτρηση δηλαδή την προσοχή του χρήστη, γεγονός που δείχνει τα επίπεδα ‘Focus‘ ή ‘προσοχή’, όπως αυτή που συμβαίνει κατά τη διάρκεια έντονης συγκέντρωσης και πνευματικής δραστηριότητας. Η κλίμακα αυτής της μέτρησης κυμαίνεται από 0 έως 100. Η απόσπαση προσοχής, η έλλειψη εστίασης ή το άγχος μπορούν να μειώσουν τα επίπεδα της μέτρησης. Όπως έχουμε ξανά πει αυτές οι τιμές λαμβάνονται ανά ένα δευτερόλεπτο από τη συσκευή.

Meditation eSense

Οι τιμές του Meditation προσδιορίζουν την τρέχουσα eSense μέτρηση δηλαδή τον ‘Διαλογισμό’ του χρήστη, το οποίο δείχνει το επίπεδο της ψυχικής ‘ηρεμίας’ ενός χρήστη ή τη ‘χαλάρωση’. Η κλίμακα αυτής της μέτρησης κυμαίνεται από 0 έως 100. Πρέπει να επισημάνουμε σε αυτό το σημείο ότι ο διαλογισμός είναι ένα μέτρο των mental levels ενός ατόμου και όχι των σωματικών επιπέδων, έτσι λοιπόν και να χαλαρώσετε όλους τους μυς του σώματος δεν οδηγεί άμεσα σε ένα αυξημένα επίπεδα διαλογισμού. Ωστόσο, για τους περισσότερους ανθρώπους με φυσιολογικές συνθήκες, χαλαρώνοντας το σώμα συχνά βοηθάμε και το μυαλό να χαλαρώσει. Ο διαλογισμός σχετίζεται επίσης και με τις μειωμένες σωματικές δραστηριότητες, μια παρατηρούμενη επίδραση είναι αυτή που όταν κλείνουμε τα μάτια είναι συχνά μια αποτελεσματική μέθοδος για την αύξηση των επιπέδων του

Διαλογισμού. Η απόσπαση προσοχής, η έλλειψη εστίασης ή το άγχος μπορούν να μειώσουν τα επίπεδα της μέτρησης.

8/16BIT_RAW Wave Value

Οι τιμές του 8Bit_Raw προσδιορίζουν τις ακατέργαστες τιμές που προέρχονται από τη συσκευή Ανάλογα τη συσκευή (Headset) που υποστηρίζει το ThinkGear λαμβάνουμε αντίστοιχα τις τιμές αυτές σε 8-bit ή σε16-bit. Αυτό καθιστά δυνατή την έξοδο των τιμών 'ακατέργαστων κυμάτων' RAW Waves έχοντας όμως περιορισμούς στο Bandwidth δίνοντας 9600 Baud Rate. Σε εφαρμογές όπως είναι η απεικόνιση σε πραγματικό χρόνο γραφημάτων των RAW Waves έχοντας 8-bit η ακρίβεια είναι επαρκής, δεδομένου ότι το ανθρώπινο μάτι τυπικά δεν μπορεί να διακρίνει γρήγορα τα pixel που αντιστοιχούν στα κατώτερα bit ακρίβειας. Εάν απαιτείται μεγαλύτερη ακρίβεια, μπορείτε να χρησιμοποιήσετε τη μορφοποίηση σε16-bit που όμως απαιτείτε υψηλότερος ρυθμός Bandwidth που εκτείνετε στις τιμές -32768 έως 32767.

EEG_POWER

Αυτές οι τιμές δεδομένων αντιπροσωπεύουν το τρέχον μέγεθος των 8 κοινώς αναγνωρισμένων ζωνών συχνοτήτων (εγκεφαλικά κύματα). Αποτελείται από οκτώ 4-byte floats με την εξής σειρά: δέλτα (0,5 - 2.75Hz), θήτα (3,5 - 6.75Hz), χαμηλής-άλφα (7,5 - 9.25Hz), υψηλής-άλφα (10 - 11.75Hz), χαμηλής-βήτα (13 - 16.75Hz), με υψηλό βήτα (18 - 29.75Hz), χαμηλής-γάμμα (31 - 39.75Hz) και μέσα γάμμα (41 - 49.75Hz). Αυτές οι τιμές δεν έχουν μονάδες και ως εκ τούτου έχουν νόημα μόνο όταν συγκρίνονται μεταξύ τους και με τον εαυτό τους, για την εξέταση της σχετικής ποσότητας και για περιοδικές διακυμάνσεις.

Η διαμόρφωση των τιμών είναι σε big-endian IEEE 754 έτσι τα 32 bytes των τιμών μπορούν, συνεπώς, να διανεμηθεί απευθείας ως float * σε C που θα χρησιμοποιείται ως πίνακας δεικτών σε float.

Eye Blink Strength

Σε αυτή την ενότητα πρέπει να σημειωθεί ότι οι τιμές του Blink είναι διαθέσιμο προς το παρόν μέσω των ThinkGear APIs. Όμως δεν είναι άμεσα διαθέσιμο ως έξοδος από οποιαδήποτε τρέχον υλικό ThinkGear. Για το TGCD, υπάρχει ο τύπος δεδομένων TG_DATA_BLINK_STRENGTH για χρήση με το TG_GetValueStatus () και TG_GetValue () functions.

Mind-wandering Level

Οι τιμές του Mind-wandering περιγράφουν την ένταση αυτού του σήματος που κυμαίνεται από 1 έως 10.

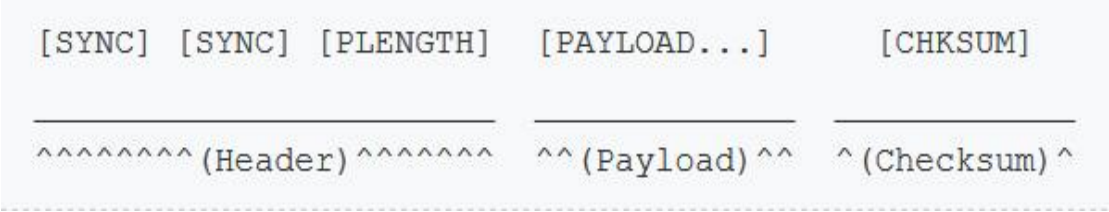
ThinkGear Packets

Τα πακέτα του ThinkGear μεταφέρουν τα ψηφιακά δεδομένα ως μια ασύγχρονη σειριακή ροή από bytes. Η σειριακή ροή που λαμβάνετε από το bio-sensor μετατρέπεται και αναλύεται σε ThinkGear Packets, προκειμένου να ερμηνευθεί μετά σε ThinkGear Values που περιεγράφηκαν παραπάνω.

Ένα τέτοιου είδους πακέτο αποτελείται από τρία μέρη, Το Header, Payload και το Payload Checksum. Η μορφή του έχει σχεδιαστεί για να είναι ισχυρή και ευέλικτη: Το Header και το Checksum παρέχουν συγχρονισμό στη ροή δεδομένων και την ακεραιότητα των δεδομένων κατά τον έλεγχο, Το Payload εξασφαλίζει στα νέα πεδία δεδομένων να μπορούν να προστεθούν στο μέλλον χωρίς να υπάρχουν απώλειες.

Δομή Πακέτου

Τα πακέτα στέλνονται ως μια ασύγχρονη σειριακή ροή από bytes. Η μεταφορά τους μπορεί να είναι UART, Serial COM, USB, Bluetooth, αρχείο. Κάθε πακέτο αρχίζει με την επικεφαλίδα του, που ακολουθείται από το ωφέλιμο φορτίο και τελειώνει με το ωφέλιμο φορτίο αθροίσματος, ως εξής:



Εικόνα 15: Packet Structure

Παράδειγμα λήψης ενός ThinkGear πακέτου

Στη παρακάτω εικόνα φαίνεται η μορφή ενός πακέτου. Εκτός από τις τιμές των bytes [SYNC], [PLENGTH], και [CHKSUM], όλα τα άλλα bytes από το [3] έως το [10] αποτελούν μέρος των δεδομένων του ωφέλιμου φορτίου του πακέτου.

```
byte: value // Explanation

[ 0]: 0xAA  // [SYNC]
[ 1]: 0xAA  // [SYNC]
[ 2]: 0x08  // [PLENGTH] (payload length) of 8 bytes
[ 3]: 0x02  // [CODE] POOR_SIGNAL Quality
[ 4]: 0x20  // Some poor signal detected (32/255)
[ 5]: 0x01  // [CODE] BATTERY Level
[ 6]: 0x7E  // Almost full 3V of battery (126/127)
[ 7]: 0x04  // [CODE] ATTENTION eSense
[ 8]: 0x12  // eSense Attention level of 18%
[ 9]: 0x05  // [CODE] MEDITATION eSense
[10]: 0x60  // eSense Meditation level of 96%
[11]: 0xE3  // [CHKSUM] (1's comp inverse of 8-bit
Payload sum of 0x1C)
```

Εικόνα 16: Παράδειγμα λήψης ενός ThinkGear Packet

5: Υλοποίηση

Στο κεφάλαιο αυτό θα δείξουμε πως έγινε η υλοποίηση και πως έγινε ο κατάλληλος συνδυασμός για το τελικό αποτέλεσμα. Θα μπορούσαμε να πούμε ότι το κεφάλαιο αυτό διακρίνετε σε δύο επιμέρους μέρη. Την υλοποίηση του παιχνιδιού στο περιβάλλον του Unity3D και το δεύτερο μέρος είναι η προσαρμογή του EEG και οι εφαρμογή του μέσα στο παιχνίδι.

5.1 Δημιουργία Γραφικού Περιβάλλοντος

Στο σημείο αυτό θα δείξουμε τη διαδικασία με την οποία υλοποιήθηκε το γραφικό περιβάλλον του παιχνιδιού. Το πρώτο πράγμα που χρειάστηκε είναι να γίνει ερευνά μοντέλων που αρμόζουν στο είδος και το ύφος του παιχνιδιού. Συνεπώς το Unity3D παρέχει στο Asset Store (Window > Asset Store) κάποια μοντέλα δωρεάν και κάποια αλλά επι-πληρωμή. Στη περίπτωση του παρόντος παιχνιδιού έγινε η αγορά ενός τέτοιου πακέτου από κτήρια (Retro City Pack) στο διαδίκτυο υπάρχουν αρκετές ιστοσελίδες που παρέχουν δωρεάν 3D μοντέλα αλλά και με πληρωμή, μερικές από αυτές τις ιστοσελίδες είναι το TF3DM, Blend Swap, Turbo Squid, Archive 3D. Έτσι λοιπόν βρίσκουμε τα μοντέλα που μας ταιριάζουν και στο επόμενο στάδιο ανοίγουμε το Unity3D και κάνουμε δοκιμές μέχρι να φτάσουμε σε ένα επιθυμητό σημείο.

Πριν ξεκινήσουμε να αναλύουμε τις τεχνικές που χρησιμοποιήθηκαν για τη δημιουργία του παιχνιδιού πρέπει να εξηγήσουμε κάποια πράγματα εκ των προτέρων για τη χρήση των μοντέλων και των αντικειμένων. Σε αυτή τη πλατφόρμα εργαζόμαστε με 3D μοντέλα. Πρέπει να υπενθυμίσουμε ότι η επιφάνεια εργασίας του Unity3D αποτελείται από τα εναλλασσόμενα παράθυρα Scene και Game όπου στο παράθυρο Scene στήνουμε τις σκηνές του παιχνιδιού και το παράθυρο Game που κάνουμε προεπισκόπηση του παιχνιδιού. Στο παράθυρο Hierarchy βλέπουμε όλα τα αντικείμενα (GameObjects) που είναι τοποθετημένα στην σκηνή. Στο παράθυρο Project βρίσκονται όλοι οι φάκελοι και τα αρχεία του παιχνιδιού. Στο παράθυρο Inspector παρουσιάζονται οι ιδιότητες για κάθε επιλεγμένο αντικείμενο και τέλος το παράθυρο Console, είναι αυτό στο οποίο εμφανίζονται όλα τα μηνύματα του συστήματος και τυχών errors, τα οποία μπορούμε εύκολα να διαχειριστούμε.

Ανοίγοντας το Unity3D πρέπει να δημιουργήσουμε ένα νέο Project και μια σκηνή. Από την έναρξη ενός νέου Project δημιουργούνται αυτόματος στο περιβάλλον εργασίας μια κάμερα και ένα Directional Light. Αμέσως μετά χρειάζεται να κάνουμε Import μερικά πακέτα που απαιτούνται.
Στο Menu του προγράμματος στη κατηγορία Assets επιλέγουμε την επιλογή Import Package και επιλέγουμε να κάνουμε Import τα εξής πακέτα Characters και Environment.

Όταν τα κάνουμε Import στην περιοχή Project δημιουργούνται φάκελοι που περιέχουν Assets. Ο φάκελος Characters περιέχει Prefabs που έχουν ενσωματωμένα Scripts που αφορούν controls του παίκτη λέγοντας αυτό εννοούμε την κίνηση και τον έλεγχο που έχουμε πάνω του με το πληκτρολόγιο. Επίσης κάτι που είναι σημαντικό και χρησιμοποιείτε συχνά στην υλοποίηση είναι τα Prefabs. Τα Prefabs είναι αντικείμενα που έχουμε δημιουργήσει, μορφοποιήσει και ορίσει κάποιες ιδιότητες αυτά τα αντικείμενα μπορούμε να τα χρησιμοποιούμε παντού στο project σε οποιαδήποτε σκηνές όπως επίσης να έχουμε πολλά στιγμιότυπα τους. Ο φάκελος Environment περιέχει αντικείμενα για το περιβάλλον περιέχει δηλαδή assets για δέντρα, έδαφος, άνεμο, νερό και αλλά.

Συνεπώς, αφού εξοικειωθούμε με το πρόγραμμα το αμέσως επόμενο βήμα είναι να δημιουργήσουμε έδαφος, για να κάνουμε κάτι τέτοιο πρέπει από το Menu να επιλέξουμε GameObject > 3D Object > Terrain. Έχοντας επιλεγμένο το Terrain από το Hierarchy στο παράθυρο Inspector βλέπουμε τις ιδιότητες του. Στη παρακάτω εικόνα είναι αριθμημένα μερικά σημεία που πρέπει να γνωρίζουμε σχεδόν για όλα τα GameObjects που χρησιμοποιούμε. Στην εικόνα έχουμε τις ειδικότητες του Terrain.

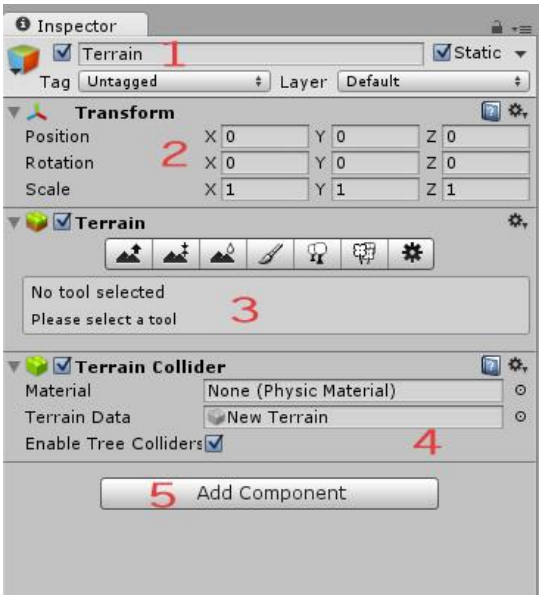
1. Σε αυτό πεδίο αναφέρετε το όνομα του αντικειμένου οπού μπορούμε να το αλλάξουμε αν θέλουμε.

