ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ
ΣΧΟΛΗ ΟΙΚΟΝΟΜΙΑΣ, ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Εργαστήριο Γνώσης και Αβεβαιότητας

Πτυχιακή εργασία

Ανάπτυξη παιχνιδιού για το Google Cardboard

Ιάσων Ζορμπάς 2025201200033

Φώτης Φωτόπουλος 2025201200115

Επιβλέπων:

Γουάλλες Εμμανουήλ
Επίκουρος καθηγητής

Τρίπολη, Μάρτιος 2018

Εγκρίθηκε από την εξεταστική επιτροπή την 15η Μαρτίου 2018.

Εμμανουήλ Γουάλλες	Γεώργιος Λέπουρας
Επίκουρος καθηγητής	Καθηγητής

.....

Φώτης Φωτόπουλος, Ιάσων Ζορμπάς
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
Πανεπιστήμιο Πελοποννήσου

Η παρούσα εργασία εκπονήθηκε σε συνεργασία με το Εργαστήριο Γνώσης και Αβεβαιότητας (ΓΑΒ LAB)

Περίληψη

Στην εργασία αυτή γίνεται μια ολοκληρωμένη καταγραφή των απαραίτητων βημάτων για την δημιουργία ενός παιχνιδιού εικονικής πραγματικότητας (VR), συγκεκριμένα για mobile λειτουργικά συστήματα (Android). Επίσης, γίνεται η παρουσίαση και επεξήγηση της χρήσης του VR Pac-Man. Πιο αναλυτικά, αρχικά παρουσιάζεται βιβλιογραφία που αφορά μια ιστορική αναδρομή για τη συγκεκριμένη τεχνολογία, πώς έχει εξελιχθεί σήμερα και ποια είναι η χρησιμότητα ενός τέτοιου συστήματος καθώς και σε ποιους τομείς έχει εφαρμογή. Έπειτα, παρουσιάζεται ένας οδηγός ο οποίος εξηγεί αναλυτικά πως λειτουργεί μια VR εφαρμογή, ποια είναι τα απαραίτητα εργαλεία για την δημιουργία της και πως μια τέτοια εφαρμογή μπορεί να εξελιχθεί στο μέλλον. Ακόμα, γίνεται παρουσίαση του VR Pac-Man, εφαρμογή που αναπτύχθηκε με την χρήση αυτών των εργαλείων ειδικά για το μέσο προβολής VR, Google Cardboard. Τέλος, παρουσιάζονται τα συμπεράσματα που προκύπτουν από τη μελέτη και τη δημιουργία του VR Pac-Man καθώς και μερικές βελτιώσεις που μπορούν να υλοποιηθούν στο μέλλον.

Abstract

This thesis makes a comprehensive record of the steps necessary to create a VR game, namely for mobile operating systems (Android). In addition, VR Pac-Man is presented and explained in the following chapters. In more detail, a literature on a historical review of this technology is presented, how it has evolved today, and what is the usefulness of such a system as well as in which areas it is applicable. Then there is a guide that explains in detail how a VR application works, which tools are necessary for its creation and how such an application can evolve in the future. Concluding, there is a presentation of VR Pac-Man, an application developed using these tools, specifically for the VR headset, Google Cardboard. Finally, hereby we present the findings that resulted from the study and creation of VR Pac-Man as well as some enhancements that can be applied in the future.

To all those who lead monotonous lives
in the hope that they may experience through Virtual Reality
the delights and dangers of adventure.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

Η εικονική πραγματικότητα (Virtual Reality - VR) είναι ένα μέσο το οποίο αποτελείται από αλληλεπιδραστικές εξομοιώσεις με υπολογιστή, οι οποίες ”αισθάνονται” την θέση και τις ενέργειες του χρήστη και αντικαθιστούν ή επαυξάνουν την ανάδραση σε μία ή παραπάνω αισθήσεις, δίνοντας το αίσθημα της εμβύθισης ή παρουσίας στην εξομοίωση (ένας εικονικός κόσμος). Με την πρόοδο της τεχνολογίας έχουν δημιουργηθεί πολλά μέσα προβολής εικονικής πραγματικότητας και μέθοδοι ανάπτυξης εφαρμογών συμβατών με αυτά, δίνοντας έτσι την δυνατότητα σε πολλούς ανθρώπους να έρθουν πιο κοντά με την τεχνολογία αυτή.

Η εικονική πραγματικότητα αποτελεί πλέων ένα σημείο τομής μεταξύ πολλών επιστημών. Τομείς όπως η ιατρική, η αεροπορία, οι εκπαιδεύσεις αστροναυτών και πολλών άλλων επαγγελμάτων χρησιμοποιούν σε καθημερινή βάση εργαλεία εικονικής πραγματικότητας. Ένας άλλος τομέας που έχει απείχες στην εικονική πραγματικότητα είναι αυτός των ηλεκτρονικών παιχνιδιών που βρίσκουν εφαρμογή στα συνεχώς εξελισσόμενα μέσα προβολής εικονικής πραγματικότητας. Μέσα όπως το Google Cardboard, Oculus Rift κ.α. είναι εύκολα αποκτήσιμα και με μια τεράστια λίστα παιχνιδιών εικονικής πραγματικότητας σε κάθε ηλεκτρονικό κατάστημα η αυξανόμενη προσοχή που λαμβάνει ο τομέας αυτός είναι αναμενόμενη.

Για να είναι όσο πιο πετυχημένη γίνεται η εμβύθιση ενός χρήστη σε ένα περιβάλλον εικονικής πραγματικότητας, είναι σημαντικό να απομονωθεί ο χρήστης και οι αισθήσεις του από το πραγματικό κόσμο, επικαλύπτοντας τα ερεθίσματα του πραγματικού κόσμου με αντίστοιχα εικονικά, φτιαγμένα από το σύστημα της Εικονικής Πραγματικότητας. Από τις πέντε (ή μήπως εφτά) αισθήσεις, οι πιο σημαντικές κατά φθίνουσα σειρά είναι η όραση, η ακοή και η αφή. Έτσι είναι πρωταρχικής σημασίας ένα σύστημα Εικονικής Πραγματικότητας να παρέχει στερεοσκοπική εικόνα, δηλαδή δύο εικόνες από διαφορετική οπτική γωνία, μία για κάθε μάτι του χρήστη, έτσι ώστε να δημιουργηθεί η αίσθηση του βάθους στο χώρο. Παράλληλα η ύπαρξη στερεοσκοπικού ήχου βοηθάει το χρήστη να κατανοεί τι γίνεται γύρω του στον εικονικό χώρο που τον περιβάλλει με πολύ φυσικό τρόπο, ενώ ταυτόχρονα αποκλείει τον χρήστη από τους ήχους του πραγματικού κόσμου, οι οποίοι θα μπορούσαν να καταστρέψουν την εικονική του εμπειρία. Τέλος η αφή, μπορεί να χρησιμοποιηθεί με κατάλληλες συσκευές είτε για να μπορεί ο χρήστης να νιώθει τον κόσμο, π.χ. να ακουμπά ένα αντικείμενο και να νιώθει αντίσταση, είτε για να καθοδηγήσουμε το χρήστη διευκολύνοντάς τον στην εκτέλεση κάποιων συγκεκριμένων ενεργειών, π.χ. μοντελοποίηση τρισδιάστατων αντικειμένων. Αν όλα τα παραπάνω συνδυαστούν και με την ανίχνευση των κινήσεων του χρήστη με κατάλληλες συσκευές ανίχνευσης, έτσι ώστε το εικονικό περιβάλλον να συμπεριφέρεται όπως και το πραγματικό, τότε η όλη εμπειρία που θα αποκτήσει ο χρήστης μπορεί να είναι άκρως ρεαλιστική

Στην παρούσα πτυχιακή εργασία θα γίνει μια πλήρης αναφορά στην εικονική πραγματικότητα και τα μέσα προβολής της. Ο αναγνώστης θα περάσει πρώτα απτό θεωρητικό κομμάτι της εικονικής πραγματικότητας μέσα από μια σύντομη αλλά πλήρης ιστορική αναδρομή ενώ στη συνέχεια θα εφοδιαστεί με τα απαραίτητα εργαλεία και προγράμματα για την ανάπτυξη μιας τέτοιας εφαρμογής. Εν κατακλείδι θα του παρουσιαστεί το VR Pacman, το αποτέλεσμα της χρήσης των εργαλείων αυτών και ένα εγχειρίδιο με οδηγίες για την κατασκευή και την πλοήγηση σε αυτό.

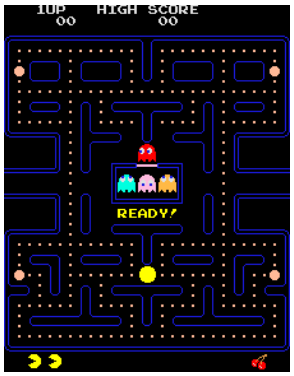
Στα κεφάλαια που ακολουθούν, θα περιγραφεί αναλυτικά ο όρος εικονική πραγματικότητα και θα παρουσιαστούν αναλυτικά βήματα για την υλοποίηση μιας VR εφαρμογής. Πιο συγκεκριμένα, στο κεφάλαιο 3 θα παρουσιαστούν θεωρητικά στοιχεία για την συγκεκριμένη τεχνολογία. Στο κεφάλαιο 4 θα αναφερθούν τα εργαλεία που χρησιμοποιήθηκαν στην συγκεκριμένη εργασία. Στο κεφάλαιο 5 θα παρουσιαστεί ο στόχος της εργασίας αυτής ενώ στη συνέχεια και στο κεφάλαιο 6 θα περιγραφεί ο τρόπος με τον οποίο αναπτύξαμε το παιχνίδι μας. Τέλος, στο κεφάλαιο 7 θα παρουσιαστεί το VR Pac-Man και ο τρόπος εγκατάστασης του και θα κλείσουμε με το κεφάλαιο 8 στο οποίο παρουσιάζουμε τα συμπεράσματα μας και σημειώνουμε πιθανές επεκτάσεις της εργασίας μας.

Κεφάλαιο 2

Pac-Man

Το Pac-Man, είναι ένα arcade παιχνίδι που αναπτύχθηκε από την εταιρεία Namco και κυκλοφόρησε για πρώτη φορά στην Ιαπωνία τον Μάιο του 1980. Δημιουργήθηκε από τον Ιάπωνα σχεδιαστή βιντεοπαιχνιδιών Toru Iwatani. Αφού εγκρίθηκε η άδεια για διανομή στις Ηνωμένες Πολιτείες από τους Midway Games, κυκλοφόρησε τον Οκτώβριο του 1980. Εξαιρετικά δημοφιλής από την αρχική κυκλοφορία του μέχρι σήμερα, το Pac-Man θεωρείται ένα από τα κλασικά βιντεοπαιχνίδια και θεωρείται μια εικόνα της λαϊκής κουλτούρας του 1980. Με την ανακοίνωση του, το παιχνίδι και στη συνέχεια, τα παράγωγα του, έγιναν ένα κοινωνικό φαινόμενο που οδήγησε σε υψηλές πωλήσεις εμπορευμάτων και ενέπνευσε μια κληρονομιά και σε άλλα μέσα, όπως η τηλεοπτική σειρά Pac-Man, δέκα Buckner κ.α.. Το Pac-Man ήταν δημοφιλής στη δεκαετία του ’80 και του ’90 και εξακολουθεί να παίζεται μέχρι σήμερα. [7]

Το παιχνίδι διαδραματίζεται σε ένα λαβύρινθο ορθογώνιου σχήματος στον οποίο ο παίκτης καλείται να καθοδηγήσει τον Pac-Man στο να ”φάει” όλες τις κίτρινες τελείες (Pac-Dots) που βρίσκονται στο μεγαλύτερο τμήμα του, αποφεύγοντας παράλληλα τα τέσσερα φαντάσματα 2.1 που έχουν ως σκοπό να αφαιρέσουν όλες τις ζωές που μένουν στον παίκτη αν ακουμπήσουν τον Pac-Man. Αν παρ’ όλα αυτά ο Pac-Man φάει μία απ’ τις μεγάλες τελείες που βρίσκονται στις τέσσερις γωνίες του λαβύρινθου (Power pills) τότε, για ένα μικρό χρονικό διάστημα τα φαντάσματα παίρνουν μπλε χρώμα και προσπαθούν να αποφύγουν τον Pac-Man, καθώς πλεον μπορεί να τα φάει αυτός. Αν ο παίκτης καταφέρει να ”φάει” όλες τις τελείες, προχωράει στον επόμενο λαβύρινθο.



Σχήμα 2.1: Pac-Man Game

Το παιχνίδι θεωρείται ως ένα από τα πιο επιδραστικά βιντεοπαιχνίδια όλων των εποχών, για αρκετούς λόγους: ο χαρακτήρας Pac-Man ήταν η πρώτη πρωτότυπη ”μασκότ” παιχνιδιού, το παιχνίδι καθιέρωσε το είδος παιχνιδιου *κυνήγι σε λαβύρινθο* , κατέδειξε το πόσο σημαντικοί είναι οι χαρακτήρες στα βιντεοπαιχνίδια, εισήγαγε τα βιντεοπαιχνίδια στα γυναικεία ακροατήρια και ήταν η πρώτη επιτυχία της αδειοδότησης των βιντεοπαιχνιδιών. Επιπλέον, ήταν το πρώτο βιντεοπαιχνίδι που περιείχε power-ups και τα επιμέρους φαντάσματα είχαν ντετερμινιστική τεχνητή νοημοσύνη ώστε να αντιδρούν στις ενέργειες των παικτών. [7]

2.1 Χαρακτήρες παιχνιδιού

Ο Pac-Man (σχ. 2.2) είναι ο πρωταγωνιστής του παιχνιδιού και ο χαρακτήρας που καθοδηγεί ο παίκτης. Ο στόχος του είναι να φάει όλες τις τελείες του λαβύρινθου αποφεύγοντας όλα τα φαντάσματα. Οι έλεγχοι του βασίζονται στις κινήσεις των χρηστών. Αριστερά, δεξιά, πάνω και κάτω. Ο Blinky (σχ. 2.3) ξεκινά κάθε επίπεδο προχωρώντας με την ίδια ταχύτητα με όλα τα άλλα φαντάσματα μέχρι ο Pac-Man να φάει ένα ορισμένο αριθμό κουκίδων όπου και αρχίζει να επιτυγχάνει. Ο Clyde (σχ. 2.4) είναι είτε τυφλός είτε ηλίθιος. Θα στριψεί συχνά στην αντίθετη κατεύθυνση αντί να προσεγγίσει τον Pac-Man. Φαίνεται σαν να μην τον ενδιαφέρει καθόλου η εξέλιξη του παιχνιδιού. Ο Inky (σχ. 2.5) είναι επικίνδυνος επειδή είναι απρόβλεπτος. Παίρνει συχνά διαφορετικές στροφές σε διαφορετικές χρονικές στιγμές. Μία θεωρία για την συμπεριφορά του Inky εξαρτάται από την σχέση του με τον Blinky σχεδόν σαν να φοβάται να δράσει μόνος του (όπως μερικοί άνθρωποι που δεν πηγαίνουν ποτέ στον κινηματογράφο μόνοι τους). Ο Pinky (σχ. 2.6) κινείται πάντα με την ίδια ταχύτητα που έχει ο Inky και ο Clyde, οπότε το δεύτερο ψευδόνυμο *Speedy* που του έχει δωθεί, είναι σαφώς παραπλανητικό. Ο Pinky φαίνεται να έχει την τάση να περνάει γύρω από τα μπλοκ του λαβίρυνθου κατά την αντίθετη φορά των δεικτών του ρολογιού, αντίθετα από τους Blinky και Clyde που φαίνεται να προτιμούν να πηγαίνουν δεξιόστροφα. Αυτό σημαίνει ότι αν ο Blinky και ο Pinky φτάσουν στην αντίθετη πλευρά ενός μπλοκ από όπου και αν βρίσκεται ο παίκτης, θα έρθουν σε αυτόν από τις αντίθετες πλευρές του. Πρόκειται για ένα θανάσιμο δίδυμο !

Σχήμα 2.2: Pac-Man

Σχήμα 2.3: Blinky

Σχήμα 2.4: Clyde

Σχήμα 2.5: Inky

Σχήμα 2.6: Pinky

Κεφάλαιο 3

Εικονική πραγματικότητα

Η Εικονική πραγματικότητα είναι η προσομοίωση ενός περιβάλλοντος από έναν υπολογιστή. Σε αυτό το κεφάλαιο θα εισάγουμε τον αναγνώστη στην έννοια της, θα προσπαθήσουμε να αναλύσουμε τους τρόπους με τους οποίους οι άνθρωποι κατάφεραν να δημιουργήσουν εικονικούς κόσμους ενώ ταυτόχρονα θα παραθέσουμε τους λογούς για τους οποίους η εικονική πραγματικότητα εξελίσσεται τόσο γρήγορα.

3.1 Ιστορική αναδρομή

Θεωρείται ότι η πρώτη προσπάθεια για την δημιουργία μιας μορφής εικονικής πραγματικότητας έγινε περίπου το 1860 [2], όταν καλλιτέχνες ξεκίνησαν να δημιουργούν ένα είδους 3D πανοραμικών τοιχογραφιών. Η σημερινή εικονική πραγματικότητα ωστόσο, είναι πολύ πιο εξελιγμένη και ίσως περισσότερο γνωστή στον κόσμο λόγω του τομέα των παιχνιδιών. Βρισκόμαστε πλέον στην εποχή που η εικονική πραγματικότητα βρίσκει εφαρμογές σε σημαντικούς για την ανθρωπότητα τομείς όπως η στρατιωτική εκπαίδευση και η θεραπεία διάφορων προβλημάτων υγείας. Στις σημερινές μέρες η τεχνολογία αυτή αρχίζει να αγγίζει τις δυνατότητες της. Παρακάτω θα δούμε ποιοί εξοπλισμοί είναι αυτοί που χρησιμοποιούνται περισσότερο στις μέρες μας και θα αναλύσουμε μερικά χαρακτηριστικά τους.

