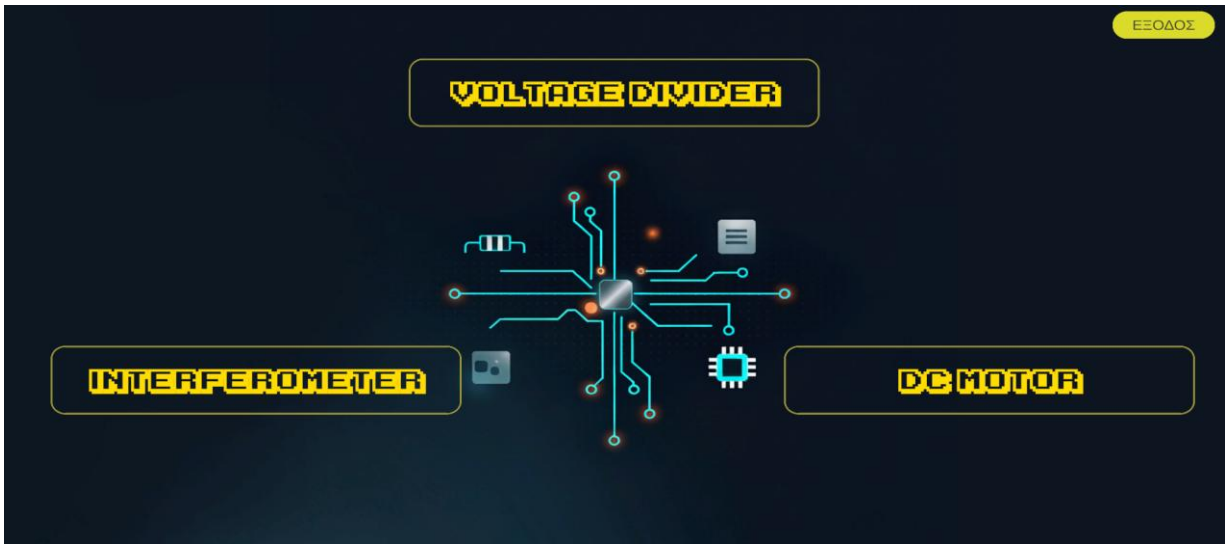
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

« Ανάπτυξη εφαρμογής με διαδραστικά εκπαιδευτικά πειράματα ηλεκτρονικών κυκλωμάτων »



Του φοιτητή
Βασιλιάν Μάρα
Αρ. Μητρώου: 518078

Επιβλέπων
Ιορδάνης Κιοσκερίδης
Βαθμίδα : Καθηγητής

Ημερομηνία 21/01/2026

Ανάπτυξη εφαρμογής με διαδραστικά εκπαιδευτικά πειράματα ηλεκτρονικών κυκλωμάτων

25121

Βασιλιάν Μάρα
Ιορδάνης Κιοσκερίδης

18/02/2025

21/01/2026

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

Η παρούσα διπλωματική εργασία είναι αφιερωμένη στους γονείς μου, για την ανεκτίμητη αγάπη, τη συνεχή υποστήριξη και την πίστη που μου έδωσαν σε κάθε βήμα αυτής της διαδρομής. Στην αδερφή μου, για την έμπνευση και την ενθάρρυνση. Αλλά και στους φίλους και συναδέλφους που γνώρισα στο Πανεπιστήμιο, γιατί έκαναν το ταξίδι των σπουδών μου πιο όμορφο, γεμάτο αξέχαστες στιγμές και νόημα.

«Αφιέρωση»

Πρόλογος

Η παρούσα διπλωματική εργασία έχει ως στόχο τη δημιουργία μιας εφαρμογής για διαδραστικά εργαστήρια ηλεκτρονικής. Η βασική ιδέα ήταν να φτιαχτεί ένα εικονικό περιβάλλον όπου ο χρήστης μπορεί να πειραματίζεται με ηλεκτρονικά κυκλώματα με ασφάλεια, ευκολία και χωρίς να χρειάζεται φυσικό εξοπλισμό. Με αυτόν τον τρόπο, η μάθηση γίνεται πιο ευχάριστη και πιο προσιτή σε όλους.

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το Unity, ένα εργαλείο που προσφέρει πολλές δυνατότητες για τη δημιουργία διαδραστικού και ρεαλιστικού περιβάλλοντος. Μέσα από το Unity χτίστηκε όλη η λειτουργία της εφαρμογής και οργανώθηκε ο τρόπος με τον οποίο ο χρήστης αλληλεπιδρά. Ο προγραμματισμός έγινε με C#, μια γλώσσα που βοήθησε στη σωστή λειτουργία των κυκλωμάτων.

Τα τρισδιάστατα μοντέλα όπως τα ηλεκτρονικά εξαρτήματα και ο χώρος του εργαστηρίου δημιουργήθηκαν στο Blender, ένα πρόγραμμα που βοήθησε στη δημιουργία ρεαλιστικών και λεπτομερών αντικειμένων, ενώ τα γραφικά και οι υφές σχεδιάστηκαν στο Krita, ένα εργαλείο που βοήθησε στο να σχεδιαστούν και να χρωματιστούν όλα τα οπτικά στοιχεία της εφαρμογής.

Η εφαρμογή έχει στόχο να δείξει πόσο σημαντική είναι η ψηφιοποίηση της εκπαίδευσης, προσφέροντας έναν σύγχρονο και ασφαλή τρόπο εκμάθησης βασικών πειραμάτων ηλεκτρονικής. Μέσα από το εικονικό περιβάλλον, οι χρήστες μπορούν να εξασκούνται, να πειραματίζονται και να μαθαίνουν με διαδραστικό και ευχάριστο τρόπο.

Περίληψη

Η παρούσα πτυχιακή εργασία αφορά την ανάπτυξη ενός διαδραστικού ψηφιακού εργαστηρίου ηλεκτρονικής και φυσικής, υλοποιημένου στο περιβάλλον Unity, με στόχο την υποστήριξη της εκπαιδευτικής διαδικασίας μέσω προσομοιώσεων υψηλής αλληλεπίδρασης. Το σύστημα περιλαμβάνει τρία ολοκληρωμένα πειράματα, σχεδιασμένα ώστε να αναπαριστούν βασικές αλλά και πιο εξειδικευμένες εργαστηριακές έννοιες. Το πρώτο πείραμα αφορά τον διαιρέτη τάσης και τον χρωματικό κώδικα αντιστάσεων, επιτρέποντας στους χρήστες να πειραματιστούν με τιμές αντιστάσεων και να παρατηρήσουν σε πραγματικό χρόνο τις μεταβολές στην έξοδο του κυκλώματος. Το δεύτερο πείραμα προσομοιώνει το συμβολόμετρο Michelson, προσφέροντας τη δυνατότητα οπτικοποίησης συμβολικών προτύπων και κατανόησης βασικών φαινομένων συμβολής. Το τρίτο πείραμα παρουσιάζει τη λειτουργία ενός κινητήρα συνεχούς ρεύματος (DC motor), επιτρέποντας στον μαθητή να εξερευνήσει την αλληλεπίδραση μεταξύ μαγνητικού πεδίου, ρεύματος και περιστροφικής κίνησης.

Κάθε πείραμα συνοδεύεται από ενσωματωμένο θεωρητικό υλικό, έτσι ώστε ο χρήστης να μπορεί να μελετήσει τη σχετική θεωρία πριν ή κατά τη διάρκεια της προσομοίωσης, ενισχύοντας την αυτοκατευθυνόμενη μάθηση. Το διαδραστικό περιβάλλον έχει υλοποιηθεί σε C#, ενώ για τη δημιουργία μέρους των τρισδιάστατων μοντέλων χρησιμοποιήθηκε το λογισμικό Blender, εξασφαλίζοντας ρεαλιστική και λειτουργική απεικόνιση των αντικειμένων. Το αποτέλεσμα είναι ένα ολοκληρωμένο ψηφιακό εργαστήριο που προσφέρει ένα ασφαλές, ευέλικτο και παιδαγωγικά αποτελεσματικό μέσο κατανόησης ηλεκτρονικών και φυσικών εννοιών. Η εργασία καταδεικνύει τον σημαντικό ρόλο των εικονικών προσομοιώσεων στη σύγχρονη εκπαίδευση και την αξία τους ως συμπληρωματικό εργαλείο στα παραδοσιακά εργαστήρια.

Development of an application with interactive educational experiments on electronic circuits

Vasilian Mara

Abstract

This thesis presents the development of an interactive digital laboratory for electronics and physics, implemented in the Unity environment, with the aim of supporting the educational process through highly interactive simulations. The system includes three fully functional experiments designed to represent both fundamental and more advanced laboratory concepts. The first experiment focuses on the voltage divider and the resistor color code, allowing users to modify resistor values and observe real-time changes in the circuit's output. The second experiment simulates the Michelson interferometer, offering visualization of interference patterns and facilitating the understanding of key wave interference phenomena. The third experiment demonstrates the operation of a direct current (DC) motor, enabling learners to explore the interaction between magnetic fields, electric current, and rotational motion.

Each experiment is accompanied by embedded theoretical material, allowing users to study relevant concepts before or during the simulation, thus promoting self-directed learning. The interactive environment was developed using C#, while part of the 3D models was created in Blender to ensure realistic and functional visual representation. The outcome is a complete digital laboratory that provides a safe, flexible, and pedagogically effective tool for understanding electronic and physical principles. The thesis highlights the significant role of virtual simulations in modern education and their value as a complementary tool to traditional laboratory practices.

Ευχαριστίες

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή μου, κ. Ιορδάνη Κιοσκερίδη, καθηγητή του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος. Η καθοδήγησή του και η εμπιστοσύνη του ήταν απαραίτητες για την ολοκλήρωση της παρούσας εργασίας. Επίσης, ευχαριστώ το σύνολο του διδακτικού προσωπικού του Τμήματος για τις πολύτιμες γνώσεις που μου παρείχε. Τέλος, απευθύνω ευχαριστίες σε όλους όσους με στήριξαν ηθικά και με ενθάρρυναν κατά τη διάρκεια της συγγραφής.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κατάλογος Εικόνων

Εισαγωγή

Η ραγδαία ανάπτυξη της τεχνολογιάς έχει αλλάξει σημαντικά τον τρόπο με τον οποίο πραγματοποιείται η εκπαίδευση την τελευταία δεκαετία. Στον τομέα της ηλεκτρονικής, όπου η κατανόηση και ο πειραματισμός με κυκλώματα αποτελούν βασικό πυλώνα εκμάθησης, η αξιοποίηση λογισμικών προσομοίωσης έχει αναδειχθεί σε ένα εξαιρετικά αποτελεσματικό εκπαιδευτικό εργαλείο [4]. Τα εικονικά εργαστήρια (virtual laboratories) επιτρέπουν την αναπαράσταση ηλεκτρονικών κυκλωμάτων και πειραματικών διαδικασιών σε ένα ασφαλές και ευέλικτο ψηφιακό περιβάλλον, ενισχύοντας την κατανόηση θεωρητικών αρχών και την ανάπτυξη πρακτικών δεξιοτήτων [2].

Σύγχρονες μελέτες δείχνουν ότι τα εικονικά εργαστήρια μπορούν να λειτουργήσουν συμπληρωματικά των φυσικών εργαστηρίων, ενισχύοντας την ενεργό μάθηση και την εννοιολογική κατανόηση των φοιτητών [1]. Η δυνατότητα επαναλαμβανόμενων πειραμάτων χωρίς περιορισμούς υλικών ή κινδύνους, καθώς και η άμεση οπτικοποίηση της συμπεριφοράς των κυκλωμάτων, καθιστούν την προσομοίωση ιδιαίτερα πολύτιμη για την εκπαιδευτική διαδικασία[6] . Επιπλέον, η προσομοίωση μειώνει σημαντικά το οικονομικό κόστος, τις ανάγκες για εξειδικευμένο εξοπλισμό και τους κινδύνους από λανθασμένους χειρισμούς, προσφέροντας ένα ασφαλές και προσβάσιμο μαθησιακό περιβάλλον [2].

Παράλληλα, παρατηρείται σημαντική αύξηση στην αποδοχή αυτών των εργαλείων από τους φοιτητές, καθώς τα συστήματα αυτά παρέχουν ευελιξία, ευχρηστία και αυτονιμία, στοιχεία που συνδέονται με υψηλότερα επίπεδα κατανόησης και ακαδημαϊκής απόδοσης [5]. Ωστόσο, τα εικονικά εργαστήρια δεν μπορούν να αντικαταστήσουν πλήρως τα φυσικά, καθώς ο κύριος περιορισμός τους είναι η έλλειψη φυσικής επαφής με τον πραγματικό εξοπλισμό. [3]. Για τον λόγο αυτό, προτείνεται συχνά ένας συνδυασμός πραγματικών και ψηφιακών εργαστηρίων, ο οποίος επιτρέπει τη βέλτιστη αξιοποίηση των πλεονεκτημάτων και των δύο προσεγγίσεων.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η δημιουργία και δοκιμή ενός ψηφιακού εργαστηρίου ηλεκτρονικής. Στο πλαίσιο αυτό, η παρούσα πτυχιακή εργασία επικεντρώνεται στην ανάπτυξη ενός διαδραστικού περιβάλλοντος εργαστηρίου ηλεκτρονικής μέσω της μηχανής Unity, στο οποίο ο χρήστης μπορεί να διεξάγει τρία βασικά πειράματα: διαιρέτη τάσης και χρωματικό κώδικα αντιστάσεων, συμβολόμετρο Michelson και λειτουργία DC κινητήρα. Κεντρικός στόχος είναι η διερεύνηση του ρόλου των ψηφιακών εργαλείων στη βελτίωση της εκπαίδευσης ηλεκτρονικών κυκλωμάτων και η αποτίμηση της αποτελεσματικότητάς τους ως σύγχρονων εργαστηριακών μέσων.

Κεφάλαιο 1ο: Ιστορική Αναδρομή & Σύγχρονες Προσεγγίσεις

1.1 Ιστορική Αναδρομή της Προσομοίωσης Κυκλωμάτων

Η ιστορία των διαδραστικών προσομοιώσεων ηλεκτρονικής ξεκινά από τις αρχές της δεκαετίας του 1970, όταν εμφανίστηκε η πρώτη γενιά λογισμικών ανάλυσης κυκλωμάτων. Το σημαντικότερο ορόσημο αυτής της περιόδου ήταν η ανάπτυξη του SPICE (Simulation Program with Integrated Circuit Emphasis) από το Πανεπιστήμιο της Καλιφόρνια στο Berkeley, το οποίο αποτέλεσε τη βάση για όλα σχεδόν τα σύγχρονα εργαλεία ηλεκτρονικής προσομοίωσης. Η πρώτη έκδοση SPICE1 παρουσιάστηκε το 1973 από τον Nagel υπό την επίβλεψη του Pederson , ενώ η βελτιωμένη έκδοση SPICE2 ακολούθησε το 1975, προσφέροντας πιο αξιόπιστα αλγοριθμικά μοντέλα και αυξημένες δυνατότητες ανάλυσης [7].

Κατά τη δεκαετία του 1980, το SPICE άρχισε να διατίθεται σε ευρύτερο ακαδημαϊκό και βιομηχανικό κοινό, οδηγώντας σε εμπορικές και εκπαιδευτικές εκδόσεις όπως PSPICE (από την OrCAD), το οποίο αποτέλεσε το πρώτο λογισμικό προσομοίωσης κυκλωμάτων με γραφικό περιβάλλον (GUI) και όχι μόνο εντολές γραμμής. Η εισαγωγή GUI υπήρξε επαναστατική, καθώς επέτρεψε στους φοιτητές να σχεδιάζουν κυκλώματα με σύρσιμο–απόθεση στοιχείων, επιτρέποντας για πρώτη φορά μια διαδραστική εκπαιδευτική εμπειρία.

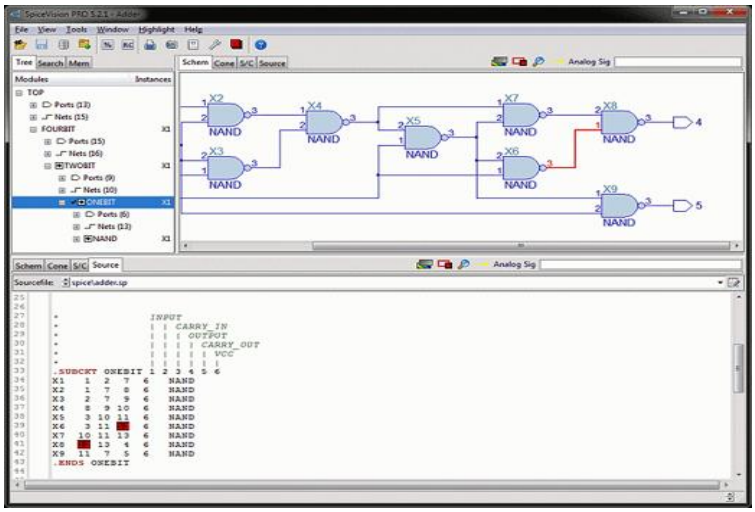
Στις αρχές της δεκαετίας 1990 εμφανίστηκαν τα πρώτα λογισμικά που συνδύαζαν γραφική αναπαράσταση, κινούμενη προσομοίωση και εκπαίδευση, όπως το Electronics Workbench, το οποίο αργότερα εξελίχθηκε στο γνωστό Multisim. Αυτές οι εφαρμογές εισήγαγαν πραγματικού χρόνου οπτικοποίηση, αλλαγή τιμών “on the fly”, και προσομοίωση πολύπλοκων οργάνων μέτρησης. Η εισαγωγή τέτοιων εργαλείων αύξησε σημαντικά την ικανότητα των φοιτητών να κατανοούν τη συμπεριφορά των κυκλωμάτων μέσω άμεσης αλληλεπίδρασης [8].

Η περίοδος 2000–2010 χαρακτηρίστηκε από την επέκταση των web-based εκπαιδευτικών εργαλείων. Σταδιακά εμφανίστηκαν διαδικτυακές πλατφόρμες όπως το iLabs του MIT, το οποίο επέτρεπε απομακρυσμένη εκτέλεση πραγματικών εργαστηριακών πειραμάτων μέσω διαδικτύου. Η εποχή αυτή σηματοδότησε τη μετάβαση από “virtual labs” σε “remote labs”, όπου οι φοιτητές δεν εκτελούσαν πλέον μόνο προσομοιώσεις, αλλά χειρίζονταν πραγματικό εξοπλισμό από απόσταση.

Στη δεκαετία του 2010 και μετά, τα διαδραστικά περιβάλλοντα εξελίχθηκαν περαιτέρω με τεχνολογίες 3D, game engines και VR. Εργαλεία βασισμένα σε Unity και Unreal Engine χρησιμοποιούνται πλέον ευρέως για εκπαιδευτικές εφαρμογές στη φυσική και την ηλεκτρονική χάρη στην υψηλή διαδραστικότητα που παρέχουν [18].

Τέλος, η πανδημία COVID-19 (2020) αποτέλεσε καθοριστικό παράγοντα για την επιτάχυνση της υιοθέτησης virtual και remote laboratories. Η τεράστια αύξηση στη χρήση εργαλείων προσομοίωσης, καθώς τα πανεπιστήμια αναγκάστηκαν να λειτουργήσουν εξ αποστάσεως [9]. Αυτό οδήγησε στη μονιμοποίηση πολλών διαδικτυακών εργαστηριακών πρακτικών και στην υιοθέτηση νέων προτύπων blended learning.