2. Στην ενότητα Transform βλέπουμε τη θέση (Position) του αντικειμένου στη περιοχή που δουλεύουμε, την περιστροφή (Rotation) του στο χώρο και τη διάσταση (Scale) του.

3. Στην ενότητα Terrain μας υπάρχουν μερικά Tools που δίνουν τη δυνατότητα να μορφοποιήσουμε τη περιοχή, δημιουργώντας υψώματα ή λακκούβες, βάζοντας Textures για την επιφάνια για παράδειγμα χώμα ή γρασίδι, δέντρα και τέλος έχουμε τα options οπού μπορούμε να κάνουμε επιπλέον αλλαγές στην επιφάνια.

4. Το Terrain Collider δίνει τη ιδιότητα στο αντικείμενο να έχει αντίσταση σε αλλά αντικείμενα που τοποθετούνται πάνω του όπως φαίνεται στην εικόνα είναι ενεργοποιημένος και για τα δέντρα. Αυτό σημαίνει ότι ο χαρακτήρας δε θα περάσει μέσα από το δέντρο αλλά θα βρει αντίσταση.

5. Τέλος, το κουμπί Add Component δίνει επιπλέον ιδιότητες για να εφαρμόσουμε στο εκάστοτε GameObject.

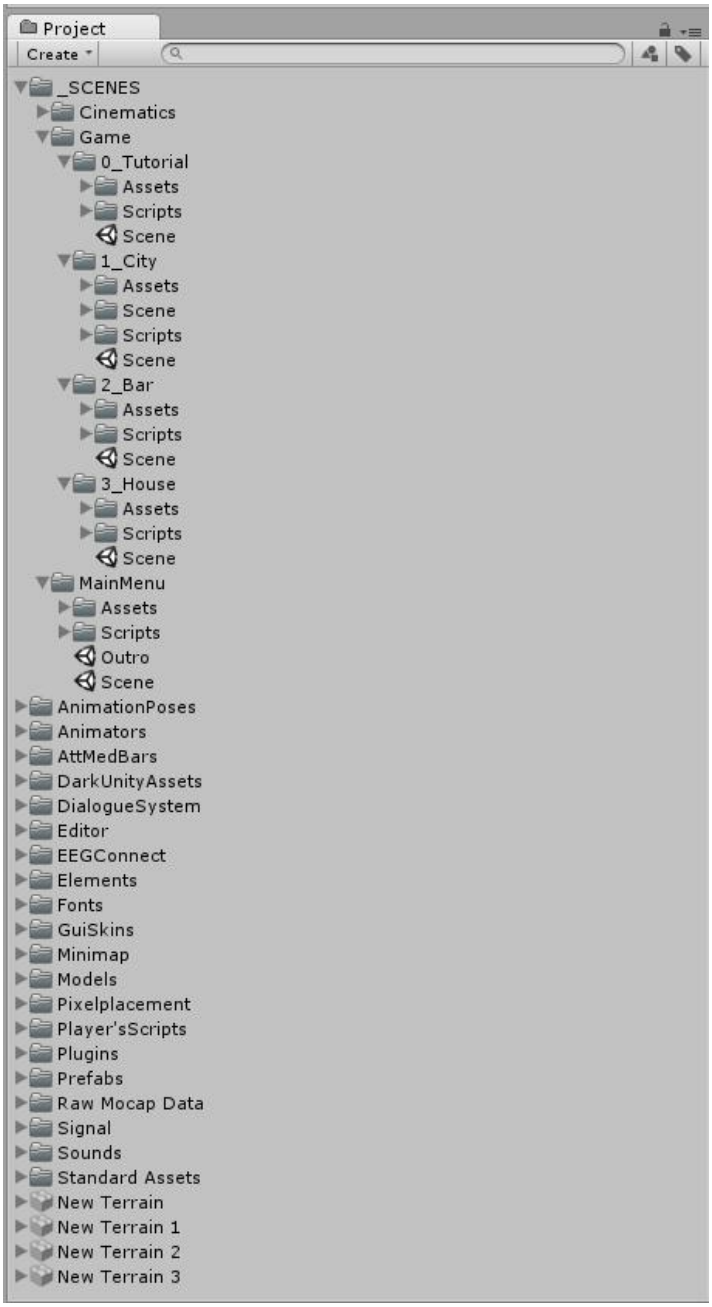


Εικόνα 17: Παράθυρο Inspector

Όπως φαίνεται στη παραπάνω εικόνα αυτές είναι οι Default ρυθμίσεις για κάθε GameObject που προστίθεται στη σκηνή μας. Αφού πλέον τελειώσουμε με τη διαμόρφωση του terrain όπως είπαμε παραπάνω, το επόμενο στάδιο είναι να βάλουμε κτήρια, δρόμους και ότι άλλο χρειάζεται για να δημιουργήσουμε μια πόλη.

Για την τοποθέτηση των αντικειμένων αυτό που πρέπει να κάνουμε είναι να επιλέξουμε το επιθυμητό αντικείμενο/μοντέλο και να το σύρουμε μέσα στο παράθυρο Scene. Εκεί μπορούμε να του αλλάξουμε μέγεθος να το στρέψουμε και να το επανατοποθετήσουμε.

Η υλοποίηση του παιχνιδιού αποτελείτε από πέντε βασικές σκηνές, τη σκηνή του Menu, το Tutorial, η πόλη, ο εσωτερικός χώρος του σπιτιού και το Bar. Υπάρχουν βέβαια και ενδιάμεσες σκηνές που είναι τα video. Κάτι που πρέπει να σημειωθεί σε αυτό το σημείο είναι ότι είναι απαραίτητη και πολύ σημαντική η αρχειοθέτηση και η ταξινόμηση των φακέλων στο παράθυρο Project.



Εικόνα 18: Ταξινόμηση Assets

Όπως φαίνεται στη εικόνα κάθε φάκελος της εκάστοτε σκηνής περιέχει ένα φάκελο με τα Assets δηλαδή τα μοντέλα που χρησιμοποιήθηκαν στη σκηνή και ένα ακόμη με τα Scripts που χρησιμοποιούνται στα αντικείμενα της.

Υπάρχουν επίσης κι άλλοι φάκελοι που περιέχουν τα Animations που εφαρμόζονται στους παίκτες, το Minimap, τους ήχους που χρησιμοποιούνται μέσα στο παιχνίδι, τον φάκελο Fonts που περιέχει γραμματοσειρές για τα GUISkins. Θα αναλυθούν πιο περισσότερο παρακάτω.

5.1.1 Κεντρικό Μενού Επιλογών

Το Menu του παιχνιδιού είναι διαμορφωμένο σε μια σκηνή στην οποία έχουμε τοποθετήσει κτήρια, δέντρα, ανθρώπους που κινούνται και συνεπώς τις επιλογές του μενού που είναι το **New Game**, τα **Options**, το **Help**, τα **Credits** και το **Exit.** Για την υλοποίηση του, η κυριά ιδέα είναι η κάμερα της σκηνής να κατευθύνετε από σημείο σε σημείο. Για να πέτυχουμε κάτι τέτοιο έγινε η χρήση της βιβλιοθήκης iTween της Pixelplacement που αναλύσαμε στο παραπάνω κεφάλαιο 'Βιβλιοθήκες'.

Για τη δημιουργία των κουμπιών χρησιμοποιήθηκαν κύβοι, GameObject > 3D Object > Cube όπου με την κατάλληλη μορφοποίηση και την δημιουργία των ανάλογων Texture στο Photoshop πήραν τη μορφή που βλέπουμε.

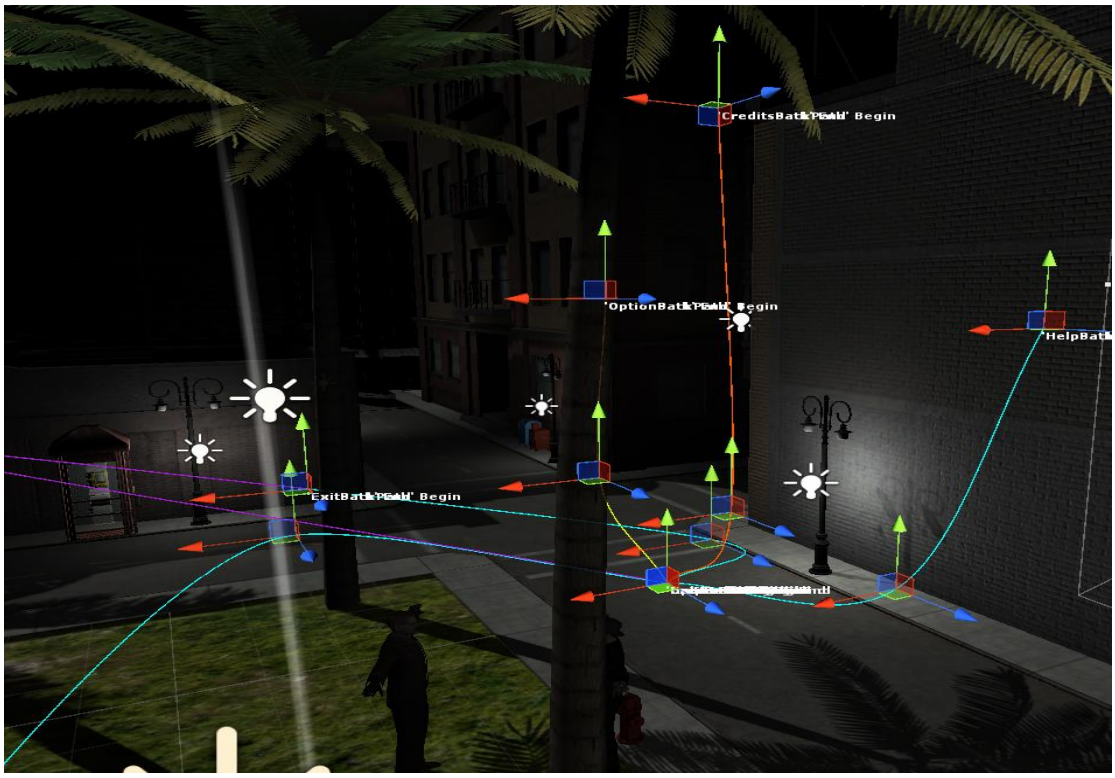
Εικόνα 19: Κεντρικό Μενού

Όταν γίνετε η έναρξη της σκηνής του Menu η κάμερα που βρίσκετε σε ένα υψηλό σημείο αρχίζει να κατευθύνετε και να σταθεροποιείται στο σημείο που βλέπουμε στη παραπάνω εικόνα. Ωστόσο, όσο η κάμερα κατεβαίνει βγαίνουν παράλληλα και τα Buttons του Menu μέσα από τον τοίχο, αντίστοιχα όταν βγαίνουμε από το μενού τα Buttons εισέρχονται πάλι μέσα στο τοίχο. Πατώντας οπουδήποτε κουμπί εκτός του New Game η κάμερα οδηγείτε σε διαφορετικά σημεία μέσα στο χώρο της σκηνής όπου εκεί επίσης ενεργοποιούνται αλλά Buttons και αρχίζουν να εμφανίζονται. Κάθε κουμπί περιέχει εκτός από τις ιδιότητές του

και ένα Script το οποίο καθορίζει τη λειτουργία του και είναι υπεύθυνο για τις λειτουργείες που εξηγήσαμε. Για παράδειγμα το κουμπί New Game πατώντας το, μας περνάει στη σκηνή του παιχνιδιού. Αντίστοιχα το Exit μας βγάζει από το παιχνίδι.

Από την άλλη πλευρά άλλο ένα εξίσου σημαντικό δομικό συστατικό του Menu είναι η κάμερα και το πιο δύσκολο στην υλοποίηση. Η δυσκολία της όφειλε λόγω των πολλών Paths που χρησιμοποιεί για να κατευθύνετε στο χώρο αλλά και να ενεργοποιεί τα κουμπιά να εξέλθουν από τον τοίχο.

Πιο συγκεκριμένα για κάθε Path που χρησιμοποιεί πρέπει να υπάρχει και ένας αντίστοιχος στόχος για εστιάσει στα σημεία που θέλουμε. Στην παρακάτω εικόνα φαίνονται τα μονοπάτια που ακολουθεί η κάμερα στο χώρο.



Εικόνα 20: Camera Paths

Στην εικόνα φαίνεται ότι έχουμε εννέα Paths και κατά συνέπεια εννέα στόχους που βρίσκονται στα σημεία που θέλουμε να εστιάσουμε κάθε φορά. Η δυνατότητα της δημιουργίας των Paths οφείλετε λόγω της χρήσης του iTween. Για να εφαρμόσουμε αυτή τη βιβλιοθήκη πρέπει από τον φάκελο Pixelplacement να βρούμε το script που λέγετε iTween path και να το σύρουμε στης ιδιότητες της κάμερας (Inspector). Εφόσον φτιάξουμε όλα τα σημεία που θέλουμε να κατευθυνθούμε μετά χρειάζεται να τα προγραμματίσουμε. Συνεπώς, πρέπει να δημιουργήσουμε ένα ακόμα Script στο οποίο περιγράφεται ο τρόπος με τον οποίο γίνετε. Επίσης πρέπει να τονιστεί ότι όλα τα προγράμματα που δημιουργούνται και επισυνάπτονται στα αντικείμενα είναι γραμμένα σε C#.

Αλγόριθμος Κάμερας

Για να δημιουργήσουμε οτιδήποτε πρέπει αρχικά να ορίσουμε κάποιες μεταβλητές οι οποίες θα είχαν νόημα σε περίπτωση που τις μεταβάλαμε θα μεταβάλλονταν και οι ιδιότητες της κίνησης της κάμερας ως προς τους στόχους. Η συνάρτηση Start() όπως υποδεικνύει και το όνομά της καλείται στην φόρτωση της σκηνής και είναι κατάλληλη για αρχικοποίηση μεταβλητών και αντικειμένων και κατά συνέπεια το πρώτο path που θέλουμε να ακολουθήσει είναι αυτό προς την περιοχή του μενού πιο συγκεκριμένα περιγράφετε στη γραμμή:

```
iTween.MoveTo(gameObject,iTween.Hash("path", iTweenPath.GetPath("CameraPath"), "time" , 15 , "easeType", iTween.EaseType.linear));
```

Έχοντας αρχικοποιήσει στην μέθοδο Start() τις μεταβλητές και το πρώτο path που θέλουμε να τρέξει η κάμερα, περνάμε στην επόμενη πολύ σημαντική μέθοδο, την Update(). Το κάθε script που γράφεται και γίνεται compiled μπορεί να περιέχει μια μέθοδο Update() και Start(). Κάνοντας το τελικό Build η κάθε Update() και Start() ενώνονται. Η Update() εκτελείται σε κάθε frame και εδώ είναι που μπαίνει ο κώδικας ο οποίος αναφέρεται στα επόμενα paths που πρόκειται να ακολουθήσει η κάμερα πατώντας κάποιο από τα buttons του μενού. Συνεπώς χρησιμοποιώντας τη μεταβλητή target η οποία όπως φαίνεται είναι public static ενημερώνετε διαρκώς με τη τοποθεσία της κάμερας.

```
using UnityEngine;
using System.Collections;

public class CameraPath : MonoBehaviour {


    public GameObject point;
    public GameObject OptionPoint;
    public static GameObject target;
    public static GameObject selected;
    public float x=0f;
    public float y=0f;
    public float z=0f;
    void Start () {
        target = new GameObject ();
        selected = new GameObject ();
        target.transform.position = GameObject.Find ("CameraPoint").transform.position;
        selected.transform.position = GameObject.Find ("CameraPoint").transform.position;
        iTween.MoveTo(gameObject, iTween.Hash("path" , iTweenPath.GetPath("CameraPath"), "time" ,15 ,"easeType", iTween.EaseType.linear));


    }

    void Update () {
        this.transform.LookAt (target.transform);
        if (target.transform.position.x != selected.transform.position.x)x = (selected.transform.position.x - target.transform.position.x)*2 / 100;
        if (target.transform.position.y != selected.transform.position.y)y = (selected.transform.position.y - target.transform.position.y)*2 / 100;
        if (target.transform.position.y != selected.transform.position.y) z = (selected.transform.position.z - target.transform.position.z)*2 / 100;
        target.transform.position = new Vector3 (target.transform.position.x + x, target.transform.position.y + y, target.transform.position.z + z);
    }

}

```

Πίνακας 5: Αλγόριθμος Κάμερας

Αλγόριθμοι Buttons

Η λειτουργεία τους όπως είπαμε είναι να στέλνουν τη κάμερα στο αντίστοιχο μονοπάτι, να εμφανίζουν και να εξαφανίζουν τα υπόλοιπα buttons, να μας εισάγουν στο κυρίως παιχνίδι όπως επίσης και να κάνουμε Exit από αυτό.