Η προσοχή της κοινής γνώμης για το VR, πέρα από κάποιες ελάχιστες αυξομειώσεις (View-Master, Virtuality, Sega VR, VFX1 Headgear), είχε εξαφανιστεί μέχρι πρόσφατα και συγκεκριμένα έως το 2012 που το ενδιαφέρον για την εικονική πραγματικότητα αυξήθηκε κατακόρυφα [1]. Το 2009, τρία χρονιά νωρίτερα, ο Palmer Luckey ένας επικεφαλής σχεδιαστής οθόνης μαζί με την εταιρία VR Aficionado, αποφάσισαν να χτίσουν απ' τήν αρχή ένα VR kit με χαμηλό κόστος και υψηλή απόδοση, χρησιμοποιώντας μια ενιαία οθόνη LCD. Έως το 2011 κατάφεραν να δημιουργήσουν το πρωτότυπο το οποία έπιασε την προσοχή του John Carmack, ιδρυτή της εταιρίας Id Software. Έτσι, δημιούργησαν μια νέα καμπάνια στο Kickstarter η οποία κατάφερε να χρηματοδοτηθεί με 2 εκατομμύρια δολάρια, αρχίζοντας έτσι το ταξίδι της εταιρίας που σήμερα είναι ευρέως γνωστή ως Oculus Rift. Μετά από έξι αναθεωρήσεις, η πρώτη έκδοση του Oculus Rift διαθέτει οθόνη OLED με ανάλυση 1080χ1200 ανά μάτι, ρυθμό ανανέωσης 90Hz, έξι βαθμούς παρακολούθησης κεφαλής και ενσωματωμένα ακουστικά με 3D ήχο. Από τον Μάιο του 2016 και μέχρι σήμερα είναι διαθέσιμο προς όλους, με την τιμή του να αγγίζει τα 600 δολάρια. [1]

Σχήμα 3.1: Oculus Rift

3.2 Google Cardboard

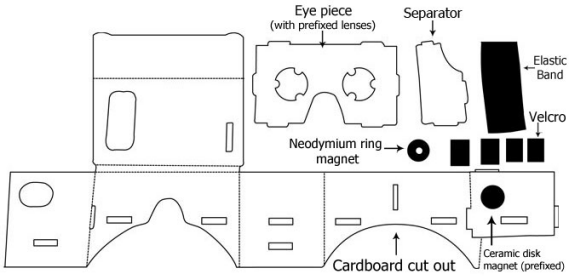
Σχήμα 3.2: Google Cardboard

Λίγο πριν την εκκίνηση του Oculus Rift στην ανοιχτή αγορά και συγκεκριμένα το καλοκαίρι του 2014 μια ομάδα Γάλλων προγραμματιστών της Google, παρουσίασαν στο συνέδριο της εταιρείας I/O το Google Cardboard. Ξεκίνησε σαν ”πλάκα” όταν δυο εργαζόμενοι της Google, ο David Coz και ο Damien Henry, αποφάσισαν να δημιουργήσουν μια πολύ φθηνή έκδοση του Oculus Rift ξοδεύοντας έτσι το δικό τους ”20 τις εκατό”, χρόνο από τις εργάσιμες ώρες τους που επιτρέπει η Google στους εργαζομένους της να αφιερώνουν στα προσωπικά τους προτζεκτ. Έχοντας πουλήσει πλέον πάνω από 10.000.000 κιτ στην τιμή των 20 δολαρίων το καθένα, το Google Cardboard, έχει ξεπεράσει κατά πολύ τις πωλήσεις του πιο υλικά εξελιγμένου και πολύ πιο ακριβού ”αδερφού” του έχοντας σχεδόν διπλάσιες πωλήσεις απ’ τό Oculus Rift σύμφωνα με τα δεδομένα της 1ης Μαΐου του 2017. Εκτός από τις πωλήσεις του όμως, η Google έχει δωρίσει δεκάδες εκατομμύρια Cardboards βοηθώντας να αυξήσει τον αριθμό συσκευών που έχουν στην διάθεση τους οι καταναλωτές. Χαρακτηριστικό παράδειγμα είναι η προσφορά που περιλάμβανε η συμφωνία του 2015 με την εφημερίδα The New York Times, κατά την οποία ο ειδησεογραφικός οργανισμός, δώρισε εκατομμύρια Cardboards, μαζί με την εφημερίδα του στους συνδρομητές της για να προωθήσει τα ντοκιμαντέρ εικονικής πραγματικότητας του [5]. Στη συνέχεια θα σας δείξουμε ποσό απλό, και φθηνό, είναι να αποκτήσει κάνεις το δικό του Google Cardboard είτε παραγγέλνοντας το απ’ το internet είτε φτιάχνοντάς το μόνος του.

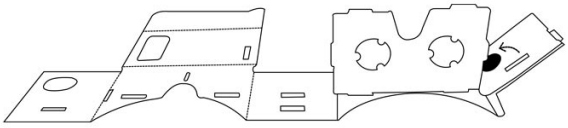
Κατασκευή Google Cardboard

Τα βήματα για να κατασκευάσει ο κάθε χρήστης το δικό του Google Cardboard είναι απλά. Παρακάτω θα σας παρουσιάσουμε μέσα από φωτογραφίες και 9 απλά βήματα πως μπορείτε να αποκτήσετε και να συναρμολογήσετε το δικό σας Google Cardboard.

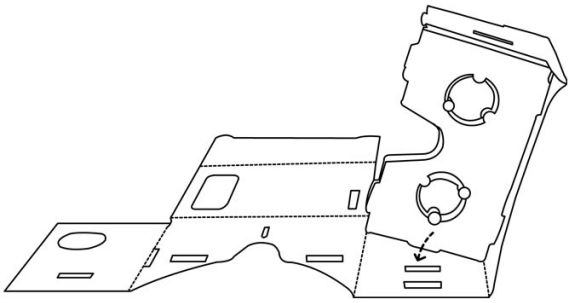
Σχήμα 3.3: First step



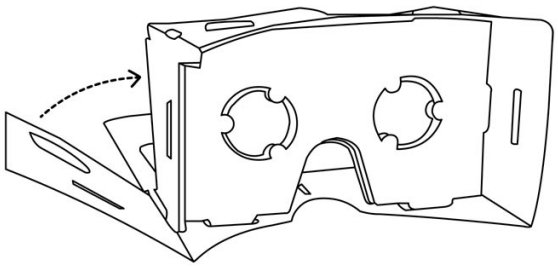
Σχήμα 3.4: Second step



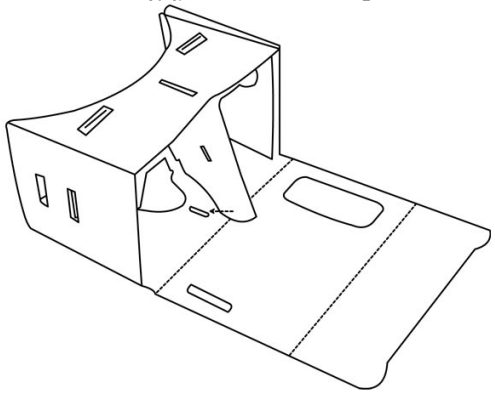
Σχήμα 3.5: Third step



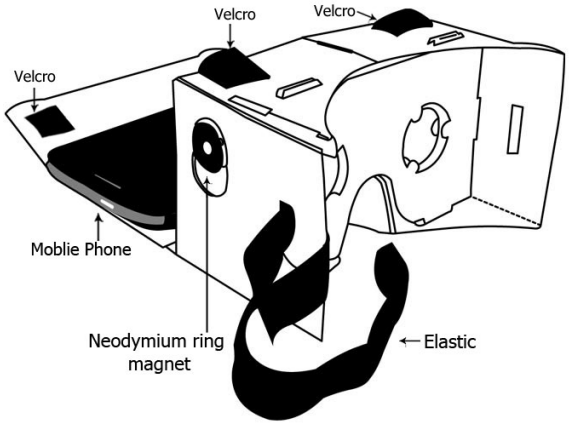
Σχήμα 3.6: Fourth step



Σχήμα 3.7: Fifth step



Σχήμα 3.8: Sixth step



1. Το πρώτο βήμα που πρέπει να κάνετε είναι να αγοράσετε μια μη συναρμολογημένη κατασκευή Cardboard. Αυτό μπορείτε να το κάνετε απευθείας από την Google και συγκεκριμένα από το παρακάτω site: Order Google Cardboard. Στο κάτω μέρος της σελίδας θα βρείτε τα μοντέλα Cardboard που υπάρχουν με τις τιμές τους δίπλα. Εκεί μπορείτε να επιλέξετε το αυθεντικό Google Cardboard στην τιμή των 15 δολαρίων.

2. Αφού το παραλάβετε και το βγάλετε από την συσκευασία του θα έχετε στα χέρια σας τα υλικά που φαίνονται στην εικόνα 3.3. Σε αυτήν την φωτογραφία βλέπετε όλα τα περιεχόμενα υλικά της συσκευασίας.

3. Η πρώτη κίνηση που πρέπει να κάνετε αφού το βγάλετε από την συσκευασία, είναι να το ξεδιπλώσετε ώστε να καταλήξετε στο παραπάνω σχήμα και στην συνέχεια ακολουθείτε τις οδηγίες που δίνονται στην φωτογραφία 3.4.

4. Στην φωτογραφία 3.5 βλέπουμε πως γυρίζουμε το παραπάνω κουμπωμένο χαρτόνι στο υπόλοιπο για να σχηματίσουμε την πάνω μεριά του Cardboard μας.

5. Στην συνέχεια και αφού κουμπώσουμε την κάτω πλευρά του χαρτονιού, ανεβάζουμε το τελευταίο κομμάτι που μένει για να ολοκληρωθεί η εξωτερική κατασκευή 3.6.

6. Προσθέτουμε το αποκολλημένο χαρτόνι στις δυο εσοχές στο πάνω και κάτω μέρος της εσωτερικής πλευράς όπως βλέπουμε στην εικόνα 3.7.

7. Στην επόμενη και τελευταία εικόνα 3.8 του τρόπου κατασκευής του Google Cardboard μπορείτε να δείτε που θα τοποθετηθούν τα σκρατς και το λάστιχο που βρήκατε στα υλικά. Επίσης βλέπετε που και από ποια μεριά μπαίνει το κινητό στην μπροστινή θήκη (αν και πλέον τα περισσότερα κινητά θα προσαρμόσουν αυτόματα την εικόνα όταν είναι γυρισμένο ανάποδα), και το σημείο που θα βάλετε τον μαγνήτη.

3.3 Εικονική πραγματικότητα και παιχνίδια

Υπάρχουν πολλά παιχνίδια εικονικής πραγματικότητας που μπορούν να χειριστούν χωρίς την χρήση πρόσθετου εξοπλισμού. Παρακάτω παρουσιάζουμε μερικά από τα οποία είναι συμβατά με το Google Cardboard.

- Κατηγορία παιχνιδιού *Action*
  Το Minos Starfighter VR είναι ένα απτά λίγα παιχνίδια εικονικής πραγματικότητας σε first-person που ανήκει στην κατηγορία Action games. Τα εκθαμβωτικά γραφικά του και η αυξανόμενη δυσκολία καθιστούν αυτό το παιχνίδι ένα απτά καλύτερα VR action games που μπορούν να παιχτούν από το Google Cardboard. Ο σκοπός του παιχνιδιού είναι να διαλύσετε εχθρικά διαστημόπλοια με συγκεκριμένα όπλα που υπάρχουν πάνω στο δικό σας πλοίο και να καταφέρετε να επιζήσετε. Βρίσκετε στο Play store για μόλις 0,99 δολάρια και είναι μια αξιοπρεπής αντιπροσώπευση του πόσο καλά μπορούν να παιχτούν τα παιχνίδια εικονικής πραγματικότητας σε ένα φθηνό kit όπως είναι το Google Cardboard.

- Κατηγορία παιχνιδιού *Puzzle-Adventure*
  Ένα από τα αγαπημένα παιχνίδια των χρηστών Google Cardboard, το Hidden Temple φέρνει

την κατηγορία Puzzle-Adventure στην εικονική πραγματικότητα του κινητού. Όλα ελέγχονται από το βλέμμα σας, δηλαδή γυρίζοντας το κεφάλι σας, καθώς κοιτάτε προσεχτικά ψάχνοντας αντικείμενα σε έναν αρχαίο ναό. Τα γραφικά του είναι πολύ καλά για παιχνίδι εικονικής πραγματικότητας στο Google Cardboard και η τιμή του αγγίζει τα 3.99 σε Play store και App store.

- Κατηγορία παιχνιδιού *Horror*
  Η κατηγορία παιχνιδιών Horror είναι η πιο ταιριαστή για την εικονική πραγματικότητα και το Chair in a Room είναι εγγυημένο οτι θα σας κάνει να φοβηθείτε. Βυθισμένο στο σκοτάδι, αντικείμενα που πετάγονται από παντού και με μόνο έναν φακό για να σας καθοδηγεί είναι σίγουρο ότι θα προσπαθήσετε πολλές φορές να πετάξετε το Google Cardboard από πάνω σας. Η ιστορία περιπλέκεται γύρω από ένα κορίτσι που χάθηκε και την συνέχεια θα ήταν καλύτερο να την δείτε από μόνοι σας. Βρίσκεται δωρεάν στο Play store ώστε ο καθένας να μπορεί να το κατεβάσει και... θα σας συνιστούσαμε αν τρομάζετε εύκολά να το αποφύγετε.

- Κατηγορία παιχνιδιού *Shooter*
  Οι παλιοί μπορεί να το θυμούνται ως NES Duck Hunt, ωστόσο οι καινούριοι θα το μάθουν ως Cyclops Duck Hunt. Είναι μια λίγο -πολύ- πιο παράξενη επανέκδοση του αρχικού Duck Hunt σε VR που λαμβάνει μέρος σε έναν κόσμο μετά την αποκάλυψη που έχει καταληφθεί από εξωγήινες πάπιες. Το παιχνίδι αρχίζει με έναν Κύκλωπα να ξυπνάει και να συνειδητοποιεί ότι ο κόσμος του έχει καταστραφεί και ο σκοπός του είναι να εξολοθρεύσει όλες τις εξωγήινες πάπιες που έχουν καταλάβει τον πλανήτη σημαδεύοντας τες για 2 δευτερόλεπτα μέσα απτό Google Cardboard. Παρά το ιδιόρρυθμο περιβάλλον και τα χιουμοριστικά χαρακτηριστικά του, είναι ένα απτά πιο ευχάριστα παιχνίδια με αρκετή δυσκολία και είναι ένα ακόμα δωρεάν παιχνίδι που υπάρχει σε Play store και App store.

- Κατηγορία παιχνιδιού *Action-Shooter*
  Αν αγαπάτε τα Star Wars και ιδιαίτερα το επεισόδιο ”The Force Awakens” τότε το Star Wars VR θα σας καταπλήξει. Το Star Wars VR ξεκινάει σε έναν γαλαξία πολύ πολύ μακριά και συγκεκριμένα στον Jakku (ο κόσμος της ερήμου που εμφανίζεται στην ταινία του 2015) οπού σας δίνεται η δυνατότητα μέσα από τον ρόλο ενός μυστικού πράκτορα αντίστασης να νιώσετε την δύναμη του Google Cardboard στα παιχνίδια εικονικής πραγματικότητας σε IPhone και Android. Το παιχνίδι είναι δωρεάν και θεωρείται ένα απτά πιο ενδιαφέροντα παιχνίδια για το 2017.

Υπάρχουν πολλά ακόμα παιχνίδια εικονικής πραγματικότητας συμβατά με το Google Cardboard και θα μπορούσαμε να συνεχίσουμε την λίστα μας για πολύ ακόμα.

Κεφάλαιο 4

Εργαλεία

Σε αυτό το κεφάλαιο παρατίθενται τα εργαλεία τα οποία συνέβαλαν στην ανάπτυξη του παιχνιδιού μας. Το τελικό αποτέλεσμα αποτελεί ένα προϊόν της εύλογης συνεργασίας μεταξύ των παρακάτω τεχνολογιών. Επιπλέον γίνεται αναφορά στο πως αυτά τα εργαλεία μπορούν να χρησιμοποιηθούν γενικότερα για την ανάπτυξη διάφορων εφαρμογών.

4.1 Unity 3D

Το Unity 3D (σχ. 4.1) είναι μια cross-platform (λογισμικό ανεξάρτητο πλατφόρμας) μηχανή δημιουργίας παιχνιδιών για διάφορες πλατφόρμες όπως PC, κονσόλες παιχνιδιών, smart-phones, tablets κ.α.. Αποτελεί ένα ολοκληρωμένο περιβάλλον ανάπτυξης (integrated development environment, IDE) και η πρώτη έκδοση του (1.0.0) δημιουργήθηκε από τους David Helgason, JoachimAnte και Nicholas Francis στη Δανία το 2004 και παρουσιάστηκε στο συνέδριο Apple's Worldwide Developers Conference το 2005 [8].