Εικόνα 1.1 Larry Nagel ο οποίος έγραψε τις πρώτες εκδόσεις του SPICE.[7]

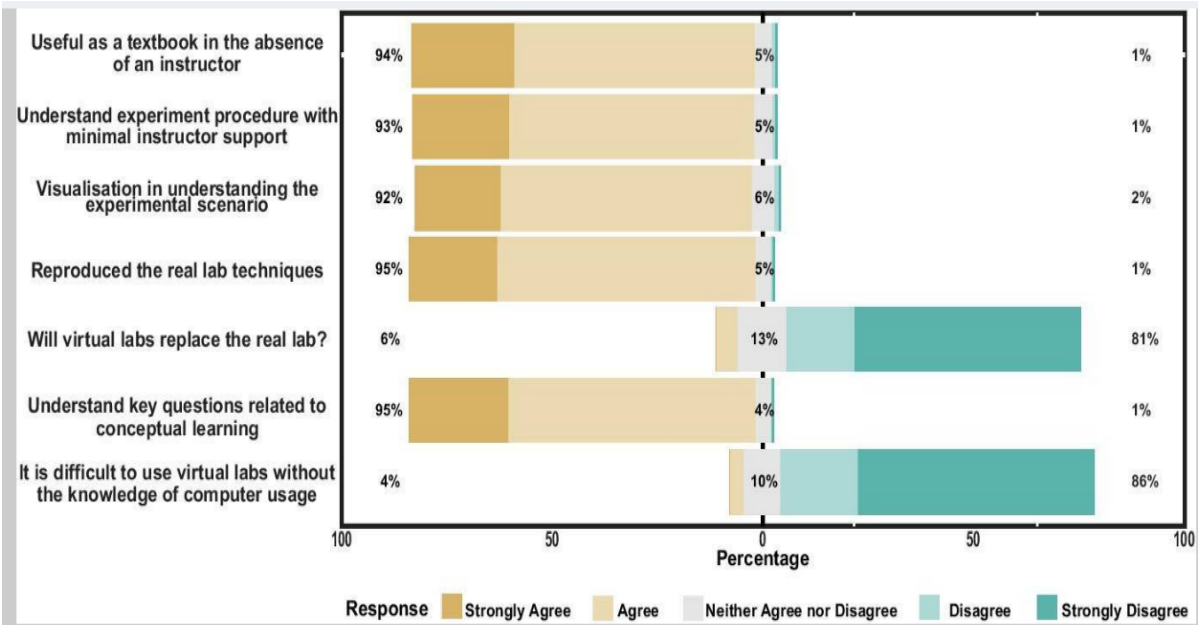


Εικόνα 1.2 Μία απο τις πρώτες εκδόσεις του SPICE.[7]

1.2 Ο ρόλος της Πανδημίας COVID-19 στη Διάδοση των Virtual / Remote Labs

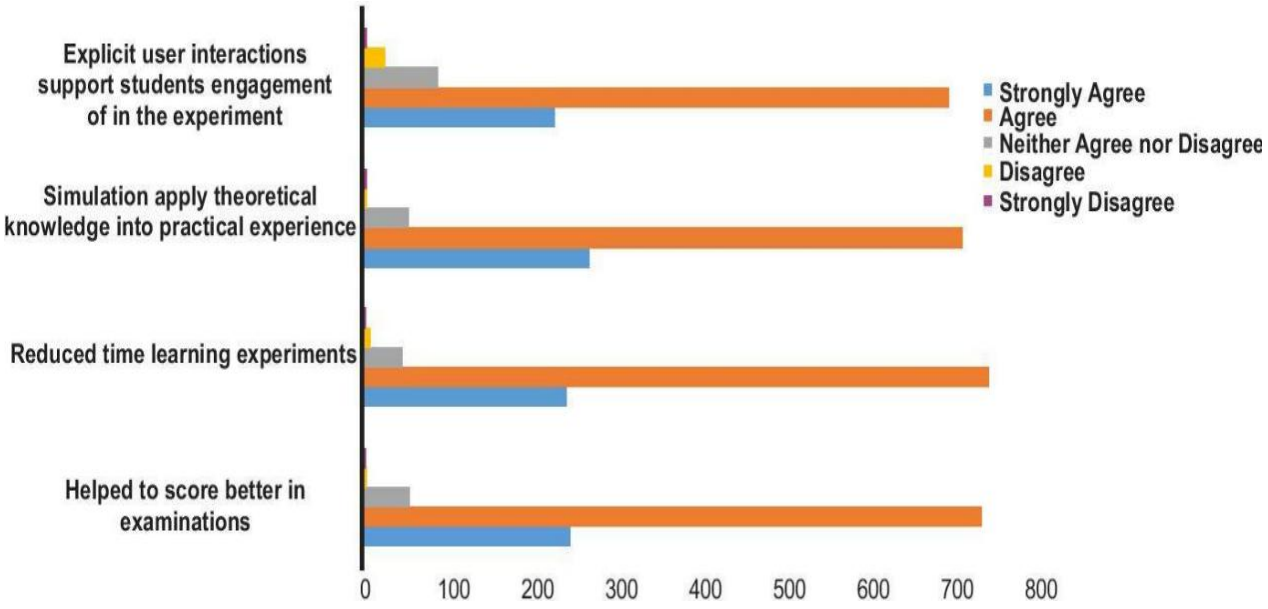
Η Η πανδημία COVID-19 έχει αναδιαμορφώσει ριζικά το τοπίο της τριτοβάθμιας εκπαίδευσης, αναγκάζοντας τα ιδρύματα παγκοσμίως να προσαρμοστούν γρήγορα σε πρωτοφανείς συνθήκες. Μία από τις πιο σημαντικές αλλαγές ήταν η ταχεία μετάβαση από την παραδοσιακή διδασκαλία πρόσωπο με πρόσωπο στην εξ αποστάσεως μάθηση, η οποία βασίζεται σε μεγάλο βαθμό σε ψηφιακά εργαλεία και διαδικτυακές πλατφόρμες. Αυτή η μετατόπιση δεν ήταν απλώς μια προσωρινή απάντηση, αλλά ένας καταλύτης για μακροπρόθεσμο μετασχηματισμό, υπογραμμίζοντας τη σημασία της τεχνολογικής ολοκλήρωσης στην εκπαίδευση.

Μια κρίσιμη πρόκληση που αντιμετωπίστηκε κατά τη διάρκεια αυτής της περιόδου ήταν η αδυναμία διεξαγωγής εργαστηριακών ασκήσεων με φυσική παρουσία, οι οποίες είναι αναπόσπαστο κομμάτι κλάδων όπως η μηχανική, η χημεία, η βιολογία και η ηλεκτρονική. Η πρακτική φύση της εργαστηριακής εργασίας είναι απαραίτητη για την ανάπτυξη πρακτικών δεξιοτήτων, την ενίσχυση του πειραματισμού και την ενθάρρυνση της κριτικής σκέψης. Η απότομη διακοπή των εργαστηριακών ασκήσεων με φυσική παρουσία υπογράμμισε την επείγουσα ανάγκη για καινοτόμες λύσεις που θα μπορούσαν να αναπαράγουν, όσο το δυνατόν πιο πιστά, την εμπειρική μάθηση των παραδοσιακών εργαστηρίων. Αυτή η αναγκαιότητα οδήγησε στην ταχεία εμφάνιση και υιοθέτηση εικονικών και εξ αποστάσεως εργαστηριακών πλατφορμών, αξιοποιώντας προηγμένες ψηφιακές τεχνολογίες για την προσομοίωση εργαστηριακών περιβαλλόντων. Αυτά τα εικονικά εργαστήρια έχουν γίνει γρήγορα απαραίτητα συστατικά της σύγχρονης εκπαίδευσης, διασφαλίζοντας τη συνέχεια και την ποιότητα της διδασκαλίας πέρα από απλά προσωρινά μέτρα [9].

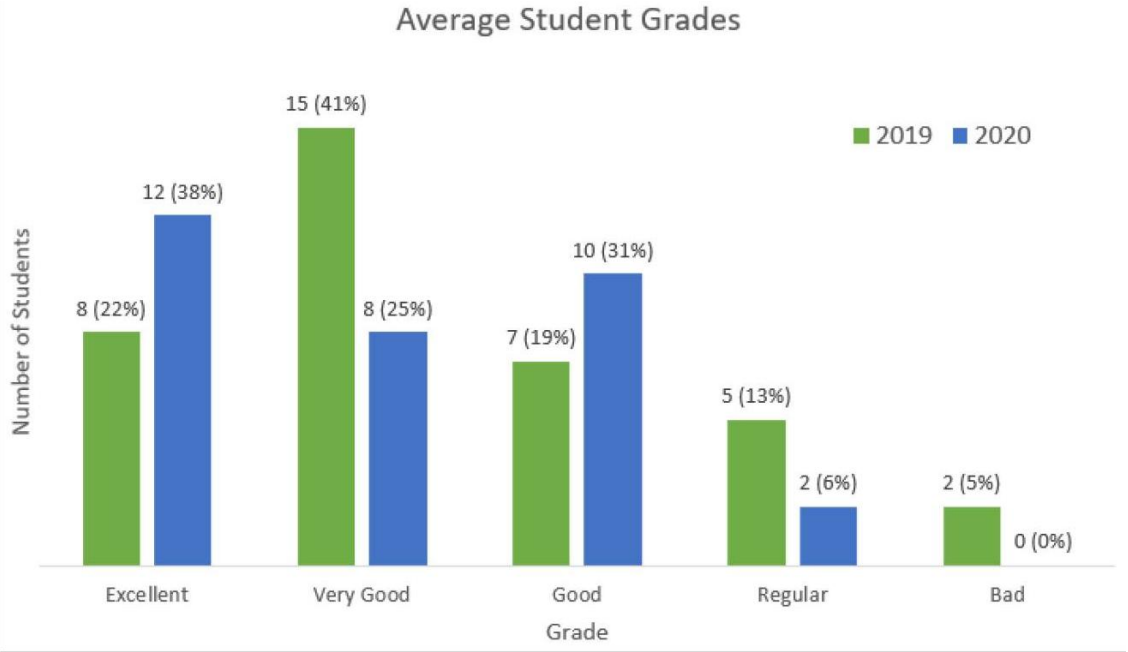


Εικόνα 1.3 Στατιστικά της εξ αποστάσεως εκπαίδευσης μέσω εικονικών εργαστηρίων [9]

Συγκριτικές μελέτες που διεξήχθησαν κατά τη διάρκεια της πανδημίας αποκαλύπτουν μια αξιοσημείωτη αύξηση στην υιοθέτηση και την εξάρτηση από ψηφιακά περιβάλλοντα. Οι μαθητές επέδειξαν μια εντυπωσιακή ικανότητα για ανεξάρτητη και ευέλικτη μάθηση, χωρίς περιορισμούς γεωγραφικούς ή φυσικούς. Σε τομείς όπως η ηλεκτρονική, η έρευνα δείχνει ότι τα καλά σχεδιασμένα διαδικτυακά πρακτικά περιβάλλοντα μπορούν να εκπληρώσουν αποτελεσματικά τους εκπαιδευτικούς στόχους χωρίς να θυσιάσουν την ανάπτυξη γνωστικών και πρακτικών δεξιοτήτων [10].



Εικόνα 1.4 Στατιστικά αλληλεπήδρασης χρηστών με την εξ αποστάσεως εκπαίδευση [9]



Εικόνα 1.5 Σύγκριση των αποδόσεων των μαθητών μεταξύ δια ζώσης και εξ αποστάσεως εκπαίδευσης [10].

Είναι ενδιαφέρον ότι πολλές μελέτες υποδηλώνουν ότι τα εικονικά εργαστήρια συχνά ενισχύουν πτυχές της εμπλοκής και της διαδραστικότητας πέρα από αυτά που μπορούν να προσφέρουν τα παραδοσιακά εργαστήρια. Οι εικονικές πλατφόρμες επιτρέπουν στους μαθητές να πειραματίζονται επανειλημμένα χωρίς τους περιορισμούς της διαθεσιμότητας εξοπλισμού, τους κινδύνους ασφαλείας ή το υψηλό κόστος που σχετίζεται με τις φυσικές εγκαταστάσεις. Αυτή η ευελιξία ενθαρρύνει μια πιο εξερευνητική και ερευνητική μαθησιακή διαδικασία, ενθαρρύνοντας τους μαθητές να πειραματίζονται ελεύθερα, να αναλύουν τα αποτελέσματα και να αναπτύσσουν δεξιότητες επίλυσης προβλημάτων σε ένα ελεγχόμενο ψηφιακό περιβάλλον [11].

Η πανδημία έχει έτσι επιταχύνει την αποδοχή και την ενσωμάτωση των εικονικών εργαστηρίων, μετατρέποντάς τα από συμπληρωματικά εκπαιδευτικά εργαλεία σε ζωτικά στοιχεία του σύγχρονου προγράμματος σπουδών. Η υιοθέτησή τους όχι μόνο ανταποκρίνεται στις άμεσες ανάγκες, αλλά ανοίγει και το δρόμο για καινοτόμες παιδαγωγικές προσεγγίσεις που μπορούν να κάνουν την εκπαίδευση στις θετικές επιστήμες πιο προσιτή, ελκυστική και αποτελεσματική στην ψηφιακή εποχή. Καθώς η τριτοβάθμια εκπαίδευση συνεχίζει να εξελίσσεται, τα εικονικά εργαστήρια είναι έτοιμα να διαδραματίσουν ολοένα και πιο κεντρικό ρόλο στη διαμόρφωση του μέλλοντος της βιωματικής μάθησης σε όλους τους κλάδους.

1.3 Οφέλη χρήσης διαδραστικών εργαστηρίων

- Προσβασιμότητα & Ευελιξία: Τα εικονικά εργαστήρια προσφέρουν τη δυνατότητα στους φοιτητές να εκτελούν πειράματα ανεξάρτητα από χωρικούς και χρονικούς περιορισμούς, γεγονός που ενισχύει τη συμμετοχή και μειώνει τα εμπόδια πρόσβασης. Μελέτες δείχνουν ότι η ευελιξία αυτή αυξάνει την αποτελεσματικότητα της μάθησης και την εκπαιδευτική ενσωμάτωση. [1][2]

- Ασφάλεια & Μείωση ρίσκου: Τα virtual labs επιτρέπουν την εκτέλεση πειραμάτων που σε πραγματικές συνθήκες θα ήταν επικίνδυνα ή απαιτητικά σε εξοπλισμό, εξαλείφοντας φυσικούς κινδύνους και επιτρέποντας την ασφαλή κατανόηση πολύπλοκων διαδικασιών. [2][3]

- Επαναληψιμότητα & Πειραματισμός: Η δυνατότητα συνεχούς αλλαγής παραμέτρων, επανάληψης πειραμάτων και δοκιμής εναλλακτικών σεναρίων χωρίς φθορά εξοπλισμού ή κατανάλωση αναλωσίμων αποτελεί ένα από τα σημαντικότερα πλεονεκτήματα των εικονικών περιβαλλόντων. Έρευνες δείχνουν ότι αυτή η δυνατότητα ενισχύει την αυτονομία και τη γνώση μέσω εξερεύνησης [14].

- Σύνδεση Θεωρίας & Πράξης: Τα εικονικά περιβάλλοντα βελτιώνουν σημαντικά τη μετάβαση από τη θεωρητική γνώση στην πρακτική εφαρμογή, επιτρέποντας στο χρήστη να δει σε πραγματικό χρόνο τη συμπεριφορά κυκλωμάτων, να λαμβάνει μετρήσεις και να παρακολουθεί την επίδραση αλλαγών. [13][14].

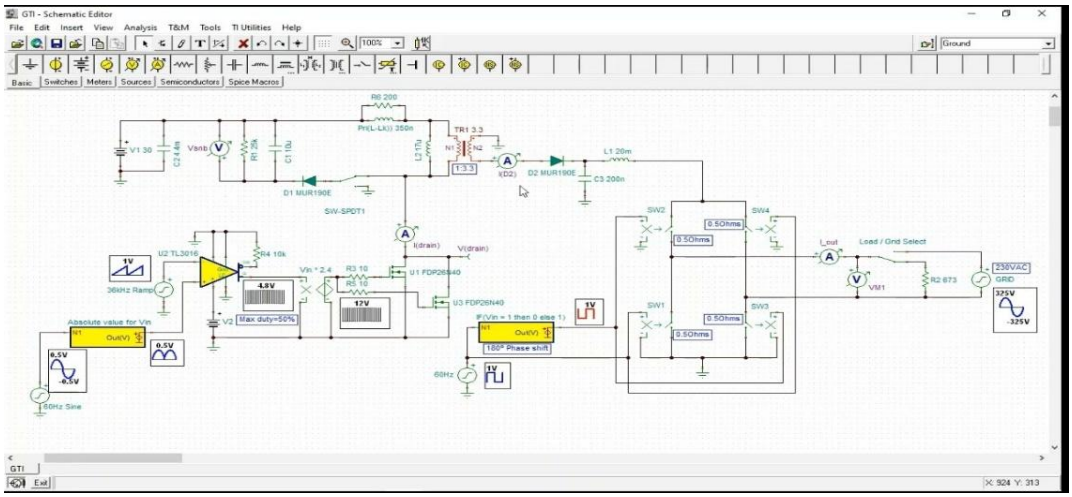
- Υποστήριξη Αυτορυθμιζόμενης Μάθησης: Οι φοιτητές μαθαίνουν με τον δικό τους ρυθμό, αναπτύσσοντας δεξιότητες αυτοκατεύθυνσης, ανάλυσης και κριτικής σκέψης [15].

- Καινοτομία & Επέκταση Πειραμάτων: Προσομοιώσεις επιτρέπουν την υλοποίηση πολύπλοκων, επικίνδυνων ή εξαιρετικά δαπανηρών πειραμάτων, τα οποία δεν μπορούν να πραγματοποιηθούν σε ένα κανονικό εργαστήριο. Αυτό ενισχύει την καινοτομία στο μάθημα και διευρύνει το εύρος της εκπαιδευτικής εμπειρίας [16][17].

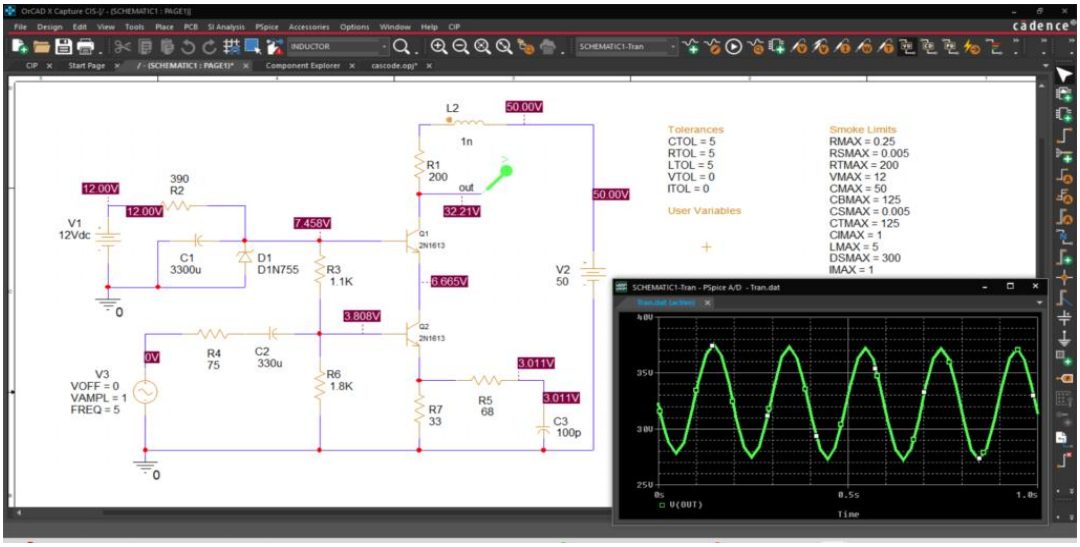
1.4 Είδη Διαδραστικών Εργαστηρίων και Εργαλείων Προσομοίωσης

Τα σύγχρονα διαδραστικά εργαστήρια ηλεκτρονικής έχουν μεταμορφώσει ριζικά τη διδασκαλία, την εκμάθηση και την πρακτική εξάσκηση στον τομέα της ηλεκτρονικής, προσφέροντας ένα ευρύ φάσμα εξελιγμένων ψηφιακών εργαλείων προσομοίωσης και εκπαιδευτικών πλατφορμών.