NewGame Button

Στη συνάρτηση Update() φαίνονται τα υπόλοιπα buttons που εισέρχονται στον τοίχο και εξαφανίζονται όπως επίσης γίνετε και το Load Level μαζί με ένα Auto fade εφέ για την είσοδο στο κυρίως μέρος του παιχνιδιού. **`AutoFade.LoadLevel (10, 2.5f, 2.5f, Color.black);`**

```
using UnityEngine;
using System.Collections;

public class BtnNewGame : MonoBehaviour {

    public bool isOver = false;
    public bool isClick = false;
    int i;

    void OnMouseEnter() {
        isOver = true;
    }
    void OnMouseExit() {
        isOver = false;
    }
    void OnMouseDown(){
        isClick = true;
    }

    void Update()
    {

        if (isClick == true && GameObject.Find ("Options").transform.position.x > 59.26f) {
                        GameObject.Find ("Options").transform.position = new Vector3 (GameObject.Find ("Options").transform.position.x - 0.01f,
                        GameObject.Find ("Options").transform.position.y, GameObject.Find ("Options").transform.position.z);
            AutoFade.LoadLevel (5, 2.5f, 2.5f, Color.black);}
        if (isClick == true && GameObject.Find ("Options").transform.position.x <= 59.26f && GameObject.Find ("Help").transform.position.x > 59.26f)
                                GameObject.Find ("Help").transform.position = new Vector3 (GameObject.Find ("Help").transform.position.x - 0.01f,
                                GameObject.Find ("Help").transform.position.y, GameObject.Find ("Help").transform.position.z);
        if (isClick == true && GameObject.Find ("Help").transform.position.x <= 59.26f && GameObject.Find ("Credits").transform.position.x > 59.26f)
                                GameObject.Find ("Credits").transform.position = new Vector3 (GameObject.Find ("Credits").transform.position.x - 0.01f,
                                GameObject.Find ("Credits").transform.position.y, GameObject.Find ("Credits").transform.position.z);
        if (isClick == true && GameObject.Find ("Credits").transform.position.x <= 59.26f && GameObject.Find ("Exit").transform.position.x > 59.26f)
                                GameObject.Find ("Exit").transform.position = new Vector3 (GameObject.Find ("Exit").transform.position.x - 0.01f,
                                GameObject.Find ("Exit").transform.position.y, GameObject.Find ("Exit").transform.position.z);
    }
}
```

Πίνακας 6: NewGame Button

Option Help, Credits, Exit Buttons

Ο κώδικας του Option Button έχει δυο βασικά σημεία που πρέπει να επισημανθούν το ένα είναι στη γραμμή κώδικα :

```
iTween.MoveTo(GameObject.Find("Main Camera"), iTween.Hash("path" ,
iTweenPath.GetPath("OptionPath"), "time" ,3 ,"easeType",
iTween.EaseType.linear));
```

Ενεργοποιεί την κάμερα να μετακινηθεί και να δείξει στο σημείο των Options και το δεύτερο είναι ο έλεγχος που γίνετε μέσα στις If για το αν τα sub-buttons του option είναι

ορατά ή όχι και εμφανίζονται στο σημείο που θέλουμε, η αλλαγή της θέσης τους γίνετε με την εντολή transform.position.

Όλα τα υπόλοιπα κουμπιά **Help, Credits, Exit** είναι στηριγμένα στην ίδια λογική με τη διαφορά ότι κάθε φορά αλλάζει το όνομα του Path στη γραμμή κώδικα που προαναφέραμε και τα αντίστοιχα sub-buttons που θέλουμε να εμφανίσουμε κάθε φορά.

```
using UnityEngine;
using System.Collections;

public class BtnOptions : MonoBehaviour {

    public bool isOver = false;
    public static bool isClick = false;

    void OnMouseEnter() {

        isOver = true;

    }
    void OnMouseExit() {
        isOver = false;
    }
    void OnMouseDown(){

        isClick = true;
        BtnOptionBack.isClick = false;
        CameraPath.selected = GameObject.Find ("OptionPoint");
        iTween.MoveTo(GameObject.Find("Main Camera"), iTween.Hash("path" , iTweenPath.GetPath("OptionPath"), "time" ,3 ,"easeType", iTween.EaseType.linear));
        }


    void Update()
    {

        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("OptionButton").transform.position.x<59.5)
            GameObject.Find ("OptionButton").transform.position =new Vector3 (GameObject.Find ("OptionButton").transform.position.x+0.006f,
            GameObject.Find ("OptionButton").transform.position.y,GameObject.Find ("OptionButton").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGOption1").transform.position.x<59.5)
            GameObject.Find ("EEGOption1").transform.position =new Vector3 (GameObject.Find ("EEGOption1").transform.position.x+0.006f,
            GameObject.Find ("EEGOption1").transform.position.y,GameObject.Find ("EEGOption1").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGConnect").transform.position.x<59.5)
            GameObject.Find ("EEGConnect").transform.position =new Vector3 (GameObject.Find ("EEGConnect").transform.position.x+0.006f,
            GameObject.Find ("EEGConnect").transform.position.y,GameObject.Find ("EEGConnect").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGDisconnect").transform.position.x<59.5)
            GameObject.Find ("EEGDisconnect").transform.position =new Vector3 (GameObject.Find ("EEGDisconnect").transform.position.x+0.006f,
            GameObject.Find ("EEGDisconnect").transform.position.y,GameObject.Find ("EEGDisconnect").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGOption2").transform.position.x<59.5)
            GameObject.Find ("EEGOption2").transform.position =new Vector3 (GameObject.Find ("EEGOption2").transform.position.x+0.006f,
            GameObject.Find ("EEGOption2").transform.position.y,GameObject.Find ("EEGOption2").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGCom1").transform.position.x<59.5)
            GameObject.Find ("EEGCom1").transform.position =new Vector3 (GameObject.Find ("EEGCom1").transform.position.x+0.006f,
            GameObject.Find ("EEGCom1").transform.position.y,GameObject.Find ("EEGCom1").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGCom2").transform.position.x<59.5)
            GameObject.Find ("EEGCom2").transform.position =new Vector3 (GameObject.Find ("EEGCom2").transform.position.x+0.006f,
            GameObject.Find ("EEGCom2").transform.position.y,GameObject.Find ("EEGCom2").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGCom3").transform.position.x<59.5)
            GameObject.Find ("EEGCom3").transform.position =new Vector3 (GameObject.Find ("EEGCom3").transform.position.x+0.006f,
            GameObject.Find ("EEGCom3").transform.position.y,GameObject.Find ("EEGCom3").transform.position.z);
        if (isClick == true && GameObject.Find ("Options").transform.position.x>=59.5 && GameObject.Find ("EEGCom4").transform.position.x<59.5)
            GameObject.Find ("EEGCom4").transform.position =new Vector3 (GameObject.Find ("EEGCom4").transform.position.x+0.006f,
            GameObject.Find ("EEGCom4").transform.position.y,GameObject.Find ("EEGCom4").transform.position.z);
    }
}
```

Πίνακας 7: Option Button

5.1.2 Επίπεδα παιγνίου: Tutorial – City – Bar – House Scenes

Η υλοποίηση των τεσσάρων αυτών σκηνών που αναφέρονται στον τίτλο του υποκεφαλαίου δημιουργήθηκαν εύκολα έχοντας συλλέξει εξ' αρχής όλα τα μοντέλα που χρειαζόμαστε για την τοποθέτησης τους στο χώρο των σκηνών. Μετά από αυτό το στάδιο χεριάζετε να μπουν φώτα για σκιές και κυρίως για να είναι φωτορεαλιστικό το περιβάλλον. Με τη χρήση του Directional Light καταφέραμε να δημιουργήσουμε τις σκιές, αυτό επιτυγχάνετε επιλέγοντας το Directional Light και στη συνέχεια από τον Inspector που περιγράφονται οι ιδιότητες από τη κατηγορία Shadow Type επιλέγουμε Soft Shadows.

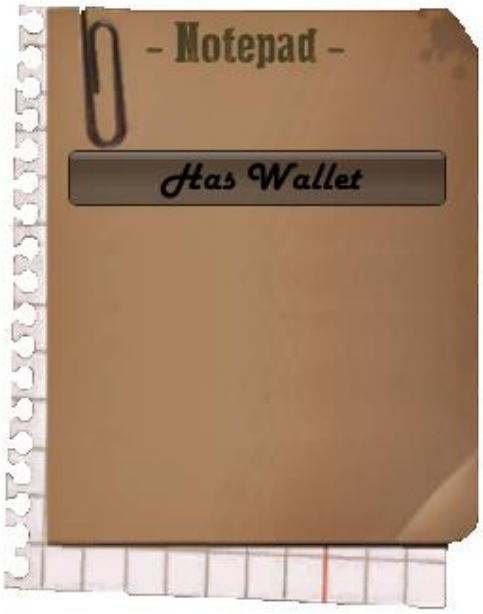
Εκτός από τα φωτά που είναι ζωτικό κομμάτι ενός παιχνιδιού, αρκετά μεγάλη βαρύτητα έχει και το κομμάτι της μουσικής που παίζει σε κάθε πιστά. Η μουσική πολλές φορές δημιουργεί συναισθήματα άγχους, ευχάριστα, μυστήριου και άλλων ανάλογα με το είδος και ύφος του εκάστοτε παιχνιδιού. Ωστόσο, μεγάλη σημασία έχουν επίσης και οι ήχοι που αφορούν μια πόλη όπως για παράδειγμα τα αμάξια, σειρήνες (περιπολικών, ασθενοφόρων), ομιλίες ανθρώπων, τριγμοί πόρτας όλα αυτά 'ζωντανεύουν' ακόμα πιο πολύ ένα παιχνίδι. Συνεπώς, για να προσαρμόσουμε και ήχο στο project χρειάστηκε να ψάξουμε μουσικά κομμάτια που προκαλούν συναισθήματα μυστήριου. Εφόσον, έγινε η έρευνα μουσικής αυτό που χρειάστηκε να γίνει στη συνέχεια είναι να δημιουργήσουμε σε κάθε πιστά ένα Game Object που θα περιέχει το γενικό μουσικό κομμάτι της εκάστοτε πίστας και να το ενσωματώσουμε στη Main Camera. Στα υπόλοιπα μοντέλα της πίστας όπως παραδείγματος χάριν τα σταματημένα αμάξια στο φανάρι ενός δρόμου μπορούμε επίσης πάνω τους να επισυνάψουμε το αντίστοιχο αρχείο ήχου. Κάτι ακόμα που πρέπει να πούμε σε αυτό το σημείο είναι ότι τα αρχεία ήχου που έχουμε τη δυνατότητα να χρησιμοποιήσουμε είναι *.mp3, .wav, .aiff* και *.ogg*.

Για την ολοκλήρωση της εμφάνισης του παιχνιδιού αποτελούν μέρος του και τα εργαλεία που χρειάζεται ένας παίκτης για να παίξει. Αυτά είναι το σύστημα διάλογων που του επιτρέπει την αλληλεπίδραση με άλλους παίκτες non-players κυρίως. Το Inventory που περιέχει όλα τα στοιχεία του συλλέγει καθ' όλη τη διάρκεια και την προβολή τους για να μπορεί να κάνει συνδυασμούς που χεριάζετε και να καταφέρει να φτάσει στη τελική λύση. Επίσης χρειάζεται να υπάρχει μίνι – χάρτης *(minimap)* που υποδεικνύει κάθε φορά το σημείο που βρίσκετε μέσα στο χώρο της πίστας. Και τέλος τα μηνύματα που εμφανίζονται σε διάφορες φάσης του παιχνιδιού που κατατοπίζουν ή υποδεικνύουν πληροφορίες στο χρήστη. Όλα τα παραπάνω επειδή χρησιμοποιούνται σε αρκετά σημεία, η υλοποίηση τους πραγματοποιήθηκε σε Prefabs, στη συνέχεια αναφέρονται αναλυτικά όπως επίσης περιέχονται και οι αλγόριθμοι που χρησιμοποιήθηκαν.

5.1.3 Inventory Manager

Σε πολλά παιχνίδια ο παίκτης χεριάζετε να μεταφέρει ένα συγκεκριμένο αριθμό αντικειμένων που έχει συλλέξει καθ' όλη τη διάρκεια του παιχνιδιού, επίσης η μορφή των Inventory Manager είναι διαφορετική κάθε φορά από παιχνίδι σε παιχνίδι. Κατά συνέπεια αυτό σημαίνει ότι ο παίκτης πρέπει να έχει τον έλεγχο των στοιχείων, προκειμένου να διαχειριστεί τη λύση ενός προβλήματος. Το είδος του παιχνιδιού που υλοποιούμε ανήκει στη κατηγορία puzzle / adventure games και απαιτείται η χρήση του.

Η μορφοποίηση του βασίστηκε στη δυνατότητα των μεθόδων *OnGUI()* και των *GUISkins*. Λέγοντας GUI *Graphical User Interface* καλείται το σύνολο γραφικών στοιχείων, τα οποία εμφανίζονται στην οθόνη κάποιας ψηφιακής συσκευής και χρησιμοποιούνται για την αλληλεπίδραση του χρήστη με τη συσκευή αυτή. Παρέχουν, μέσω γραφικών, ενδείξεις και εργαλεία προκειμένου αυτός να φέρει εις πέρας κάποιες επιθυμητές λειτουργίες. Το GUI Skin είναι μια συλλογή από GUIStyles που μπορούν να εφαρμοστούν και να αλλάξουν την εμφάνιση των gui elements που χρησιμοποιούνται, gui elements είναι τα buttons, text box, labels και άλλα. Για τη δημιουργία GUISkin, πρέπει να επιλέξουμε από το μενού *Assets > Create > GUI Skin.*



Εικόνα 21: Inventory Manager

Συνεπώς, το πρώτο βήμα είναι η δημιουργία της εμφάνισης του. Η επιλογή και η σύνθεση της εικόνας που φαίνεται παραπάνω σχεδιαστικέ στο Photoshop και εφαρμόστηκε ως texture στο GuiBox μέσω του GUI Skin. Τα στοιχεία που εμφανίζονται μέσα σε αυτό είναι buttons τα οποία δημιουργήθηκαν μέσω της μεθόδου OnGUI() και σηματοδοτούν τα

αντικείμενα που έχει βρει ο παίκτης στη διάρκεια του παιχνιδιού. Πατώντας πάνω σε κάθε ένα από τα εμφανίζετε στο κέντρο της οθόνης το αντικείμενο που ποιο πριν είχε συλλεχθεί από το χρήστη. Επίσης υπάρχει η δυνατότητα εμφάνισης και απομάκρυνσης του από την οθόνη πατώντας το πλήκτρο 'N'. Στη συνέχεια θα παρουσιαστεί ο κώδικας της υλοποίησης.