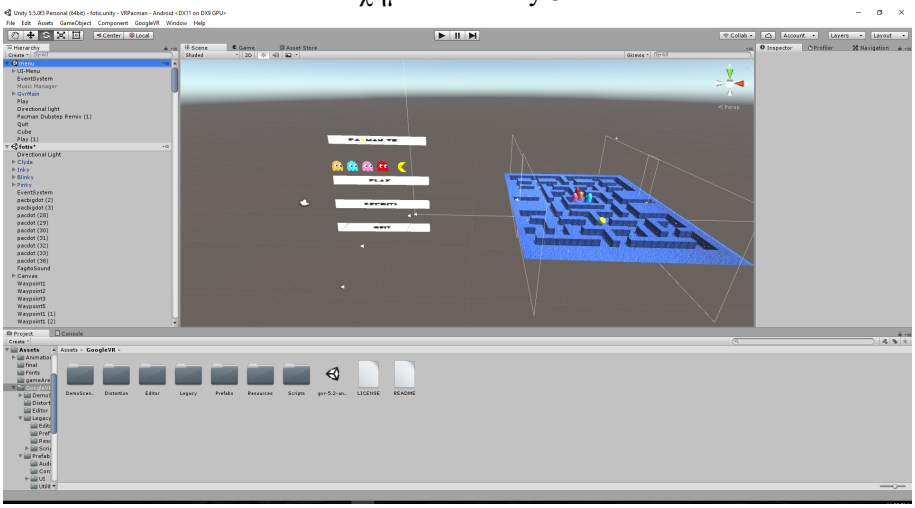
Το Unity υποστηρίζει γραφικά 2D καθώς και 3D και παρέχει την δυνατότητα προγραμματισμού διάφορων σεναρίων χρησιμοποιώντας τη γλώσσα προγραμματισμού C#. Αυτή τη στιγμή υποστηρίζει 27 πλατφόρμες και λειτουργικά το οποίο το καθιστά ένα από τα πιο χρησιμοποιημένα εργαλεία για την ανάπτυξη παιχνιδιών. Αδιάψευστο μάρτυρα αποτελεί η λίστα [3] μερικών απ' τα πολλά παιχνίδια που έχουν δημιουργηθεί με την συγκεκριμένη μηχανή.

4.2 Visual Studio

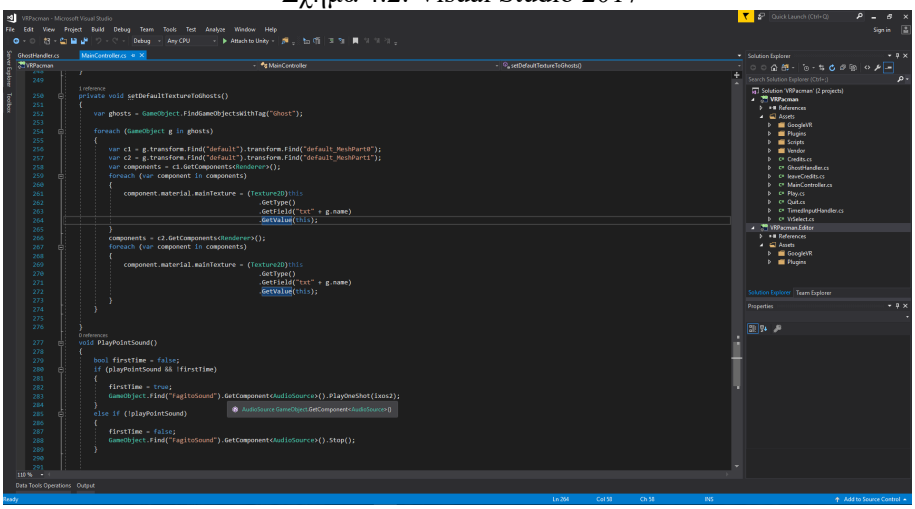
Το Visual Studio (σχ. 4.2) αποτελεί ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού, το οποίο χρησιμοποιείται για την ανάπτυξη προγραμμάτων ηλεκτρονικών υπολογιστών, ιστοσελίδων καθώς και εφαρμογών και υπηρεσιών διαδικτύου. Υποστηρίζει 36 προγραμματιστικές γλώσσες όπως τις: C, C++, C# κ.α. και δίνει στους προγραμματιστές την δυνατότητα να δημιουργήσουν ένα μεγάλο πλήθος από διάφορους τύπους εφαρμογών όπως εφαρμογές των Windows Store, εφαρμογές Windows Phone, εφαρμογές γραφείου κ.α.. Η τελεύταια έκδοση του έχει την ονομασία Visual Studio 2107 και κυκλοφόρησε τον Μάρτιο του 2017. Από τότε λαμβάνει συνεχώς νέες αναβαθμίσεις, όπως αυτή που κυκλοφόρησε τον Ιανουάριο του 2018, οι οποίες βελτιώνουν και διορθώνουν διάφορες λειτουργίες καθώς και επίσης παρουσιάζουν νέες. [6]

Το Unity3D έχει ενσωματωμένο έναν δικό της μεταγλωττιστή C# βασισμένο στο .NET Framework το οποίο καθιστά εύκολη τη χρήση του Visual Studio για την επεξεργασία και δημιουργία των κλά-

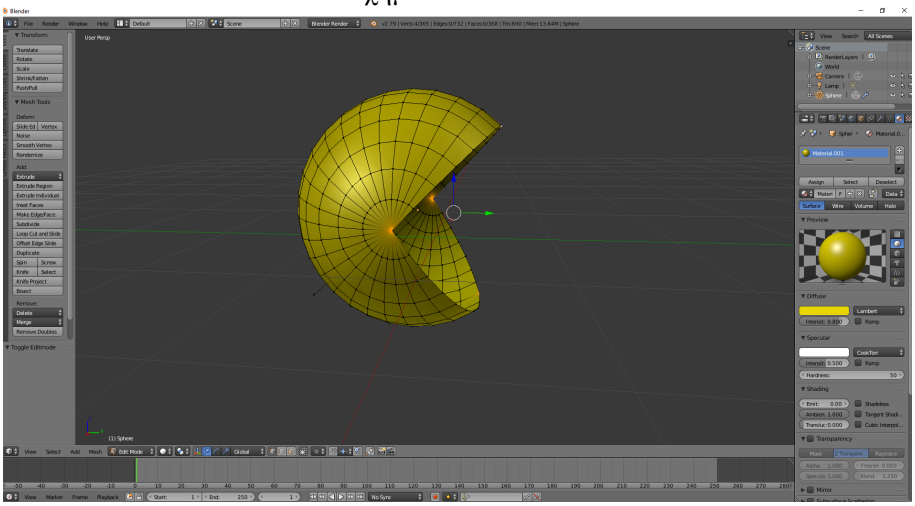
Σχήμα 4.1: Unity 3d



Σχήμα 4.2: Visual Studio 2017



Σχήμα 4.3: Blender



σεων, αλγόριθμων και προγραμματισμό των διάφορων σεναρίων που χρησιμοποιούνται για την υλοποίηση ενός παιχνιδιού.

4.3 Blender

Το Blender (σχ. 4.3) είναι ένα λογισμικό ανοικτού κώδικα (open-source software) το οποίο παρέχεται δωρεάν. Η πρώτη έκδοση δημιουργήθηκε το 1995 από το Ολλανδικό studio Neo Geo ενώ το 2002 ανακοινώθηκε ότι ο πηγαίος κώδικας του θα ήταν ελεύθερος [4]. Το συγκεκριμένο εργαλείο χρησιμοποιείται για τη δημιουργία 3D Model,οπτικών εφέ, εφαρμογών τρισδιάστατης διάδρασης, ταινιών με κινούμενα σχέδια κ.α..

Λόγω της φύσης του ανοιχτού κώδικα του Blender, υπήρξαν διάφορες προσπάθειες ανάπτυξης τροποποιημένων ελάχιστα εκδόσεων ως ξεχωριστά προγράμματα με σκοπό να επωφεληθούν από την επιτυχία του Blender. Παραδείγματα περιλαμβάνουν τα IllusionMage, 3DMofun, 3DMagix και Fluid Designer. Το Blender έχει χρησιμοποιηθεί για τηλεοπτικές διαφημίσεις σε διάφορα μέρη του κόσμου, όπως η Αυστραλία, η Ισλανδία, η Βραζιλία, η Ρωσία και η Σουηδία. Επιπλέον χρησιμοποιήθηκε και από τη NASA για να αναπτύξει μια διαδραστική διαδικτυακή εφαρμογή για τον εορτασμό της 3ης επετείου της προσγείωσης του Curiosity rover στον Άρη. [4]

4.4 Google VR SDK

Το Google Cardboard υποστηρίζει τόσο το Android όσο και το iOS και επιτρέπει την εμβυσματική εμπειρία VR με τη σύντηξη δεδομένων από τους αισθητήρες του τηλεφώνου για να προβλέψει τη θέση του χρήστη στο κεφάλι τόσο στον πραγματικό όσο και στον εικονικό κόσμο. Για το λόγο αυτό η Google παρέχει διάφορα εργαλεία για την εύκολη ανάπτυξη εφαρμογών εικονικής πραγματικότητας σε διάφορα περιβάλλοντα ανάπτυξης όπως το Unity. Συγκεκριμένα το Google VR SDK (Gvr SDK) είναι ένα τέτοιο βοηθητικό εργαλείο ανάπτυξης λογισμικού VR.

Με την ενσωμάτωση του Gvr SDK στο Unity διευκολύνεται η ανάπτυξη νέων εφαρμογών εικονικής πραγματικότητας για τα Google Cardboard και Daydream VR Headsets ή η μετατροπή

παλαιότερων εφαρμογών σε εφαρμογές εικονικής πραγματικότητας. Η υποστήριξη του Gvr SDK από τη συσκευή προϋποθέτει ότι το λειτουργικό Android που είναι εγκατεστημένο σε αυτή, είναι τουλάχιστον το Android API SDK version 21 γνωστό και ως ”Lollipop”. Επιπλέον η συσκευή είναι απαραίτητο να διαθέτει ένα γυροσκόπιο για την ομαλή λειτουργεία των δυνατοτήτων που παρέχει το Gvr SDK.

4.5 Android SDK

Το Android SDK είναι ένα σύνολο εργαλείων ανάπτυξης που χρησιμοποιούνται για την ανάπτυξη εφαρμογών για την πλατφόρμα Android. Η παροχή αυτών των εργαλείων βοηθάει τους προγραμματιστές εξασφαλίζοντας ότι ή διαδικασία ανάπτυξης λογισμικού για το Android, θα είναι όσο το δυνατόν πιο ομαλή. Επιπλέον παρέχει δυνατότητες όπως τη πρόσβαση σε μοναδικές λειτουργίες του λειτουργικού συστήματος καθώς και ενός εξομοιωτή για την δοκιμή των εφαρμογών. Συγκεκριμένα, το πακέτο περιλαμβάνει τα εξής:

- Βιβλιοθήκες
- Εργαλείο αποσφαλμάτωσης
- Εξομοιωτή
- Οδηγίες για το τρόπο χρήσης της διεπαφής προγραμματισμού εφαρμογών (Android APIs)
- Δείγματα πηγαίου κώδικα
- Εκπαιδευτικό υλικό για τη πλατφόρμα Android

Τα εργαλεία αυτά είναι πολύ σημαντικά για την ανάπτυξη εφαρμογών μέσω του Unity. Ειδικά, μέσω αυτών το Unity παράγει το τελικό αρχείο (.apk) για την εγκατάσταση της εφαρμογής στις συσκευές Android.

Κεφάλαιο 5

Στόχος

Το κλασσικό arcade παιχνίδι Pac-Man διαδραματίζεται σε ένα δισδιάστατο εικονικό κόσμο και βασίζεται στη κίνηση και την αλληλοεπίδραση του χαρακτήρα *Pacman* με το περιβάλλον. Ως επί το πλείστων, ο παίκτης «ελέγχει» και δίνει εντολές στον χαρακτήρα χρησιμοποιώντας διάφορες συσκευές εισόδου όπως: joystick , πληκτρολόγιο κ.α.. Το Google Cardboard χρησιμοποιείται κατά κύριο λόγο για τη προβολή και αναπαραγωγή περιεχομένου εικονικής πραγματικότητας χωρίς να παρέχει στον χρήστη επιπλέον δυνατότητες όπως αυτές μιας συσκευής εισόδου, οι οποίες διευκολύνουν την επικοινωνία μεταξύ συσκευής και ανθρώπου.

Στόχος της παρούσας πτυχιακής εργασίας είναι η ανάπτυξη του παιχνιδιού Pac-Man σε εικονική πραγματικότητα για την χρήση του στο Google Cardboard. Σκοπός είναι οι χρήστες να μπορούν να παίξουν και να πλοηγηθούν στον τρισδιάστατο εικονικό κόσμο, αλληλοεπιδρώντας με τα εικονικά αντικείμενα, χωρίς να είναι απαραίτητη η χρήση πρόσθετου εξοπλισμού (π.χ. joystick) παρά μόνο ενός smartphone και της πλατφόρμας εικονικής πραγματικότητας από τη Google.

Για να μεταφερθεί ένα τυπικό δισδιάστατο arcade παιχνίδι σε εικονική πραγματικότητα και συγκεκριμένα για χρήση στο Google Cardboard, χρειάζεται να υλοποιηθούν εκ νέου οι προϋποθέσεις οποιασδήποτε πιθανής κίνησης και χειρισμού σε συνδυασμό με τις δυνατότητες που παρέχει η πλατφόρμα της Google όπως: προβολή τρισδιάστατου εικονικού κόσμου σε 360 μοίρες κ.α.. Το τελικό αποτέλεσμα θα πρέπει να προσφέρει, εκτός από τον επαυξημένο εικονικό κόσμο, ήχο, εικόνα και χειρισμό παιχνιδιού, όσο είναι δυνατόν πλησιέστερα στο κλασσικό Pac-Man έτσι ώστε να επιτευχθεί μια πλήρης εμπειρία, αντίστοιχη του αυθεντικού παιχνιδιού.

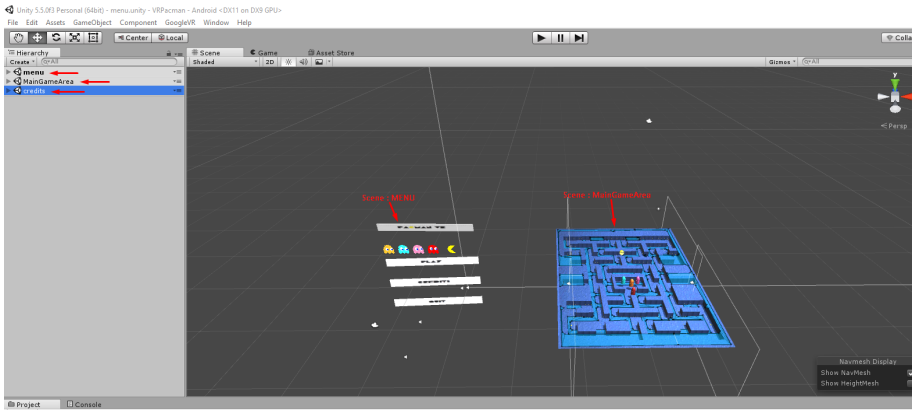
Κεφάλαιο 6

Ανάπτυξη του VR Pac-Man

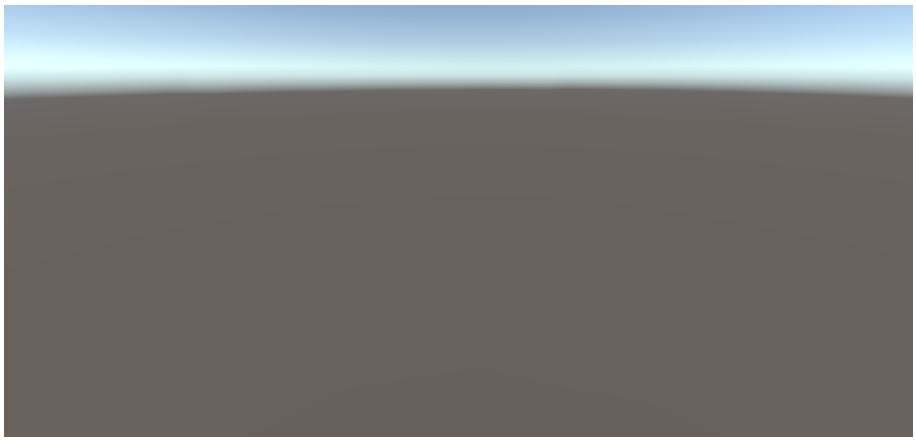
Με τη δημιουργία ενός νέου Unity Project, το Unity δημιουργεί αυτόματα για εμάς μια νέα σκηνή (η οποία θα ονομάζεται εκατέρωθεν scene). Μια unity scene αποτελεί ουσιαστικά ένα επίπεδο για το παιχνίδι καθώς μπορεί να περιέχει διάφορα GameObjects (6.2). Το VR Pac-Man αποτελείτε από τρία scene (σχ. 6.1):

- Menu
- MainGameArea
- Credits

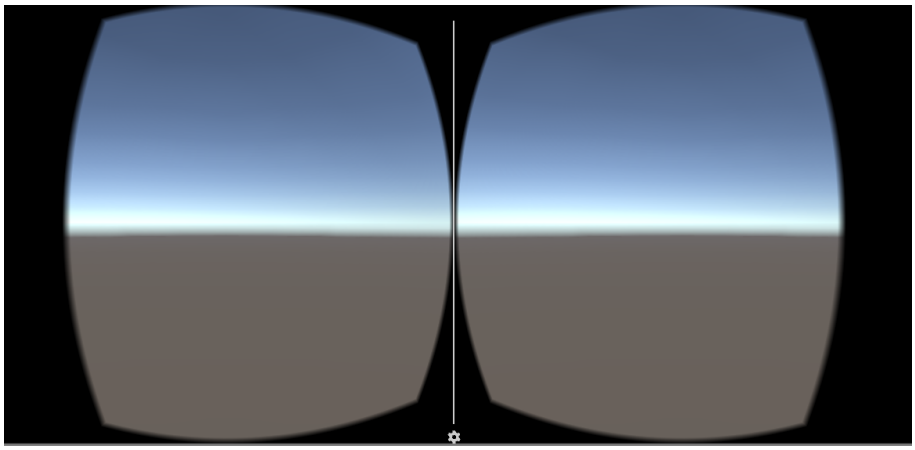
Στις παρακάτω σελίδες θα ασχοληθούμε με τον τρόπο υλοποίησης της σκηνής MainGameArea η οποία και αποτελεί τη βασική πίστα του παιχνιδιού VR Pac-Man. Δημιουργώντας ένα νέο Unity Project τα μόνα GameObject που περιέχει η scene που δημιουργήθηκε αυτόματα είναι μια κάμερα η οποία αναλαμβάνει την απεικόνιση του χώρου στο χρήστη (δηλαδή αυτό που θα βλέπει ο χρήστης την ώρα που θα παίζει) και ένα αντικείμενο φωτισμού στο χώρο. Το πρώτο βήμα είναι να εισάγουμε σαν Asset το πακέτο GoogleVRForUnity το οποίο βρίσκεται στα αρχεία λήψης του Google VR SDK και το οποίο μας δίνει πρόσβαση σε διάφορα Objects που μπορούμε να εισάγουμε στη σκηνής μας. Εισάγοντας το αντικείμενο GvrViewerMain στη σκηνή και μέσω του C# Script που έχει γραφτεί η κάμερα μετατρέπει αυτόματα τον τρόπο που προβάλει το χώρο στο χρήση απ' τον προκαθορισμένο σε Virtual Reality Mode (σχ. 6.2, σχ. 6.3).