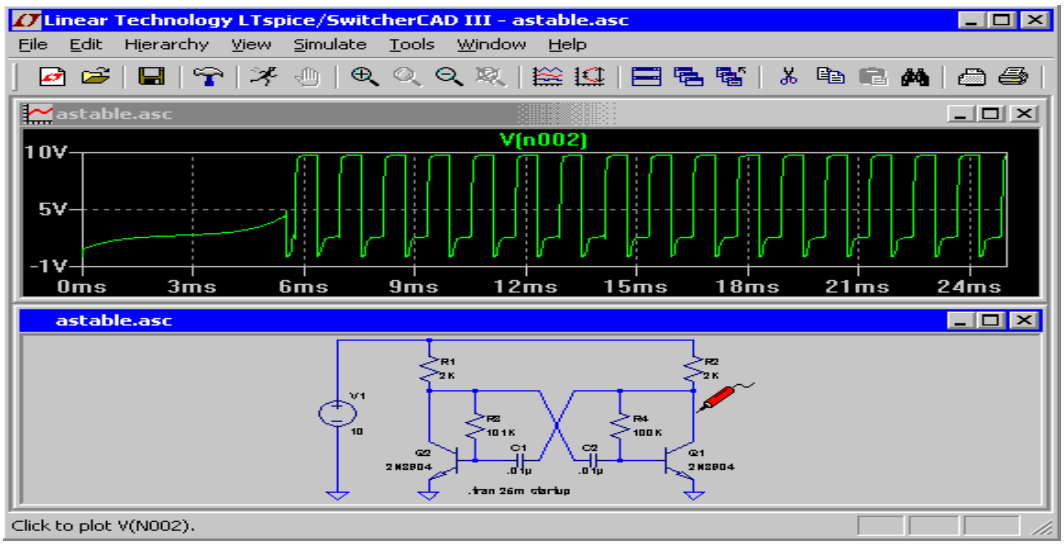
Στον πυρήνα της ακαδημαϊκής και επαγγελματικής εκπαίδευσης βρίσκονται τα κλασικά, ισχυρά εργαλεία ανάλυσης κυκλωμάτων. Προγράμματα όπως το SPICE (με τις ποικίλες εκδοχές του, π.χ., PSpice, LTspice) και το TINA-TI επιτρέπουν τη λεπτομερή και υψηλής ακρίβειας προσομοίωση τόσο αναλογικών όσο και ψηφιακών κυκλωμάτων. Αυτά τα εργαλεία χρησιμοποιούνται ευρέως σε πανεπιστημιακά εργαστήρια για τη διδασκαλία θεμελιωδών εννοιών, όπως η απόκριση συχνοτήτων, η μεταβατική απόκριση και η λειτουργία ημιαγωγών, καθώς και στην επαγγελματική εκπαίδευση για τον σχεδιασμό και τον έλεγχο πραγματικών κυκλωμάτων [19]. Η δυνατότητα ανάλυσης Fourier, θερμικής συμπεριφοράς και θορύβου τα καθιστά απαραίτητα για σοβαρό ηλεκτρονικό σχεδιασμό.



Εικόνα 1.6 Περιβάλλον TINA-TI

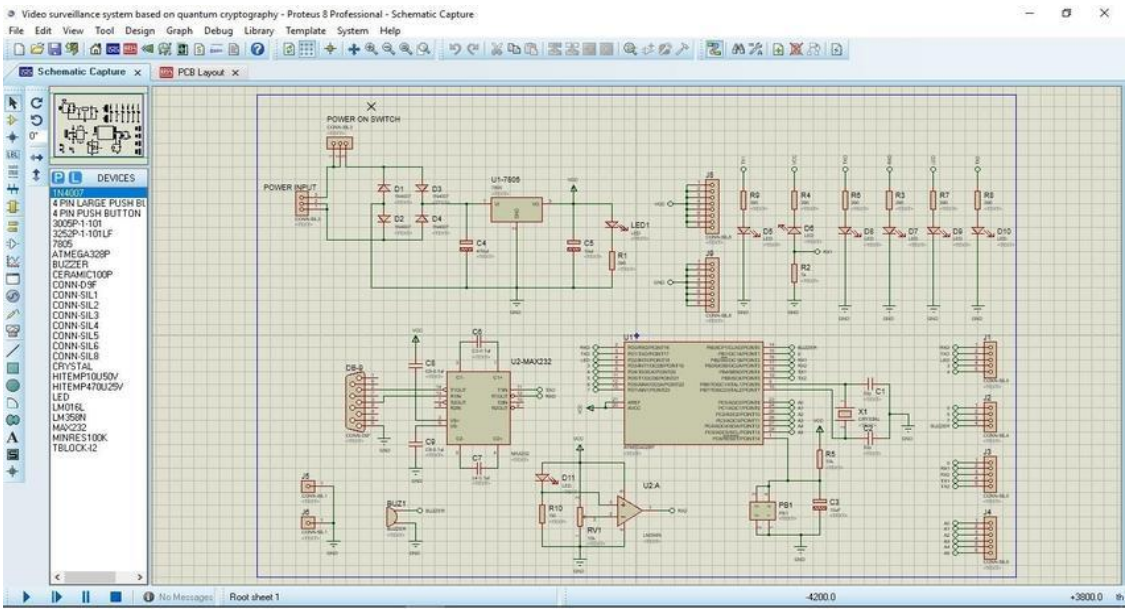


Εικόνα 1.7 Περιβάλλον PSpice

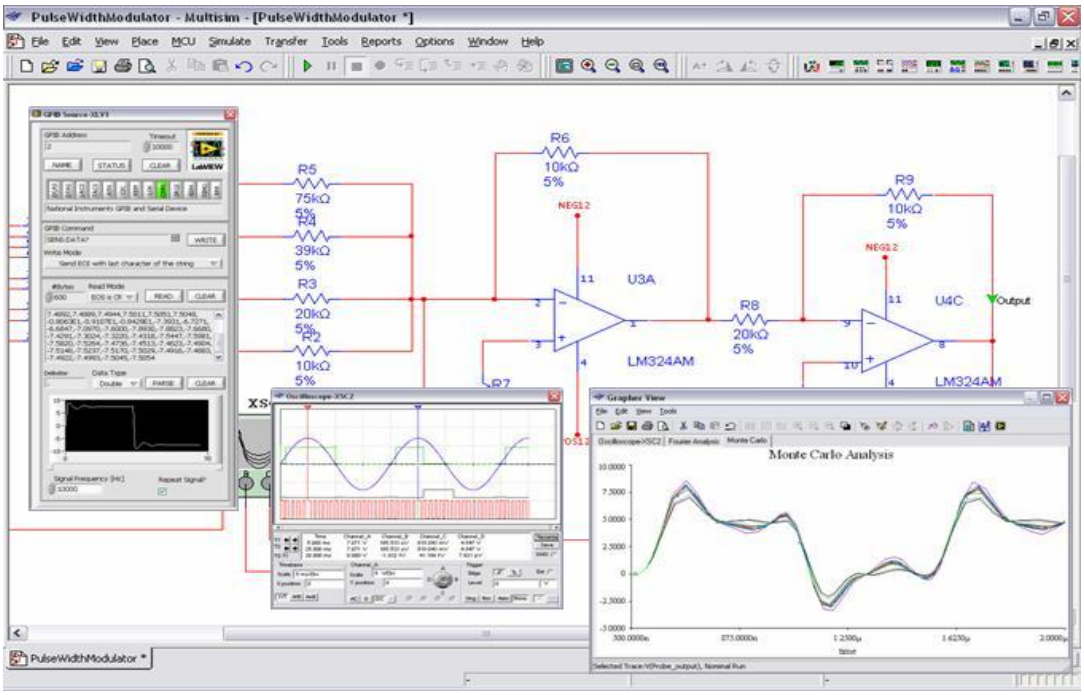


Εικόνα 1.8 Περιβάλλον LTSpice

Για μια πιο ολοκληρωμένη και συστημική προσέγγιση, εργαλεία όπως το Proteus (ISIS/ARES) και το NI Multisim συνδυάζουν την προχωρημένη προσομοίωση κυκλωμάτων με την υποστήριξη μικροελεγκτών και την αυτόματη σχεδίαση πλακέτας τυπωμένων κυκλωμάτων (PCB). Αυτές οι πλατφόρμες είναι ιδιαίτερα πολύτιμες για τη διδασκαλία και την πρακτική εξάσκηση σε ενσωματωμένα συστήματα. Επιτρέπουν τη δοκιμή κώδικα και τη λειτουργική προσομοίωση δημοφιλών πλατφορμών όπως Arduino, PIC και AVR, χωρίς καμία ανάγκη για φυσικό υλικό, παρέχοντας ένα ασφαλές περιβάλλον για πειραματισμό και ανάπτυξη προγραμμάτων [20].

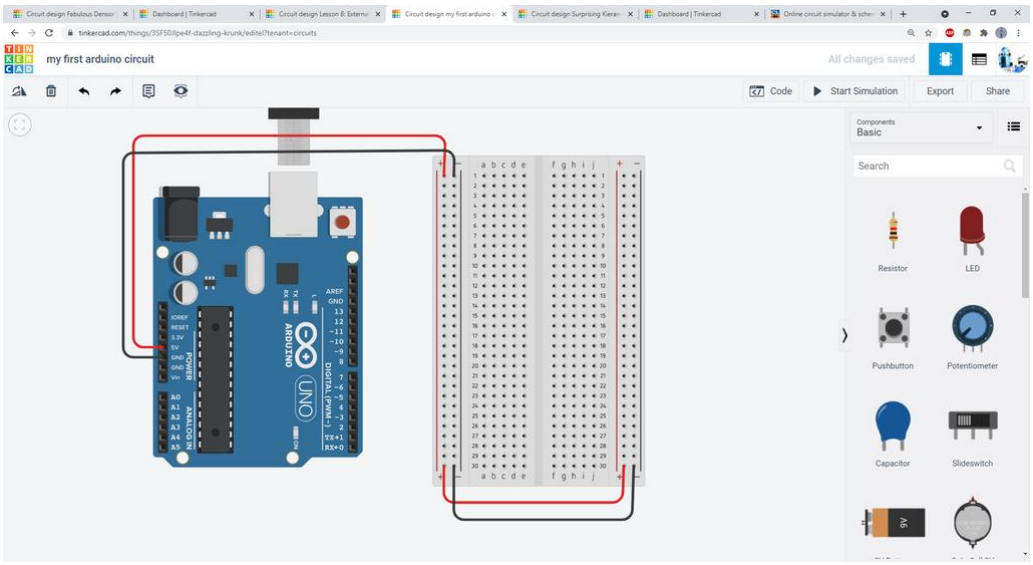


Εικόνα 1.9 Περιβάλλον Proteus



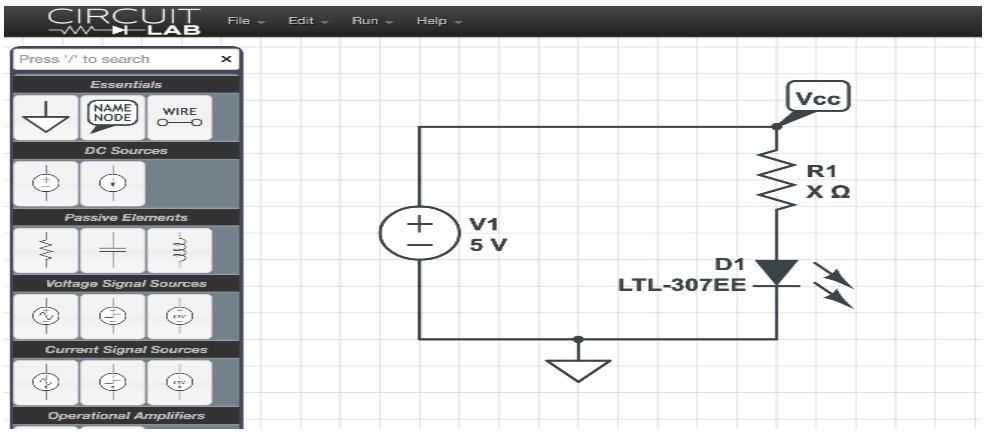
Εικόνα 1.10 Περιβάλλον NI Multisim

Στον τομέα της εκπαίδευσης αρχάριων και της διαδικτυακής μάθησης, έχουν αναδυθεί προσβάσιμες και φιλικές προς το χρήστη πλατφόρμες. Το Tinkercad Circuits της Autodesk, για παράδειγμα, είναι ένα ιδανικό εργαλείο εισαγωγής. Προσφέρει ένα εύχρηστο, διαδικτυακό περιβάλλον όπου οι μαθητές μπορούν να "συναρμολογήσουν" βασικά κυκλώματα με drag-and-drop, να προγραμματίσουν βασικές λειτουργίες (συχνά με βάση το Arduino) και να παρακολουθήσουν αμέσως τα αποτελέσματα. Αυτό το χαμηλό φράγμα εισόδου ενθαρρύνει την εξερευνητική μάθηση και καλλιεργεί την κατανόηση θεμελιωδών αρχών [21].

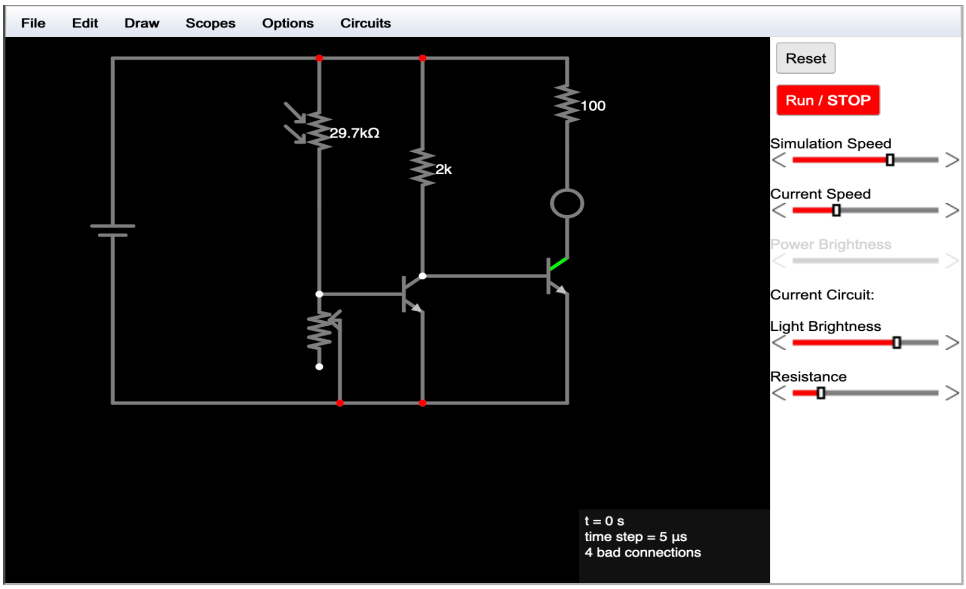


Εικόνα 1.11 Περιβάλλον TinkerCad

Επιπλέον, υπάρχουν ειδικά online εργαλεία γρήγορης προσομοίωσης και απεικόνισης, όπως το Falstad Circuit Simulator και το CircuitLab. Το Falstad, με τη χαρακτηριστική του κινούμενη απεικόνιση του ρεύματος και της τάσης, βοηθάει εξαιρετικά στην οπτικοποίηση και την κατανόηση εννοιών όπως το νόμο του Ohm, οι χρονικές σταθερές και τα φασματικά περιεχόμενα. Το CircuitLab προσφέρει ένα γρήγορο και αποτελεσματικό περιβάλλον για την ανάλυση πιο πολύπλοκων κυκλωμάτων απευθείας από το πρόγραμμα περιήγησης [22]. Αυτά τα εργαλεία είναι ιδανικά για γρήγορη επαλήθευση ιδεών, διερευνητική μάθηση και επεξηγήσεις κατά τη διάρκεια διαλέξεων.



Εικόνα 1.12 Περιβάλλον CircuitLab



Εικόνα 1.13 Περιβάλλον Falstad Circuit

Συνολικά, τα πλεονεκτήματα της χρήσης αυτών των διαδραστικών ψηφιακών εργαστηρίων είναι πολλαπλά και σημαντικά:

1. Άμεση Σύνδεση Θεωρίας και Πράξης: Οι μαθητές μπορούν να δουν άμεσα πώς οι μαθηματικές εξισώσεις και οι θεωρητικές αρχές μεταφράζονται σε συμπεριφορά κυκλωμάτων.
2. Ενίσχυση της Επαναληψιμότητας και της Αυτορρύθμισης: Οι μαθητές μπορούν να επαναλάβουν πειράματα άπειρες φορές, να αλλάξουν παραμέτρους και να παρατηρήσουν τις επιπτώσεις με δικό τους ρυθμό, ενισχύοντας τη βαθύτερη κατανόηση.
3. Μείωση Κόστους και Κινδύνων: Εξαλείφεται η ανάγκη για ακριβό φυσικό εξοπλισμό, ανταλλακτικά και χώρους εργαστηρίων. Επιπλέον, εξασφαλίζεται απόλυτη ασφάλεια, καθώς δεν υπάρχει κίνδυνος βραχυκυκλώματος, υπερθέρμανσης ή ηλεκτροπληξίας.
4. Πρόσβαση και Διανεμομένη Μάθηση: Δημιουργούνται δυνατότητες για εξ αποστάσεως εκπαίδευση και μαζικά ανοικτά διαδικτυακά μαθήματα (MOOCs), καθώς όλο το εργαστήριο βρίσκεται μέσα στον υπολογιστή.
5. Προετοιμασία για Βιομηχανικές Διαδικασίες: Η χρήση επαγγελματικών εργαλείων προσομοίωσης (όπως PSpice, Proteus) εξοικειώνει τους μαθητές με τις πραγματικές ροές εργασίας στον ηλεκτρονικό σχεδιασμό, αυξάνοντας την εργασιακή τους ελκυστικότητα.

Η εξέλιξη αυτών των πλατφορμών συνεχίζεται, με τάσεις να ενσωματώνουν τεχνητή νοημοσύνη για βελτιστοποίηση σχεδιασμού, εικονική και επαυξημένη πραγματικότητα για πιο αφαιρετικά εργαστήρια και ακόμη πιο στενή ενσωμάτωση λογισμικού και υλικού (π.χ., μέσω πλατφορμών όπως το Raspberry Pi Pico). Το σύγχρονο διαδραστικό εργαστήριο ηλεκτρονικής δεν είναι πλέον απλώς ένα εργαλείο, αλλά ένα ολοκληρωμένο, δυναμικό και απαραίτητο εκπαιδευτικό περιβάλλον για τον 21ο αιώνα.