Αλγόριθμος Inventory Manager

Για τη δημιουργία του χρειάστηκε να γίνει συνδυασμός από Scripts. Το πρώτο αναφέρεται στην εμφάνιση και των γραφικών του στοιχείων στην οθόνη, το δεύτερο αφορά τη συλλογή των αντικειμένων (Pick Up Script) και τέλος η αποθήκευσή τους σε μια κλάση, για την ομαλή χρήση τους και πρόσβαση σε αυτά.

```
using UnityEngine;
using System.Collections;

public class NotepadGuiPlus : MonoBehaviour {

    public GUISkin NotepadSkin;
    bool opt1 = false;
    bool opt2 = false;
    bool opt3 = false;
    bool opt4 = false;
    bool opt5 = false;
    bool opt6 = false;
    bool opt7 = false;
    public Texture2D CardIcon;
    public Texture2D LetterIcon;
    public Texture2D NewspaperIcon;
    public Texture2D ReciptIcon;
    public Texture2D BloodHandIcon;
    public Texture2D GunIcon;
    public Texture2D BulletIcon;
    public int currentEPosition=0;

    public bool displayInventory =false;

    void Update() {

        if (Input.GetKeyDown (KeyCode.N))
        {
            displayInventory = !displayInventory;
        }

    }


    void OnGUI()
    {

        if (displayInventory) {

            GUI.skin = NotepadSkin;
            GUI.Box (new Rect (0, 0, 258, 347), "- Notepad -");


            if (PlayerInventory.hasReciept == true) {

                if (GUI.Button (new Rect (20, 90, 200, 30), "Wallet") && PlayerInventory.hasReciept == true) {
                    opt5 = !opt5;
                }

                if (opt5) {
                    if (GUI.Button (new Rect (Screen.width / 2 - 250, Screen.height / 2 - 250,ReciptIcon.width, ReciptIcon.height),ReciptIcon,"")) {

                    }
                }
            }
            if (PlayerInventory.hasBullet == true) {
                if (GUI.Button (new Rect (20, 120, 200, 30), "Bullet") && PlayerInventory.hasBullet == true) {
                    opt7 = !opt7;
                }

                if (opt7) {
                    if (GUI.Button (new Rect (Screen.width / 2 - 250, Screen.height / 2 - 250,BulletIcon.width, BulletIcon.height),BulletIcon,"")) {

                    }
                }
            }
            if (PlayerInventory.hasGun == true) {
                if (GUI.Button (new Rect (20, 150, 200, 30), "Gun") && PlayerInventory.hasGun == true) {
                    opt6 = !opt6;
                }

                if (opt6) {
                    if (GUI.Button (new Rect (Screen.width / 2 - 250, Screen.height / 2 - 250,GunIcon.width, GunIcon.height),GunIcon,"")) {

                    }
                }
            }
```

```
                }
            }
        }
        if (PlayerInventory.hasNewspaper == true) {
            if (GUI.Button (new Rect (20, 180, 258, 30), "Newpaper")) {
                opt1 = !opt1;
            }

            if (opt1) {
                if (GUI.Button (new Rect (Screen.width / 2 - 250, Screen.height / 2 - 250, NewspaperIcon.width, NewspaperIcon.height),NewspaperIcon,"")) {

                }
            }
        }
        if (PlayerInventory.hasCard == true) {
            if (GUI.Button (new Rect (20, 210, 200, 30), "Card") && PlayerInventory.hasCard == true) {
                opt4 = !opt4;
            }

            if (opt4) {
                if (GUI.Button (new Rect (Screen.width / 2 - 250, Screen.height / 2 - 250,CardIcon.width, CardIcon.height),CardIcon,"")) {

                }
            }
        }
        if (PlayerInventory.hasLetter == true) {
            if (GUI.Button (new Rect (20, 240, 200, 30), "Letter") && PlayerInventory.hasLetter == true) {
                opt3 = !opt3;
            }

            if (opt3) {
                if (GUI.Button (new Rect (Screen.width / 2 - 250, Screen.height / 2 - 250,LetterIcon.width, LetterIcon.height),LetterIcon,"")) {

                }
            }
        }
        if (PlayerInventory.hasCube == true) {
            if (GUI.Button (new Rect (0, 140, 258, 60), "Has Cube") && PlayerInventory.hasCube == true) {
                opt2 = !opt2;
            }

            if (opt2) {
                if (GUI.Button (new Rect (Screen.width / 2 - 250, Screen.height / 2 - 250, 500, 500), "Cube")) {

                }
            }
        }
    }
}
```

Πίνακας 8: Αλγόριθμος Inventory Manager

Αλγόριθμος Pick Up στοιχείων

Σε κάθε στοιχείο ξεχωριστά πρέπει να επισυνάψουμε το παρακάτω Script το οποίο δίνει οδηγίες για την αποθήκευση του και την καταγραφή του στη κλάση PlayerInventory που προαναφέραμε για τη διαχείριση του. Η συνάρτηση OnMouseDown() λειτουργεί όταν κάνουμε click με το ποντίκι πάνω στο συγκεκριμένο αντικείμενο.

```
using UnityEngine;
using System.Collections;

public class PickUpLetter : MonoBehaviour {

    public GameObject DisLetter;
    public bool isOver = false;
    public bool isClick = false;
    public GameObject item;
    public bool PickUp = false;

    void Start () {
        DisLetter= GameObject.Find ("DisplayLetter");
        DisLetter.SetActive (false);
    }
    void Update(){
        if (Input.GetKeyDown (KeyCode.Escape))
        {
            DisLetter.SetActive (false);
        }
    }
    void OnMouseEnter() {

        isOver = true;

    }
    void OnMouseExit() {
        isOver = false;
    }
    void OnMouseDown(){

        isClick = true;
        DisLetter.SetActive (true);
        PickUp = true;
        PlayerInventory.hasLetter = true;
        Debug.Log (PlayerInventory.hasLetter);
    }
}
```

Πίνακας 9: Αλγόριθμος Pick Up στοιχείων

Αλγόριθμος για τη συλλογή στοιχείων

```
using UnityEngine;
using System.Collections;

public class PlayerInventory : MonoBehaviour {

    public static bool hasReciept;
    public static bool hasCard;
    public static bool hasNewspaper;
    public static bool hasBloodHand;
    public static bool hasLetter;
    public static bool hasGun;
    public static bool hasCube;
    public static bool hasBullet;

}
```

Πίνακας 10: Αλγόριθμος για τη συλλογή στοιχείων

5.1.4 Μικρογραφία Χάρτη

Ένας μίνι-χάρτης ή minimap είναι μια μικρογραφία χάρτη που συχνά τοποθετείται σε μια γωνία της οθόνης στα βίντεο παιχνίδια για να βοηθήσει τους παίκτες στο να προσανατολιστούν μέσα στον κόσμο του παιχνιδιού. Συχνά είναι μόνο ένα μικρό τμήμα της οθόνης και κατά συνέπεια πρέπει να είναι επιλεκτική σε ό, τι εμφανίζουν λεπτομέρειες. Στοιχεία που συνήθως περιλαμβάνονται σε Μίνι-χαρτών ποικίλλουν από είδος παιχνιδιού βίντεο. Ωστόσο, συμπεριλαμβάνονται συχνά χαρακτηριστικά είναι η θέση του χαρακτήρα παίκτη, συμμαχικές μονάδες ή δομές, τους εχθρούς, τους στόχους και τις γύρω έδαφος. Μίνι-χάρτες έχουν κυρίως τα παιχνίδια στρατηγικής MMORPG επειδή χρησιμεύουν ως ένδειξη για το πού η τρέχουσα οθόνη βρίσκεται εντός του πεδίου εφαρμογής της στον κόσμο του παιχνιδιού. Επίσης τα FPS παιχνίδια έχουν επίσης κάποια εκδοχή ή παραλλαγή minimap, συχνά δείχνει τους εχθρούς σε πραγματικό χρόνο.

Εικόνα 22: Minimap

Αλγόριθμος Minimap

Για την υλοποίηση χρειάστηκε να γίνει ο συνδυασμός δυο scripts, το πρώτο πραγματοποιεί την εμφάνισή του και την τοποθέτησή του μέσα στο παιχνίδι και το δεύτερο την λειτουργία της κάμερας που ακολουθεί τον παίκτη κατά τη διάρκεια του παιχνιδιού. Το βέλος που δείχνει την κατεύθυνση του παίκτη είναι ένα Plane GameObject που του έχουμε διαμορφώσει το material με την εικόνα βέλους στη συνέχεια αυτό το το επισυνάψαμε πάνω από τον παίκτη και κατά συνέπεια να ακολουθεί τον παίκτη μαζί με την κάμερα.

Στην Update() πραγματοποιείτε έλεγχος για την κάμερα, αν δεν υπάρχει, το πρόγραμμα σταματάει να λειτουργεί αν υπάρχει συνεχίζει παρακάτω και τοποθετεί τον χάρτη στην οθόνη. Η μεταβλητή adjustSize κάνει το minimap responsive δηλαδή να μπορεί να προσαρμόζετε κάθε φορά με το μέγεθος της εκάστοτε οθόνης.

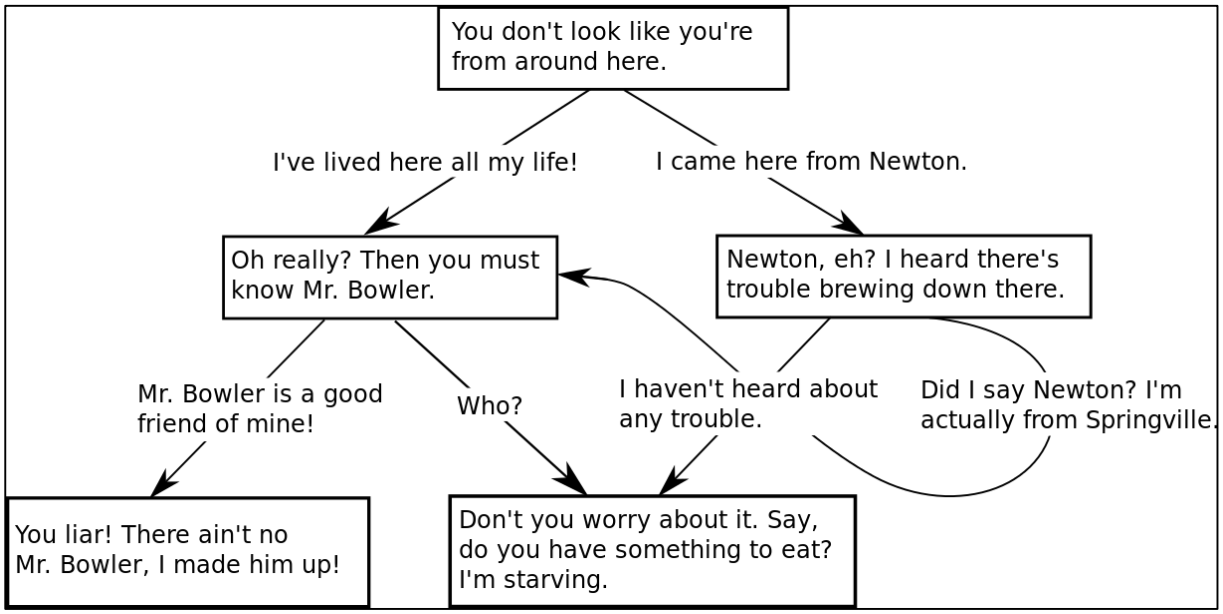
Στην μέθοδο OnGUI() και συγκεκριμένα στη γραμμή: **minimapCamera.Render** (); περιγράφετε οτι, ό,τι καταγραφεί η κάμερα να το προβάλει πάνω στο texture (**GUI.DrawTexture**) που δώσαμε στις οδηγίες. Το δεύτερο script που αναφέρθηκε έγινε Import από τα Assets του Unity, όπως έχουμε ξανά αναφέρει υπάρχουν μερικά έτοιμα πράγματα που μπορούμε να χρησιμοποιήσομε από τη πλατφόρμα, έτσι λοιπόν χρησιμοποιήσαμε το SmoothFollow.js το οποίο επισυναπτηκε στη κάμερα και ορίσαμε τον target το οποίο θέλουμε να ακολουθεί δηλαδή τον παίκτη του παιχνιδιού.

```
@script ExecuteInEditMode()

var minimapSize : float = 2.0;
var offsetX : float = 10.0;
var offsetY : float = 10.0;
private var adjustSize : float = 0.0;


var borderTexture : Texture;
var effectTexture : Texture;
var minimapCamera : Camera;


function Update() {
    if (minimapCamera == null) return;

    adjustSize = Mathf.RoundToInt(Screen.width/10);
    minimapCamera.pixelRect = new Rect((Screen.width -(minimapSize * adjustSize))-offsetX,  offsetY, minimapSize * adjustSize, minimapSize * adjustSize);

}

function OnGUI(){

    minimapCamera.Render ();
    GUI.DrawTexture(Rect((Screen.width -(minimapSize * adjustSize))-offsetX,
    (Screen.height -(minimapSize * adjustSize)) - offsetY, minimapSize * adjustSize, minimapSize * adjustSize), effectTexture);
    GUI.DrawTexture(Rect((Screen.width -(minimapSize * adjustSize))-offsetX,
    (Screen.height -(minimapSize * adjustSize)) - offsetY, minimapSize * adjustSize, minimapSize * adjustSize), borderTexture);
}
```

Πίνακας 11: Αλγόριθμος Minimap

5.1.5 Σύστημα Διαλόγων Παιγνίου

Ένα παράθυρο διαλόγου δέντρο ή συνομιλία είναι ένας μηχανισμός που χρησιμοποιείται σε πολλά παιχνίδια περιπέτειας και όχι μόνο. Κατά την αλληλεπίδραση του παίκτη με ένα non-player character (Έτσι λέγονται οι χαρακτήρες που αλληλοεπιδρούν με τον κυρίως παίκτη χωρίς να έχουν ιδιαίτερους ρόλους στο παιχνίδι), δίνεται στο παίκτη η επιλογή να επιλέξει τις απαντήσεις του για το τι θα κάνει στη συνέχεια μέχρι να λήξει η συζήτηση. Ορισμένα είδη των παιχνιδιών, όπως οπτικά μυθιστορήματα και dating sims, περιστρέφονται σχεδόν εξ ολοκλήρου γύρω από αυτές τις αλληλεπιδράσεις χαρακτήρα και Dialogue tree.