Σχήμα 6.1: VR Pac-Man, Game Scenes



Σχήμα 6.2: Προβολή κόσμου χωρίς το GvrViewerMain



Σχήμα 6.3: Προβολή κόσμου με GvrViewerMain

Με αυτό τον τρόπο ο χρήστης έχει πλέον τη δυνατότητα να παρατηρήσει τον εικονικό κόσμο, όπως ακριβώς θα έκανε και στον πραγματικό, δηλαδή στρέφοντας το κεφάλι του πάνω, κάτω, δεξία και αριστερά μετακινώντας το Google Cardboard προς την αντίστοιχη κατεύθυνση.

6.1 Η κίνηση στον εικονικό χώρο

Έχοντας δώσει στο χρήστη τη δυνατότητα να παρατηρεί το εικονικό περιβάλλον, είναι πλέον απαραίτητο να υλοποιήσουμε μια τεχνική κίνησης στο χώρο ώστε το παιχνίδι να προσφέρει μια παρόμοια εμπειρία με αυτή του κλασσικού Pac-Man. Σε πολλές εφαρμογές εικονικής πραγματικότητας ο τρισδιάστατος εικονικός κόσμος είναι μεγαλύτερος απ' τον χώρο στον οποίο βρίσκεται ο χρήστης της εφαρμογής. Το ίδιο συμβαίνει και στη περίπτωση του παιχνιδιού VR Pac-Man, όπου αν υποθέσουμε ότι ο χαρακτήρας Pac-Man έχει περίπου το μέγεθος ενός ανθρώπου, τότε η πίστα ξεπερνάει κατά πολύ το μέγεθος ενός απλού δωματίου.

Με την πάροδο των χρόνων έχουν αναπτυχθεί και σχεδιαστεί διάφορες τεχνικές οι οποίες διευκολύνουν την εξερεύνηση μεγάλων εικονικων περιβαλλόντων. Ωστόσο αρκετές απ' αυτές χρησιμοποιούν πρόσθετο εξοπλισμό όπως ειδικά σχεδιασμένους διαδρόμους, παπούτσια με αισθητήρες κίνησης κ.α.. Για την αποφύγη χρήσης πρόσθετου εξοπλισμού σχεδιάζουμε μια τεχνική η οποία το μόνο που απαιτεί να έχει ο παίκτης είναι απλά το κινητό του και το Google CardBoard.

6.1.1 Σχεδιασμός κίνησης

Υποθέτουμε ότι ο παίκτης και συγκεκριμένα το κεφάλι του, στο οποίο και στηρίζεται το Google Cardboard, είναι σημείο του άξονα X (καρτεσιανό σύστημα συντεταγμένων) και θεωρούμε πως κοιτάζοντας ευθεία η γωνία περιστροφής του κεφαλιού του είναι 0 μοίρες. Κοιτάζοντας ευθεία ξεκινά αυτόματα η κίνηση στο εικονικό περιβάλλον και συνεχίζεται όσο το κεφαλι του είναι στραμμένο προς τα επάνω ή προς σημεία που βρίσκονται έως το πολύ 20 μοίρες προς τα κάτω.

Κατά τη διάρκεια της κίνησης ο παίκτης έχει τη δυνατότητα να περιηγηθεί στο χώρο, όπου αυτό είναι εφικτό από άποψη γραφικών, στρέφοντας απλά το κεφάλι του αριστερά ή δεξιά. Στην περίπτωση που ο παίκτης στρέψει το κεφάλι του προς τα κάτω με γωνία μεγαλύτερη των 20 μοιρών, σηματοδοτείται και το τέλος της κίνησης. Η τεχνική αυτή παρέχει μια πλήρη εμπειρία εξερεύνησης του εικονικού κόσμου χρησιμοποιώντας μόνο το γυροσκόπιο του κινητού που βρίσκεται εντός του Google Cardboard.

6.1.2 Υλοποιήση μέσω Unity 3D

Για την υλοποίηση της παραπάνω τεχνικής χρησιμοποιούμε το βοηθητικό εργαλείο Character Controller που παρέχει το Unity. Το εργαλείο αυτό, όταν ενσωματωθεί σε ένα GameObject, βοηθάει στην αυτοματοποίηση της κίνησης του. Παρέχει διάφορες λειτουργικότητες όπως βαρύτητα, εντοπισμό σύγκρουσης με άλλα GameObjects και φυσικά κίνηση του αντικειμένου προς τη γωνία κατεύθυνσης που επιθυμούμε.

Για να συνδυάσουμε τη τεχνική κίνησης που σχεδιάσαμε και την αυτοματοποιήση που παρέχει το Character Controller αρχικά ορίζουμε πέντε μεταβλητές (κωδ. 6.1). Η μεταβλητή *mainCamera* αφορά την βασική κάμερα του παιχνιδιού δηλαδή αυτό που βλέπει ο παίκτης. Η *moveAngle* είναι η γωνία με την οποία θα συγκρίνουμε , σε κάθε καρέ του παιχνιδιού, την γωνία περιστροφής του κεφαλιού του χρήστη (χρησιμοποιώντας την μεταβλ. mainCamera) για να του επιτρέψουμε ή όχι να κινηθεί στον χώρο. Η μεταβλητή *speed* χρησιμοποιείται για να ορίσει την ταχύτητα με την οποία θα προχωράει ο παίκτης ενώ η *moveAvailable* είναι μια μεταβλητή ελέγχου η οποία βοηθάει στο να γνωρίζουμε σε κάθε σημείο του παιχνιδιού αν ο χρήστης είναι στάσιμος ή όχι. Για το λόγο αυτό, η μεταβλητή αυτή ορίζεται με **private set** ώστε να παρέχεται μόνο η πληροφορία κίνησης του παίκτη χωρίς να είναι εφικτό να αλλάξει η τίμη της από κάποια άλλη κλάση. Τέλος η *cc* μας δίνει πρόσβαση στο βοηθητικό εργαλείο Character Controller, το οποίο έχουμε ενσωματώσει στη κάμερα, ώστε να χρησιμοποιήσουμε την συνάρτηση **public bool SimpleMove(Vector3 speed)** που παρέχει, για να ξεκινήσουμε την κίνηση του παίκτη. Για να ελέγχξουμε αν ο παίκτης μπορεί να προχωρήσει προς τη κατεύθυνση που έχει στρέψει το κεφάλι του χρησιμοποιούμε την συνάρτηση **void Update()** η οποία καλείται (αυτόματα) μία φορά για κάθε καρέ. Για την υλοποιήση του σχεδιασμού που περιγράφεται στο 6.1.1 γράφουμε τον κώδικα που απεικονίζεται στο κωδ. 6.2.

```
// Main Game's Camera
private Transform mainCamera = transform;

//Camera's CharacterController component
private CharacterController cc =
    GetComponent<CharacterController>();

//Gwnia stin opoia stamataei/ksekinaei i kinisi
private float moveAngle = 20.0f;
```

```
// Taxitita kinisis
private float speed = 0.1f;

// Metavliti elegxou dinatotitas kinisis tou paikti
public bool moveAvailable {get; private set;}
```

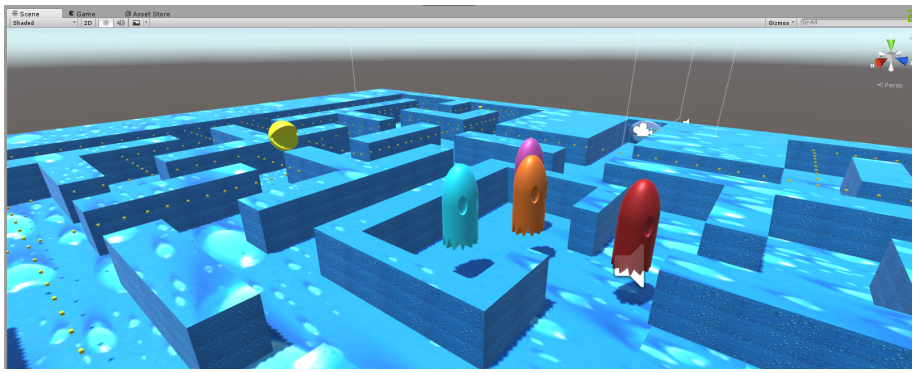
Κώδικας C# 6.1: Properties για την υλοποιήση κίνησης

```
//Update is called once per frame
void Update()
{
    //elegxos an o xristis exei girisei to kefali pros ta katw ( >
       20 moires)
    if (mainCamera.eulerAngles.x >= moveAngle &&
       mainCamera.eulerAngles.x < 90.0f)
        // Stop moving
        moveAvailable = false;
    else
        // Available to move
        moveAvailable = true;

    // Otan teleiwsei to intro song ksekinaei to Game
    if (moveAvailable &&
       !gameObject.GetComponent<AudioSource>().isPlaying)
    {
        //ksekinaei to sound ton ghost
        if (!gameStarted)
            GameObject.Find("GhostSound").GetComponent<AudioSource>().Play();

        gameStarted = true;
        Destroy(GameObject.Find("GetReady"));
        // vriskei ti gwnia gia movement pros ta empros
        Vector3 fw =
           mainCamera.TransformDirection(Vector3.forward);
        // Enarksi kinisis
        cc.SimpleMove(fw * speed);
    }
}
```

Κώδικας C# 6.2: Υλοποιήση κίνησης

6.2 GameObjects

Αφού ολοκληρώσουμε τις απαραίτητες ενέργειες για να δώσουμε στον χρήστη τη δυνατότητα κίνησης και εξερεύνησης του εικονικού κόσμου, χρείαζεται να εισάγουμε στη σκηνή τα GameObjects που θα χρειαστούμε για το παιχνίδι. Τα GameObject είναι μία από τις πιο σημαντικές λειτουργίες που παρέχει το Unity. Κάθε αντικείμενο στο παιχνίδι αποτελεί ένα GameObject, από χαρακτήρες και συλλεκτικά αντικείμενα σε φώτα, κάμερες και ειδικά εφέ.



Σχήμα 6.4: VR Pac-Man, τελικά GameObjects με textures

Ωστόσο, ένα GameObject δεν μπορεί να κάνει τίποτα μόνο του. Για να μπορέσει να γίνει ένας χαρακτήρας, ένα περιβάλλον ή ένα ειδικό αποτέλεσμα, πρέπει να του δωθούν ιδιότητες όπως για παράδειγμα προγραμματισμός των διάφορων σεναρίων με τα οποία μπορεί να έρθει αντιμέτωπο.

6.2.1 Βασικά GameObjects

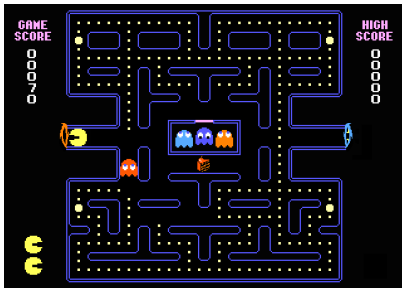
Τα βασικά GameObjects της εφαρμογής μας είναι τα παρακάτω:

- Πίστα-Λαβύρινθος
- Pacman
- Φαντάσματα(Blinky, Inky, Clyde, Pinky)

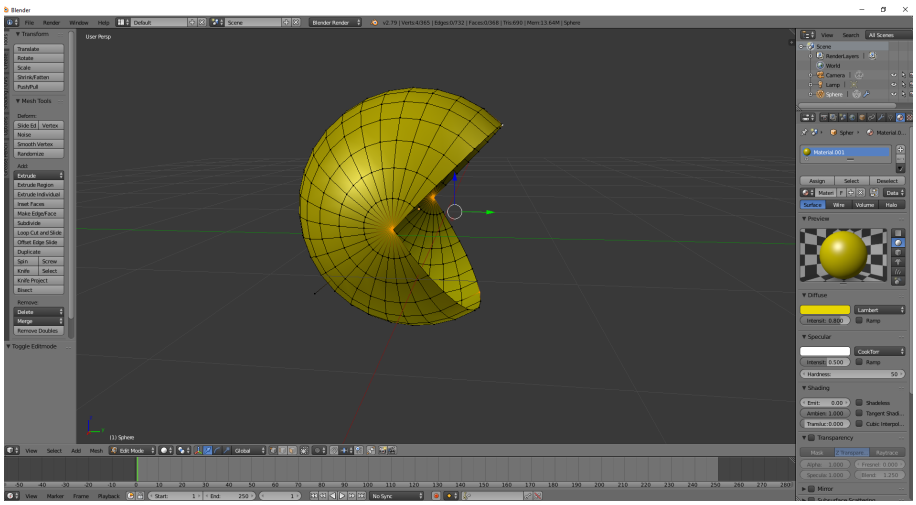
Τα παραπάνω 3D Model-GameObjects δημιουργούνται με τη βοήθεια του προγράμματος Blender. Για τη δημιουργία του Pacman και του μοντέλου που χρησιμοποιείται για τα τέσσερα φαντάσματα χρησιμοποιούμε ένα αντικείμενο UV Sphere το οποίο προσαρμόζουμε (π.χ. αφαίρεση στόματος για τον Pacman) μέσω του editing mode που προσφέρει το blender (σχ. 6.5, σχ. 6.6). Για την πίστα-λαβύρινθος, αρχικά εισάγουμε ένα αντικείμενο Plane (σχ. 6.7), του οποίου μεγάλωνουμε τις διαστάστεις και του δίνουμε σαν texture μια φωτογραφία από το χάρτη του Pacman. Έπειτα με μετασχηματισμό διάφορων αντικειμένων τύπου Cube τοποθετημένων πάνω στο αντικείμενο Plane, φτιάχνουμε τους τοίχους βασισμένοι στη φωτογραφία της πίστας (σχ. 6.8). Το τελικό αποτέλεσμα φαίνεται στο σχ. 6.4.

6.2.2 Πύλη μεταφοράς

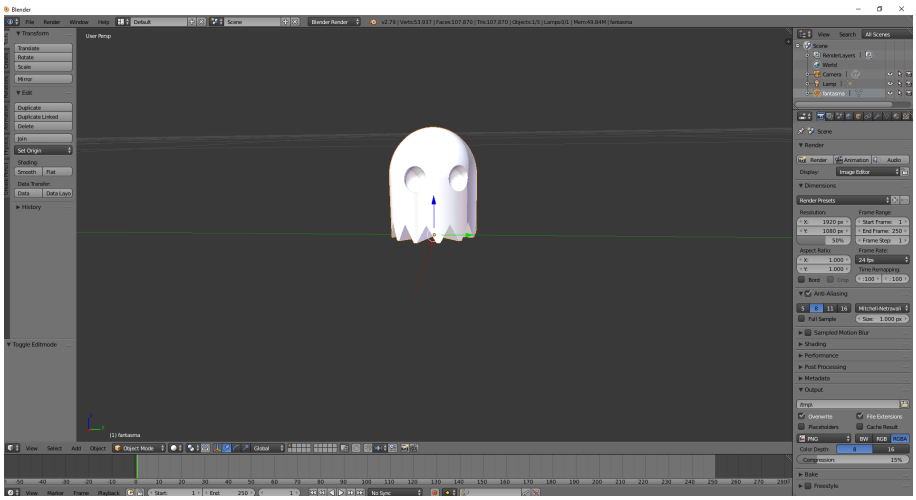
Υπάρχουν δύο επιπλέον GameObjects τα οποία πρέπει να δημιουργηθούν. Ο λόγος για τις δύο πύλες μεταφοράς (portals) οι οποίες βρίσκονται στην αριστερή και δεξιά άκρη του λαβύρινθου αντίστοιχα (σχ. 6.9). Τα δύο αυτά portal υπάρχουν με σκοπό να βοηθήσουν τον Pacman να κινηθεί στο λαβύρινθο και να αποφύγει τα φαντάσματα πιο εύκολα καθώς όταν περάσει μέσα από μια πύλη, μεταφέρεται απευθείας στην αντίθετη πλευρά της πίστας δηλαδή στην απέναντι πύλη. Επιπλέον αν περάσει ένα φάντασμα μέσα από ένα portal τότε