Πίνακας 1.1 Συγκρίσεις Διαδραστικών Εργαλείων Προσομοίωσης

Εργαλείο	Τύπος	Κύρια Χρήση	Πλεονεκτήματα	Μειονεκτήματα
Tinkercad Circuits	Online / Εκπαιδευτικό	Αρχάριοι & Εκμάθηση Arduino	Πολύ εύκολο, δωρεάν, ασφαλές για πειράματα	Περιορισμένες δυνατότητες, λίγα εξαρτήματα
Falstad Simulation	Online / Διερευνητικό	Γρήγορη διερεύνηση εννοιών και απλών κυκλωμάτων	Άμεση οπτικοποίηση ρευμάτων, απλό, δωρεάν	Μη επαγγελματική ακρίβεια, μικρά κυκλώματα
CircuitLab	Online CAD / SPICE	Γρήγορη σχεδίαση και ανάλυση στο browser	Χωρίς εγκατάσταση, γρήγορο για απλά σχέδια	Περιορισμός σε πολυπλοκότητα κυκλώματος
Multisim	SPICE-based (Εκπαιδευτικό)	Πανεπιστημιακά εργαστήρια	Ρεαλιστικά εικονικά όργανα, καλή διδασκαλία	Μεσαία δυσκολία, ιδιόκτητο (συνήθως μέσω πανεπιστημίου)
PSpice	SPICE-based (Cadence)	Εκπαίδευση και επαγγελματική ανάλυση	Πλούσια βιβλιοθήκη, ακριβής, καλή τεκμηρίωση	Μεσαία δυσκολία, ιδιόκτητο (ακριβό) χρειάζεται χρόνος για να το μάθεις
LTspice	SPICE-based (Analog Devices)	Ηλεκτρονική Ισχύος	Δωρεάν, εξαιρετικά γρήγορο, πολύ δυνατό	Περιβάλλον λίγο περίπλοκο για αρχάριους
TINA-TI	SPICE-based (Texas Instruments)	Εκμάθηση και δοκιμή κυκλωμάτων με εξαρτήματα TI	Εύκολο περιβάλλον, δωρεάν, εξαιρετικό για TI	Περιορισμένο κυρίως σε βιβλιοθήκη TI, λιγότερο για μη-TI
Proteus	Προσομοίωση και Μικροελεγκτές	Προσομοίωση Μικροελεγκτών (Arduino κτλπ)	Συνδυάζει hardware & software	Ιδιόκτητο (απαιτεί άδεια), βαρύ για πολύπλοκα κυκλώματα

1.5 Στόχος της Εργασίας

Στόχος της παρούσας διπλωματικής εργασίας είναι ο σχεδιασμός και η υλοποίηση ενός διαδραστικού εικονικού εργαστηρίου, το οποίο αξιοποιεί σύγχρονες τεχνολογίες τρισδιάστατης προσομοίωσης και διαδραστικής μάθησης. Η εργασία αποσκοπεί στη δημιουργία ενός ψηφιακού περιβάλλοντος στο οποίο οι φοιτητές μπορούν να μελετήσουν θεωρητικές έννοιες και να τις εφαρμόσουν πρακτικά μέσω προσομοιωμένων πειραμάτων, χωρίς τους περιορισμούς ενός φυσικού εργαστηρίου. Στο πλαίσιο αυτό, η εργασία έγκειται στην ενσωμάτωση τριών αντιπροσωπευτικών πειραμάτων, διαιρέτη τάσης και χρωματικού κώδικα αντιστάσεων, συμβολόμετρο Michelson και λειτουργία κινητήρα συνεχούς ρεύματος, σε ένα ενιαίο, τρισδιάστατο και διαδραστικό περιβάλλον, σχεδιασμένο με γνώμονα την ευχρηστία και την εκπαιδευτική αποτελεσματικότητα. Επιπλέον, η χρήση εργαλείων όπως η Unity για

την ανάπτυξη της εφαρμογής και το Blender για τη δημιουργία τρισδιάστατων αντικειμένων, η εργασία φιλοδοξεί να συμβάλει στη διερεύνηση των δυνατοτήτων που προσφέρουν τα διαδραστικά εργαστήρια στην εκπαίδευση της ηλεκτρονικής, προσφέροντας ένα λειτουργικό παράδειγμα εφαρμογής που μπορεί να αξιοποιηθεί τόσο σε εξ αποστάσεως όσο και σε υβριδικά εκπαιδευτικά εργαστήρια.

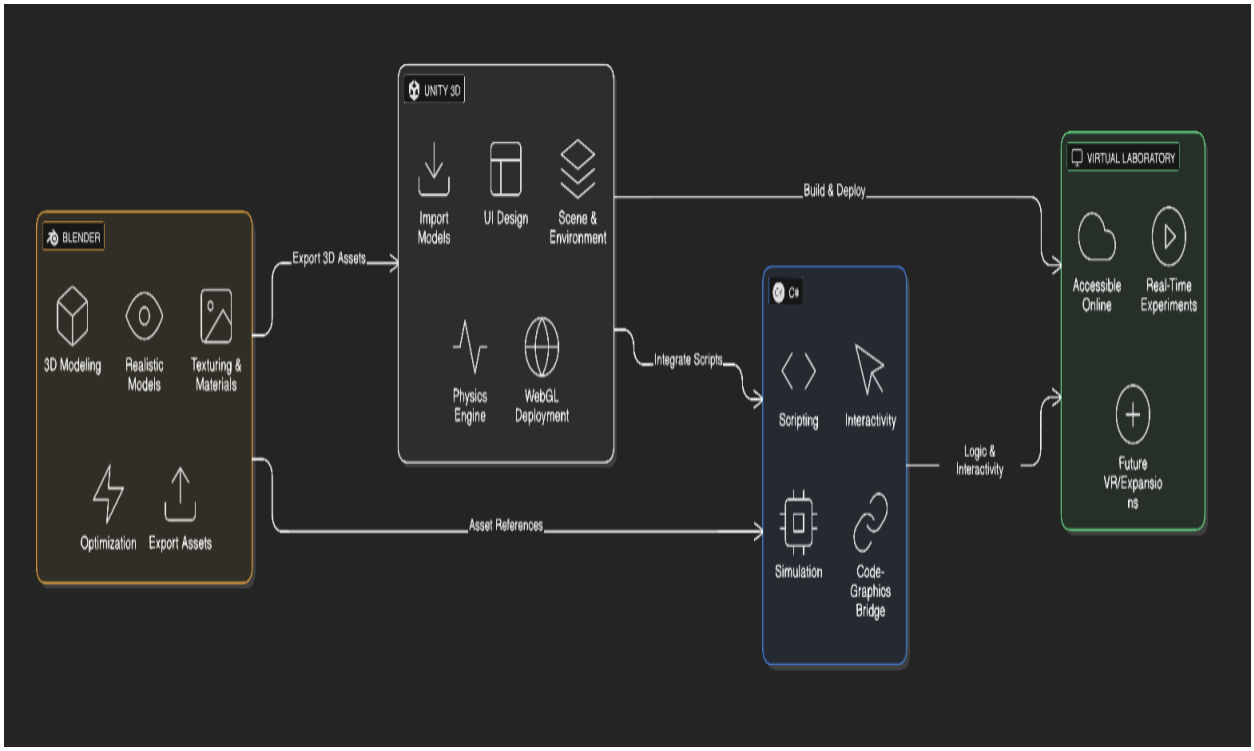
Κεφάλαιο 2ο: Unity και τεχνικά χαρακτηριστικά

2.1 Εισαγωγή

Όσον αφορά τη δημιουργία ενός εικονικού εργαστηρίου, η επιλογή του κατάλληλου λογισμικού είναι απολύτως απαραίτητη για την εξασφάλιση της επιτυχίας. Μετά από προσεκτική εξέταση διαφόρων επιλογών, συμπεριλαμβανομένων διαφορετικών εργαλείων προσομοίωσης και δημοφιλών μηχανών παιχνιδιών, η Unity ξεχώρισε ως η καλύτερη επιλογή. Αυτή η απόφαση βασίστηκε στα εντυπωσιακά τεχνικά χαρακτηριστικά της Unity, τα οποία επιτρέπουν τη δημιουργία ρεαλιστικών και διαδραστικών περιβαλλόντων. Επιπλέον, η αποδεδειγμένη αποτελεσματικότητά της ως εκπαιδευτικό εργαλείο βοηθά τους μαθητές να μαθαίνουν με τον πιο εύκολο τρόπο. Επιπλέον, επειδή η Unity χρησιμοποιείται ευρέως σε ακαδημαϊκά περιβάλλοντα, διαθέτει μια ισχυρή κοινότητα και άφθονους πόρους, καθιστώντας την μια αξιόπιστη πλατφόρμα για όλα τα στάδια ανάπτυξης.

2.2 Ανάλυση Τεχνικών Χαρακτηριστικών

Η ανάπτυξη του εικονικού εργαστηρίου βασίζεται σε μια στενά συνδεδεμένη τριάδα τεχνολογιών: Unity 3D, C# και Blender. Αυτά τα εργαλεία συνδέονται σε ένα ενιαίο και αρμονικό σύστημα ανάπτυξης, το οποίο έχει σχεδιαστεί ειδικά για να ανταποκρίνεται στις πολύπλοκες ανάγκες μιας διαδραστικής και σύγχρονης εκπαιδευτικής πλατφόρμας. Η συγκεκριμένη αρχιτεκτονική τεχνολογιών εξασφαλίζει τη δημιουργία ενός συστήματος που είναι ταυτόχρονα τεχνικά άρτιο, οπτικά πιστό και παιδαγωγικά αποτελεσματικό.



Σχήμα 2.1: Σύνδεση Unity, Blender, C#

2.2.1 Unity

Η Unity δεν αποτελεί απλώς μια μηχανή γραφικών, αλλά το κεντρικό περιβάλλον ενσωμάτωσης (Integration Environment) όπου συγκλίνουν όλα τα επιμέρους στοιχεία της εφαρμογής.

- Προηγμένη Φυσική και Δυναμική Μοντελοποίηση: Η χρήση της μηχανής φυσικής PhysX επιτρέπει την εξομοίωση σύνθετων φαινομένων με υψηλό βαθμό ρεαλισμού. Πέρα από τη βασική κινηματική και τις συγκρούσεις στερεών σωμάτων, η Unity επιτρέπει την επέκταση του μοντέλου φυσικής μέσω προσαρμοσμένων κωδίκων[24][26]. Αυτό είναι κρίσιμο για την αναπαράσταση μη-μηχανικών φαινομένων, όπως η ηλεκτρομαγνητική επαγωγή ή η θερμοδυναμική συμπεριφορά, εξασφαλίζοντας ότι η απόκριση του εικονικού πειράματος ταυτίζεται με τα αναμενόμενα πειραματικά δεδομένα.

- Προηγμένη Τρισδιάστατη Απόδοση και Οπτικός Ρεαλισμός: Ο κινητήρας της Unity χρησιμοποιεί σύγχρονες γραφικές τεχνικές, όπως το Physically-Based Rendering (PBR), δυναμικό φωτισμό (Real-time Global Illumination) και προηγμένα συστήματα σκίασης και εφέ μετάβασης. Αυτές οι δυνατότητες εξασφαλίζουν την παραγωγή οπτικά ρεαλιστικών και συναρπαστικών εικονικών περιβαλλόντων, τα οποία ενισχύουν την αίσθηση της παρουσίας (Sense of Presence) και της εμβάπτισης (Immersion) του χρήστη. Ο οπτικός ρεαλισμός είναι καθοριστικός για τη δημιουργία μιας αυθεντικής εργαστηριακής ατμόσφαιρας και την ακριβή αναπαράσταση εξοπλισμού[24].

- Αρχιτεκτονική Prefabs και Επεκτασιμότητα: Η μεθοδολογία των Prefabs (προκατασκευασμένων αντικειμένων) προωθεί την επαναχρησιμοποίηση του κώδικα και των πόρων. Κάθε εργαστηριακό όργανο μοντελοποιείται ως μια αυτόνομη οντότητα που φέρει τις δικές της ιδιότητες και συμπεριφορές. Αυτή η προσέγγιση διευκολύνει την κλιμάκωση του εργαστηρίου, επιτρέποντας την προσθήκη νέων πειραματικών διατάξεων χωρίς την ανάγκη ανασχεδιασμού της βασικής δομής[25].

- Βελτιστοποίηση για το Διαδίκτυο μέσω WebGL: Στο πλαίσιο της καθολικής προσβασιμότητας, η υποστήριξη του WebGL είναι στρατηγικής σημασίας. Επιτρέπει τη μεταγλώττιση της εφαρμογής σε κώδικα που εκτελείται απευθείας στον περιηγητή (browser), καταργώντας την ανάγκη για εγκατάσταση λογισμικού και μειώνοντας τα εμπόδια εισόδου για φοιτητές με περιορισμένους υπολογιστικούς πόρους[25][26]. Παράλληλα υποστηρίζεται ανάπτυξη για Windows, macOS, Linux και κινητές συσκευές.

- Ευέλικτο Σύστημα Προγραμματισμού και Επέκτασης: Ως κύρια γλώσσα προγραμματισμού υποστηρίζεται η C#, μια ισχυρή, αντικειμενοστρεφής γλώσσα με τεράστια βιβλιοθήκη κλάσεων (.NET Framework/Mono). Το εκτεταμένο και καλά τεκμηριωμένο API της Unity παρέχει άμεση πρόσβαση σε όλες τις λειτουργίες της μηχανής. Επιπλέον, η πλατφόρμα υποστηρίζει διάφορα πρωτόκολλα δικτύωσης και επικοινωνίας (TCP/IP, UDP, WebSockets, MQTT), επιτρέποντας την ενσωμάτωση με εξωτερικό υλικό (π.χ., Arduino, Raspberry Pi), την ανάπτυξη πολυχρηστικών λειτουργιών ή τη σύνδεση με διαδικτυακές υπηρεσίες και βάσεις δεδομένων, γεγονός που ανοίγει το δρόμο για υβριδικά (εικονικά/πραγματικά) και απομακρυσμένα εργαστήρια[29][30].

- Εγγενής Υποστήριξη για Επαυξημένη και Εικονική Πραγματικότητα(AR/VR): Η Unity προσφέρει εγγενή υποστήριξη για τις πιο δημοφιλείς πλατφόρμες Επαυξημένης Πραγματικότητας (AR) (ARKit για iOS, ARCore για Android) και Εικονικής Πραγματικότητας (VR) (OpenXR, Oculus SDK, SteamVR). Αυτό σημαίνει ότι το εικονικό εργαστήριο μπορεί να σχεδιασθεί εξαρχής με τη δυνατότητα μελλοντικής επέκτασής του σε αυτές τις τεχνολογίες,

χωρίς να απαιτείται ουσιαστικός ανασχεδιασμός. Η χρήση AR/VR μπορεί να προσφέρει παντελώς νέες διαστάσεις εμβάπτισης και αλληλεπίδρασης, καθιστώντας τα πειράματα ακόμη πιο ενδιαφέροντα και ρεαλιστικά [27].

Ολοκληρωμένο Περιβάλλον Υποστήριξης και Ανάπτυξης (Unity)

Η επιλογή μιας τεχνολογικής πλατφόρμας εξαρτάται όχι μόνο από τις βασικές τεχνικές δυνατότητές της, αλλά και από ευρύτερα πρότυπα, συμπεριλαμβανομένης της αξιολόγησης του συνολικού πλαισίου υποστήριξης. Για την πλατφόρμα Unity, το καλά οργανωμένο και ολοκληρωμένο περιβάλλον υποστήριξης αποτελεί σημαντικό πλεονέκτημα, ειδικά σε ακαδημαϊκές εφαρμογές. Αυτό το πλαίσιο αποτελείται από τέσσερις διασυνδεδεμένους πυλώνες που συλλογικά βοηθούν στη μείωση του χρόνου, των τεχνικών και των οικονομικών κινδύνων που σχετίζονται με την ανάπτυξη λογισμικού.

Unity Asset Store

Το Unity Asset Store είναι ένα βασικό εργαλείο για τη βελτίωση της αποτελεσματικότητας της ανάπτυξης. Είναι ένα πλούσιο και καλά οργανωμένο προκαθορισμένο αποθετήριο πόρων, συμπεριλαμβανομένων ακριβών τρισδιάστατων μοντέλων επιστημονικών οργάνων, προσχεδιασμένων συστημάτων διεπαφής χρήστη και ειδικών βιβλιοθηκών κώδικα. Με αυτούς τους πόρους, η εστίαση στην ανάπτυξη μπορεί να μετατοπιστεί από τη δημιουργία βασικών στοιχείων στην ταχεία δημιουργία πρωτοτύπων, την προσαρμογή και την καινοτομία. Οι στρατηγικές επαναχρησιμοποίησης λογισμικού θεωρούνται ευρέως κλειδί για τον έλεγχο του κόστους και τη βελτιστοποίηση των χρονοδιαγραμμάτων ανάπτυξης, επιτρέποντας στους ερευνητές να επικεντρωθούν στην παιδαγωγική αξία των εκπαιδευτικών εργαλείων.

Unity Learn

Η ποιότητα και η ευκολία χρήσης της τεκμηρίωσης επηρεάζουν άμεσα την εκμάθηση και την εφαρμογή νέων τεχνολογιών. Η Unity διαθέτει ένα πολυεπίπεδο εκπαιδευτικό σύστημα, που περιλαμβάνει ολοκληρωμένη τεκμηρίωση διεπαφής προγραμματισμού εφαρμογών (API), διαδραστικά σεμινάρια και μαθήματα βασισμένα σε έργα που προσφέρονται μέσω της πλατφόρμας Unity Learn. Αυτή η δομημένη υποδομή διευκολύνει αποτελεσματικά την απόκτηση γνώσεων, βοηθώντας τόσο τους αρχάριους χρήστες όσο και τους έμπειρους προγραμματιστές να βρουν τις βέλτιστες πρακτικές. Η ύπαρξη τέτοιων συστηματικών εκπαιδευτικών πόρων συσχετίζεται θετικά με την επιτυχή ενσωμάτωση σύνθετων εργαλείων σε ακαδημαϊκά περιβάλλοντα.

Κοινότητα Χρηστών και Μηχανισμοί Υποστήριξης

Η ύπαρξη μιας ενεργής και τεχνικά καταρτισμένης παγκόσμιας κοινότητας μπορεί να προσφέρει ανεκτίμητη υποστήριξη. Πλατφόρμες όπως τα επίσημα φόρουμ της Unity, το Stack Overflow και διάφορες ομάδες κοινωνικών μέσων αποτελούν ένα δυναμικό δίκτυο για την ανταλλαγή γνώσεων και την επίλυση προβλημάτων. Στην πραγματικότητα, αυτό σημαίνει ότι μια μεγάλη ποικιλία τεχνικών ζητημάτων και ανησυχιών διερευνάται και επιλύεται σε ευρύτερο πλαίσιο. Η έρευνα επιβεβαιώνει ότι τέτοιες διαδικτυακές κοινότητες μπορούν να παρέχουν γρήγορες και αποτελεσματικές απαντήσεις,

μειώνοντας σημαντικά τον χρόνο που απαιτείται για τον εντοπισμό και τη διόρθωση σφαλμάτων, κάτι που είναι ιδιαίτερα σημαντικό για έργα που απαιτούν χρόνο.

Οικονομικά Βιώσιμο και Κλιμακούμενο Μοντέλο Άδειας Χρήσης

Το μοντέλο εμπορικής αδειοδότησης της Unity έχει σχεδιαστεί για να καλύπτει τις ανάγκες διαφορετικών σταδίων ανάπτυξης. Η δωρεάν Προσωπική Έκδοση παρέχει πλήρη πρόσβαση σε βασικές λειτουργίες, επαρκείς για ακαδημαϊκή έρευνα, διδασκαλία και μη εμπορική ανάπτυξη, με χαμηλό κόστος εισόδου. Για πιο απαιτητικά έργα ή λειτουργίες επαγγελματικής υποστήριξης, η Pro και η Enterprise εκδόσεις προσφέρουν σαφείς διαδρομές αναβάθμισης. Το μοντέλο "Freemium" έχει αποδειχθεί αποτελεσματικό στην προώθηση της εφαρμογής σε ακαδημαϊκό και ερευνητικό περιβάλλον.