Εικόνα 23: Παράδειγμα Dialogue tree

Το πρώτο σύστημα διαλόγου γράφτηκε από τον Joseph Weizenbaum μεταξύ 1964 και 1966 σε ELIZA, μια πρωτόγονη φυσική γλώσσα προγράμματος επεξεργασίας υπολογιστή. Το πρόγραμμα μιμήθηκε την αλληλεπίδραση μεταξύ του χρήστη και ενός υπολογιστή. Με την έλευση των video games και της διαδραστικής ψυχαγωγίας έχουν προσπαθήσει να ενσωματώσουν σημαντικές αλληλεπιδράσεις με τους εικονικούς χαρακτήρες.

Ο παίκτης συνήθως μπαίνει στη διαδικασία να μιλήσει με έναν χαρακτήρα non-Player, και στη συνέχεια, επιλέγει μια σειρά από απαντήσεις και εκτυλίσσετε ο διάλογος. Μετά την επιλογή των απαντήσεων, ο χαρακτήρας non-Player ανταποκρίνεται στον παίκτη, και ο παίκτης απαντά επίσης σε κάτι άλλο. Ο κύκλος αυτός συνεχίζεται μέχρι να λήξει η συζήτηση. Η συζήτηση μπορεί να λήξει όταν ο παίκτης επιλέγει ένα αποχαιρετιστήριο

μήνυμα, ο χαρακτήρας non-Player δεν έχει τίποτα περισσότερο να προσθέσει και τελειώνει τη συνομιλία, επίσης κάποιες φορές η συζήτηση τερματίζετε όταν ο παίκτης κάνει μια 'κακή' επιλογή που ίσως εξαγριώσει τον non-Player και να αφήσει τη συνομιλία.

Συχνά ο μηχανισμός αυτός επιτρέπει στους σχεδιαστές παιχνιδιών να παρέχουν διαδραστικές συνομιλίες με τους non-Player χαρακτήρες χωρίς να χρειάζεται να αντιμετωπίσουν τις προκλήσεις της επεξεργασία φυσικής γλώσσας/φωνής στον τομέα της τεχνητής νοημοσύνης. Σε παιχνίδια όπως το Monkey Island, αυτές οι συνομιλίες μπορούν να φανερώσουν την προσωπικότητα ορισμένων χαρακτήρων.

Για την υλοποίηση του Dialogue System του παιχνιδιού η λογική που ακολουθείτε είναι βασισμένη σε steps δηλαδή το πρώτο step περιλαμβάνει την πρώτη ερώτηση και απάντηση μεταξύ του παίκτη και του non-player χαρακτήρα και συνεχίζετε στα επόμενα πατώντας το κουμπί 'next' που βρίσκεται μέσα στην περιοχή συνομιλίας. Το Dialogue System ενεργοποιείτε όταν ο παίκτης βρίσκεται μέσα στη περιοχή του non-player χαρακτήρα με τη χρήση των μεθόδων OnTriggerEnter() και OnTriggerExit() εφόσον συμβεί αυτό μετά λαμβάνει χωρά η μέθοδος OnGUI() που είναι υπεύθυνη για την εμφάνιση των μηνυμάτων στην οθόνη.

Αλγόριθμος Dialogue System

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class Dialogue1 : MonoBehaviour {
5
6      public bool startDialog= false;
7      public int currentStep=0;
8      public GUISkin skin;
9
10     void OnTriggerEnter(){
11         startDialog = true;
12     }
13     void OnTriggerExit(){
14         startDialog = false;
15     }
16
17     void OnGUI()
18     {
19         if (!startDialog)
20             return;
21         GUI.skin = skin;
22
23         if (currentStep == 0)
24         {
25             GUILayout.BeginArea (new Rect (Screen.width / 2 - 400, Screen.height/2 - 50, 800, 150));
26             GUI.Box (new Rect (0, 0, 800, 150), "");
27             GUILayout.Label ("- Hello, Mr. Christopher Clark.");
28             GUILayout.Label ("  We got an anonymous call which declared the murder.");
29             GUILayout.Label ("- What hour was performed the incident?");
30             if (GUI.Button (new Rect (700, 100, 80, 50), "Next")) currentStep++;
31             GUILayout.EndArea ();
32             return;
33         }
```

Πίνακας 12: Αλγόριθμος Dialogue System

5.1.6 Message Triggers

Message Triggers είναι τα μηνύματα που εμφανίζονται στον παίκτη κατά τη διάρκεια του παιχνιδιού. Η λειτουργεία τους είναι απλή, όταν ο παίκτης περάσει από μια περιοχή οπού έχουμε τοποθετήσει ένα διάφανο Plane GameObject ενεργοποιείτε το μήνυμα. Από τις ιδιότητες του Plane έχουμε ενεργοποιήσει την επιλογή στον Collider να είναι Is Trigger με αυτό τον τρόπο χρησιμοποιούμε τις μεθόδους OnTriggerEnter() οπού έχουμε βάλει στο εσωτερικό τους μια Boolean μεταβλητή την displayMessage η οποία δίνει στη συνέχεια την οδηγία στη μέθοδο OnGui() να εμφανίσει το εκάστοτε μήνυμα. Βέβαια το μήνυμα μετά από ένα ορισμένο χρονικό διάστημα εξαφανίζετε.

Αλγόριθμος Message Trigger

```
using UnityEngine;
using System.Collections;

public class messageTrigger : MonoBehaviour {

    public GUISkin GuiSkinTest;
    public bool passed=false;
    public float startTime =0f;
    public float displayTime =10f;
    public bool displayMessage =false;
    public string text = "this is a message";
    public int currentMPosition=0;

    void OnTriggerEnter(Collider other)
    {
        if (displayMessage == false && passed == false)
        {
            displayMessage = true;
            startTime = Time.time;
            MessagePool.LoadedMessages += 1;
            currentMPosition = MessagePool.LoadedMessages;
        }
    }
    void OnGUI()
    {
        GUI.skin = GuiSkinTest;
        if (displayMessage && passed==false)
        {
            if ((Time.time - startTime) < displayTime)
            {
                GUI.Box (new Rect (10, currentMPosition*10+(currentMPosition-1)*50, Screen.width-10, 50), text);
            }
            else
            {
                displayMessage = false;
                passed=true;
                MessagePool.LoadedMessages=-1;
            }
        }
    }
}
```

Πίνακας 13: Αλγόριθμος Message Trigger

5.1.7 Pause Menu

Σε ένα παιχνίδι προφανώς δε πρέπει να λείπει ένα Pause menu, είναι σημαντικό στα παιχνίδια που έχουν χρόνο μέτρησης, με αυτό τον τρόπο ο χρήστης μπορεί να σταματήσει το παιχνίδι για οποιουσδήποτε λογούς και να συνεχίσει αργότερα χωρίς να έχει χάσει ή ο περασμένος χρόνος να χρησιμοποιηθεί εναντίον του. Το παιχνίδι που παρουσιάζετε χρησιμοποιεί χρόνο σε ένα συγκεκριμένο σημείο οπότε το Pause είναι απαραίτητο. Πατώντας το πλήκτρο Esc 'παγώνουν' τα πάντα στη εκάστοτε πιστά, παράλληλα στην οθόνη προβάλετε ένα μήνυμα του ρωτάει το χρήστη αν θέλει να παραμείνει στο παιχνίδι ή όχι αυτό γίνετε σε περίπτωση που πραγματικά ο χρήστης θέλει να αποχωρίσει από το παιχνίδι. Παρακάτω παρουσιάζεται ο κώδικας για την υλοποίηση του.

Αλγόριθμος Pause Menu

```
using UnityEngine;
using System.Collections;

public class PauseMenu : MonoBehaviour {

    public bool displayPause= false;
    public GUISkin EscapeSkin;

    void Update () {

        if (Input.GetKeyDown (KeyCode.Escape))
        {
            displayPause = !displayPause;
        }
    }

    void OnGUI(){
        GUI.skin = EscapeSkin;

        if (displayPause)
        {
            GameObject.Find("Player").GetComponent<MouseLook>().enabled=false;

            GUILayout.BeginArea(new Rect(Screen.width/2-250,Screen.height/2-150,500,300));

            GUI.Box(new Rect(0,0,500,500), "Do you want really exit?");
            Time.timeScale = 0.0f;
                if(GUI.Button(new Rect(200,80,80,50), "Yes"))
                {
                    displayPause=false;
                    Time.timeScale = 1f;
                    AutoFade.LoadLevel(13,3,2,Color.black);
                }
                if(GUI.Button(new Rect(300,100,80,50), "No")) {
                    Application.CancelQuit();
                }
            GUILayout.EndArea();
        }
        else
        {
            Time.timeScale = 1f;
            GameObject.Find("Player").GetComponent<MouseLook>().enabled=true;
        }
    }
}
```

Πίνακας 14: Αλγόριθμος Pause Menu

5.1.8 Εμφάνιση Δεδομένων Εγκεφαλογράφου

Η υλοποίηση του παιχνιδιού εμπεριέχει λειτουργίες που απαιτούν τη χρήση του EEG για την επίλυση προβλημάτων και την κατεύθυνση προς τον τερματισμό του. Παραδείγματος χάριν, σε κάποια σημεία ο χρήστης καλείται να βρει κάποια αντικείμενα αρκεί οι τιμές του Attention να είναι σε υψηλά επίπεδα. Έτσι λοιπόν αυτά τα δεδομένα που δεχόμαστε από τον ηλεκτροεγκεφαλογράφο (EEG Mindwave) πρέπει να τα εμφανίσουμε και στην οθόνη για να γνωρίζει ο χρήστης τις βιομετρικές του τιμές, να έχει τον έλεγχο αλλά και να παρατηρήσει πως αντιδρά όταν ψάχνει να βρει ένα κρυμμένο αντικείμενο στο χώρο, σε αυτή τη περίπτωση οι τιμές του Attention θα έχουν πολύ υψηλά επίπεδα.

Εικόνα 24: Display EEG Data

Στη παραπάνω εικόνα εμφανίζονται οι τιμές του Meditation με μπλε χρώμα, οι τιμές του Attention με κόκκινο αντίστοιχα και το σήμα που αλλάζει αν χαθεί λίγο αλλά και όταν χαθεί εντελώς. Στο κεφάλαιο 4.2.2.2 ThinkGear έχουμε αναλύσει πως κυμαίνονται οι τιμές αυτές. Στην αρχή του παιχνιδιού όταν πραγματοποιούμε τη σύνδεση του EEG από τα Options ο αντάπτορας wireless που είναι συνδεμένος στον υπολογιστή αλλάζει χρώμα, από κόκκινο γίνετε μπλε αυτό μας δείχνει ότι η σύνδεση έγινε επιτυχώς, εξίσου το ίδιο συμβαίνει και στη συσκευή του EEG που τοποθετείτε στο κεφάλι του χρήστη.

```
using UnityEngine;
using System.Collections;

public class AttentionBar : MonoBehaviour {

    public GUIStyle AttBar;
    public GUIStyle EscapeSkin;
    public Texture2D Back;
    public Texture2D Front;
    public Texture2D icon;
    int Attention = int.Parse(EEGLib.attention);

    void Update(){
        Attention = int.Parse(EEGLib.attention);
        Debug.Log (EEGLib.attention + "\t-\t"+EEGLib.meditation);
    }

    void OnGUI(){

        GUI.BeginGroup (new Rect (50, Screen.height-30, 300, 32));
        GUI.Box (new Rect (0, 0, 300, 32), Back, AttBar);
        GUI.BeginGroup (new Rect (0, 0, Attention*3, 32));
        GUI.Box (new Rect (0, 0, 300, 32), Front, AttBar);
        GUI.EndGroup ();
        GUI.EndGroup ();
        GUI.Button (new Rect (10, Screen.height-35, icon.width, icon.height),icon,"");
    }
}
```

Πίνακας 15: Κώδικας εμφάνισης Attention

Οι μεταβλητές που χρησιμοποιούμε είναι τα Textures που εικονίζουν το χρώμα τις μπάρας, το πάνω μέρος οπού είναι το κόκκινο χρώμα (Front) και ανεβοκατεβαίνει ανάλογα με την μεταβλητή Attention και το πίσω χρώμα που παραμένει σταθερό (Back). Όπως φαίνεται στις μεταβλητές χρησιμοποιείτε και ένα επιπλέον GuiSkin (GUIStyle) όπου από τις ρυθμίσεις του δίνουμε επιπλέον μορφοποίηση πέραν από αυτά που μπορούμε να κάνουμε μέσω του κώδικα.

Η μεταβλητή int Attention = int.Parse(EEGLib.attention) προέρχεται από την βιβλιοθήκη EEGLib που δημιουργήθηκε ειδικά για το παιχνίδι. Ακριβώς ο ίδιος κώδικας ισχύει και για το Meditation όπου αλλάζουν οι εικόνες των textures και αντί για τη μεταβλητή attention χρησιμοποιείτε η int Meditation = int.Parse(EEGLib.meditation);

```
using UnityEngine;
using System.Collections;

public class Signal : MonoBehaviour {

    public Texture2D icon1;
    public Texture2D icon2;
    public Texture2D icon3;
    public Texture2D icon4;
    public Texture2D icon5;
    public Texture2D icon;
    int sig = int.Parse(EEGLib.signal);