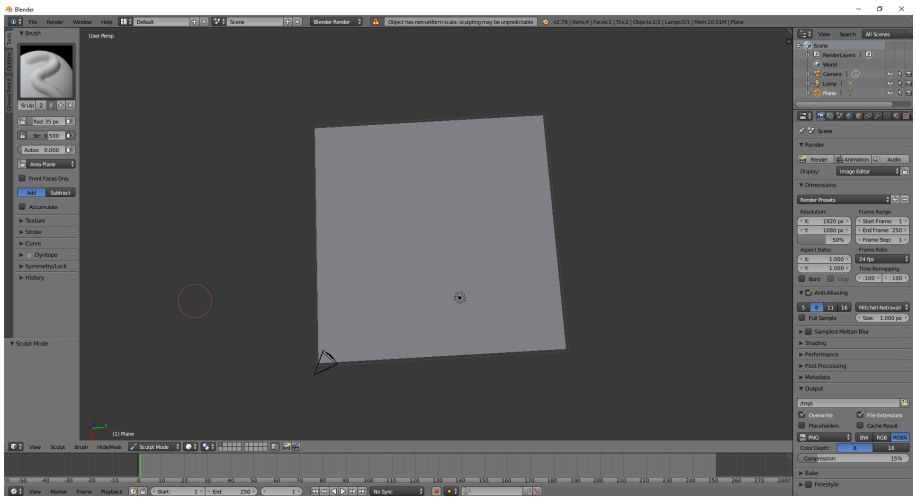
Σχήμα 6.9: Πύλες μεταφοράς PacMan



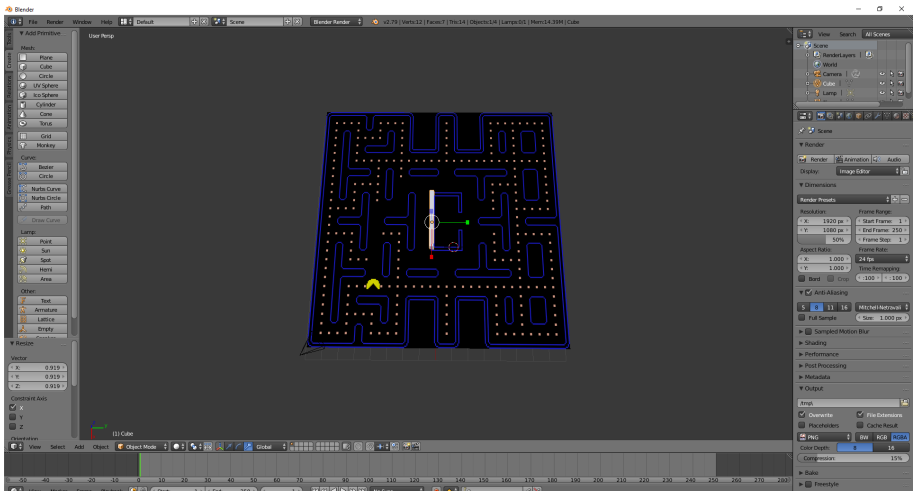
Σχήμα 6.5: Δημιουργία Pac-Man μέσω Blender



Σχήμα 6.6: Δημιουργία Μοντέλου για τα φαντάσματα μέσω Blender



Σχήμα 6.7: Αρχικό αντικείμενο Plane χωρίς texture



Σχήμα 6.8: Αντικείμενο Plane με τη φωτογραφία του λαβύρινθου για texture

παρόλο που θα μεταφερθεί και αυτό απέναντι η ταχύτητα με
την οποία κινείται μειώνεται για ένα χρονικό διάστημα.

Για την σχεδίαση των portal χρησιμοποιούμε δύο Unity GameObjects τύπου Sphere(σφαίρα) και δύο κάμερες. Αρχικά μετασχηματίζουμε τις σφαίρες στο μέγεθος που επιθυμούμε και τις τοποθετούμε στις πλευρές του λαβύρινθου, μία αριστερά και μία δεξιά αντίστοιχα. Για να βοηθήσουμε τον παίκτη να αποφασίσει αν είναι καλή επιλογή να περάσει μέσα από μία πύλη, υλοποιούμε έναν τρόπο απεικόνισης του χώρου που θα μεταφερθεί όταν και αν το επιλέξει. Αυτό πραγματοποιείται με τη βοήθεια των δύο καμερών που αναφέραμε παραπάνω. Συγκεκριμένα τοποθέτουμε μία κάμερα μπροστά απο κάθε πύλη η οποία προβάλει ότι ακριβώς υπάρχει μπροστά απ' το portal σε πρώτο πρόσωπο. Τέλος με τη βοήθεια του Unity που δίνει την δυνατότητα να ορίσουμε μία κάμερα ως texture σε ένα αντικείμενο, θέτουμε για texture στην πύλη που βρίσκεται αριστερά την κάμερα που βρίσκεται μπροστά από την πύλη στα δεξία και το αντίστροφο όπως φαίνεται στο σχ. 6.10. Με αυτό το τρόπο ο παίκτης γνωρίζει ανά πάσα στιγμή τι θα αντικρίσει όταν περάσει μέσα από ένα portal.

6.3 Η κλάση MainController.cs

Η κλάση αυτή αποτελεί την πιο σημαντική κλάση του παιχνιδιού καθώς είναι ενσωματωμένη στη βασική Camera (κάμερα του παίκτη) και διαχείριζεται όλες του τις ενέργειες όπως τις αλληλεπιδράσεις του με τα αντικείμενα στο χώρο, την συγκέντρωση πόντων, τον έλεγχο για το αν ο παίκτης έχει κερδίσει η έχει χάσει, την επανεκκίνηση του παιχνιδιού κ.α.. Επιπλέον περιέχει το κώδικα που αφορά την κίνηση του παίκτη που περιγράφηκε στο 6.1.2.

Είναι το κύριο συστατικό για την διαχείριση των διάφορων σεναρίων του παιχνιδιού, καθώς ανταποκρίνεται στις εισροές από τον παίκτη και κανονίζει τα γεγονότα στο παιχνίδι να συμβαίνουν όταν πρέπει. Στη συνέχεια θα περιγράψουμε τη λειτουργικότητα των βασικών συναρτήσεων που περιέχει παραθέτοντας κομμάτια κώδικα για την καλύτερη κατανόηση τους.

6.3.1 Η συνάρτηση Update

Όπως αναφέρθηκε στο 6.1.2 η συνάρτηση αυτή καλείται αυτόματα για κάθε καρέ του παιχνιδιού. Αυτό την καθιστά μια πολύ σημαντική συνάρτηση καθώς σε αυτή γίνονται δύο βασικοί έλεγχοι για το παιχνίδι:

- Έλεγχος δυνατότητας κίνησης και κίνηση του παίκτη 6.1.2
- Έλεγχος αν ο παίκτης έχει κερδίσει

Για να καταφέρει ο παίκτης να κερδίσει πρέπει να "φάει" όλες τις τελείες οι οποίες βρίσκονται στο μεγαλύτερο κομμάτι του λαβύρινθου. Για την υλοποίηση της παραπάνω λογικής χρησιμοποιούμε τα Unity tags. Ενα unity tag είναι μια ετικέτα που μπορεί να αντιστιχοιθεί σε ένα η περισσότερα GameObjects με σκοπό την ομαδοποιήση τους ώστε να μπορούν να εντοπιστούν εύκολα. Αρχικά δίνουμε στα GameObject που αφορούν τις μικρές κίτρινες τελείες την ετικέτα *pacdot* (σχ. 6.12) ενώ στις μεγάλες την ετικέτα *PacBigDot* (σχ. 6.13). Επειτα ελέγχουμε σε κάθε καρέ αν ο συνολικός αριθμός των GameObject που υπάρχουν στη σκηνή και η ετικέτα τους είναι *pacdot* ή *PacBigDot* είναι ίσος με μηδέν (κωδ. 6.3). Αν ισχύει αυτη η συνθήκη θέτουμε τη μεταβλητή moveAvailable = false ώστε ο παίκτης να μείνει στάσιμος και καλούμε τη συνάρτηση *void Winner()* (κωδ. 6.4) η οποία όπως προδίδει και το όνομα της είναι υπεύθυνη για τη ειδοποιήση του χρήστη ότι έχει κερδίσει. Η συνάρτηση αυτή αρχικά βρίσκει όλα τα ηχητικά εφέ που ακούγονται και τα σταματάει. Στη συνέχεια παίζει τον ήχο νίκης εμφανίζοντας ταυτόχρονα το κείμενο *Congratulations!!! You Win!!*. Τέλος επιστρέφει τον παίκτη στο αρχικό Menu μετά απο πέντε δευτερόλεπτα.

```
//winner winner chicken dinner
 if ((GameObject.FindGameObjectsWithTag("PacDot").Length +
    GameObject.FindGameObjectsWithTag("PacBigDot").Length)==0)
 {
     //arxikopoiisi ton ponton
     points = 0;
     //stop movement
     moveAvailable = false;
     Winner();
 }
```

Κώδικας C# 6.3: Ελεγχος νικητή

```
 void Winner()
 {
     //epistrofi olon ton ixitikon gameobject
     var allAudioSources = FindObjectsOfType(typeof(AudioSource))
        as AudioSource[];
     foreach (AudioSource audioS in allAudioSources
                                    .Cast<AudioSource>()
                                    .Where(x=> x.isPlaying))
     {
         audioS.Stop();
     }
     if
        (!GameObject.Find("YouWin").GetComponent<AudioSource>().isPlaying)
```

```
            GameObject.Find("YouWin").GetComponent<AudioSource>().Play();

        GameObject.Find("txtYouWin").GetComponent<Text>().enabled =
            true;//show you win txt
        //epistrofi sto menu meta apo 5 sec
        Invoke("QuitToMenu", 5);
    }
```

Κώδικας C# 6.4: Συνάρτηση Winner

6.3.2 Η συνάρτηση OnControllerColliderHit

Ο ρόλος της συγκεκριμένης συνάρτησης είναι πολύ σημαντικός για την εξέλιξη του παιχνιδιού καθώς καλείται αυτόματα κάθε φορά που ο χρήστης έρχεται σε σύγκρουση με κάποιο άλλο αντικείμενο στο χώρο. Με άλλα λόγια είναι υπεύθυνη για την αλληλοεπίδραση του παίκτη με το περιβάλλον στο οποίο βρίσκεται και χωρίς αυτή το παιχνίδι δε θα ήταν καθόλου διαδραστικό. Συγκεκριμένα, στην συνάρτηση αυτή βρίσκεται ο χειρισμός των παρακάτω σεναρίων:

1. **Επαφή του παίκτη με τις κίτρινες τελείες (πόντους)**
   Αν ο Pacman ακουμπίσει μία κίτρινη τελεία που βρίσκεται στο λαβύρινθο τότε την ”τρώει” και αν αυτή είναι μικρή τελεία κερδίζει 10 πόντους. Διαφορετικά αν η τελεία είναι Power-Pill τότε κερδίζει 50 πόντους και μπορεί επιπλέον για ένα χρονικό διάστημα, να ”φάει” και τα φαντάσματα αν έρθει σε επαφή μαζί τους χωρίς αυτά να μπορούν να του κάνουν κάτι.

   Για την μεταφορά των παραπάνω περιπτώσεων σε κώδικα αρχικά ορίζουμε στην κλάση μας μια μεταβλητή ελέγχου *isPowerPilled* η οποία μας βοηθάει να γνωρίζουμε σε οποιοδήποτε σημείο του παιχνιδιού αν ο παίκτης έχει ”φάει” ένα Power-Pill. Στην συνέχεια ελέγχουμε αν το αντικείμενο με το οποίο έχει έρθει σε επαφή ο Pacman έχει την ετικέτα *pacdot* η *PacBigDot*. Σε περίπτωση που αυτό ισχύει, προσθέτουμε 10 πόντους στον παίκτη, παίζουμε το ηχητικό εφέ του ”φαγητού” και καταστρέφουμε (αφαιρούμε απ’ τη σκηνή) τη τελεία την οποία έφαγε το οποίο μας βοηθάει στο να διαπιστώσουμε αν ο παίκτης έχει κερδίσει όπως περιγράφουμε στο 6.3.1. Αν η τελεία έχει το tag *PacBigDot* τότε υλοποιούμε τα παρακάτω βήματα:

   (α) Προσθέτουμε επιπλέον 40 πόντους

   (β) Θέτουμε την *isPowerPilled = true*

   (γ) Παίζουμε το ηχητικό εφέ το οποίο ειδοποιεί τον χρήστη οτί μπορεί να ”φάει” τα φαντάσματα

   (δ) Δίνουμε σαν texture στα φαντάσματα το χρώμα σκούρο μπλε.

   Η υλοποιήση σε κώδικα C# μπορεί να βρεθεί στο κωδ. 6.5

```
if (hit.gameObject.tag == "PacBigDot" || hit.gameObject.tag ==
    "PacDot")
{
    score += 10;
    //paixe ton ixo tou pontou
    if
        (!GameObject.Find("ChompDotSound").GetComponent<AudioSource>().isPlaying)
```

```
            GameObject.Find("ChompDotSound").GetComponent<AudioSource>().Play();

        //afairesi pacdot apo to scene
        Destroy(hit.gameObject);

        //eidiko case gia PowerPills
        if (hit.gameObject.tag == "PacBigDot")
        {
            score += 40;
            isPowerPilled = true;
            GameObject.Find("ScaredSound").GetComponent<AudioSource>().Play();
            var ghosts = GameObject.FindGameObjectsWithTag("Ghost");

            //allagi tou main texture ton ghost sto scared (blue)
               texture
            foreach (GameObject g in ghosts)
            {
                var c1 =
                   g.transform.Find("default").transform.Find("default_MeshPart0");
                var c2 =
                   g.transform.Find("default").transform.Find("default_MeshPart1");
                var components = c1.GetComponents<Renderer>();
                foreach (var component in components)
                    component.material.mainTexture = scaredtexture;

                components = c2.GetComponents<Renderer>();

                foreach (var component in components)
                    component.material.mainTexture = scaredtexture;
            }
        }
    }
```

Κώδικας C# 6.5: Σύγκρουση με τελεία

2. **Σύγκρουση του παίκτη με φάντασμα**
   Όταν ο Pacman έρθει σε σύγκρουση με κάποιο φάντασμα τότε του αφαιρείται μία ζωή. Αν δεν έχει καμία ζωή υπόλοιπο τότε το παιχνίδι σταματάει και ο παίκτης χάνει. Αν ο Pacman ακουμπίσει ένα φάντασμα ενώ προηγουμένως έχει φάει μία μεγάλη τελεία (Power Pill σεν. 1) τότε αντί να χάσει ζωή ή το παιχνίδι ”τρώει” το φάντασμα και κερδίζει 200 πόντους.

   Για την υλοποίηση της παραπάνω λογικής, αρχικά ορίζουμε μια μεταβλητή με το όνομα lives η οποία αρχικοποιείται με τον αριθμό 2. Έπειτα δημιουργούμε ένα κενό Unity GameObject και το τοποθετούμε στο κέντρο του λαβύρινθου δίνοντας του το όνομα **GhostSpawnPos** (σχ. 6.11). Αυτό μας βοηθάει στο να μεταφέρουμε γρήγορα το φάντασμα που θα ”φάει” ο Pacman στην αρχική του θέση για να ξεκινήσει ξανά απ’ την αρχή. Τέλος δίνουμε στα GameObjects που αποτελούν φαντάσματα την ετικέτα Ghost για την διευκόλυνση μας στον εντοπισμό τους. Στη περίπτωση που το αντικείμενο με το οποίο συγκρούστηκε ο Pacman έχει ετικέτα *Ghost* χειριζόμαστε τα παρακάτω σενάρια:

(α) Ο παίκτης έχει ”φάει” Power-Pill

(β) Ο παίκτης δεν έχει ”φάει” Power-Pill

i. Ο παίκτης έχει ζωές

ii. Ο παίκτης δεν έχει ζωές

Αν ο χρήστης έχει φάει Power-Pill κατά τη σύγκρουση του με ένα φάντασμα τότε, προσθέτουμε 200 πόντους και μεταφέρουμε το φάντασμα στην αρχική του θέση στο λαβύρινθο. Στη περίπτωση που ισχύει το σεν. 2(β)i αφαιρούμε μία ζωή απ’ τον παίκτη, θέτουμε την μεταβλητή *moveAvailable = false* ώστε να μείνει στάσιμος, και καλούμε μετά απο 2 δευτερόλεπτα την συνάρτηση *resetMovement()* η οποία επαναφέρει την δυνατότητα κίνησης στον παίκτη και μεταφέρει τον Pacman και τα φαντάσματα στην αρχική τους θέση στο λαβύρινθο. Διαφορετικά αν ισχύει το σεν. 2(β)ii τότε θέτουμε και πάλι την μεταβλητή *moveAvailable = false*, εμφανίζουμε στον χρήστη το κείμενο **GAME OVER** παίζοντας παράλληλα το αντίστοιχο ηχητικό εφέ και τέλος τον επιστρέφουμε στο αρχικό μενού του παιχνιδιού. Η υλοποιήση σε κώδικα C# μπορεί να βρεθεί στο κωδ. 6.6

```
if (hit.gameObject.tag == "Ghost")
{
    if (!isPowerPilled)
    {
        if (lives > 0)
        {
            lives--;
            moveAvailable = false;
            GameObject.Find("LoseLifeSound").GetComponent<AudioSource>().Play();
            GameObject.Find("GhostSound").GetComponent<AudioSource>().Stop();
            Invoke("resetMovement", 2);
        }
        else
        {
            moveAvailable = false;
            GameObject.Find("txtGameOver").GetComponent<Text>().enabled
                = true;
            GameObject.Find("GameOverSound").GetComponent<AudioSource>().Play();
            GameObject.Find("GhostSound").GetComponent<AudioSource>().Stop();
            Invoke("QuitToMenu", 2);
        }
    }
    else
    {
        score += 200;
        togglePlus200();//dikse to + 200
        Invoke("togglePlus200", 2);
        GameObject.Find("ChompSound").GetComponent<AudioSource>().Play();
        \\metafora tou ghost stin arxiki thesi
        hit.gameObject.transform.position =
            GameObject.Find("GhostSpawnPos").transform.position;
    }
}
```