Αξιολόγηση και Μείωση Συνολικού Κινδύνου

Όταν συγκρίνουμε την Unity με άλλες επιλογές που μπορεί να βασίζονται σε λιγότερο ανεπτυγμένα συστήματα, η Unity ξεχωρίζει πραγματικά χάρη στην ωριμότητα του υποστηρικτικού της πλαισίου. Κάθε κομμάτι αυτού του πλαισίου έχει σχεδιαστεί για να αντιμετωπίζει συγκεκριμένους κινδύνους. Για παράδειγμα, το Asset Store βοηθάει να μειωθεί ο κίνδυνος καθυστερήσεων και υπερβάσεων κόστους, ενώ η καλά οργανωμένη τεκμηρίωση και εκπαίδευση ανακουφίζουν από τις τεχνικές δυσκολίες. Επιπλέον, η ενεργή κοινότητα της Unity είναι εκεί για να μας στηρίξει και να μας βοηθήσει να αποφύγουμε αδιέξοδα. Με λίγα λόγια, η Unity δεν είναι απλώς μια πλατφόρμα, είναι ένα ολόκληρο οικοσύστημα που μας στηρίζει σε κάθε βήμα της δημιουργικής μας διαδικασίας.

2.2.2 C#: Η Γλώσσα Προγραμματισμού και Επιχειρησιακής Λογικής

Αν η Unity είναι το σώμα του εργαστηρίου, η γλώσσα C# είναι το μυαλό του. Είναι η γλώσσα που δίνει εντολές και ορίζει τους κανόνες του παιχνιδιού..

• Αλληλεπίδραση μέσω του Unity API: Η C# στην Unity δεν είναι απλώς μια γλώσσα γενικού σκοπού, αλλά προσαρμόζεται στις ανάγκες του project μέσω εξειδικευμένων βιβλιοθηκών. Χρησιμοποιώντας την κλάση MonoBehaviour και τη βιβλιοθήκη UnityEngine, ο κώδικας αποκτά άμεση πρόσβαση στα αντικείμενα της κάθε σκηνής[27].

Εξειδικευμένες Βιβλιοθήκες

- UnityEngine.Physics: Επιτρέπει τον έλεγχο των δυνάμεων και των αλληλεπιδράσεων σε πραγματικό χρόνο.
- UnityEngine.UI: υπεύθυνη για τη δημιουργία των μενού, των κουμπιών και των ψηφιακών οθονών των οργάνων μέτρησης

• Αλγοριθμική Ακρίβεια: Στο επίπεδο του scripting υλοποιούνται οι επιστημονικοί υπολογισμοί Στο επίπεδο του scripting υλοποιούνται οι πυρήνες των πειραμάτων. Η C# διαχειρίζεται τη ροή των δεδομένων και εκτελεί αλγορίθμους βασισμένους σε επιστημονικά μοντέλα, όπως ο υπολογισμός της συμπεριφοράς ενός ολοκληρωμένου κυκλώματος [24]. Αυτό εξασφαλίζει ότι η εικονική προσομοίωση δεν είναι απλώς μια οπτική αναπαράσταση, αλλά μια πιστή ψηφιακή αντιγραφή (Digital Twin) της πραγματικής επιστημονικής διαδικασίας.

2.2.3 Blender: Το Εργαλείο Δημιουργίας Περιεχομένου και Οπτικού Σχεδιασμού

Το Blender αναλαμβάνει την παραγωγή όλων των τρισδιάστατων περιεχομένων, διασφαλίζοντας την οπτική ακρίβεια και ρεαλισμό του εργαστηρίου. Ο ρόλος του είναι καθοριστικός, καθώς αναλαμβάνει να μετατρέψει τις ψυχρές μαθηματικές παραμέτρους σε αναγνωρίσιμα ψηφιακά αντικείμενα. Δεν πρόκειται απλώς για ένα εργαλείο σχεδίασης, αλλά για μια ολοκληρωμένη πλατφόρμα που διασφαλίζει ότι ο φοιτητής θα αντικρίσει ένα οικείο περιβάλλον. Είναι υπεύθυνο για το πώς "φαίνεται" το εργαστήριο.

- Μοντελοποίηση Ακρίβειας και Τοπολογία (Topology): Κάθε αντικείμενο, από μια απλή αντίσταση έως έναν σύνθετο ψηφιακό παλμογράφο, σχεδιάζεται με γνώμονα τον ρεαλισμό αλλά και την υπολογιστική απόδοση. Η διαδικασία ξεκινά με τη δημιουργία μοντέλων υψηλής ανάλυσης (High-Poly), όπου αποδίδεται κάθε κατασκευαστική λεπτομέρεια. Στη συνέχεια, μέσω της τεχνικής του Retopology, δημιουργούνται βελτιστοποιημένα μοντέλα (Low-Poly) που διατηρούν τη γεωμετρική ακεραιότητα του πρωτοτύπου, ενώ είναι αρκετά ελαφριά ώστε να εκτελούνται απρόσκοπτα σε περιβάλλοντα WebGL χωρίς να προκαλούν καθυστερήσεις (latency) στην προσομοίωση.

- Ροή Εργασίας PBR (Physically Based Rendering): Για την επίτευξη φωτορεαλισμού, χρησιμοποιείται η μεθοδολογία PBR, η οποία διασφαλίζει ότι τα υλικά αντιδρούν στο φως βάσει φυσικών κανόνων. Μέσω του Blender, δημιουργούνται πολλαπλοί χάρτες υφών (Albedo, Normal, Roughness, Metallic). Η τεχνική του Texture Baking επιτρέπει τη μεταφορά των λεπτομερειών (όπως σκιές, χαράξεις και ανακλάσεις) από το High-Poly μοντέλο στο Low-Poly, προσφέροντας μια εξαιρετικά πιστή οπτική αναπαράσταση με ελάχιστο υπολογιστικό κόστος. Αυτό είναι κρίσιμο για την αναγνώριση των οργάνων και των ενδείξεων από τους φοιτητές, ενισχύοντας τη γνωστική εμβύθιση [28].

- UV Unwrapping και Διαχείριση Χώρου: Η ορθή ανάπτυξη των επιφανειών στο δισδιάστατο επίπεδο (UV Unwrapping) επιτρέπει την τοποθέτηση λεπτομερών σημάνσεων πάνω στα όργανα, όπως κλίμακες μέτρησης και λογότυπα κατασκευαστών. Αυτή η λεπτομέρεια είναι απαραίτητη για την εκπαιδευτική διαδικασία, καθώς ο φοιτητής πρέπει να είναι σε θέση να διαβάζει τιμές και ρυθμίσεις ακριβώς όπως θα έκανε σε ένα φυσικό περιβάλλον.

- Rigging και Τεχνική Κίνηση (Hard Surface Animation): Πέρα από την εξωτερική εμφάνιση, το Blender χρησιμοποιείται για τον καθορισμό των κινητών μερών των οργάνων. Μέσω της δημιουργίας ψηφιακών σκελετών (Rigging) και περιορισμών κίνησης (Constraints), ορίζεται ο τρόπος με τον οποίο περιστρέφονται οι διακόπτες ή κινούνται οι βελόνες των αναλογικών οργάνων. Αυτές οι πληροφορίες εξάγονται στην Unity, όπου η C# αναλαμβάνει να κινήσει τα μέρη αυτά βάσει των υπολογισμών του πειράματος, δημιουργώντας μια απόλυτα συνεκτική και διαδραστική εμπειρία.

- Βελτιστοποίηση για Πραγματικό Χρόνο (Real-time Optimization): Η διαλειτουργικότητα με την Unity ενισχύεται μέσω της εξαγωγής σε μορφές αρχείων που υποστηρίζουν ιεραρχίες και animations (όπως .fbx ή .gltf). Η χρήση των LODs (Levels of Detail), που σχεδιάζονται στο Blender, επιτρέπει στην Unity να προβάλλει απλουστευμένες εκδοχές των αντικειμένων όταν

αυτά βρίσκονται μακριά από την εικονική κάμερα, εξοικονομώντας πόρους και διασφαλίζοντας σταθερό ρυθμό ανανέωσης πλαισίων (frames per second), στοιχείο καθοριστικό για την αποφυγή της κόπωσης του χρήστη (cyber-sickness) [26].

2.3 Η Αρχιτεκτονική του Εργαστηρίου

Η κατασκευή του εικονικού εργαστηρίου βασίζεται σε τρία επίπεδα: την Εικόνα (πώς φαίνεται), τον Χώρο (πού βρίσκεται) και τη Λογική (πώς λειτουργεί). Η αρχιτεκτονική του εικονικού εργαστηρίου προσεγγίζεται ως ένα ολοκληρωμένο ψηφιακό οικοσύστημα, όπου η αισθητική αρτιότητα, η χωρική οργάνωση και η επιστημονική εγκυρότητα συνυπάρχουν αρμονικά. Η υλοποίηση αυτής της δομής ακολουθεί μια συστηματική ροή εργασίας (pipeline), η οποία διασφαλίζει ότι η εμπειρία του φοιτητή θα είναι ταυτόχρονα ρεαλιστική και εκπαιδευτικά ωφέλιμη.

1. Η Δημιουργική Βάση: Μορφοποίηση και Οπτική Ταυτότητα (Blender)

Η διαδικασία ξεκινά με τη χρήση του Blender, το οποίο λειτουργεί ως το ψηφιακό εργαστήριο κατασκευής του εξοπλισμού. Σε αυτό το στάδιο, η έμφαση δίνεται στη δημιουργία αντικειμένων που δεν είναι απλώς οπτικά ελκυστικά, αλλά γεωμετρικά ορθά.

- Σχεδιασμός και Εξαγωγή: Κάθε εργαστηριακό όργανο μοντελοποιείται με βάση τις πραγματικές του προδιαγραφές. Η χρήση των μορφοτύπων .fbx και .obj είναι συμαντική καθώς επιτρέπει τη μεταφορά των μοντέλων στην Unity διατηρώντας την ποιότητα των υλικών και των σημείων περιστροφής (pivots), τα οποία είναι απαραίτητα για την μετέπειτα κίνησή τους. Για αντικείμενα όπως οι βελόνες των αναλογικών οργάνων ή οι αντιστάσεις, ώστε να περιστρέφονται σωστά γύρω από τον άξονά τους στην Unity.

- PBR Texturing: Η δημιουργία χαρτών υφών (Normal maps, Metallic, Roughness) διασφαλίζει ότι τα υλικά θα αντιδρούν ρεαλιστικά στον φωτισμό της Unity, ενισχύοντας τη γνωστική εμβύθιση του φοιτητή [28].

- Βελτιστοποίηση: Μέσω της τεχνικής του low-poly modeling, διασφαλίζεται ότι τα assets είναι "ελαφριά" και αποδοτικά, επιτρέποντας στο εργαστήριο να λειτουργεί απρόσκοπτα σε περιβάλλον browser χωρίς να απαιτείται ισχυρό υλικό από την πλευρά του χρήστη [26, 28].

2. Ο Κεντρικός Κόμβος: Οργάνωση και Ενοποίηση (Unity 3D)

Μετά τη δημιουργία των assets, η Unity 3D αναλαμβάνει τον ρόλο του "ενορχηστρωτή". Είναι ο χώρος όπου η στατική γεωμετρία αποκτά υλική μορφή και αλληλεπιδρά.

- Δόμηση Σκηνών (Scenes): Το εργαστήριο οργανώνεται σε Scenes, οι οποίες λειτουργούν ως ανεξάρτητες εκπαιδευτικές ενότητες. Μέσα σε αυτές, ορίζονται ο φωτισμός, οι φυσικοί νόμοι και οι κάμερες που καθοδηγούν το βλέμμα του φοιτητή.

- Αρχιτεκτονική Prefabs: Η Unity χρησιμοποιεί το σύστημα των Prefabs, το οποίο επιτρέπει την οργάνωση των αντικειμένων ως πρότυπα. Αυτό σημαίνει ότι κάθε μοντέλο μετατρέπεται σε

Prefab, μια ψηφιακή "σφραγίδα" που περιέχει όλα τα οπτικά στοιχεία και τα scripts. Αυτό επιτρέπει τη μαζική αναβάθμιση των οργάνων: αν αλλάξει μια γραμμή κώδικα στο Prefab, η αλλαγή εφαρμόζεται αυτόματα σε όλες τις σκηνές του εργαστηρίου, γεγονός που καθιστά το σύστημα εξαιρετικά επεκτάσιμο και εύκολο στη συντήρηση [25].

3. Η Νοημοσύνη του Συστήματος: Προγραμματισμός και Λογική (C# & Visual Studio)

Το τελικό και πιο ουσιαστικό επίπεδο της αρχιτεκτονικής είναι η απόδοση "νοημοσύνης" στα ψηφιακά αντικείμενα. Αυτό επιτυγχάνεται μέσω της γλώσσας C# στο περιβάλλον ανάπτυξης του Visual Studio.

- Αλληλεπίδραση μέσω Visual Studio: Η σύνδεση της Unity με το Visual Studio επιτρέπει τη συγγραφή scripts που κληρονομούν από την κλάση MonoBehaviour. Αυτό δίνει τη δυνατότητα στον κώδικα να παρεμβαίνει σε κάθε "καρέ" (frame) της προσομοίωσης, ενημερώνοντας τις τιμές των οργάνων.

- Σύνδεση Logic και Visuals: Μέσω του Visual Studio, αναπτύσσονται εξειδικευμένα scripts τα οποία "κουμπώνουν" πάνω στα αντικείμενα της Unity. Ο προγραμματισμός εδώ δεν αφορά μόνο την κίνηση, αλλά την υλοποίηση της επιστημονικής λογικής του πειράματος.

- Μαθηματική Μοντελοποίηση Πειράματος: Στο επίπεδο αυτό υλοποιούνται οι φυσικοί νόμοι. Για παράδειγμα, ένας αλγόριθμος σε C# υπολογίζει τη διαφορά δυναμικού σε ένα κύκλωμα και μεταφράζει αυτό το νούμερο σε κίνηση της βελόνας ενός εικονικού βολτομέτρου [24, 27].

- Debugging και Βελτιστοποίηση: Το Visual Studio παρέχει εργαλεία εντοπισμού σφαλμάτων σε πραγματικό χρόνο, διασφαλίζοντας ότι η λογική του πειράματος είναι αλάνθαστη πριν την τελική εξαγωγή.

Η συνέργεια αυτών των τριών επιπέδων της μορφής από το Blender, του χώρου από την Unity και της λογικής από τη C#, δημιουργεί ένα στιβαρό ακαδημαϊκό εργαλείο. Το τελικό Build ολοκληρώνει την αρχιτεκτονική, προσφέροντας μια λύση που είναι ταυτόχρονα τεχνικά προηγμένη και εύκολα προσβάσιμη από τον καθένα.

2.4 Διαδικασία της τελικής εξαγωγής (Build)

Η ολοκλήρωση της αρχιτεκτονικής ανάπτυξης επισφραγίζεται με τη διαδικασία της τελικής παραγωγής (Build), η οποία συνιστά τη μετάβαση από το περιβάλλον ανάπτυξης σε ένα λειτουργικό και διαλειτουργικό εκπαιδευτικό προϊόν. Η διαδικασία αυτή δεν αποτελεί μια απλή εξαγωγή δεδομένων, αλλά μια σύνθετη μεταστοιχείωση (compilation) του πηγαίου κώδικα C# και των τρισδιάστατων assets σε βελτιστοποιημένα σύνολα δεδομένων, ικανά να εκτελεστούν σε ποικίλα υπολογιστικά περιβάλλοντα.

Κεντρικό πυλώνα αυτής της στρατηγικής αποτελεί η μορφή WebGL, η οποία, μέσω της χρήσης του μεταγλωττιστή IL2CPP, μετατρέπει την εφαρμογή σε έναν συνδυασμό WebAssembly και JavaScript. Η επιλογή αυτή υπηρετεί τον ακαδημαϊκό στόχο της καθολικής προσβασιμότητας (Universal Access), καθώς επιτρέπει την εκτέλεση σύνθετων επιστημονικών πειραμάτων εντός του περιηγητή,

εκμηδενίζοντας τις τεχνικές φραγές που σχετίζονται με την εγκατάσταση λογισμικού ή τη συμβατότητα λειτουργικών συστημάτων.

Παράλληλα, η ευελιξία της αρχιτεκτονικής διασφαλίζει ότι το εργαστήριο μπορεί να εξελίσσεται συμβαδίζοντας με τις τεχνολογικές τάσεις. Πέρα από τη διαδικτυακή του μορφή, το σύστημα επιτρέπει την εξαγωγή αυτόνομων εκδόσεων (Standalone), οι οποίες προσφέρουν τη μέγιστη υπολογιστική ισχύ όπου αυτό απαιτείται, ενώ η υποστήριξη του προτύπου OpenXR ανοίγει τον δρόμο για τη μελλοντική μετάβαση σε περιβάλλοντα πλήρους εμβύθισης μέσω Εικονικής Πραγματικότητας (VR). Με αυτόν τον τρόπο, η διαδικασία του Build λειτουργεί ως ο τελικός συνδετικός κρίκος που μεταφέρει την επιστημονική εγκυρότητα του προγραμματισμού και του σχεδιασμού απευθείας στον χρήστη. Ουσιαστικά, το εργαστήριο παύει να είναι μια στατική εφαρμογή και μετατρέπεται σε ένα δυναμικό εκπαιδευτικό εργαλείο, το οποίο παραμένει επίκαιρο και ικανό να προσαρμόζεται στις ανάγκες της σύγχρονης τριτοβάθμιας εκπαίδευσης, ανεξάρτητα από το μέσο πρόσβασης που θα επιλέξει ο φοιτητής.

Κεφάλαιο 3ο: Σχεδιασμός και ανάλυση πειραμάτων εικονικού εργαστηρίου

3.1 Εισαγωγή

Το τρέχον κεφάλαιο αναλύει διεξοδικά τη διαδικασία σχεδίασης, ανάπτυξης και εκτέλεσης των διαδραστικών πειραμάτων που συνθέτουν το εικονικό εργαστήριο αυτής της διπλωματικής εργασίας. Ο σκοπός του κεφαλαίου είναι να αποδειχθεί η μετάβαση από τη θεωρητική έννοια των πειραμάτων στην πρακτική και εκπαιδευτικά χρήσιμη εφαρμογή τους, αξιοποιώντας σύγχρονες τεχνολογίες τρισδιάστατης απεικόνισης και προγραμματισμού. Η ανάλυση υιοθετεί μια σταθερή προσέγγιση, που περιλαμβάνει την τρισδιάστατη μοντελοποίηση των στοιχείων στο λογισμικό Blender, την ενσωμάτωσh τους στο περιβάλλον Unity 3D και την εφαρμογή της λογικής αλληλεπίδρασης και των υπολογιστικών διεργασιών μέσω προγραμματισμού στη C#. Στο πλαίσιο αυτής της εφαρμογής πραγματοποιήθηκαν τρία συνολικά διαδραστικά πειράματα, που αγγίζουν διαφορετικές αλλά συμπληρωματικές γνωστικές σφαίρες της ηλεκτρονικής και της φυσικής.

Το αρχικό πείραμα σχετίζεται με τον διαιρέτη τάσης και τον χεωματικό κώδικα αντιστάσεων, δίνοντας τη δυνατότητα στον χρήστη να εξερευνήσει θεμελιώδεις έννοιες ηλεκτρικών κυκλωμάτων με παραμετρικές μεταβολές και άμεση οπτική αναπαράσταση των αποτελεσμάτων.

Το δεύτερο πείραμα εστιάζει στην προσομοίωση του συμβολόμετρου Michelson, προσφέροντας τη δυνατότητα κατανόησης φαινομένων υπέρθεσης φωτός και εξερεύνησης της επίδρασης μεταβολών σε σημαντικούς παράγοντες του συστήματος.

Το τρίτο πείραμα αναπαριστά τη λειτουργία ενός κινητήρα συνεχούς ρεύματος (DC motor), δίνοντας τη δυνατότητα ανάλυσης της σχέσης ανάμεσα σε ηλεκτρικά και μηχανικά μεγέθη, όπως είναι η τροφοδοτούμενη τάση και η περιστροφική ταχύτητα.