    void Update(){
        sig = int.Parse(EEGLib.signal);
    }

    void OnGUI(){
        GUI.Button (new Rect (10, Screen.height-100, icon.width, icon.height),icon,"");
        if (sig==0)
        {
            GUI.Button (new Rect (50, Screen.height-90, icon2.width, icon2.height),icon2,"");
        }
        if(sig>1 && sig<70)
        {
            GUI.Button (new Rect (50, Screen.height-90, icon3.width, icon3.height),icon3,"");
        }
        if(sig>70 && sig<120)
        {
            GUI.Button (new Rect (50, Screen.height-90, icon4.width, icon4.height),icon4,"");
        }
        if(sig>120 && sig<170)
        {
            GUI.Button (new Rect (50, Screen.height-90, icon5.width, icon5.height),icon5,"");
        }
        if(sig==200)
        {
            GUI.Button (new Rect (50, Screen.height-90, icon1.width, icon1.height),icon1,"");
        }
    }
}
```

Πίνακας 16: Κώδικας Signal

Το Poor Signal Quality είναι η τιμή που χειριζόμαστε στον παρακάτω κώδικα περιγράφεται από ένα ακέραιο αριθμό που κυμαίνεται από 0 έως 255. Οι μη μηδενικές τιμές υποδεικνύουν ότι έχουμε κάποιο είδος θορύβου. Όσο υψηλότερες είναι οι τιμές, τόσο περισσότερος θόρυβος ανιχνεύεται. Η τιμή 200 έχει ιδιαίτερη σημασία, γιατί σημαίνει ότι

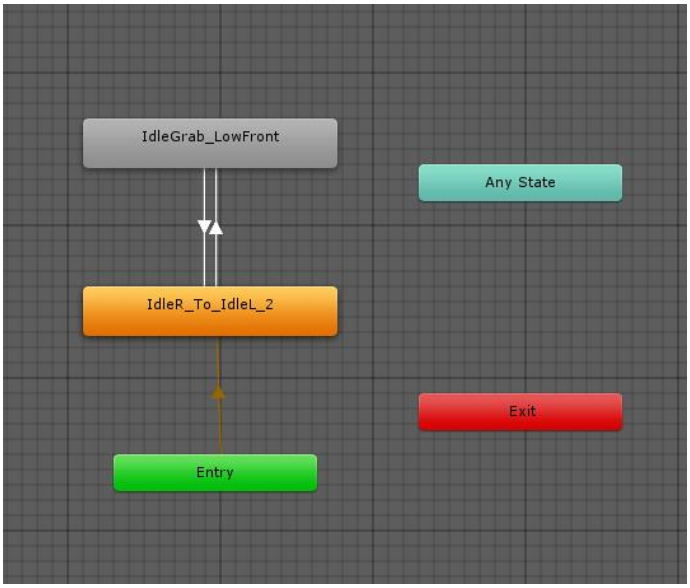
τα ηλεκτρόδια δεν είναι σε επαφή με το δέρμα του χρήστη. Κάποιοι επίσης λόγοι που μπορούν να προκαλέσουν μερική απώλεια σήματος είναι η κακή επαφή των ηλεκτροδίων με το κεφάλι του χρήστη, η υπερβολική κίνηση του κεφαλιού ή του σώματος του χρήστη.

Οι μεταβλητές που χρησιμοποιούμε είναι τα Textures, απεικονίζουν τα icons που αλλάζουν ανάλογα με τις τιμές του Signal που προέρχεται από το EEG και σαφώς τη δήλωση της μεταβλητής του σήματος που λαμβάνουμε από τη βιβλιοθήκη που δημιουργήσαμε int Signal = int.Parse(EEGLib.signal);.

Στο παραπάνω κομμάτι κώδικα φαίνεται η κατηγοριοποίηση εύρους του σήματος ανάλογα με τις τιμές που προέρχονται από τη συσκευή και την εκπροσώπησή τους από τα αντίστοιχα icons.

5.2 Animations

Σε αυτό το υποκεφάλαιο θα ασχοληθούμε με την εφαρμογή των κινήσεων (Animation Clips) που χρησιμοποιήθηκαν μέσα στο παιχνίδι για την κίνηση των παικτών και των αντικειμένων. Το Unity3D παρέχει ένα πολύ σημαντικό εργαλείο που λέγεται Animator Controller στο οποίο γίνετε ο συνδυασμός κινήσεων και η εφαρμογή τους στον εκάστοτε χαρακτήρα ή αντικείμενο. Οι κινήσεις αυτές εναλλάσσονται μεταξύ τους όταν συμβεί κάποιο γεγονός, για παράδειγμα η εναλλαγή από περπάτημα σε τρέξιμο όταν πατιέται το πλήκτρο Shift ωφελείτε σε αύτη τη σύνθεση κινήσεων που αναφέραμε παραπάνω. Πολλές φορές είναι αναγκαίο να χρησιμοποιούνται πάνω από μια κινήσεις σε ένα Animator Controller.



Εικόνα 25: Παράδειγμα χρήσης Animator Controller

Στη παραπάνω εικόνα περιγράφονται δύο Animation Clips τα οποία είναι συνδεδεμένα μεταξύ τους, αυτό σημαίνει ότι όταν τελειώσει το πρώτο clip θα συνεχίσει το επόμενο και ούτω καθεξής. Το Clip που έχει πορτοκαλί χρώμα υποδεικνύει ότι είναι η Default κίνηση του χαρακτήρα, το Exit υποδηλώνει τον τερματισμό όλων των κινήσεων όταν συμβεί μια δράση που τερματίζει το χαρακτήρα, παραδείγματος χάριν όταν ο παίκτης πεθάνει με κάποιο τρόπο το τελευταίο clip θα πρέπει να συνδεθεί με το exit.

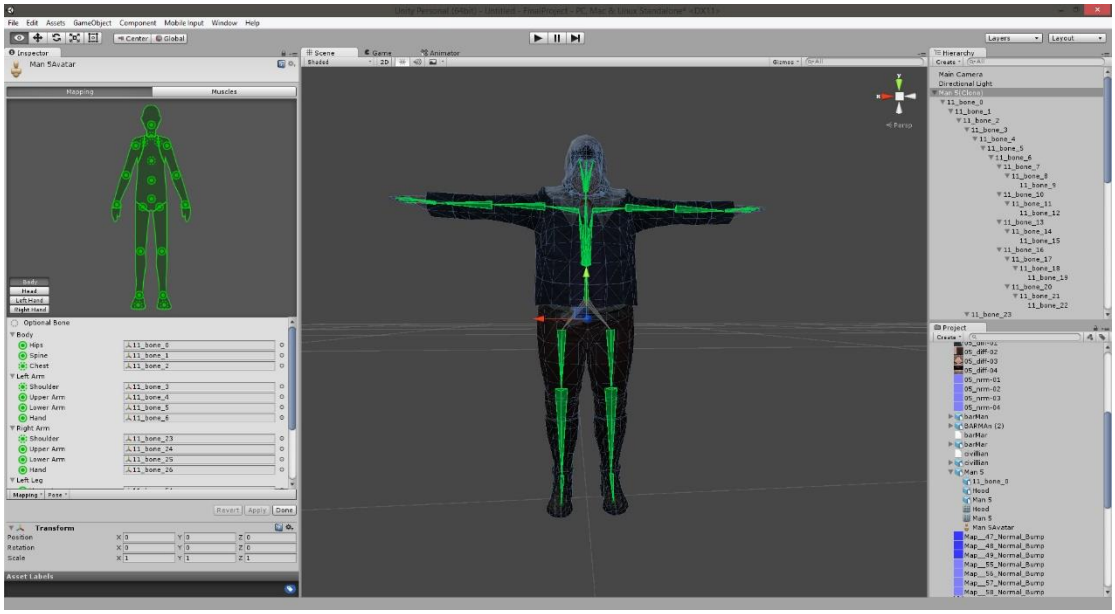
Στη παραπάνω παράγραφο έγινε αναφορά για Animation Clips. Το Animation Clip είναι ένα είδος κίνησης λέγοντας αυτό εννοούμε ότι είναι ένα clip βηματισμού ή για παράδειγμα η κίνηση του μοντέλου που πιάνει κάτι από το έδαφος.

Η διαδικασία εφαρμογής του Animator Controller σε ένα χαρακτήρα προϋποθέτει ο χαρακτήρας να καλύπτει κάποια χαρακτηρίστηκα. Το μοντέλο πρέπει να έχει τον κατάλληλο αριθμό από bones συνήθως από 13 και άνω, ο λόγος που πρέπει να συμβαίνει αυτό είναι γιατί χρειάζεται να γίνει αντιπροσωπευτικός ο σκελετός του. Πρέπει δηλαδή να υπάρχουν 6 bones για τα ποδιά, 4 για τα χεριά και 3 για τον κορμό αυτός ο αριθμός bones είναι ο μικρότερος δυνατός για να αντιπροσωπεύσουν ένα μοντέλο με βασικές κινήσεις.

Εφαρμογή Animator Controller

Η εφαρμογή του Animator Controller στις ιδιότητες ενός μοντέλου απαιτεί τα έξεις βήματα:

- Η εύρεση μοντέλου ή η δημιουργία του.
- Η εισαγωγή (Import) του στο Unity3d.
- Επιλογή του μοντέλου από τα Assets του παραθύρου Project και στη συνέχεια στο Hierarchy επιλέγουμε το μενού Rig.
- Αμέσως μετά στην επιλογή Animation Type επιλέγουμε Humanoid, αυτό σημαίνει ότι το μοντέλο θα κάνει ανθρωπόμορφη κίνηση.
- Παρακάτω η επιλογή Avatar Definition δημιουργεί ένα Avatar για το εκάστοτε μοντέλο.
- Στη συνέχεια πατώντας το κουμπί Configure βλέπουμε την παρακάτω εικόνα. Σε αυτή τη φάση το μοντέλο είναι έτοιμο για την εφαρμογή Animation Clip.



Εικόνα 26: Δημιουργία Avatar

- Στη συνέχεια από το παράθυρο Project πατώντας δεξί click δημιουργούμε έναν Animator Controller ο οποίος δεν περιέχει τίποτα προς το παρόν, αυτό που χρειάζεται να γίνει είναι να κάνουμε import μερικά animation clips που θέλουμε να κάνει ο χαρακτήρας.

- Τέλος, επιλέγοντας το μοντέλο από τα Assets της σκηνής, πηγαίνουμε στις ιδιότητες του (Hierarchy) και στο πεδίο Controller κάνουμε Drag n Drop τον animator controller που μόλις δημιουργήσαμε.

- Πατώντας play πλέον μπορούμε να δούμε τον χαρακτήρα να κινείτε στο χώρο.

Animation Clips

Η δημιουργία animations είναι μια χρονοβόρα διαδικασία και πολλές φορές αρκετά δύσκολη να εφαρμοστεί ακριβώς όπως θέλουμε πάνω στο μοντέλο. Στη παρούσα πτυχιακή εργασία χρησιμοποιήθηκαν έτοιμα animations clips. (ACCAD , n.d.)

Το Asset Store του Unity3d παρέχει μια βιβλιοθήκη που περιέχει βασικά clips όπως είναι για παράδειγμα το περπάτημα, το τρέξιμο, άλματα και άλλα. Η βιβλιοθήκη λέγετε Raw Mocap Data και παρέχετε δωρεάν, στο διαδίκτυο επίσης υπάρχουν βιβλιοθήκες με κινήσεις που επίσης εφαρμόσαμε στα μοντέλα του παιχνιδιού και λέγετε Motion Capture Labs. Τα αρχεία αυτά έχουν κατάληξη .bvh, Biovision Hierarchy (BVH). Αναπτύχθηκε από τη Biovision εταιρεία παροχής υπηρεσιών καταγραφή κίνησης.

Η εισαγωγή των παραπάνω animation clips απαιτεί πρώτα την επεξεργασία τους στο Motion Builder όπως αναφέραμε στο υποκεφάλαιο του 4.3 Motion Builder. Η απευθείας εισαγωγή στο Unity3d δε δουλεύει.

5.3 Πείραμα Χρωμάτων

Σε αυτό το υποκεφάλαιο ασχολούμαστε με την εφαρμογή του ηλεκτροεγκεφαλογράφου στο παιχνίδι. Σκοπός είναι η αξιοποίηση των δεδομένων που μας προσφέρει η συσκευή για την δημιουργία επιπέδων δυσκολίας αλλά και την μελέτη κάποιων βασικών συμπερασμάτων τα οποία μπορούν να χρησιμοποιηθούν από κλάδους του μάρκετινγκ αλλά ακόμα και για ιατρικούς λόγους.

5.3.1 Τα Χρώματα και ο Άνθρωπος

Η αναγνώριση των χρωμάτων συνδέεται με βασικές λειτουργίες του εγκεφάλου. Το περιβάλλον του ατόμου είναι γεμάτο από χρώματα. Η φύση, από μόνη της, είναι ένας πίνακας γεμάτος από χρώματα, το κίτρινο του ήλιου, το μπλε της θάλασσας… Επίσης οι κοινωνικές επιρροές σε κάποια μέρη το μαύρο αντιπροσωπεύει το πένθος ενώ σε άλλα μέρη το λευκό. Συνεπώς, ο ανθρώπινος εγκέφαλος είναι «προγραμματισμένος» να αντιδρά στα χρώματα, τα οποία επηρεάζουν τόσο τις σκέψεις όσο και τα συναισθήματα του ατόμου. (Η Ψυχολογία των Χρωμάτων, n.d.)

Ενώ η αντίληψη του χρώματος είναι υποκειμενική, κάποιες χρωματικές επιρροές είναι κοινές παγκοσμίως. Τα χρώματα είναι από τα πιο δυνατά κομμάτια της μη λεκτικής επικοινωνίας. Μεταφέρουν, στιγμιαία, έννοιες και μηνύματα.

Η θεωρία των χρωμάτων είναι μια επιστήμη από μόνη της με τη μελέτη της επιρροής των χρωμάτων σε διαφορετικά άτομα, είτε αυτά είναι απομονωμένα ή αν ανήκουν σε μια ομάδα. Μερικές φορές, η αλλαγή της απόχρωσης ή του κορεσμού ενός χρώματος μπορεί να προκαλέσει εντελώς διαφορετικά συναισθήματα.

Η αντίδραση του χρώματος στην ψυχολογία μας είναι κληρονομική αλλά και αποτέλεσμα μάθησης. Εξαρτάται από διάφορους παράγοντες όπως είναι το φύλο, η ηλικία, η ευφυΐα και η μόρφωση. Επίσης η θερμοκρασία, το κλίμα, το κοινωνικοοικονομικό περιβάλλον και οι τοπικές παραδόσεις έχουν επίδραση στον τρόπο που αντιδρούμε στα χρώματα.

Όταν η κοινωνική μας θέση μεταβάλλεται οι προτιμήσεις μας στα χρώματα προσαρμόζονται στις καινούργιες συνθήκες ζωής μας. Το ίδιο συμβαίνει όταν αλλάζει το εισόδημα μας ή η επαγγελματική μας κατάσταση.

Κατά κανόνα όμως οι άνθρωποι αντιδρούν με προβλέψιμο τρόπο στα χρώματα. Τα

εργαστηριακά τεστ και η εμπείρα έχουν αποδείξει ότι υπάρχει ενέργεια στα χρώματα, η οποία επιδρά στην υγεία, την ευεξία, την απόδοση στην εργασία, στην ασφάλεια μας, στον τρόπο που αγοράζουμε καταναλωτικά αγαθά και υπηρεσίες.

Οι υπεύθυνοι του μάρκετινγκ χρησιμοποιούν τα χρώματα για να πουλούν διάφορα προϊόντα σε υψηλότερες τιμές. Η έρευνα έχει δείξει ότι σε μεγάλο βαθμό η αποδοχή του προϊόντος από τους καταναλωτές εξαρτάται από την αρχική εντύπωση που δίνει το χρώμα.

Σε μια δημοσκόπηση που έγινε οι καταναλωτές δοκίμασαν τον ίδιο καφέ από κουτιά διαφορετικού χρώματος και βρήκαν ότι είχε διαφορετική γεύση. Είπαν ότι ο καφές από το κίτρινο κουτί είχε πολύ αδύνατη γεύση, ενώ ο ίδιος καφές από ένα σκούρο καφετί κουτί ήταν ιδιαίτερα δυνατός. Από το κόκκινο κουτί ήταν «πλούσιος σε άρωμα» και από το μπλε κουτί ήταν «απαλός»