Κώδικας C# 6.6: Σύγκρουση με φάντασμα

3. **Μεταφορά του παίκτη όταν περάσει μέσα από ένα portal**
Όπως περιγράφηκε στο 6.2.2 όταν ο παίκτης περάσει μέσα από μια πύλη μεταφοράς πρέπει να μεταφερθεί στο απέναντι σημείο του λαβύρινθου όπου βρίσκεται η άλλη πύλη. Για να το πετύχουμε αυτό, δίνουμε αρχικά στα δύο Portals την ετικέτα *portal* και ονομάζουμε τη κάμερα που βρίσκεται μπροστά από την πύλη με το όνομα portal1, **camPortal1** και τη κάμερα που βρίσκεται μπροστά από την πύλη με το όνομα portal2, **camPortal2**. Έπειτα ελέγχουμε αν ο παίκτης έρθει σε επαφή με αντικείμενο το οποίο έχει ετικέτα *portal* και διαπιστώνουμε αν η πύλη έχει όνομα portal1 η portal2 για να τον μεταφέρουμε στην τοποθεσία της αντίστοιχης κάμερας που βρίσκεται μπροστά από την απέναντι πύλη μεταφοράς (κωδ. 6.7).

```
if (hit.gameObject.tag == "Portal")
{
    if (hit.gameObject.name == "portal1")
        gameObject.transform.position =
           GameObject.Find("camportal2").transform.position;
    else if (hit.gameObject.name == "portal2")
        gameObject.transform.position =
           GameObject.Find("camportal1").transform.position;
}
```

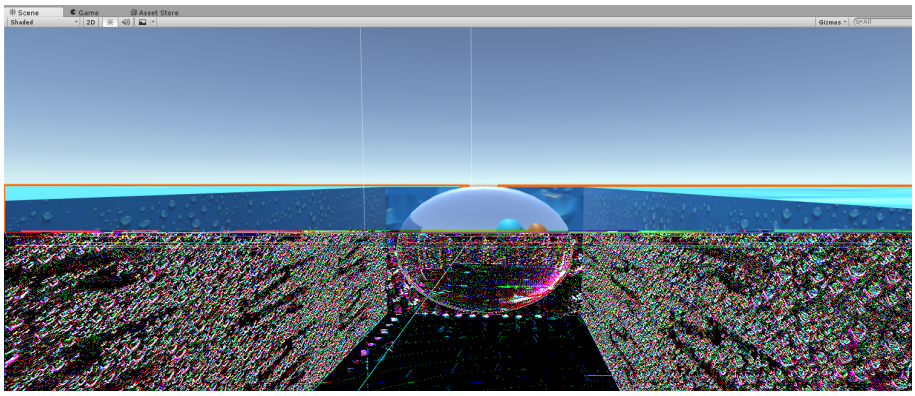
Κώδικας C# 6.7: Μεταφορά παίκτη μέσα από πύλη

6.4 Η κλάση GhostHandler.cs

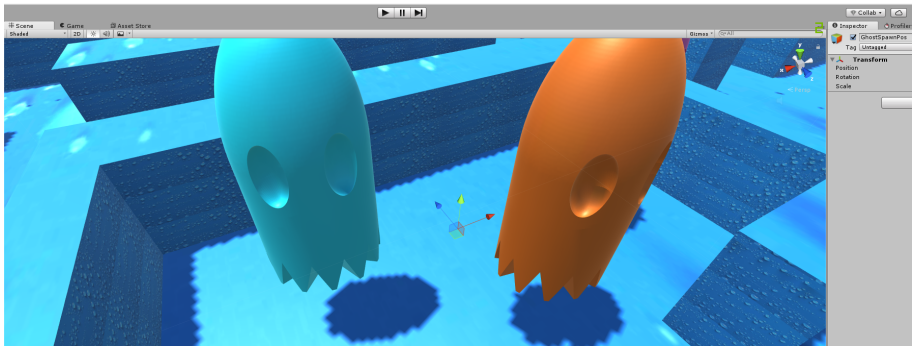
Η κλάση αυτή, όπως προδίδει και το όνομα της, είναι ενσωματωμένη στα GameObjects που αποτελούν τα φαντάσματα του παιχνιδιού. Στα περιεχόμενα της βρίσκεται η υλοποιήση του αλγόριθμου της κίνησης των εχθρών του PacMan, το οποίο συμβάλει στην βελτίωση της διαδραστικότητας που προσφέρει το παιχνίδι και το καθιστά αρκετά πιο ενδιαφέρον απ' ότι θα ήταν στη περίπτωση που τα φαντάσματα ήταν στάσιμα. Στις παρακάτω σελίδες θα αναλύσουμε την τεχνική με την οποία τα φαντάσματα κινούνται στο χώρο, το τρόπο με τον οποίο υλοποείται στο παιχνίδι μέσω του Unity και θα παρουσιάσουμε τις πιο σημαντικές συναρτήσεις της κλάσης GhostHandler.cs.

6.4.1 Η κίνηση των φαντασμάτων

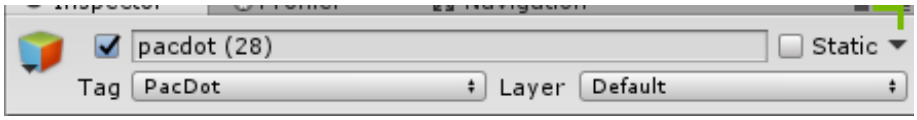
Όπως αναφέραμε στο 2.1, τα φαντάσματα (Ghosts) αποτελούν ένα εμπόδιο για τον παίκτη και αποσκοπούν στο να τον αποτρέψουν απ' το να "φάει" όλες τις κίτρινες τελείες και να κερδίσει. Ενα στατικό εμπόδιο παρ' όλ' αυτά δεν αποτελεί και τον μεγαλύτερο κίνδυνο τις περισσότερες φορές. Για το λόγο αυτό σχεδιάζουμε μια τεχνική αυτοματοποιημένης κίνησης στο χώρο για κάθε φάντασμα, με σκοπό να δυσκολέψουμε τον παίκτη και να μεταφέρουμε ένα ακόμη κύριο συστατικό του αυθεντικού Pac-Man στο παιχνίδι μας. Η τεχνική αυτή, βασίζεται στο σύστημα κίνησης με σημεία αναφοράς πορείας (waypoint system). Ένα waypoint μπορεί να είναι ένας τόπος σε μια διαδρομή ή μια γραμμή διαδρομής, ένα σημείο στάσης ή ένα σημείο στο οποίο αλλάζει η πορεία.



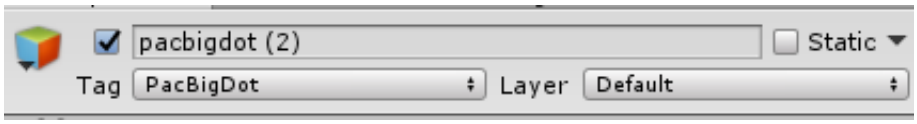
Σχήμα 6.10: Πύλη μεταφοράς στο VR Pac-Man



Σχήμα 6.11: Θέση επαναφοράς φαντασμάτων



Σχήμα 6.12: Unity pacdot tag



Σχήμα 6.13: Unity PacBigDot tag

Τοποθετώντας σημεία αναφοράς σε διάφορα μέρη στο λαβύρινθο ορίζουμε το τελικό σημείο μιας πορείας που μπορεί να ακολουθήσει ένα φάντασμα. Όσο το φάντασμα απέχει μια συγκεκριμένη απόσταση απο αυτό το σημείο μπορεί να συνεχίσει να κινείται προς αυτό. Αν το πλησιάσει με μία συγκεκριμένη απόσταση τότε ορίζουμε στο φάντασμα ένα διαφορετικό σημείο αναφοράς στο οποίο πρέπει και πάλι να φτάσει. Με αυτό το τρόπο τα φαντάσματα κινούνται συνεχώς στον εικονικό κόσμο χωρίς να χρείαζεται η παρέμβαση του ανθρώπου-χρήστη.

6.4.2 Υλοποίηση μέσω Unity 3D

Για την υλοποίηση της κίνησης των εχθρών του PacMan, αρχικά τοποθετούμε empty GameObjects σε διάφορα σημεία στον λαβύρινθο όπως απεικονίζεται στο σχ. 6.14. Έπειτα δίνουμε σε αυτά τα αντικείμενα την ετικέτα *waypoint* ώστε να μπορούμε να τα εντοπίσουμε αργότερα στον κώδικα. Για να αυτοματοποιήσουμε την κίνηση των φαντασμάτων πρός ένα waypoint χρησιμοποιούμε το NavMeshAgent, ένα βοητικό εργαλείο που παρέχει το Unity. Το NavMeshAgent δίνει την δυνατότητα κίνησης ενός GameObject προς ένα στόχο αυτόματα και παρέχει διάφορες λειτουργικότητες όπως αποφυγή εμποδίων κ.α.. Στη συνέχεια ορίζουμε τέσσερις μεταβλητές (κωδ. 6.8) οι οποίες είναι απαραίτητες για τον χειρισμό της κίνησης του κάθε φαντάσματος. Παρακάτω θα περιγράψουμε πως αυτές οι μεταβλητές σε συνδυασμό με τις συναρτήσεις void Start() και void Update() χρησιμοποιούνται ώστε να υλοποιηθεί η αυτοματοποιημένη κίνηση των φαντασμάτων.

6.4.3 Η συνάρτηση Start

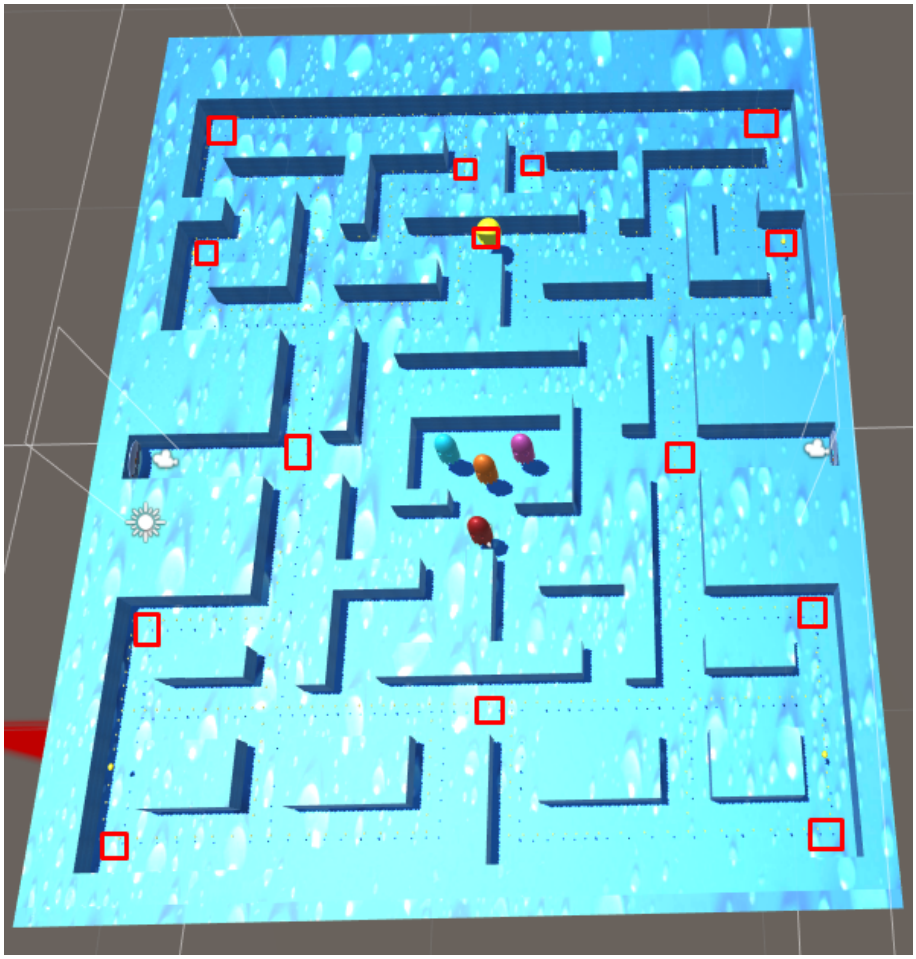
Η συνάρτηση Start καλείται αυτόματα όταν η κλάση-script του GameObject ενεργοποιηθεί. Καλείτε πριν τη συνάρτηση Update() και χρησιμοποείται για την αρχικοποιήση μιας κλάσης. Τρέχει μόνο μία φορά για κάθε κλάση και αποτελεί μια πολύ σημαντική συνάρτηση.

Στη συνάρτηση αυτή αρχικοποιούμε τις μεταβλητές (κωδ. 6.8). Αρχικά στον πίνακα *ghosts* βάζουμε όλα τα φαντάσματα εκτός απ' τον φάντασμα "ιδιοκτήτη" της κλάσης. Έπειτα στο πίνακα *waypoints* βάζουμε όλα τα waypoints που βρίσκονται στο λαβύρινθο εντοπίζοντας τα με την ετικέτα *waypoint*. Τέλος στη μεταβλητή currentWp (current waypoint), η οποία αποτελεί το σημείο αναφοράς στο οποίο πρέπει να φτάσει κάθε φάντασμα, ορίζουμε τυχαία ένα σημείο αναφοράς απ' το πίνακα *waypoints* με τη βοήθεια του property *int getNewWaypoint()* η οποία επιστρέφει τυχαία έναν αριθμό απ' το 0 εώς το μέγεθος του πίνακα *waypoints*. Ο κώδικας της συνάρτησης Start και του property getNewWaypoint απεικονίζεται στο κωδ. 6.9.

```
//to waypoint pros to opoio kineitai to fantasma
public GameObject currentWp {get; set;};
//pinakas me ola ta waypoints
private GameObject[] waypoints;
//pinakas me ola ta ghost(!this)
private GameObject[] ghosts;
```

Κώδικας C# 6.8: Properties της κλάσης GhostHandler.cs

```
void Start()
{
    ghosts = GameObject.FindGameObjectsWithTag("Ghost")
                       .Cast<GameObject>()
                       .Where(g=> g.name!= this.name)
```



Σχήμα 6.14: Σημεία αναφοράς στο λαβύρινθο

```
                        .ToArray();
    waypoints = GameObject.FindGameObjectsWithTag("Waypoint");
    currentWp = waypoints[this.getNewWaypoint];
}

private int getNewWaypoint
{
    get
    {
        return Random.Range(0, waypoints.Length - 1);
    }
}
```

Κώδικας C# 6.9: Συνάρτηση Start

6.4.4 Η συνάρτηση Update

Όπως και για τη κλάση MainController.cs η συνάρτηση Update παίζει ενα πολύ σημαντικό ρόλο για την GhostHandler.cs καθώς είναι υπεύθυνη για τον χειρισμό της κίνησης των φαντασμάτων. Εδώ ορίζουμε το σημείο αναφοράς το οποίο πρέπει να φτάσει το κάθε φάντασμα όσο το παιχνίδι εξελίσσεται. Επιπλέον ελέγχουμε αν η απόσταση του απ' το σημείο αναφοράς είναι μικρότερη από ένα συγκεκριμένο αριθμό όπου και του αλλάζουμε το σημείο το οποίο πρέπει να φτάσει.