Η κοινή αρχιτεκτονική ανάπτυξης που χρησιμοποιείται και στα τρία πειράματα εξασφαλίζει τη συνοχή του εικονικού εργαστηρίου και προσφέρει ομοιόμορφη εμπειρία χρήσης, ενώ ταυτόχρονα διευκολύνει την μελλοντική ανάπτυξη της εφαρμογής με νέα πειράματα ή δυνατότητες. Σε αυτό το κεφάλαιο αξιολογούνται τα τεχνικά στοιχεία της εφαρμογής καθώς και οι εκπαιδευτικές διάστασεις κάθε πειράματος.

3.2 Διαιρέτης Τάσης και Χρωματικός Κώδικας

Ο διαιρέτης τάσης είναι ένα από τα πιο βασικά κυκλώματα στην ηλεκτρονική, βασισμένο στην αρχή της κατανομής της τάσης σε αντιστάσεις που είναι συνδεδεμένες σε σειρά. Αυτό το φαινόμενο, αποτυπώνεται στον απλό μαθηματικό τύπο $V_{out} = V_{in} \times (R_2/(R_1 + R_2))$ και είναι ιδιαίτερα ενδιαφέρον λόγω της διπλής του φύσης: αφενός, η διατύπωση αντιπροσωπεύει μια θεμελιώδη αρχή της θεωρίας κυκλωμάτων, αφετέρου, είναι διαρκώς παρόν στην πρακτική σχεδίαση συστημάτων. Η αξία του ενισχύεται από το ότι πολλά πολύπλοκα ηλεκτρονικά συστήματα, όπως ενισχυτές, μετατροπείς και συστήματα αισθητήρων, στηρίζονται σε παραλλαγές αυτού του κυκλώματος. Παράλληλα, ο χρωματικός κώδικας των αντιστάσεων είναι ένα απλό αλλά ευφυές σύστημα οπτικής μεταφοράς τεχνικών πληροφοριών. Μέσω μιας ακολουθίας πολύχρωμων δακτυλίων, κάθε αντιστάτης διαβιβάζει τρεις καίριες πληροφορίες: την αριθμητική τιμή (τα δύο πρώτα χρώματα), τον πολλαπλασιαστή (το τρίτο χρώμα) και την ανοχή (το τέταρτο χρώμα). Αυτή η κωδικοποίηση λύνει ένα σημαντικό πρακτικό ζήτημα όπως τον τρόπο που γράφουμε μικρούς αριθμούς σε μικροσκοπικά μέρη. Ο κώδικας λειτουργεί

ως μια παγκόσμια τεχνική γλώσσα, δηλαδή ένας μηχανικός από οποιαδήποτε περιοχή του κόσμου έχει τη δυνατότητα να εκτιμήσει την αξία μιας αντίστασης.

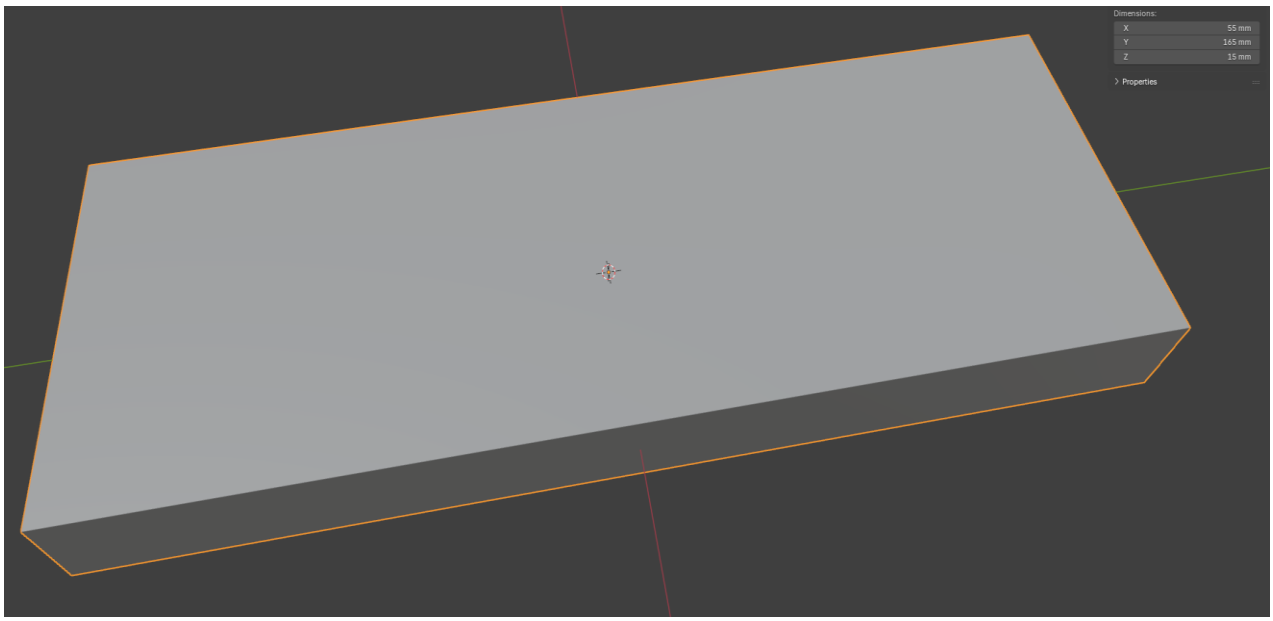
3.2.1 Σχεδίαση και Τρισδιάστατη Μοντελοποίηση στο Blender

Στο πλαίσιο του πρώτου πειράματος το μόνο τρισδιάστατο μοντέλο που σχεδιάστηκε εξ' αρχής στο Blender ήταν breadboard με σκοπό την παραγωγή ενός αντικειμένου που να πλησιάζει τόσο σε γεωμετρία όσο και σε λειτουργικότητα ένα πραγματικό εργαστηριακό breadboard. Τα υπόλοιπα εξαρτήματα του κυκλώματος συγκεκριμένα η αντίσταση, το τροφοδοτικό και το πολύμετρο[31][32][33] τοποθετήθηκαν στο περιβάλλον ως προσχεδιασμένα μοντέλα σε μορφή .fbx, τα οποία στη συνέχεια παραμετροποιήθηκαν και προσαρμόστηκαν στις ανάγκες του πειραματισμού.

Πώς δημιουργήθηκε το Breadboard βήμα-βήμα.

1. Δημιουργία της Βάσης

Η διαδικασία ξεκίνησε με την εισαγωγή ενός βασικού κύβου (Cube), ο οποίος αποτέλεσε τη βάση για το μοντέλο. Στο αντικείμενο αυτό δόθηκαν οι πραγματικές διαστάσεις ενός τυπικού breadboard, ώστε η κλίμακα να είναι ακριβής. Συγκεκριμένα, ορίστηκε το μήκος στα 16.5 cm (άξονας Y), το πλάτος στα 5.5 cm (άξονας X) και το ύψος στα 1.5 cm (άξονας Z). Αυτή η ρύθμιση ήταν απαραίτητη για να εξασφαλιστεί ότι το Breadboard θα έχει την ίδια εμφάνιση και αναλογίες με ένα αληθινό.



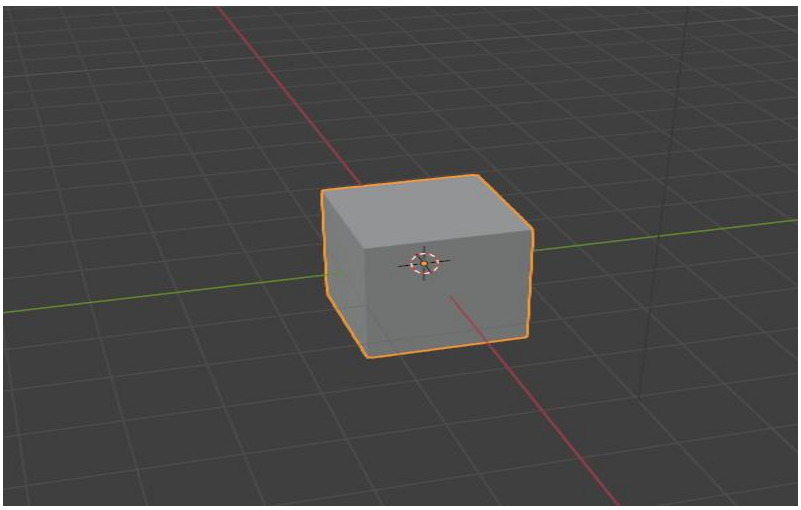
Εικόνα 3.1 Βάση Breadboard

2. Δημιουργία Πλέγματος

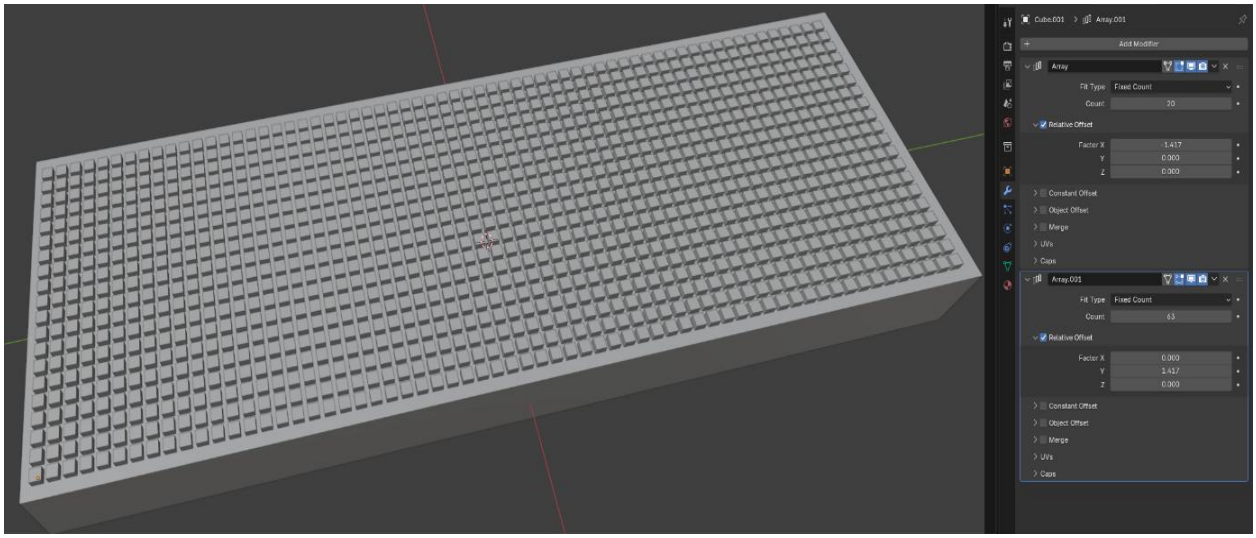
   Τοποθετήθηκε ένας νέος, μικρότερος κύβος που θα αποτελέσει το πρότυπο για μια οπή. Αυτός κλιμακώθηκε σε 1.8 mm × 1.8 mm και τοποθετήθηκε στην επιφάνεια της βάσης. Επειτα προστέθηκε Array modifier στον κύβο, με τις εξής παραμέτρους:

- 20 αντίγραφα στον άξονα X
- 63 αντίγραφα στον άξονα Y
- Απόσταση 2.55 mm μεταξύ των κέντρων (0.1 ίντσας)

Ο παράγοντας (Relative Offset – Factor X / Y / Z) στον Array Modifier δεν είναι μέτρηση σε χιλιοστά. Αποτελεί παράγοντα πολλαπλασιασμού του αρχικού μεγέθους του αντικειμένου. Το Factor ενημερώνει το Blender «πόσες φορές το μέγεθος του αντικειμένου να μετακινηθεί το επόμενο αντίγραφο». Η απόσταση μεταξύ των οπών καθορίζεται μέσω του Relative Offset του Array Modifier, που λειτουργεί ως παράγοντας του μεγέθους του βασικού αντικειμένου. Δεδομένου ότι το μέγεθος της οπής ορίστηκε στα 1,8 mm και η πραγματική απόσταση μεταξύ διαδοχικών οπών είναι 2,55 mm, το αντίστοιχο factor υπολογίστηκε ως 2,55/1,8 ≈ 1,417.



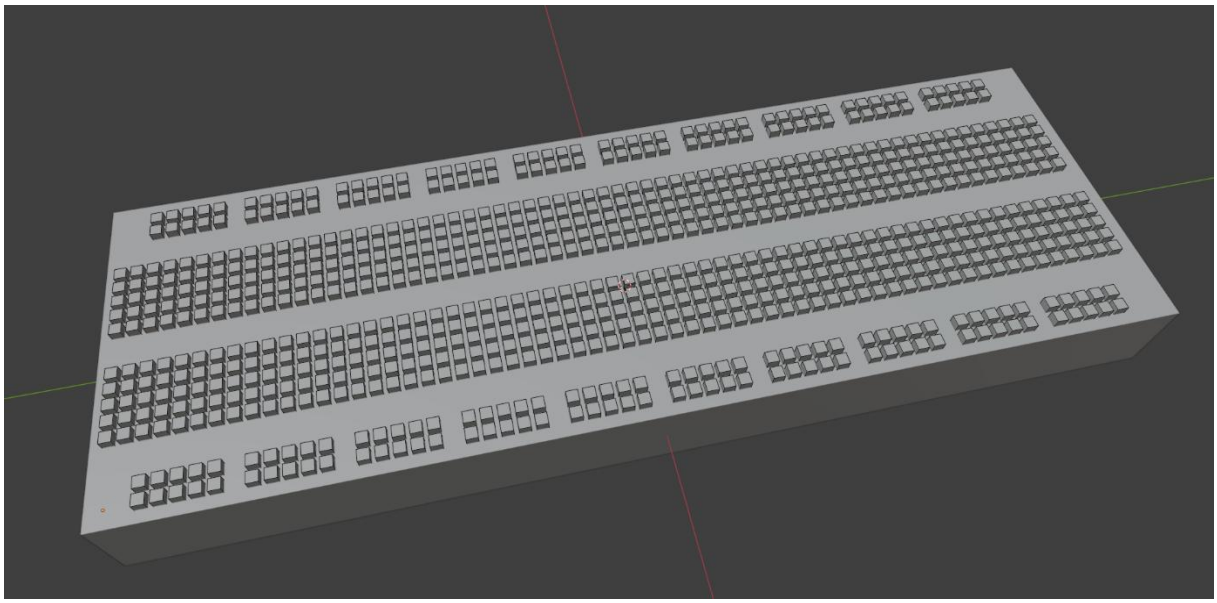
Εικόνα 3.2 Πρότυπος κύβος για τις οπές.



Εικόνα 3.3 Η βάση όπου έχουν τοποθετηθεί οι οπές μαζί και οι ρυθμίσεις του Array Modifier.

3. Τροποποίηση Πλέγματος

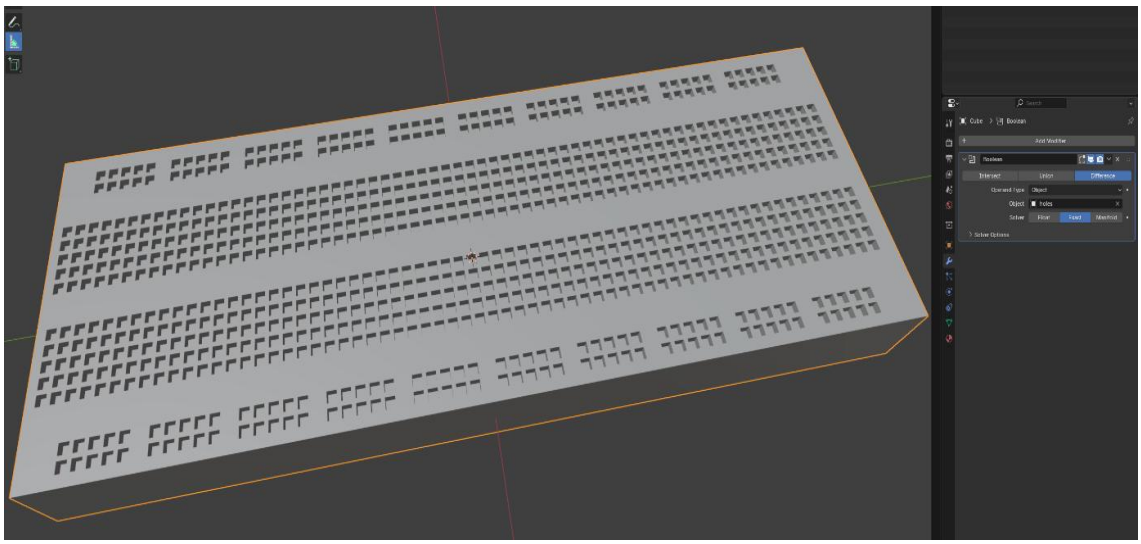
Για να μοιάζει με το χαρακτηριστικό σχέδιο ενός αληθινού breadboard, αφαιρέθηκαν κάποιες ομάδες οπών από συγκεκριμένες περιοχές, διαγράφοντας τους αντίστοιχους κύβους.



Εικόνα 3.4 Διαγραφή επιλεγμένων ομάδων κύβων και η μορφή του Breadbord.

4. Δημιουργία Οπών

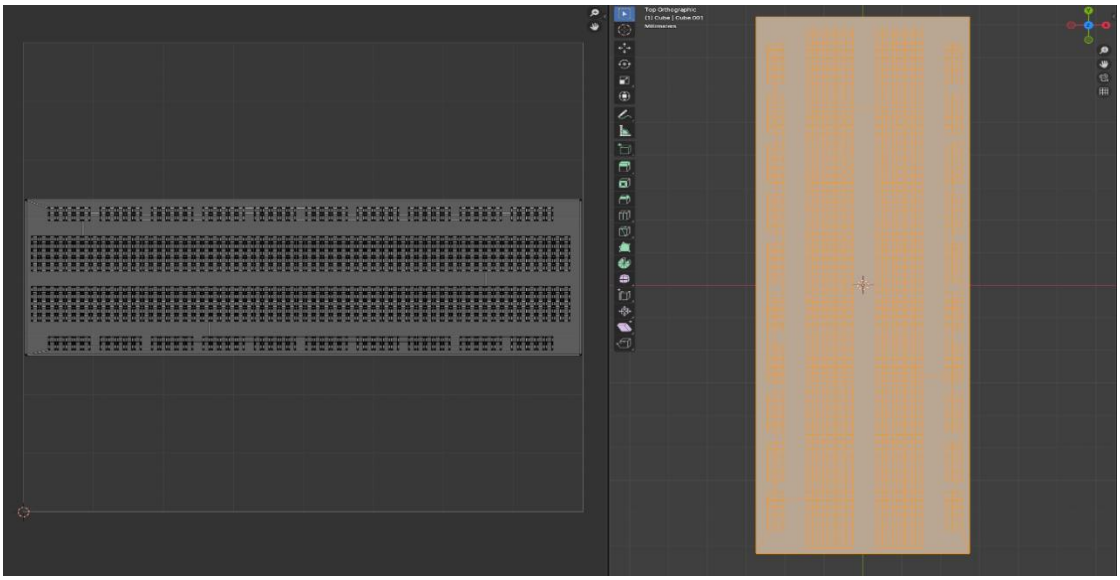
Στη βάση του breadboard εφαρμόστηκε Boolean modifier με λειτουργία Difference, επιλέγοντας ως αντικείμενο αφαίρεσης το πλέγμα των μικρών κύβων. Αυτό δημιούργησε τις οπές στη συμπαγή βάση σύμφωνα με τις πραγματικές διαστάσεις του breadboard. Η χρήση του solver Exact εξασφάλισε καθαρή τοπολογία και αξιόπιστο αποτέλεσμα, κατάλληλο για εισαγωγή και περαιτέρω επεξεργασία στο περιβάλλον της Unity



Εικόνα 3.5 Εφαρμογή Boolean Modifier στη βάση.

5. Τελική μορφή

Μετά την εφαρμογή του Boolean modifier και την διαγραφή του πλέγματος των μικρών κύβων ελέγχθηκε η ορθότητα των οπών και έγινε εξαγωγή του μοντέλου σε μορφή .fbx για επεξεργασία στο πειβάλλον της Unity.