Τα χρώματα κάνουν τους καταναλωτές να προτιμούν ορισμένες μάρκες ή κατηγορίες εμπορευμάτων. Για παράδειγμα στο χώρο των αναψυκτικών, η πασίγνωστη σε όλους μάρκα cola δεν είναι τυχαίο που έχει κόκκινο χρώμα.

Οι χρωματικοί τόνοι της γης, όπως είναι το καφέ, το σκούρο πορτοκαλί και το χρυσαφί δείχνει στους καταναλωτές ένα φυσικό, χωρίς συντηρητικά προϊόν.

Το μπλε συνδέεται με τη σόδα, το αποβουτυρωμένο γάλα χαμηλής περιεκτικότητας σε θερμίδες και το τυρί cottage. Το πράσινο, με τα λαχανικά και την τσίχλα, ενώ είναι ακατάλληλο για ζυμαρικά ή κρέατα επειδή θυμίζει στους υποψήφιους καταναλωτές τη μούχλα.

Αναλύοντας τα παραπάνω κατανοούμε την αναγκαιότητα μελέτης της εγκεφαλικής δραστηριότητας των ανθρώπων στα χρώματα.

5.3.2 Ορισμός Πειράματος

Το πείραμα σχετίζετε με την τυχαία απεικόνιση συγκεκριμένων μεμονωμένων χρωμάτων που βρίσκονται στο παιχνίδι και τις μετρήσεις από τα ερεθίσματα (attention/meditation) που προκαλούν σε έναν αριθμό ατόμων σε διαφορετικές ώρες της μέρας και σαφώς διαφορετικών ψυχολογικών διαθέσεων παραδείγματος χάριν κούραση, χαρά κ.λπ.

Τα χρώματα που χρησιμοποιήθηκαν είναι 5. Η επιλογή τους έγινε με βάση την ψυχολογική επίδραση που δημιουργούν.

Αναλυτικά παρουσιάζονται τα χρώματα και οι επιδράσεις που έχουν στον άνθρωπο:

Χρώμα	Επίδραση
κόκκινο	Το κόκκινο είναι ένα ιδιαιτέρα δυνατό χρώμα που διεγείρει και ενεργοποιεί το σώμα, αυξάνοντας τον καρδιακό ρυθμό, την αρτηριακή πίεση και την αναπνοή. Μελέτες δείχνουν ότι το κόκκινο αυξάνει την αθλητική ικανότητα, αλλά επίσης έχει συνδεθεί με αυξημένη επιθετικότητα, με αδυναμία συγκέντρωσης, ακόμη και με πονοκέφαλο. Η έκθεση σε κόκκινο χρώμα έχει βρεθεί να επηρεάζει αρνητικά τις σχολικές και ακαδημαϊκές επιδόσεις των παιδιών.
καφέ	Το φιλικό και φιλόξενο καφέ, έχει έναν ιδιαίτερο κοινωνικό χαρακτήρα γιατί εμπνέει τη διαπροσωπική επικοινωνία και κάνει τους ανθρώπους να νιώθουν άνετα. Όπως και το κίτρινο, το πολύ πορτοκαλί μπορεί να προκαλέσει υπερδιέγερση για αυτό και πρέπει να χρησιμοποιείται με φειδώ.
πράσινο	Το πράσινο συμβολίζει τη φύση και επομένως συνδέεται με τη γαλήνη και την ηρεμία. Τα πράσινο έχει βρεθεί να σχετίζεται με την υγεία και την ευημερία. Έχει μια κατευναστική επίδραση στο μυαλό και το σώμα, μειώνει το άγχος και ενισχύει τη συγκέντρωση. Έρευνες έχουν δείξει πως η έκθεση στο πράσινο χρώμα βελτιώνει την ικανότητα ανάγνωσης. Σε μελέτη βρέθηκε πως τοποθετώντας ένα διαφανές πράσινο φύλλο πάνω σε ένα κείμενο, βελτιώνεται η ταχύτητα ανάγνωσης και η κατανόηση του κειμένου.
μπλε	Το μπλε, είναι το ακριβώς αντίθετο από το κόκκινο. Ηρεμεί το μυαλό και το σώμα, μειώνει την αρτηριακή πίεση, τον καρδιακό ρυθμό, την αναπνοή, τα συναισθήματα άγχους και επιθετικότητας. Το μπλε περιβάλλον, μπορεί να βοηθήσει τα παιδιά που έχουν προβλήματα ύπνου ή είναι επιρρεπή σε ξεσπάσματα. Σκεφτείτε τη θετική επίδραση που έχει η όψη του γαλάζιου ουρανού και της θάλασσας. Παρόλα αυτά όμως, το σκούρο μπλε μπορεί να θυμίσει επικείμενη καταιγίδα και να προκαλέσει αρνητικά συναισθήματα.
μαύρο	Το μαύρο είναι ένα ουδέτερο χρώμα το οποίο δεν προκαλεί συγκεκριμένη ψυχολογική διάθεση στους ανθρώπους όμως αυξάνει την ήδη υπάρχουσα.

Πίνακας 17: Πίνακας Χρωμάτων

Το δείγμα ανθρώπων που χρησιμοποιήθηκαν είναι άτομα ηλικιών 20-30, διαφορετικής κοινωνικής καταγωγής και επιρροής. Λόγω έλλειψης υλικού το δείγμα ανθρώπων περιορίστηκε στα 10 άτομα. Ωστόσο η μελέτη μπορεί να θεωρηθεί ακριβής αφού τα μέσα και ο τρόπος υλοποίησης της έρευνας έγιναν με έγκυρες επιστημονικές μεθόδους.

5.3.3 Μέθοδος Υλοποίησης Πειράματος

Για την υλοποίηση αυτού του πειράματος χρησιμοποιήθηκαν αλγόριθμοι από τον κλάδο της Αναγνώρισης Προτύπων κάποιες στατιστικές μεθόδους για την υποβοήθηση των προηγούμενων αλγορίθμων ώστε να επιτευχθεί το καλύτερο δυνατό αποτέλεσμα.

Η Αναγνώριση Προτύπων (Pattern Recognition), ασχολείται με την ανάπτυξη αλγορίθμων για την αυτοματοποιημένη απόδοση κάποιας τιμής ή διακριτικού στοιχείου σε εισαγόμενα δεδομένα, συνήθως είναι κωδικοποιημένα ως αλληλουχίες αριθμών. Κατ' αυτόν τον τρόπο, ενδεικτικά, τα δεδομένα αυτόματα ταξινομούνται σε κατηγορίες ή διαχωρίζονται σε ομάδες με βάση κάποια κριτήρια, ακόμα και υπό την παρουσία θορύβου ο οποίος δυσκολεύει την αναγνώριση, ωθώντας συνήθως τα δεδομένα να μοιάζουν περισσότερο τυχαία απ' όσο πραγματικά είναι.

Τις περισσότερες φορές τα σύνολα δεδομένων χρειάζονται να αναλυθούν, για να αναγνωρίσουμε το είδος της πληροφορίας που περιέχουν. Στη συγκεκριμένη περίπτωση δε χρειάστηκε μεγάλη προεπεξεργασία δεδομένων εφόσον η ίδια η συσκευή μας προσφέρει πολύ καλά αποτελέσματα. Σε κάποιες περιπτώσεις χρειάστηκε να αντιμετωπιστεί το πρόβλημα των outliers. Λέγοντας outliers εννοούμε μερικές λανθασμένες τιμές που μπορούν να απομακρύνουν το μέσο όρο από μια αντιπροσωπευτική τιμή κατηγορίας. Τέτοιες τιμές απομακρύνθηκαν στο στάδιο της προεπεξεργασίας.

Γνωρίζοντας ότι κάθε πρότυπο εκπροσωπείται από τις χαρακτηριστικές τιμές του οι οποίες είναι κατα βάση αριθμητικές, μπορούν να χρησιμοποιηθούν από ένα σύστημα αναγνώρισης προτύπων προκειμένου να γίνει ο χωρισμός των προτύπων σε κλάσεις (ομάδες).

Τα μέτρα διασποράς που χρησιμοποιήθηκαν για την αφαίρεση των outliers είναι ο Mean Absolute Deviation που περιγραφει το Μέσο όρο των αποστάσεων από τον μέσο. Όσον αφορά τις στατιστικές, η μέση απόλυτη απόκλιση (MAD) είναι ένας ισχυρός δείκτης της μεταβλητότητας μιας μονο μεταβλητής.

Επίσης ένα ακόμα σημαντικό μέτρο διασποράς που χρησιμοποιήθηκε είναι η Τυπική απόκλιση ,το οποιο είναι ένα ευρέως χρησιμοποιούμενο μέτρο διασποράς.

K- Means

Για την επίλυση του προβλήματος χρησιμοποιήθηκε ο αλγόριθμος K- Means ο οποίος στοχεύει στη κατανομή των δεδομένων σε ομάδες, η κάθε ομάδα ανήκει στη περιοχή με το πλησιέστερο μέσο. Ο αλγόριθμος k-means (k-μέσων) είναι ένας αλγόριθμος (MacQueen, 1967) που ομαδοποιεί αντικείμενα βάσει των χαρακτηριστικών των k μεριδίων. Αποτελεί μεταβλητή του αλγόριθμου μεγιστοποίησης αναμονής (expectation-maximization algorithm-EM), όπου σκοπός είναι να οριστεί ο k-means δεδομένων που προήλθαν από Gaussian κατανομές. Ο αλγόριθμος υποθέτει ότι τα χαρακτηριστικά του αντικειμένου δημιουργούν ένα χώρο διανυσμάτων και ο σκοπός του είναι να ελαχιστοποιήσει τη συνολική διακύμανση της ομάδας ή τη συνάρτηση τετραγωνικού σφάλματος.

Τα βασικά βήματα του αλγόριθμου είναι τα εξής:

- Επιλογή του αριθμού των ομάδων.
- Τυχαία δημιουργία k ομάδων και ορισμός των κεντροειδών των ομάδων.
- Μεταβίβαση του κάθε σημείου στο κεντροειδές της κοντινότερης ομάδας.
- Υπολογισμός των νέων κεντροειδών των ομάδων.
- Επανάληψη μέχρι να συγκλίνει ο αλγόριθμος σε κάποιο κριτήριο.

Ο αλγόριθμος ξεκινά διαχωρίζοντας τα αρχικά σημεία σε k αρχικά σύνολα είτε τυχαία είτε χρησιμοποιώντας ευριτικά δεδομένα. Στη συνέχεια υπολογίζει το μεσαίο ή το κεντροειδές του κάθε συνόλου, υλοποιεί νέο διαχωρισμό ώστε το κάθε σημείο να σχετίζεται με το κοντινότερο κεντροειδές. Έπειτα τα κεντροειδή ξαναϋπολογίζονται για τις νέες ομάδες, ο αλγόριθμος επαναλαμβάνει τα δυο βήματα ωσότου τα σημεία δεν μπορούν να αλλάξουν ομάδες (ή εναλλακτικά τα κεντροειδή παραμένουν αμετάβλητα).

Ο αλγόριθμος αυτός παραμένει διάσημος επειδή τείνει σε κάποιο όριο πολύ γρήγορα. Όσον αφορά την απόδοση ο αλγόριθμος δεν εγγυάται ότι θα αγγίξει το βέλτιστο. Η ποιότητα της τελική λύσης εξαρτάται πολύ από το αρχικό σύνολο ομάδων και μπορεί να είναι πολύ χαμηλότερη από το συνολικό βέλτιστο.

Για την σύγκριση των αποτελεσμάτων του k-Means χρησιμοποιήθηκε και ένας ακόμα αλγόριθμος ο οποίος αν και το μαθηματικό μοντέλο του μας προτρέπει για καλύτερα αποτελέσματα δυστυχώς λόγο έλλιψης αρκετών δειγμάτων δεν λειτούργησε όπως θα έπρεπε.

Fuzzy C-means

Ο αλγόριθμος αυτός είναι μια μέθοδος ομαδοποίησης που επιτρέπει τα δεδομένα να ανήκουν σε περισσότερες από μια ομάδες. Αυτή η μέθοδος (που αναπτύχθηκε αρχικά από τον Dunn (1973) και βελτιώθηκε από τον Bezdek (1981)) χρησιμοποιείται συχνά στην

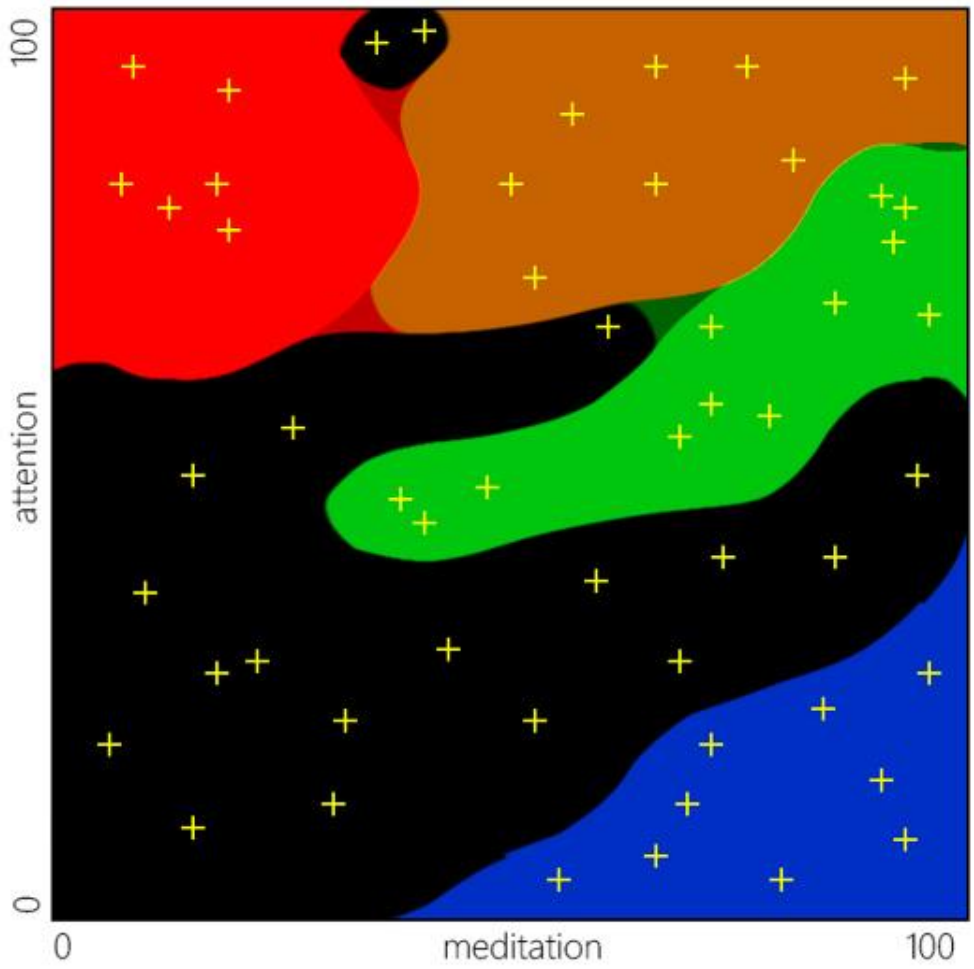
αναγνώριση προτύπων. Τα δεδομένα μεταπηδούν σε κάθε ομάδα βάσει μιας συνάρτησης μέλους που απεικονίζει τη fuzzy συμπεριφορά του αλγόριθμου.

Μια ιδιαίτερη χρήση των αποτελεσμάτων του πειράματος θα ήταν η κλιμακωτή αύξηση δυσκολίας των επιπέδων του παιχνιδιού.

Πιο συγκεκριμένα τα πρώτα επίπεδα θα μπορούν να είναι πιο εύκολα στον χρήση υποβοηθώντας τον να συγκεντρωθεί ώστε η εύρεση των αντικειμένων στο χώρο να γίνεται με σχετική ευκολία. Καθώς ο χρήστης ανεβαίνει επίπεδα θα μπορούσαμε να δυσκολεύουμε την εύρεση αντικειμένων εμφανίζοντας χρώματα όπου θα του αποσπούσαν την προσοχή.

5.3.4 Αποτελέσματα Πειράματος

Τα αποτελέσματα του πειράματος εμφανίζονται στην παρακάτω εικόνα, όπου παρατηρούνται τα πρότυπα (άνθρωποι) και τα χρώματα του πειράματος:



Εικόνα 27: Χάρτης Χρωμάτων

Κάθε φορά που κάποιος κοιτούσε:

- Το κόκκινο χρώμα η τιμή του Attention ήταν υψηλή και η τιμή του meditation ήταν χαμηλή.
- Το καφέ χρώμα η τιμή του attention ήταν υψηλή αλλά και η τιμή του meditation.