Με τη βοήθεια του εργάλειου NavMeshAgent και συγκεκριμένα θέτοντας τη μεταβλήτη του *destination* ίση με το currentWp, το GameObject-Ghost αρχίζει να κινείται αυτόματα προς αυτό το waypoint. Για να ελέγξουμε αν το φάντασμα έχει πλησιάσει χρησιμοποιούμε τη συνάρτηση *Distance()* η οποία ανοίκει στον τύπο-struct Vector3 και ουσιαστικά αφαιρεί το διάνυσμα του currentWp (x,y,z) απ' αυτό του φαντάσματος και επιστρέφει τη ρίζα του (x*x+y*y+z*z) του τελικού διανύσματος. Αν αυτός ο αριθμός είναι μικρότερος απο 0.7 τότε θέτουμε το currentWp ίσο με ένα νέο τυχάιο waypoint. Ο κώδικας τη συνάρτησης Update απεικονίζεται στο κωδ. 6.10

```
void Update()
{
    float distance = Vector3.Distance(transform.position ,
       currentWp.transform.position);
    //otan plisiasei sto current waypoint
    if (distance < 0.7)
        currentWp = waypoints[getNewWaypoint()];

    //an plisiasei kapoio allo ghost
    foreach(var g in ghosts)
    {
        if((transform.position - g.transform.position).magnitude < 0.6)
            currentWp = waypoints[getNewWaypoint()];
    }
    //otan to intro song teleiwsei , ksekinane ta ghosts
    if
       (!GameObject.Find("pacman").GetComponent<AudioSource>().isPlaying)
        transform.GetComponent<NavMeshAgent>().destination =
           currentWp.transform.position;
}
```

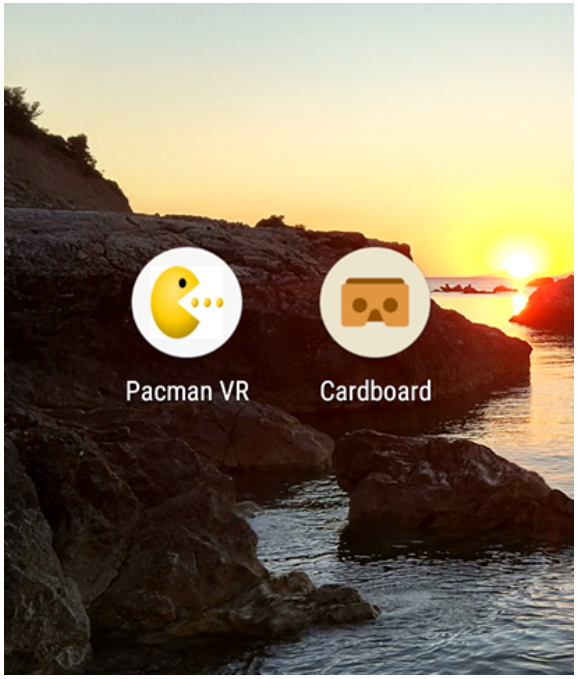
Κώδικας C# 6.10: Συνάρτηση Update

Κεφάλαιο 7

Χρήση του VR Pac-Man

7.1 Εγκατάσταση VR Pac-Man

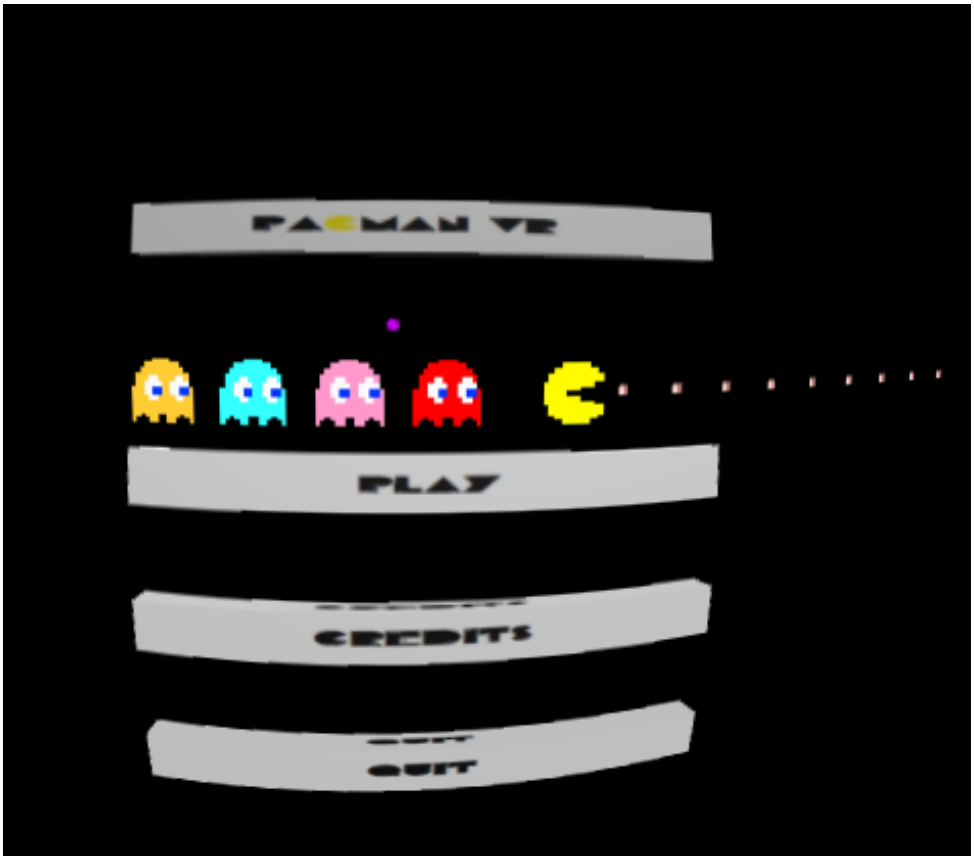
Σε αυτό το σημείο θα δείξουμε στους αναγνώστες πως να κατεβάσουν την εφαρμογή μας, VR Pac-Man, μέσα από μία σειρά φωτογραφιών και γραπτών οδηγιών.



Σχήμα 7.1: Οθόνη κινητού με τις απαραίτητες εφαρμογές

1. Η εφαρμογή μας προς το παρών είναι συμβατή μόνο με λειτουργικά συστήματα Android. Αφού ανοίξετε το κινητό σας και προηγηθείτε στον Browser που χρησιμοποιείτε, μπορείτε να μεταβείτε στο παρακάτω link για να βρείτε και να κατεβάσετε τη τελευταία **stable** έκδοση (v3.6) του παιχνιδιού μας https://drive.google.com/open?id=1eUBc2Cg8QMArOU-wUGwmvqMaLRzj
   Το ιστορικό των εκδόσεων του VR Pac-Man μπορεί να βρεθεί εδώ : VR Pac-Man Releases ή https://drive.google.com/open?id=1XgAXAxvmJy8XAweVvcT8Kp7bZvaVhKdG

2. Αφού κατεβάσετε την εφαρμογή απτό παραπάνω link θα χρειαστεί να προηγηθείτε στο Play



Σχήμα 7.2: Ανοίγοντας την εφαρμογή.

Store και να κατεβάσετε την εφαρμογή Cardboard, ο ρόλος της οποίας είναι να προσαρμόσει το κινητό σας με το Cardboard.

3. Εφοσον εχετε ολοκληρώσει τα παραπάνω βήματα αυτό που θα πρέπει να βλέπετε στην οθόνη του κινητού σας απεικονίζεται στην εικόνα 7.1.

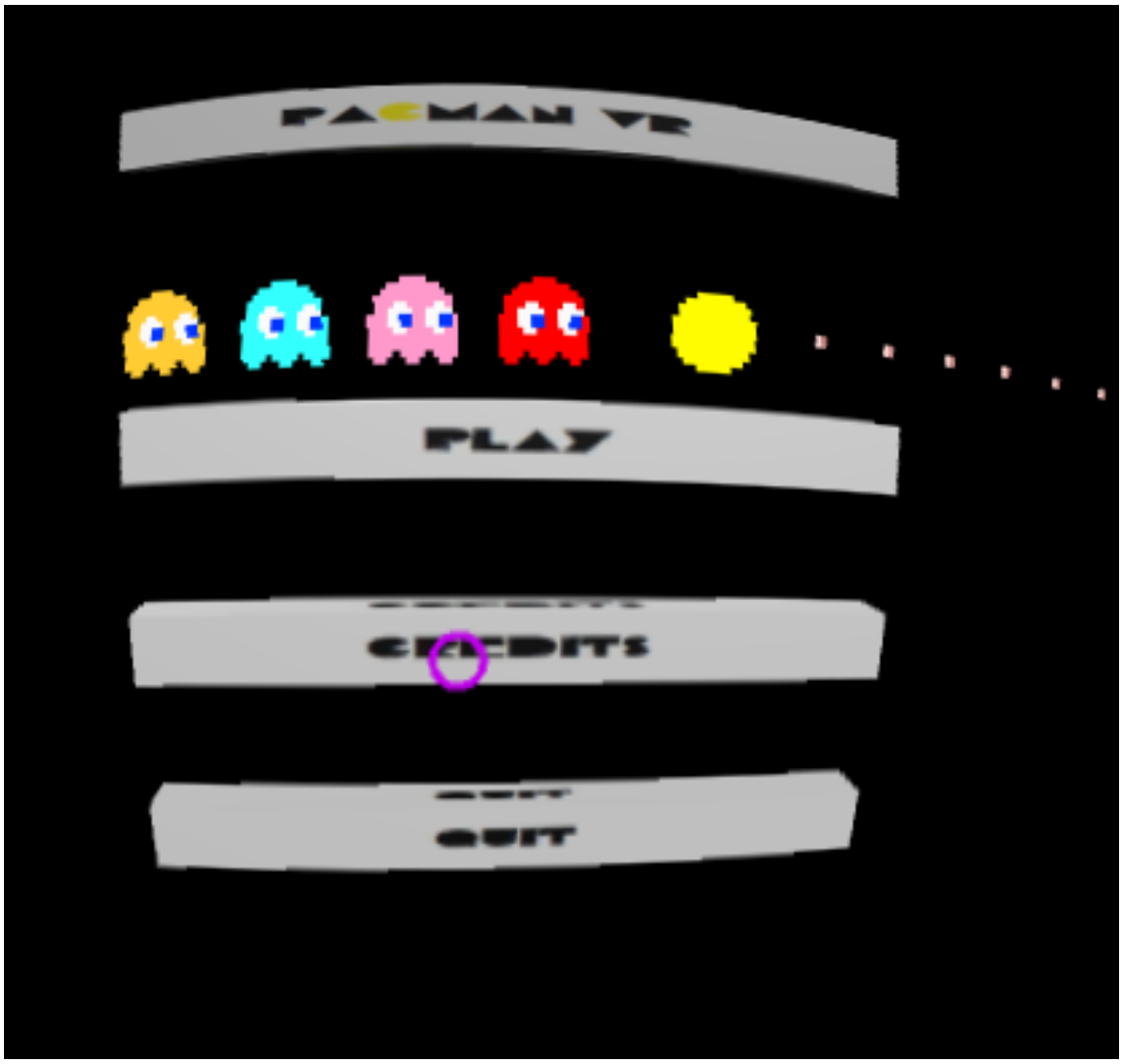
4. Πλέον είστε έτοιμοι να ανοίξετε την εφαρμογή μπορείτε να παίξετε το παιχνίδι μας.

5. Enjoy!

7.2 Εκτέλεση VR Pac-Man

Αφού ανοίξουμε την εφαρμογή και προσαρμώσουμε το Google Cardboard στο κεφάλι μας, το πρώτο πράγμα που βλέπουμε είναι το Menu όπως φαίνεται και στην εικόνα 7.2. Το Menu αποτελείτε από τρία κουμπιά : Play, Credits, Quit. Αρχικά το πρώτο πράγμα παρατηρούμε είναι μια ροζ κουκίδα στο κέντρο. Η κουκίδα αυτή είναι ουσιαστικά ο αντίστοιχος κέρσορας ενός ηλεκτρονικού υπολογιστή και η λειτουργία της είναι αντίστοιχη με αυτή ενός ποντικιού, που αντί να ανταποκρίνετε με το πάτημα ενός κλικ, ανταποκρίνετε με βάση τον χρόνο που μένει πάνω από ένα κουμπί. Ο λόγος που δημιουργήθηκε είναι για να μπορέσει ο χρήστης να ”πατήσει” τα κουμπιά του μενού.

Όπως μπορούμε να δούμε και στην εικόνα 7.3 η κουκίδα όταν μένει πάνω από ένα ”κουμπί” ανοίγει και σχηματίζει έναν κύκλο. Μετά από 5 δευτερόλεπτα και αφού η κουκίδα έχει γίνει πλέον κύκλος, ο χρήστης μεταφέρεται στην ανάλογη λειτουργία του κουμπιού. Είναι ένας έξυπνος και



Σχήμα 7.3: Η κουκίδα γίνεται κύκλος μετά απο 5 δευτερόλεπτα.

διαδραστικός τρόπος να πλοηγηθεί ένας χρήστης Google Cardboard σε μία εφαρμογή εικονικής πραγματικότητας αντί να κουνήσει τον μαγνήτη που υπάρχει στο πλάι του Cardboard.

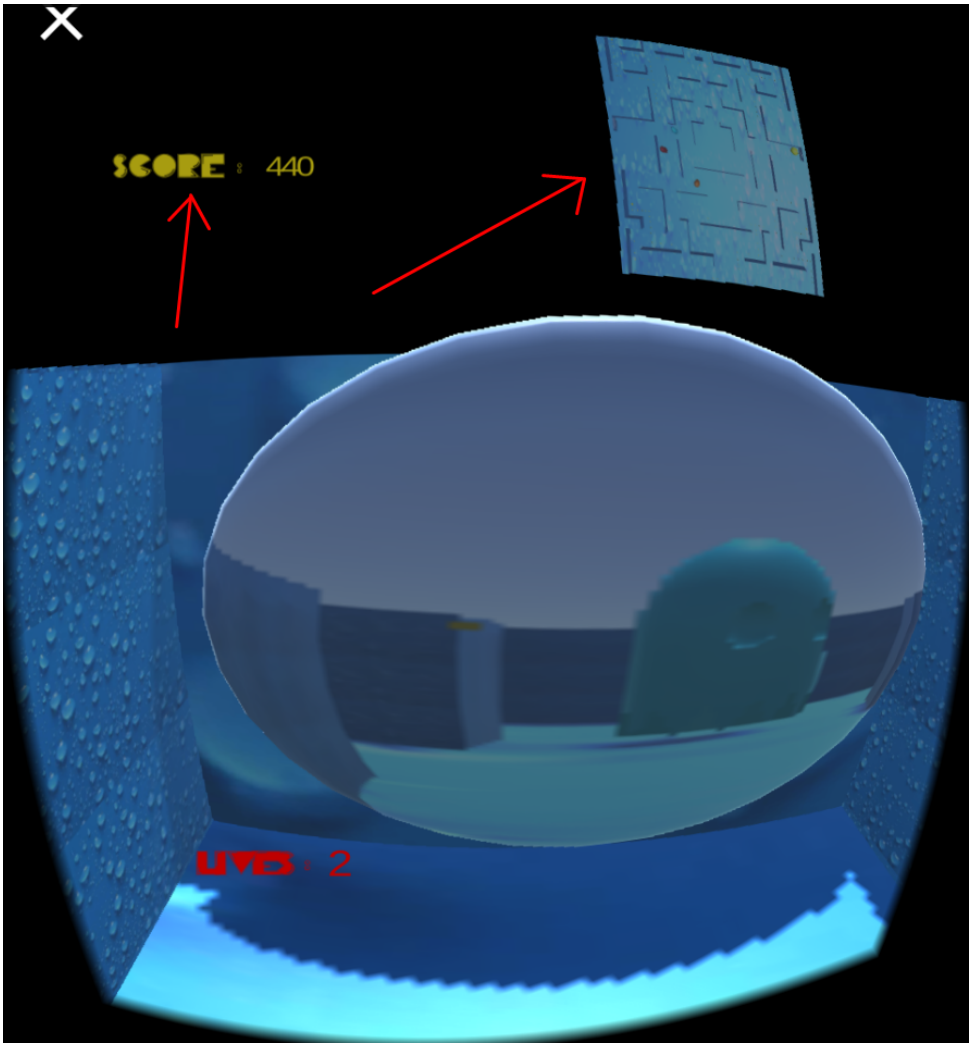
Στην εικόνα 7.4 μπορούμε να δούμε τα credits που δημιουργήσαμε. Αφού ο χρήστης έχει αφήσει την κουκίδα πάνω απτο κουμπί "Credits" θα μεταφερθεί στον παρακάτω εικονικό χώρο και μία μουσική που θυμίζει σε πολλούς έναν άλλο εικονικό κόσμο θα αρχίσει να παίζει. Να σημειώσουμε εδώ ότι και η αρχική οθόνη και τα credits είναι "κινούμενα" και όχι έτσι όπως φαίνεται στις φωτογραφίες, αλλά αυτό θα το παρατηρήσουμε κατά την διάρκεια της χρήσης της εφαρμογής.

Κάνοντας την αντίστοιχη κίνηση, αφήνοντας δηλαδή την κουκίδα πάνω απτο κουμπί Play μεταφερόμαστε στο παιχνίδι. Όπως βλέπετε στην φωτογραφία 7.5, η αρχική οθόνη μοιάζει κάπως έτσι. Το κλασσικό μήνυμα "Get Ready" εμφανίζεται στην οθόνη και μετά από 2 δευτερόλεπτα, αφού ακουστεί ο χαρακτηριστικός ήχος του Pacman, το παιχνίδι ξεκινάει.

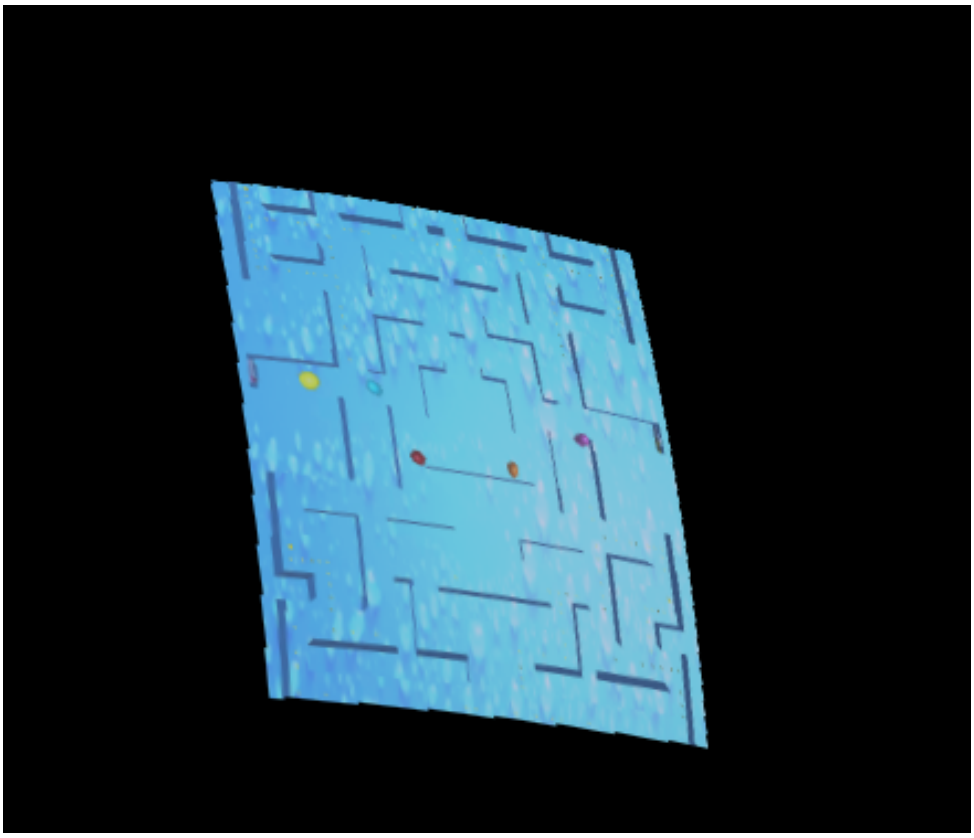
Τα πρώτα δυο πράγματα που παρατηρούμε είναι το Score και ο χάρτης (εικόνα 7.6). Το Score βρίσκεται στην πάνω αριστερή γωνία και αυξάνει κατά 10 κάθε φορά που ο Pacman "τρώει" ένα τυράκι. Αν ωστόσο φάει την μεγάλη μπάλα που υπάρχει στις 4 γωνίες του χάρτη, τα φαντάσματα θα αλλάξουν χρώμα και θα γίνουν μπλε και ο Pacman ακουμπώντας τα μπορεί να τα σκοτώσει και

Σχήμα 7.4: Τα Credits του παιχνιδιού.