Εικόνα 3.6 Τελική μορφή Breadboard

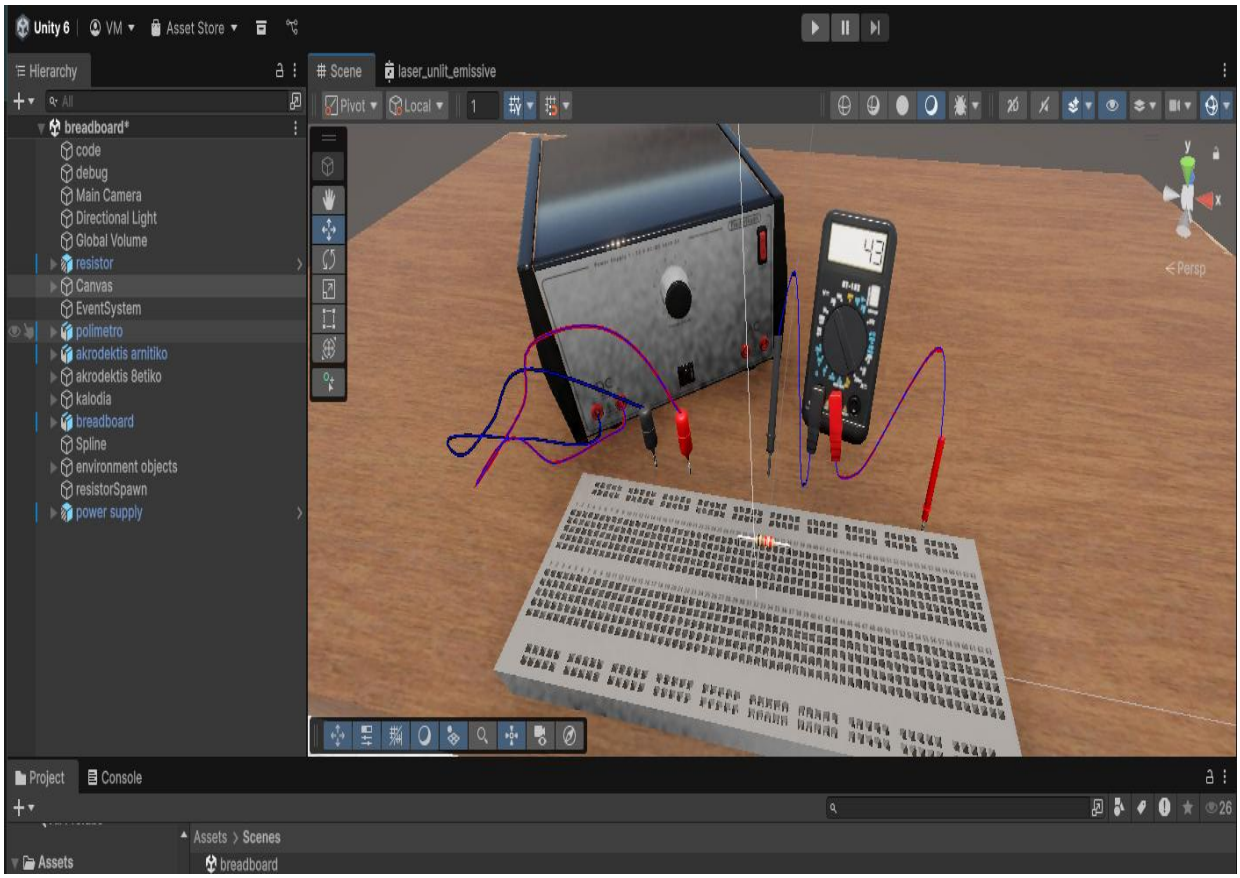
3.2.2 Δομή και Οργάνωση του Διαιρέτη Τάσης στο Unity

Μετά την εξαγωγή του μοντέλου breadboard από το Blender, προχωρήσαμε στην οργανωμένη εισαγωγή όλων των μοντέλων στο περιβάλλον Unity. Η ανάπτυξη του πειράματος στη Unity ξεκίνησε με τη δημιουργία μιας νέας σκηνής (Scene) (όπου ένα Scene είναι ένα σύνολο από αντικείμενα σε ιεραρχεία) με την ονομασία breadboard, η οποία αποτελεί το βασικό περιβάλλον δημιουργίας μέσα στο οποίο συνυπάρχουν και αλληλεπιδρούν όλα τα τρισδιάστατα αντικείμενα, τα φώτα, η κάμερα και τα scripts της εφαρμογής. Η σκηνή λειτουργεί ως ψηφιακή αναπαράσταση του φυσικού εργαστηριακού χώρου.

Αφού δημιουργήθηκε η βασική σκηνή σε αυτό το στάδιο ορίστηκαν:

- η βασική κάμερα (Main Camera), που είναι υπεύθυνη για την οπτική αναπαράσταση της σκηνής,
- το Directional Light, προκειμένου να αναπαρασταθεί ο βασικός φωτισμός του εργαστηρίου,
- Καθώς και ένα Global Volume, για τη ρύθμιση βασικώνν παραμέτρων φωτισμού και αντίθεσης.

Η ιεραρχία της σκηνής δομήθηκε σε συνεπείς ομάδες αντικειμένων (environment objects, experiment objects, UI), ενισχύοντας τη διαχείριση και τη μελλοντική επεκτασιμότητα



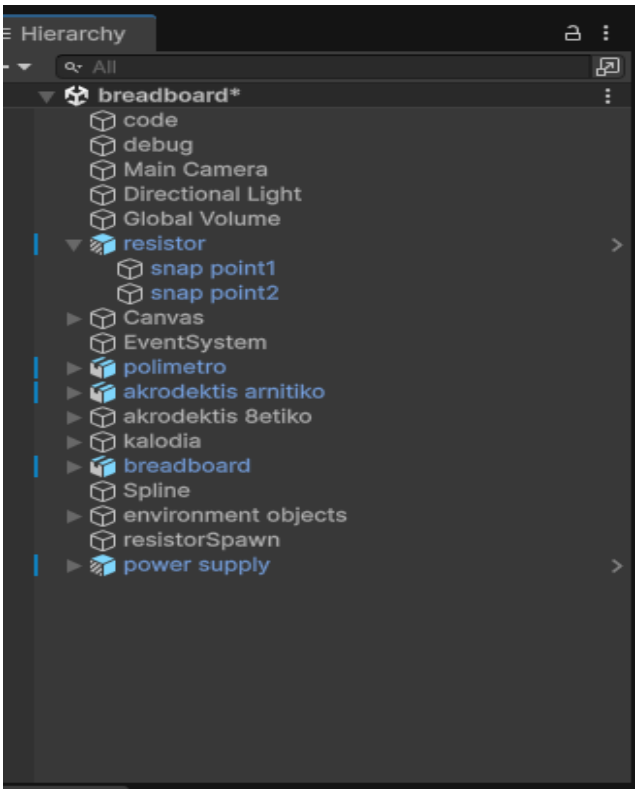
Εικόνα 3.7 Πείραμα διαιρέτη τάσης στο scene της Unity

Αρχικό βήμα ήταν το breadboard, που είναι και ο βασικός πυρήνας στο Unity, όπου:

- η σωστή κλίμακα διατηρήθηκε (1 μονάδα Unity = 1 μέτρο),
- ρυθμίστηκε ο άξονας κατεύθυνσης,
- και προστέθηκε κατάλληλος Mesh Collider για την υποστήριξη της αλληλεπίδρασης.

Το breadboard τοποθετήθηκε στο κέντρο της σκηνής, ενεργώντας ως το κύριο στοιχείο στο οποίο αναπτύσσονται τα ηλεκτρονικά κυκλώματα. Ακολούθως, συμπληρώθηκαν τα επιμέρους αντικείμενα (items) του εργαστηρίου, όπως το τροφοδοτικό σταθερού ρεύματος, το ψηφιακό πολύμετρο, οι αντιστάσεις και τα καλώδια σύνδεσης (όπου σχεδιάστηκαν με την δημιουργία καμπύλης, επιλέγοντας το Draw Splines Tool μέσω δεξιού κλικ και στη συνέχεια τροποποιόντας τις θέσεις των κόμβων από τον Inspector δόθηκε το σχήμα τους). Τα εν λόγω αντικείμενα προέρχονται από έτοιμα τρισδιάστατα μοντέλα (.fbx), τα οποία μεταφέρθηκαν στο Unity και τροποποιήθηκαν όσον αφορά τα υλικά, τα χρώματα και τη γεωμετρία τους, ώστε να επιτευχθεί οπτική συνοχή με το υπόλοιπο περιβάλλον. Ιδιαίτερη προσοχή δόθηκε στην σωστή διάταξη των αντικειμένων γύρω από το breadboard, ώστε να αναπαρίσταται μια πραγματική εργαστηριακή διάταξη.

Για την καλύτερη οργάνωση της σκηνής, τα αντικείμενα ταξινομήθηκαν ιεραρχικά στο παράθυρο Hierarchy



Εικόνα 3.8 Παάθυρο Hierarchy

Όπως φαίνεται στην Εικόνα 20, όλα τα επιμέρους στοιχεία του πειράματος συγκεντρώνονται κάτω από έναν κεντρικό κόμβο (breadboard), ο οποίος δρα ως λογική και λειτουργική ομπρέλα για την σκηνή. Αυτή η μέθοδος διευκολύνει την παρακολούθηση, τη φροντίδα και την μελλοντική ανάπτυξη της εφαρμογής. Με απλά λόγια στην Hierarchy αναπτύσεται μια λίστα αντικειμένων σκηνής Unity με:

Κύρια Ομάδα Αντικειμένων

- breadboard - Το κύριο αντικείμενο της σκηνής που περιέχει όλα τα υπόλοιπα.
- code - Φάκελος με όλα τα προγράμματα (scripts) της εφαρμογής.
- debug - Ειδικά εργαλεία για τον έλεγχο και τη διόρθωση του κώδικα.

Σύστημα Οθόνης και Φωτισμού

- Main Camera - Η κύρια κάμερα που βλέπει ο χρήστης.
- Directional Light - Το κύριο φως της σκηνής (σαν τον ήλιο).
- Global Volume - Ειδικά εφέ για την εικόνα (χρώματα, φωτεινότητα).

Ηλεκτρονικά Εξαρτήματα

- resistor - Μία αντίσταση για τα κυκλώματα.

  snap point1 - Σημείο σύνδεσης 1 για την αντίσταση.

  snap point2 - Σημείο σύνδεσης 2 για την αντίσταση.

UI (Οθόνη Ελέγχου)

- Canvas - Η οθόνη όπου εμφανίζονται τα κουμπιά και τα μενού. Το Canvas περιέχει τα γραφικά στοιχεία διεπαφής (UI), όπως ενδείξεις και πληροφορίες μετρήσεων όλα τα στοιχεία UI τοποθετούνται σε ένα "Canvas", το οποίο δημιουργείτε με δεξί κλικ στο Hierarchy (UI ->

Canvas) και επεξεργάζεστε μέσω του Rect Transform, που προσφέρει ειδικά εργαλεία για τη μορφοποίηση των γραφικών.

- EventSystem - Το σύστημα που διαχειρίζεται τα κλικ του ποντικιού. Το EventSystem χειρίζεται τα γεγονότα εισόδου του χρήστη (π.χ. κλικ, επιλογές), διευκολύνοντας τη διαδραστική λειτουργία του εικονικού εργαστηρίου.

Εργαλεία Μετρήσεων

- polimetro - Το πολύμετρο για μετρήσεις τάσης.
- akrodektis arnitiko - Ο αρνητικός ακροδέκτης (μαύρος).
- akrodektis 8etiko - Ο θετικός ακροδέκτης (κόκκινος).

Συσκευές και Περιβάλλον

- kalodia - Τα καλώδια για τις συνδέσεις.
- breadboard - Ο πίνακας συναρμολόγησης κυκλωμάτων.
- Spline - Το σύστημα για την δημιουργία καμπυλωτών καλωδίων.
- environment objects - Τα αντικείμενα του δωματίου (τραπέζι, τοίχοι).
- resistorSpawn - Η περιοχή όπου εμφανίζονται νέες αντιστάσεις.
- power supply - Η πηγή τάσης για τα κυκλώματα.

3.2.3 Ανάπτυξη κώδικα και λειτουργία πειράματος του διαιρέτη τάσης (C#).

Το εικονικό εργαστήριο διαιρέτη τάσης υλοποιείται μέσω δύο βασικών κλάσεων που συνεργάζονται για την προσομοίωση ηλεκτρικών κυκλωμάτων:

1. grid - Διαχειρίζεται το ηλεκτρικό πλέγμα του breadboard
2. dragAndDrop - Επιτρέπει τη διαδραστική τοποθέτηση εξαρτημάτων

Η βασική αρχιτεκτονική του συστήματος βασίζεται στην αφηρημένη κλάση ElectronicComponent, η οποία υλοποιείται σε C# ως κλάση που κληρονομεί από το MonoBehaviour του Unity. Κάθε συγκεκριμένο ηλεκτρονικό στοιχείο (όπως Resistor ή VoltageSource) επεκτείνει αυτήν την κλάση, διατηρώντας μια κοινή δομή με λίστα ακροδεκτών (Terminals). Στο περιβάλλον Unity, κάθε στοιχείο αναπαρίσταται ως GameObject, ενώ οι ακροδέκτες του συνδέονται μέσω κώδικα σε κόμβους (Nodes) που αντιστοιχούν σε ομάδες ηλεκτρικά συνδεδεμένων οπών του ψηφιακού breadboard. Αυτή η προσέγγιση επιτρέπει τον ξεκάθαρο διαχωρισμό λογικής και αναπαράστασης: η αφηρημένη κλάση διαχειρίζεται τη συμπεριφορά και τα δεδομένα του στοιχείου, ενώ το Unity GameObject και οι κόμβοι του breadboard διασφαλίζουν τη οπτική και διαδραστική προσομοίωση της φυσικής σύνδεσης του κυκλώματος.

ElectronicComponent.cs

```
1   using System.Collections.Generic;
2   using UnityEngine;
3
    Unity Script | 13 references
4   public abstract class electronicComponent : MonoBehaviour
5   {
6       [SerializeField]
7       public List<Terminal> Terminals = new List<Terminal>();
8   }
9
```

Χρησιμοποιείται μια αφηρημένη κλάση (abstract). Αυτό σημαίνει ότι ο κώδικας δεν ξέρει αν το αντικείμενο είναι αντίσταση ή πηγή, ξέρει μόνο ότι έχει "πόλους" (Terminals). Μπορείς να προσθέσεις αντικείμενα χωρίς να αλλάξεις τον βασικό κώδικα.

Terminal.cs

```
using UnityEngine;

Unity Script (10 asset references) | 6 references
public class Terminal : MonoBehaviour
{
    public electronicComponent eComponent;
}
```

Κάθε ακροδέκτης λειτουργεί ως "γέφυρα". Συνδέει τη φυσική θέση στο χώρο με το λογισμικό εξάρτημα.

Η προσομοίωση στηρίζεται σε δυναμική ανίχνευση επαφών, κατασκευή κόμβων και αριθμητική επίλυση των εξισώσεων κυκλώματος με επαναληπτική τεχνική, χωρίς την εκ των προτέρων χρήση καθορισμένων εξισώσεων για συγκεκριμένα κυκλώματα

Το Σύστημα Πλέγματος (grid.cs)

Ο grid.cs είναι ο "εγκέφαλος" του εργαστηρίου. Είναι αυτός που:

- Δημιουργεί το ψηφιακό αντίγραφο του φυσικού breadboard
- Κρατάει όλες τις θέσεις των τρυπών
- Υπολογίζει τις ηλεκτρικές τάσεις και ρεύματα
- Διαχειρίζεται τους ηλεκτρικούς κόμβους

Βασικά Μέρη του Κώδικα:

Αρχικοποίηση

Φτιάχνει έναν "ψηφιακό χάρτη" με 1260 θέσεις (63×20) που ακριβώς αντιστοιχούν στις τρύπες του πραγματικού breadboard.

```
Unity Script (1 asset reference) | 22 references
public class grid : MonoBehaviour
{
    public static grid instance;
    public Transform gridStartGuide;
    public float gridSpacing;
    public int nWidth ;
    public int nHeight ;

    public float[] zValues;
    public float[] xValues;

    public Vector3[,] gridPoints; // store positions for gizmos
    public List<RectInt> validPoints = new List<RectInt>(); // FIXED: proper type declaration

    public List<Node> nodes = new List<Node>();
```

Η διαδικασία δημιουργίας του πλέγματος υλοποιεί έναν αλγόριθμο παραμετρικής παραγωγής σημείων που εξασφαλίζει μετρική ακρίβεια και επεκτασιμότητα:

```
Unity Message | 0 references
void Start()
{
    instance = this;

    // Initialize arrays
    xValues = new float[nWidth];
    zValues = new float[nHeight];
    gridPoints = new Vector3[nWidth, nHeight];

    // Fill xValues (columns)
    for (int x = 0; x < nWidth; x++)
    {
        xValues[x] = gridStartGuide.position.x + (x * gridSpacing);
    }

    // Fill zValues (rows)
    for (int z = 0; z < nHeight; z++)
    {
        zValues[z] = gridStartGuide.position.z + (z * gridSpacing);
    }

    // Store positions for gizmos
    for (int x = 0; x < nWidth; x++)
    {
        for (int z = 0; z < nHeight; z++)
        {
            gridPoints[x, z] = new Vector3(xValues[x], gridStartGuide.position.y, zValues[z]);
        }
    }
    AddBreadboardRects();
    InitializeNodes();
}
```