- Το πράσινο χρώμα η τιμή του attention κυμαίνεται σε μεσαία επίπεδα, η τιμή του meditation είχε υψηλά επίπεδα.
- Το μαύρο χρώμα κατείχε μεσαίες τιμές και στις δυο χαρακτηριστικές τιμές.
- Το μπλε χρώμα είχε χαμηλές τιμές attention και υψηλές τιμές του meditation.

Ο συνδυασμός αυτός παρατηρήθηκε σε όλα τα άτομα με μικρές αποκλίσεις, χωρίς βέβαια να αναιρούμε το γεγονός πως υπήρχαν και αρκετές περιπτώσεις outliers. Αυτό συνέβη γιατί η συσκευή έχανε το σήμα, δεν είχε καλή επαφή με το μέτωπο των ατόμων, όπως επίσης ο χρήστης απομακρυνόταν αρκετά από τη συσκευή Bluetooth με αποτέλεσμα να χάνει το σήμα.

Μια ιδιαίτερη χρήση των αποτελεσμάτων του πειράματος θα ήταν η κλιμακωτή αύξηση δυσκολίας των επιπέδων του παιχνιδιού.

Πιο συγκεκριμένα τα πρώτα επίπεδα θα μπορούν να είναι πιο εύκολα στον χρήση υποβοηθώντας τον να συγκεντρωθεί ώστε η εύρεση των αντικειμένων στο χώρο να γίνεται με σχετική ευκολία. Καθώς ο χρήστης ανεβαίνει επίπεδα θα μπορούσαμε να δυσκολεύουμε την εύρεση αντικειμένων εμφανίζοντας χρώματα όπου θα του αποσπούσαν την προσοχή.

5.3.5 Επιπλέον Λειτουργίες Εγκεφαλογράφου Στο Παιχνίδι

Μια ακόμη προσέγγιση έχει να κάνει ως επιτοπλείστον με τις τιμές του attention και meditation που προέρχονται από τη συσκευή και τη χρήση τους ώστε να επηρεάσουν κάποιο αντικείμενο στο χώρο της σκηνής.

Έχοντας αυτή τη δυνατότητα κατασκευάστηκαν εργαλεία με τα οποία ο χρήστης μπορεί να ανιχνεύσει πιο ευκολά κρυμμένα αντικείμενα που βρίσκονται στο χώρο της πίστας και είναι μείζονος σημασίας για τη λύση μιας υπόθεσης. Τα αντικείμενα αυτά είναι άμεσα συνδεδεμένα με το attention που έχει ο χρήστης την εκάστοτε στιγμή. Όσο πιο μεγάλες τιμές έχει τόσο πιο εμφανή είναι τα αντικείμενα εάν προφανώς βρίσκετε κοντά σε αυτά. Δε μπορεί παραδείγματος χάριν να τα εντοπίσει εάν βρίσκετε σε άλλο σημείο της πίστας (Εικόνες 32 και 33).

Μια ακόμη χρήση της συσκευής είναι η δυνατότητα της εκκίνησης ενός χρονομέτρου (Timer) το οποίο θα τρέχει πιο γρήγορα ή πιο αργά ανάλογα με το meditation του χρήστη. Έτσι εάν τα επίπεδα είναι χαμηλά το χρονόμετρο θα τρέχει γρηγορότερα σε αντίθετη περίπτωση θα κυλάει πιο αργά. Τέλος, μια ακόμη λειτουργία με τη χρήση των μετρήσεων που αναφέρονται στο meditation είναι ο επηρεασμός των ιδιοτήτων αντικειμένων θα επηρεάζεται σε μερικά αντικείμενα το Opacity τους με αυτό τον τρόπο θα γίνονται ορατά και μη (Εικόνα 30).

6: Αποτελέσματα

Στο παρόν κεφάλαιο αναλύονται τα αποτελέσματα της πτυχιακής εργασίας και θα παρουσιαστούν διάφορες μελλοντικές αναβαθμίσεις που θα μπορούσαν να γίνουν ώστε το συγκεκριμένο ηλεκτρονικό παιχνίδι να πλησιάζει τα πλαίσια εικονικής πραγματικότητας.

Στην εργασία επιτευχθήκαν οι αρχικοί στόχοι που τέθηκαν ως ερωτήματα όπως είναι η σωστή λειτουργία του παιχνιδιού, η δημιουργία όμορφου περιβάλλοντος που ο χρήστης θα διασκεδάσει και θα περάσει ευχάριστα το χρόνο του και εκτός από αυτό θέλουμε να απευθύνετε σε άτομα όλων των ηλικιών.

Ένας ακόμη στόχος μελλοντικής επέκτασης του παιχνιδιού είναι ο χρήστης να είναι σε θέση να ελέγχει το παιχνίδι εξολοκλήρου χωρίς τη χρήση πληκτρολογίου ή joystick αλλά με τη δραστηριότητα του εγκεφάλου και κινήσεων.

Πιο συγκεκριμένα επιτευχθήκαν:

- Το αποτέλεσμα της εργασίας πλησιάζει τα πλαίσια της επαυξημένης πραγματικότητας με ένα καλύτερο εξοπλισμό.

- Η χρήση του πειράματος χρωμάτων εφαρμόστηκε με επιτυχία και απέδωσε το συμπέρασμα ότι τα θερμά χρώματα αυξάνουν την προσοχή και αντιθέτως τα πιο ψυχρά τη χαλάρωση.

- Ένα τέτοιου είδους παιχνίδι μπορεί να έχει εκπαιδευτικό χαρακτήρα για τη χρήση σε άτομα με μειωμένη προσοχή.

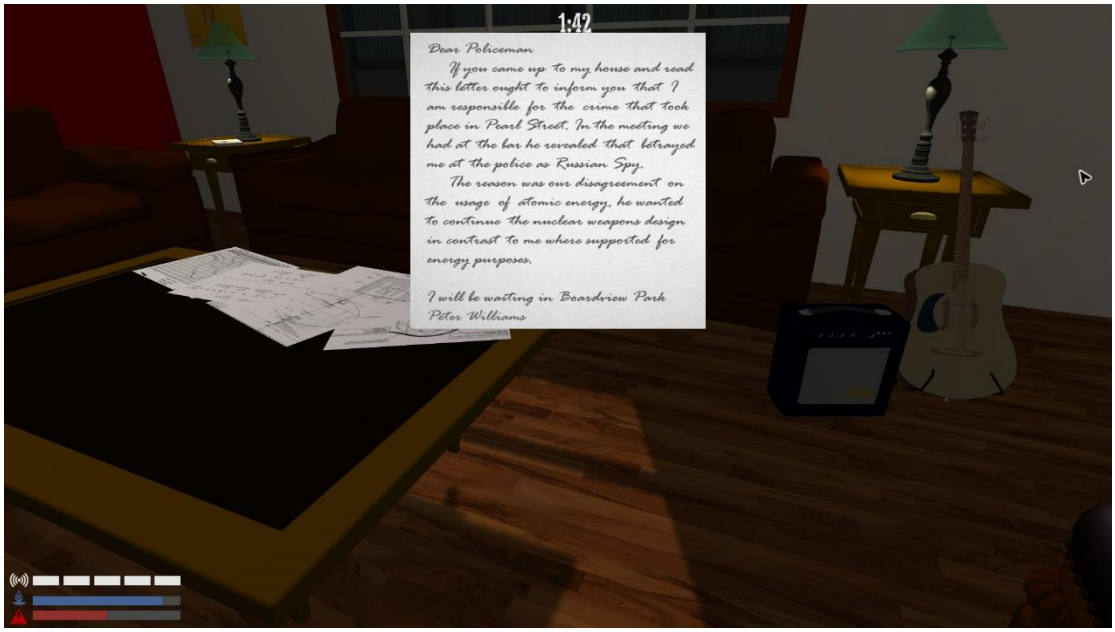
- Επιπλέον, μπορεί να συμβάλει στη βελτίωση ανθρώπων με πραγματικά προβλήματα συγκέντρωσης και την εκπαίδευσή τους για τη βελτίωση του προβλήματος.

6.1 Παρουσίαση Δείγματος Λειτουργίας

Στο σημείο αυτό θα παρουσιαστούν μερικές εικόνες από τη λειτουργία του παιχνιδιού και τα εργαλεία που χρησιμοποιεί ο παίκτης για τη λύση του προβλήματος.

Εικόνα 28: City Scene –guide message

Εικόνα 29: Door Trigger



Εικόνα 30: Display Time and Element by inventory

Εικόνα 31: Dialogue System

Εικόνα 32: Attention Glow (Full Signal)

Εικόνα 333: Attention Glow (Medium Signal)

6.2 Μελλοντικές Επεκτάσεις Και Έρευνα

Το πώς παρουσιάζεται ένα ηλεκτρονικό παιχνίδι όσο αφορά τον τομέα των γραφικών είναι ένα πολύ σημαντικό κεφάλαιο γιατί επηρεάζει άμεσα το πώς αντιλαμβάνεται ο παίκτης το παιχνίδι. Βέβαια δεν πρέπει να συγχέονται τα γραφικά του παιχνιδιού με την αισθητική. Τα γραφικά αφορούν στο πόσο καλά αναπαρίσταται ο κόσμος όσο αφορά κυρίως την γεωμετρία του κόσμου και των αντικειμένων. Η αισθητική είναι επίσης, ίσως και ακόμα πιο σημαντική, από την σωστή γραφική αναπαράσταση.

Όσον αφορά τα γραφικά και τα μοντέλα, αποκτώντας παραπάνω εμπειρία στον τομέα του 3d modeling τα μοντέλα και τα αντικείμενα θα μπορούν να αναπαρασταθούν με πολύ μεγαλύτερη λεπτομέρεια, κάτι το οποίο είναι απαραίτητο για ένα παιχνίδι με εκπαιδευτικό προσανατολισμό. Όσον αφορά την αισθητική, τα φίλτρα που εφαρμόστηκαν στην αρχική σκηνή του παιχνιδιού δεν μπορούν να εφαρμοστούν και στο υπόλοιπο παιχνίδι λόγο ελλιπούς υπολογιστικής ισχύος.

Ο τομέας του sound design σε ένα παιχνίδι είναι σίγουρα από τους πιο εξεζητημένους και απαιτητικούς. Για να αποκτήσει ένα παιχνίδι τον δικό του χαρακτήρα χρειάζεται σίγουρα κάποιον συνθέτη που θα ασχοληθεί εξολοκλήρου με το soundtrack και με τα ηχητικά εφέ που θα ζωντανέψουν τον κόσμο του παιχνιδιού. Ο επαγγελματισμός στον ήχο κάνει τις περισσότερες φορές την διαφορά σε κάποιο ηλεκτρονικό παιχνίδι. Ένα καλοστημένο soundtrack, καθαρά ηχογραφημένα ηχητικά εφέ και δυναμικός ήχος ο οποίος μεταβάλλεται ανάλογα με το παιχνίδι θα δώσει στο παιχνίδι τον επαγγελματικό αέρα που χρειάζεται για να ξεχωρίσει. Όμως και εδώ η έλλειψη κατάλληλου υλικού εξοπλισμού μπορεί πολύ εύκολα να σταθεί εμπόδιο.

Συνοψίζοντας βλέπουμε πως όλοι οι τομείς του παιχνιδιού μπορούν να επωφεληθούν από την προσθήκη παραπάνω λεπτομερειών, όπως μηχανισμοί χρόνου και πόντων που θα προσθέσουν replayability και θα δώσουν στον παίκτη την επιθυμία να ξεκλειδώσει όλες τις λεπτομέρειες.

Όσον αφορά τα γραφικά την αισθητική και τον ήχο, η έλλειψη κατάλληλου εξοπλισμού εμφανίζει σημαντικά εμπόδια. Τέλος το παιχνίδι μπορεί να επωφεληθεί, όχι μόνο από ένα ολοκληρωμένο σενάριο, αλλά κάποιους βοηθητικούς χαρακτήρες αλλά και μικρά mini games τα οποία θα ζωντανέψουν το παιχνίδι.

Η χρήση των βίο-αισθητήρων και την ενσωμάτωση τους σε εφαρμογές όπως είναι τα παιχνίδια μπορούν να προσφέρουν ένα άλλο είδος ψυχαγωγίας πολύ πιο ποιοτικό, διασκεδαστικό, αλλά μπορούν επίσης να βελτιώσουν τις επιδώσεις συγκέντρωσης και χαλάρωσης. Μια επιπλέον επέκταση που θα μπορούσε να εισαχθεί στην υλοποίηση είναι η χρήση του Kinect, ο συνδυασμός των δυο αυτών συσκευών μπορεί να πλαισιώσει τον ορισμό της επαυξημένης πραγματικότητας.

7: Βιβλιογραφία

ACCAD . (n.d.). (Motion Capture Lab) Ανάκτηση 2015, από http://accad.osu.edu/research/mocap/mocap_data.htm

Augmented reality. (n.d.). Ανάκτηση 2015, από https://en.wikipedia.org/wiki/Augmented_reality

Berkebile, B. (n.d.). *iTween for Unity* . (Pixelplacement) Ανάκτηση 2015, από http://itween.pixelplacement.com/index.php

Eck, R. V. (2006). *Digital Game- Based Learning - It's Not Just the Digital Natives Who Are Restless.*

Electroencephalography. (n.d.). (Wikipedia) Ανάκτηση 2015, από https://en.wikipedia.org/wiki/Electroencephalography

Emmanuel John L. Bagacina, J. R. (2013). *Peripheral Control Using EEG Signals and Facial Artifacts.* Philippines: Department of Electronics, Computer and Communications.

First video game. (n.d.). (wikipedia) Ανάκτηση 2015, από https://en.wikipedia.org/wiki/First_video_game

Game Engine. (n.d.). (wikipedia) Ανάκτηση 2015, από https://en.wikipedia.org/wiki/Game_engine

Gibson, J. (2014). *Introduction to Game Design, Prototyping, and Development: From Concept to Playable Game with Unity and C#.*

Hunt, W. (2014). *Electroencephalography (EEG) Headset Brain - Computer Interface (BCI).*

Konzack, L. (2002). *Computer Game Criticism: A Method for Computer Game Analysis.*

Mindwave. (n.d.). (Neurosky) Ανάκτηση 2015, από http://store.neurosky.com/products/mindwave-1

Rollings A., A. E. (2006). *Fundamentals of Game Design.*

Salabun, W. (2014). *Processing and spectral analysis of the raw EEG signal from the MindWave.* West Pomeranian University of Technology, Szczecin.

Thinkgear Communications Protocol [NeuroSky Developer - Docs]. (n.d.). (Mindwave) Ανάκτηση από http://developer.neurosky.com/docs/doku.php?id=thinkgear_communications_protocol

Unity - Game Engine. (n.d.). (Unity3D) Ανάκτηση 2015, από https://unity3d.com/

Unity Manual. (n.d.). (Unity3D) Ανάκτηση 2015, από http://docs.unity3d.com/Manual/

Η Ψυχολογία των Χρωμάτων. (n.d.). (Συμβουλευτικο Κεντρο Επαγγελματικου Προσανατολισμου & Ψυχικης Υγειας) Ανάκτηση 2015, από http://www.skepsy.gr/index.php/el/articles/138-i-psychologia-twn-xrwmatwn