Σχήμα 7.5: Μπαίνοντας στο παιχνίδι.



Σχήμα 7.6: Το Score και ο χάρτης.



Σχήμα 7.7: Ο χάρτης σε κοντινό πλάνο.

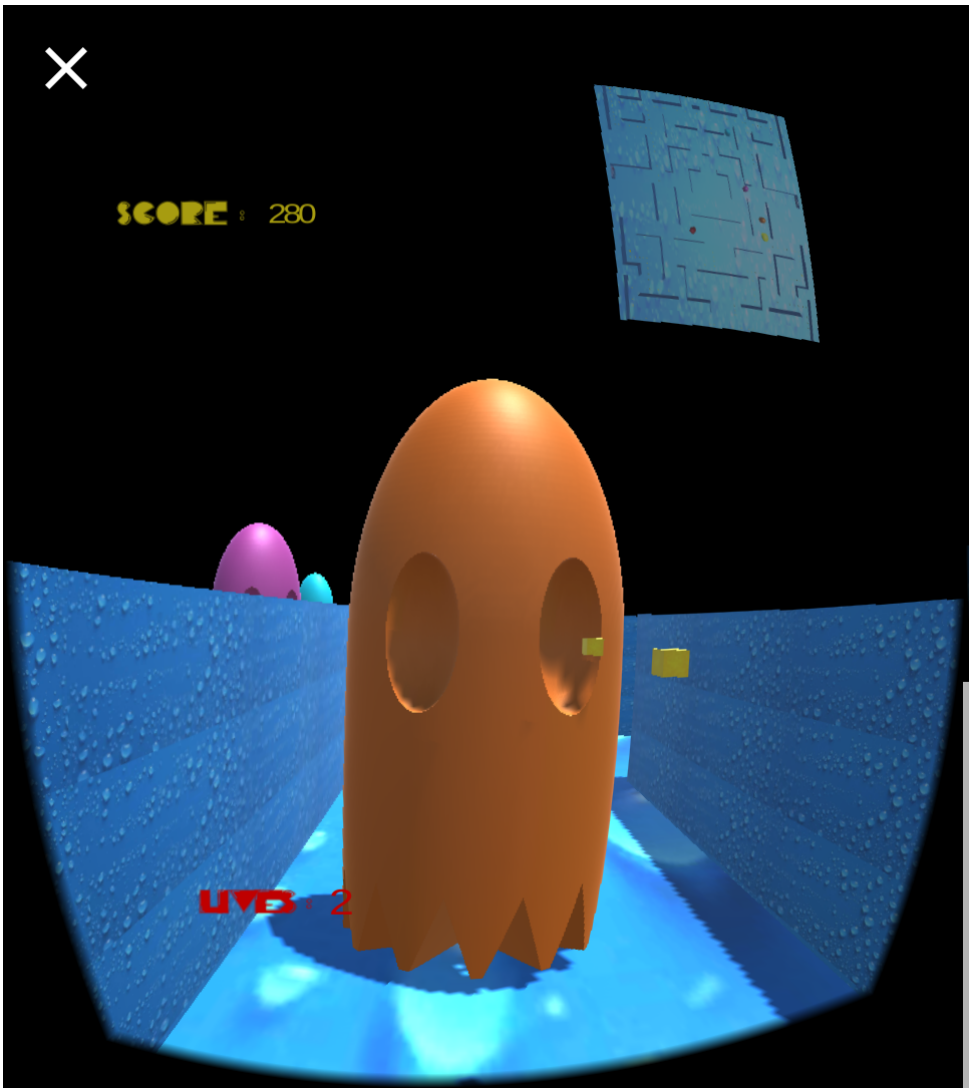
να κερδίσει έξτρα 200 πόντους για το καθένα.

Στην εικόνα 7.7 βλέπουμε μια κοντινή φωτογραφία του χάρτη. Επιλέξαμε να δημιουργήσουμε τον χάρτη διότι όταν ο χρήστης παίζει σε First Person με το Google Cardboard δεν θα μπορούσε να έχει την δυνατότητα να ξέρει που είναι τα φαντάσματα καθώς ουσιαστικά «βλέπει» μέσα από τον Pacman. Έχει τοποθετηθεί στο συγκεκριμένο σημείο έτσι ώστε να μην εμποδίζει το οπτικό πεδίο του χρήστη και ταυτόχρονα να μπόρει ο χειρηστής του Pacman να καταλάβει τη θέση του αλλά και την θέση των φαντασμάτων.

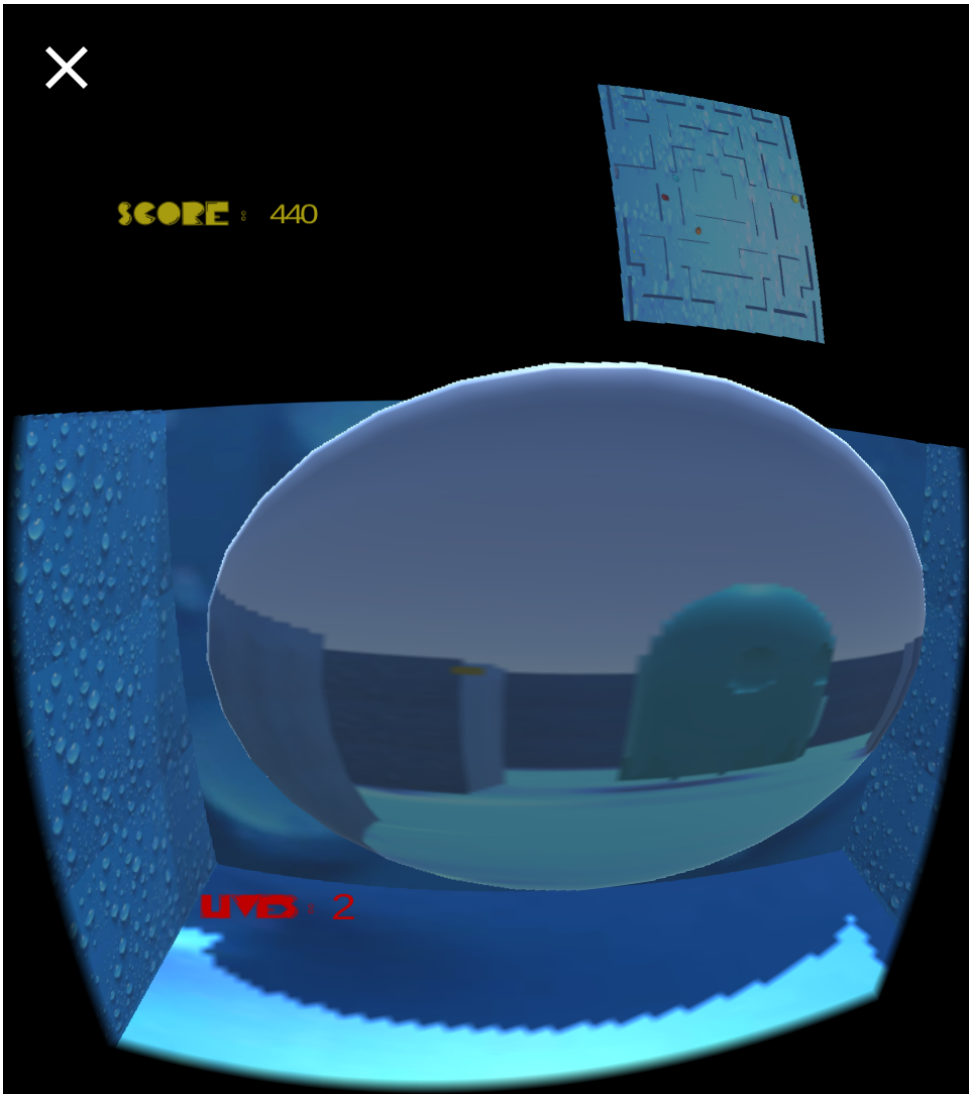
Τα φαντασματάκια μοιάζουν κάπως έτσι (εικόνα 7.8) και είναι χτισμένα γύρω απτο μοντέλο που δείξαμε στο προηγούμενο κεφάλαιο. Κινούνται προς τυχαίες κατευθύνσεις και σκοπός τους είναι να πίασουν τον Pacman. Η ταχύτητα με την οποία κινούνται ωστόσο είναι ελάχιστα πιο χαμηλή από του Pacman έτσι ώστε να είναι λίγο πιο εύκολο στον χρήστη να τα αποφύγει. Ο λόγος που σχεδιάστηκε έτσι η λειτουργία τους είναι έτσι ώστε στο ενδεχόμενο το οποίο ο χρήστης χρειαστεί να κάνει αναστροφή για να γλυτώσει, να έχει αρκετό χρόνο να γυρίσει το κεφάλι του και να τρέξει.

Κάπου κοντά στο κέντρο της πίστας μπορούμε να δούμε οτι και στις δύο πλευρές υπάρχει από ένα ”Portal”. Η χρησιμότητα του Portal, είναι να μπορούμε να τηλεμεταφερθούμε στην ακριβώς απέναντι μεριά από αυτή που είμαστε. Για να γίνει αυτό απλά προχωράμε ευθεία στο σημείο που βλέπουμε στην εικόνα 7.9 και αυτόματα μεταφερόμαστε στην αντίθετη πλευρά.

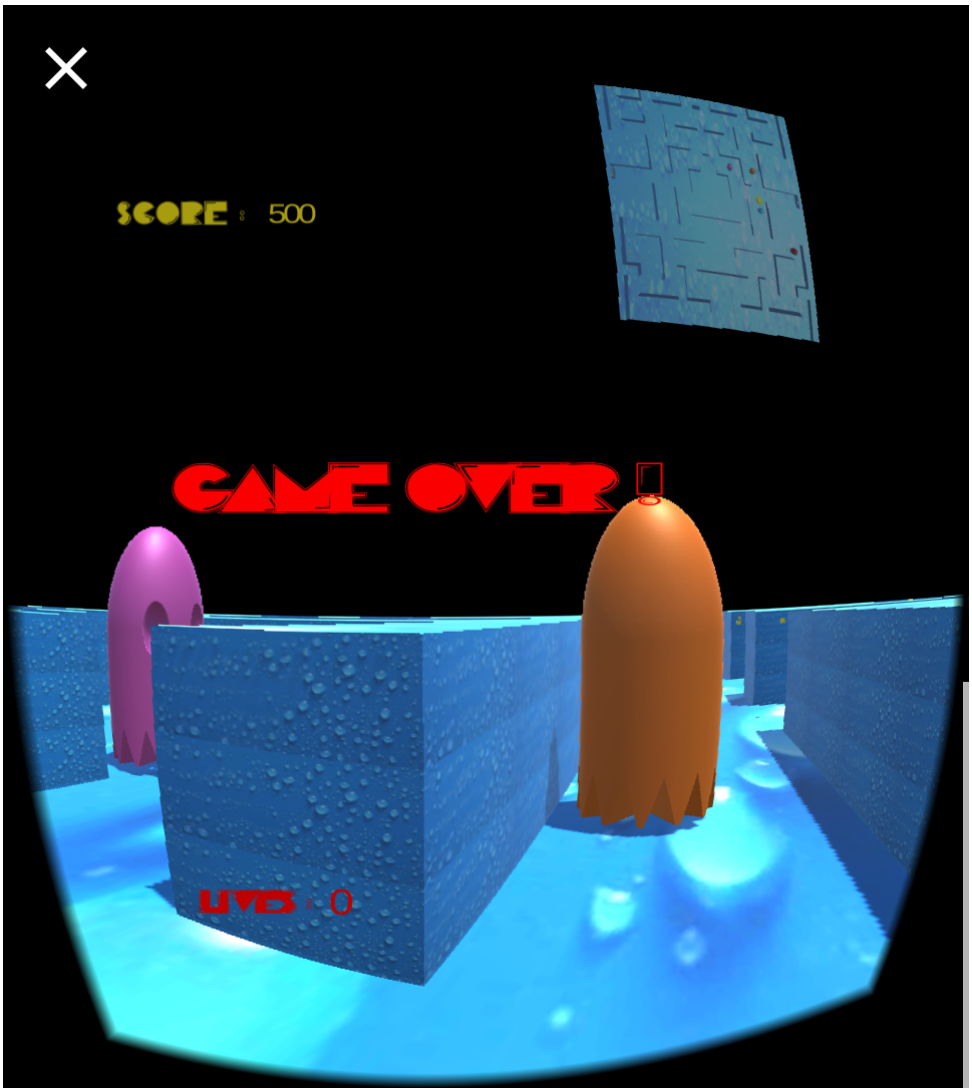
Αν σας αγγίξει κάποιο φάντασμα ενώ δεν έχετε υπόλοιπο ζωών το μήνυμα που φαίνεται στην εικόνα 7.10 θα φανεί στην οθόνη και θα μεταφερθείτε στο αρχικό Menu.



Σχήμα 7.8: Ένα εκ των φαντασμάτων.



Σχήμα 7.9: Ένα εκ των δύο Portal.



Σχήμα 7.10: Το μήνυμα Game Over.

Κεφάλαιο 8

Συμπεράσματα

Σε αυτή τη πτυχιακή εργασία αναπτύξαμε ένα παιχνίδι εικονικής πραγματικότητας, το οποίο βασίζεται στο γνωστό παιχνίδι Pac-Man, για τη χρήση του με το μέσο προβολής περιεχομένου εικονικής πραγματικότητας Google Cardboard. Το τελικό αποτέλεσμα μετατρέπει το κλασσικό δυσδιάστατο παχνίδι σε τρισδιάστατο και δίνει την δυνατότητα στον χρήστη να βρεθεί στη θέση του χαρακτήρα Pac-Man σε πρώτο πρόσωπο, επαυξάνοντας την εμπειρία του παιχνιδιού και εξαλείφοντας την ανάγκη χρήσης πρόσθετου εξοπλισμού όπως ενός joystick.

Παίζοντας και εξερευνώντας το παιχνίδι που σχεδιάσαμε, παρατηρούμε το πόσο διαδραστικό μπορεί να αποτελέσει ένα περιβάλλον εικονικής πραγματικότητας σε συνδυασμό με τη χρήση του Google Cardboard. Απ' τη πρώτη στιγμή που καλούμαστε να χειριστούμε τον Pac-Man σε πρώτο πρόσωπο, συνειδητοποιούμε ότι το VR Pac-Man προσφέρει μια εμπειρία παρόμοια με αυτή του κλασσικού παιχνιδιού καταφέρνοντας παράλληλα να την αναβαθμίσει σε διάφορα σημεία. Συγκεκριμένα, μας εκπλήσει το γεγονός οτι μας προκαλεί διάφορα συναισθήματα όπως αυτό του άγχους, όταν για παράδειγμα μας κυνηγάει ένα φάντασμα ή όταν δεν γνωρίζουμε τι θα αντικρίσουμε σε κάθε γωνία του λαβύρινθου.

Ολοκληρώνοντας κρίνουμε σημαντικό να αναφερθούμε σε κάποιες βελτιώσεις που θα μπορούσαν να υλοποιηθούν. Το Pac-Man είναι ένα από τα πρώτα παιχνίδια το οποίο χρησιμοποιούσε ένα είδος τεχνητής νοημοσύνης για τα φαντάσματα το οποίο είναι ένα στοιχείο που λείπει απ' την εφαρμογή μας όπου τα φαντάσματα κινούνται τυχαία στο χώρο. Μια υλοποιήση σαν και αυτή καθιστά τα φάντασματα περισσότερο επίκυνδυνα και συνεπώς συμβάλει στην ενίσχυση της συνολικής εμπειρίας που προσφέρει το παιχνίδι. Μια ακόμα βελτίωση που θα μπορούσαμε να σημειώσουμε είναι η εισαγωγή επιπλέον επιπέδων όπως και στο κλασσικό Pac-Man όπου ο παίκτης συνεχίζει σε νέο λαβύρινθο όταν κερδίσει. Τέλος θα θεωρούσαμε σημαντική την ανάπτυξη του συγκεκριμένου παιχνιδιού και σε λειτουργικά IOs καθώς μέχρι στιγμής η εφαρμογή διανέμεται μόνο σε λειτουργικά Android.

Βιβλιογραφία

[1] IGN. «The Evolution Of Virtual Reality». Στο: (Απρ. 2016). URL: http://www.ign.com/articles/2016/01/14/the-evolution-of-virtual-reality.

[2] redorbit.com. «The History Of Virtual Reality». Στο: (Φεβ. 2017). URL: http://www.redorbit.com/reference/the-history-of-virtual-reality.

[3] Unity. *Unity Gallery*. URL: https://unity3d.com/showcase/gallery.

[4] Wikipedia. *Blender (software)*. Φεβ. 2018. URL: https://en.wikipedia.org/wiki/Blender_(software).

[5] Wikipedia. *Google Cardboard*. Δεκ. 2017. URL: https://en.wikipedia.org/wiki/Google_Cardboard.

[6] Wikipedia. *Microsoft Visual Studio*. Φεβ. 2018. URL: https://en.wikipedia.org/wiki/Microsoft_Visual_Studio.

[7] Wikipedia. *Pac-Man*. Φεβ. 2018. URL: https://en.wikipedia.org/wiki/Pac-Man.

[8] Wikipedia. *Unity (game engine)*. Φεβ. 2018. URL: https://en.wikipedia.org/wiki/Unity_(game_engine).