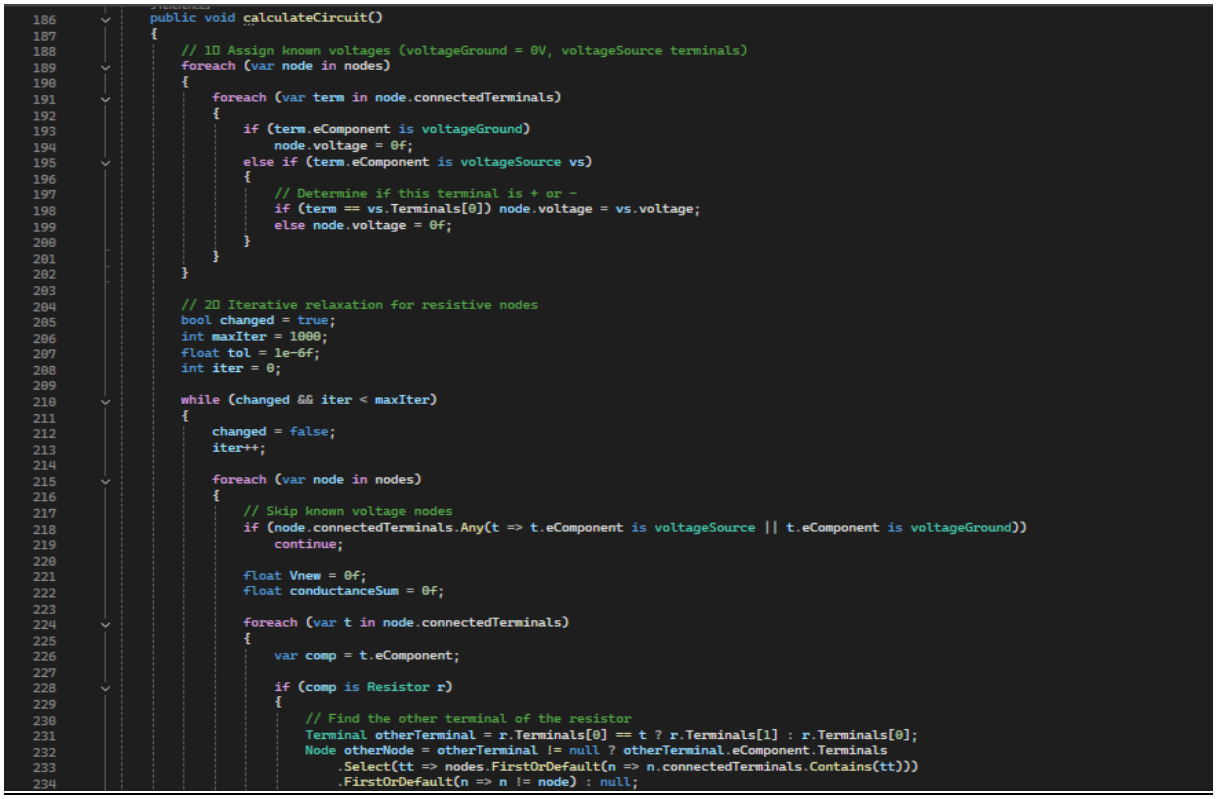
Δημιουργία Ηλεκτρικών Κόμβων

Το σύστημα κόμβων υλοποιεί έναν μηχανισμό αντιστοίχισης θέσης-συνδεσιμότητας που αντικατοπτρίζει την αρχιτεκτονική φυσικών breadboards:

```
5 references
public Node getNode(int xIndex, int yIndex)
{
    // --- Power rails (each full row is one node) ---
    if (yIndex == 0 || yIndex == 1 || yIndex == 18 || yIndex == 19)
    {
        int railIndex = yIndex switch
        {
            0 => 0,
            1 => 1,
            18 => 2,
            19 => 3,
            _ => -1
        };

        if (railIndex < 0)
            return null;

        EnsureNodeListSize(railIndex + 1);

        if (nodes[railIndex] == null)
            nodes[railIndex] = new Node();

        return nodes[railIndex];
    }

    // --- Main breadboard top section (y = 4-8) ---
    if (yIndex >= 4 && yIndex <= 8)
    {
        int nodeIndex = 4 + xIndex; // top section offset after 4 rails
        EnsureNodeListSize(nodeIndex + 1);

        if (nodes[nodeIndex] == null)
            nodes[nodeIndex] = new Node();

        return nodes[nodeIndex];
    }

    // --- Main breadboard bottom section (y = 11-15) ---
    if (yIndex >= 11 && yIndex <= 15)
    {
        int nodeIndex = 4 + 64 + xIndex; // offset for bottom section
        EnsureNodeListSize(nodeIndex + 1);

        if (nodes[nodeIndex] == null)
            nodes[nodeIndex] = new Node();

        return nodes[nodeIndex];
    }

    // If not in a valid node region
    return null;
}
```

Αλγόριθμος Προσομοίωσης Κυκλωμάτων

Η συνάρτηση calculateCircuit υλοποιεί την αριθμητική επίλυση του κυκλώματος μέσω της μεθόδου των κομβικών τάσεων, χρησιμοποιώντας έναν επαναληπτικό αλγόριθμο χαλάρωσης (Relaxation Method) που συγκλίνει στη βέλτιστη κατανομή δυναμικού βάσει του Νόμου των Ρευμάτων του Kirchhoff



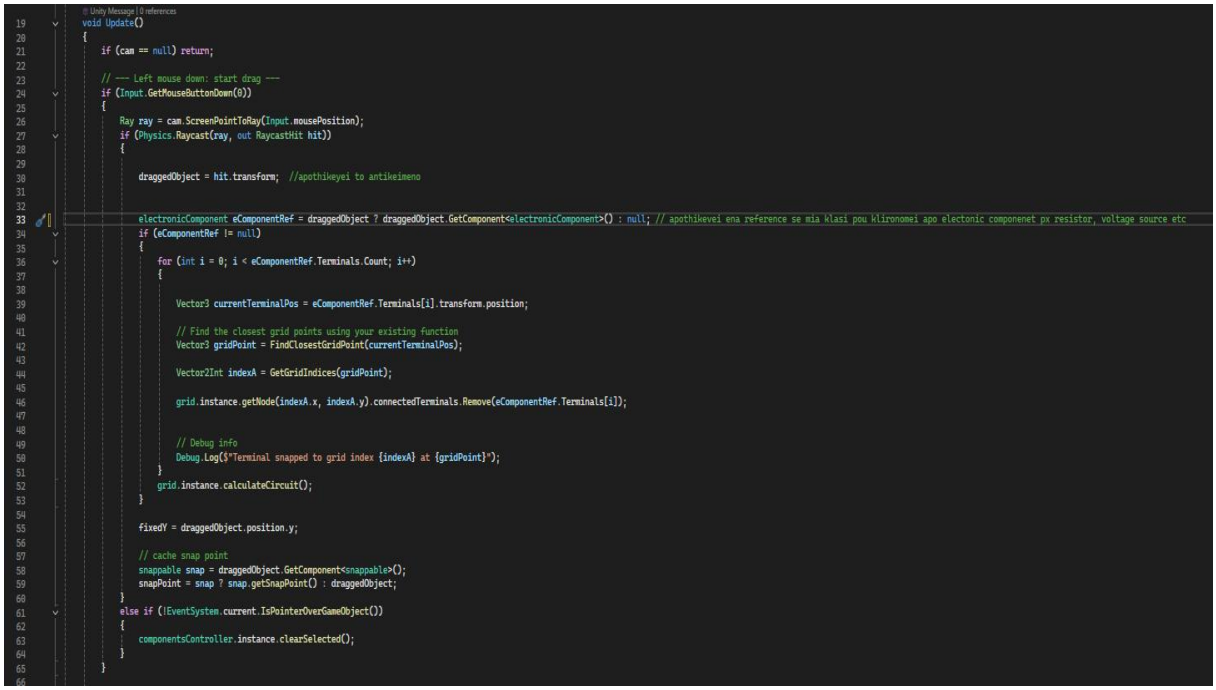
```
public void calculateCircuit()
{
    // 1️⃣ Assign known voltages (voltageGround = 0V, voltageSource terminals)
    foreach (var node in nodes)
    {
        foreach (var term in node.connectedTerminals)
        {
            if (term.eComponent is voltageGround)
                node.voltage = 0f;
            else if (term.eComponent is voltageSource vs)
            {
                // Determine if this terminal is + or -
                if (term == vs.Terminals[0]) node.voltage = vs.voltage;
                else node.voltage = 0f;
            }
        }
    }

    // 2️⃣ Iterative relaxation for resistive nodes
    bool changed = true;
    int maxIter = 1000;
    float tol = 1e-6f;
    int iter = 0;

    while (changed && iter < maxIter)
    {
        changed = false;
        iter++;

        foreach (var node in nodes)
        {
            // Skip known voltage nodes
            if (node.connectedTerminals.Any(t => t.eComponent is voltageSource || t.eComponent is voltageGround))
                continue;

            float Vnew = 0f;
            float conductanceSum = 0f;

            foreach (var t in node.connectedTerminals)
            {
                var comp = t.eComponent;

                if (comp is Resistor r)
                {
                    // Find the other terminal of the resistor
                    Terminal otherTerminal = r.Terminals[0] == t ? r.Terminals[1] : r.Terminals[0];
                    Node otherNode = otherTerminal != null ? otherTerminal.eComponent.Terminals
                        .Select(tt => nodes.FirstOrDefault(n => n.connectedTerminals.Contains(tt)))
                        .FirstOrDefault(n => n != node) : null;
```

```
                    if (otherNode == null) continue;

                    float g = 1f / Mathf.Max(r.resistance, 1e-9f);
                    Vnew += g * otherNode.voltage;
                    conductanceSum += g;
                }
                else if (comp is Wire w)
                {
                    Terminal otherTerminal = w.Terminals[0] == t ? w.Terminals[1] : w.Terminals[0];
                    Node otherNode = otherTerminal != null ? otherTerminal.eComponent.Terminals
                        .Select(tt => nodes.FirstOrDefault(n => n.connectedTerminals.Contains(tt)))
                        .FirstOrDefault(n => n != node) : null;

                    if (otherNode == null) continue;
                    float g = 1e6f; // ideal wire
                    Vnew += g * otherNode.voltage;
                    conductanceSum += g;
                }
            }

            if (conductanceSum > 0)
            {
                Vnew /= conductanceSum;
                if (Mathf.Abs(Vnew - node.voltage) > tol)
                {
                    node.voltage = Vnew;
                    changed = true;
                }
            }
        }
    }
```

Αυτή είναι η μαθηματική υλοποίηση του Νόμου του Kirchhoff. Δείχνει ότι η νέα τάση προκύπτει από την εξίσωση ισορροπίας και ότι ο αλγόριθμος "καταλαβαίνει" πότε έφτασε στη σωστή λύση.

- Ανάλυση Συστήματος Χειρισμού Αντικειμένων (dragAndDrop.cs)

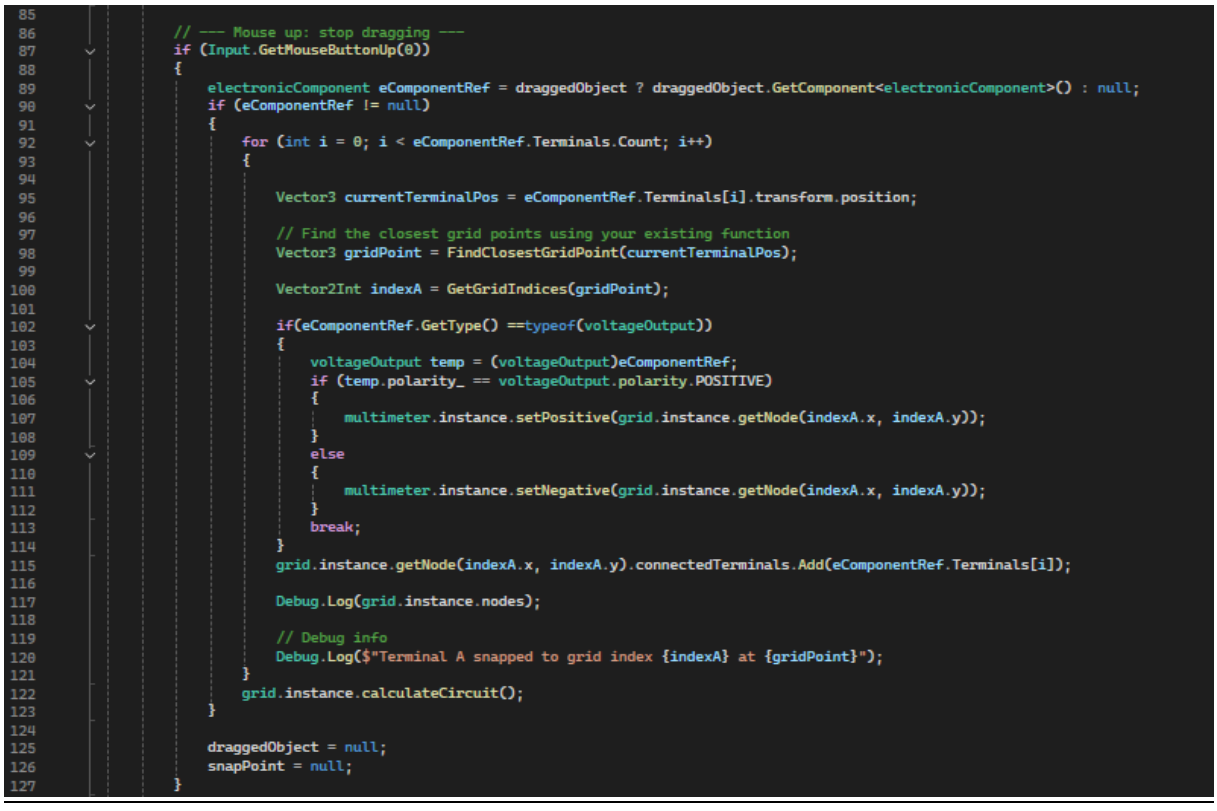
Μηχανισμός Έναρξης Μεταφοράς



```
0 Unity Message | 0 references
void Update()
{
    if (cam == null) return;

    // --- Left mouse down: start drag ---
    if (Input.GetMouseButtonDown(0))
    {
        Ray ray = cam.ScreenPointToRay(Input.mousePosition);
        if (Physics.Raycast(ray, out RaycastHit hit))
        {

            draggedObject = hit.transform;  //apothikeyei to antikeimeno


            electronicComponent eComponentRef = draggedObject ? draggedObject.GetComponent<electronicComponent>() : null; // apothikevei ena reference se mia klasi pou kliironomei apo electonic componenet px resistor, voltage source etc
            if (eComponentRef != null)
            {
                for (int i = 0; i < eComponentRef.Terminals.Count; i++)
                {

                    Vector3 currentTerminalPos = eComponentRef.Terminals[i].transform.position;

                    // Find the closest grid points using your existing function
                    Vector3 gridPoint = FindClosestGridPoint(currentTerminalPos);

                    Vector2Int indexA = GetGridIndices(gridPoint);

                    grid.instance.getNode(indexA.x, indexA.y).connectedTerminals.Remove(eComponentRef.Terminals[i]);


                    // Debug info
                    Debug.Log($"Terminal snapped to grid index {indexA} at {gridPoint}");
                }
                grid.instance.calculateCircuit();
            }

            fixedY = draggedObject.position.y;

            // cache snap point
            snappable snap = draggedObject.GetComponent<snappable>();
            snapPoint = snap ? snap.getSnapPoint() : draggedObject;
        }
        else if (!EventSystem.current.IsPointerOverGameObject())
        {
            componentsController.instance.clearSelected();
        }
    }
```

Αυτό το τμήμα υλοποιεί ένα προληπτικό σύστημα διαχείρισης συνδεσιμότητας. Πριν από την έναρξη της μεταφοράς, όλοι οι ακροδέκτες αποσυνδέονται από τους τρέχοντες κόμβους, αποτρέποντας λανθασμένες ηλεκτρικές συνδέσεις κατά τη διάρκεια της κίνησης. Η άμεση κλήση του calculateCircuit() διασφαλίζει συνεπή ηλεκτρική κατάσταση σε πραγματικό χρόνο.

Μηχανισμός Ολοκλήρωσης Μεταφοράς



```
        // --- Mouse up: stop dragging ---
        if (Input.GetMouseButtonUp(0))
        {
            electronicComponent eComponentRef = draggedObject ? draggedObject.GetComponent<electronicComponent>() : null;
            if (eComponentRef != null)
            {
                for (int i = 0; i < eComponentRef.Terminals.Count; i++)
                {

                    Vector3 currentTerminalPos = eComponentRef.Terminals[i].transform.position;

                    // Find the closest grid points using your existing function
                    Vector3 gridPoint = FindClosestGridPoint(currentTerminalPos);

                    Vector2Int indexA = GetGridIndices(gridPoint);

                    if(eComponentRef.GetType() ==typeof(voltageOutput))
                    {
                        voltageOutput temp = (voltageOutput)eComponentRef;
                        if (temp.polarity_ == voltageOutput.polarity.POSITIVE)
                        {
                            multimeter.instance.setPositive(grid.instance.getNode(indexA.x, indexA.y));
                        }
                        else
                        {
                            multimeter.instance.setNegative(grid.instance.getNode(indexA.x, indexA.y));
                        }
                        break;
                    }
                    grid.instance.getNode(indexA.x, indexA.y).connectedTerminals.Add(eComponentRef.Terminals[i]);

                    Debug.Log(grid.instance.nodes);

                    // Debug info
                    Debug.Log($"Terminal A snapped to grid index {indexA} at {gridPoint}");
                }
                grid.instance.calculateCircuit();
            }

            draggedObject = null;
            snapPoint = null;
        }
```

Η διαδικασία λήξης της μεταφοράς υλοποιεί πολυεπίπεδη επανασύνδεση με διαφοροποιημένη συμπεριφορά ανάλογα με τον τύπο του εξαρτήματος. Η έλεγχος τύπου κατά το χρόνο εκτέλεσης (GetType() == typeof(voltageOutput)) επιτρέπει εξειδικευμένες ενέργειες για ειδικές κατηγορίες εξαρτημάτων, όπως η αυτόματη ρύθμιση του πολύμετρου για πηγές τάσης.

Αλγόριθμος Αυτόματης Ευθυγράμμισης

```
4 references
public Vector3 FindClosestGridPoint(Vector3 position)
{
    Vector3[,] points = grid.instance.gridPoints;

    int width = points.GetLength(0);//63
    int height = points.GetLength(1);//20

    Vector3? closest = null;
    float closestDist = float.MaxValue;

    for (int x = 0; x < width; x++)
    {
        for (int z = 0; z < height; z++)
        {
            // Skip invalid grid holes
            if (!grid.instance.isValid(x, z))
                continue;

            Vector3 p = points[x, z];
            float dist = Vector3.SqrMagnitude(p - position);

            if (dist < closestDist)
            {
                closestDist = dist;
                closest = p;
            }
        }
    }

    // If no valid point found, fallback to first grid point
    if (closest.HasValue)
        return closest.Value;

    return points[0, 0];
}
```

Ο αλγόριθμος χρησιμοποιεί τετραγωνικές αποστάσεις (SqrMagnitude) για βελτιστοποίηση επιδόσεων, αποφεύγοντας την υπολογιστικά ακριβή τετραγωνική ρίζα. Η διπλή επανάληψη επιτρέπει πλήρη εξερεύνηση του χώρου αναζήτησης, ενώ η συνθήκη isValid εγγυάται ότι μόνο οι λειτουργικές οπές λαμβάνονται υπόψη. Το μοτίβο Nullable (Vector3?) προσφέρει ασφαλή χειρισμό περιπτώσεων όπου δεν βρίσκεται έγκυρο σημείο.

Δυναμική Προβολή και Ισοπέδωση

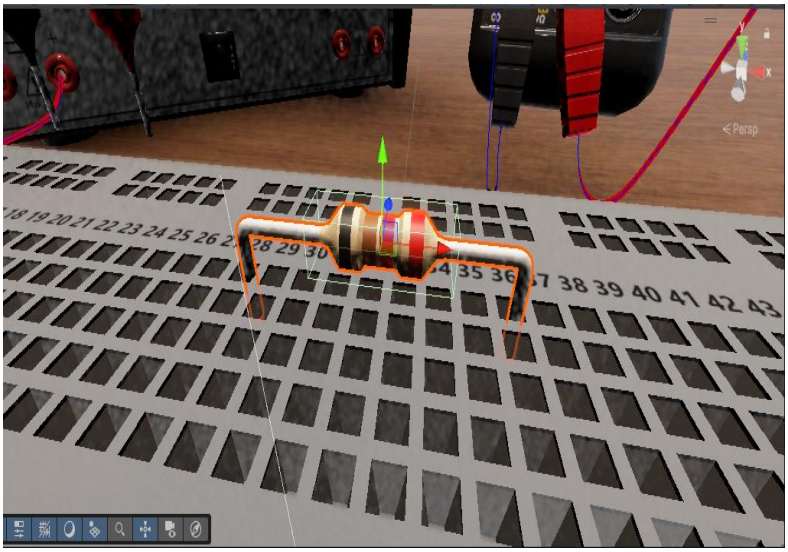
```
}

1 reference
private Vector3 ProjectMouseToPlane(float y)
{
    Ray ray = cam.ScreenPointToRay(Input.mousePosition);
    Plane plane = new Plane(Vector3.up, new Vector3(0, y, 0));
    if (plane.Raycast(ray, out float enter))
    {
        return ray.GetPoint(enter);
    }
    return Vector3.zero;
}
```

Η μέθοδος εφαρμόζει γεωμετρική απεικόνιση σε οριζόντιο επίπεδο, μετατρέποντας τις δισδιάστατες συντεταγμένες του ποντικιού σε ένα τρισδιάστατο σημείο στον ψηφιακό χώρο. Αυτή η μαθηματική περιγραφή εξασφαλίζει ομαλή και φυσική κίνηση των αντικειμένων, αποφεύγοντας ζητήματα βάθους και προοπτικής.

Resistor και χρωματικός κώδικας

- Σχεδιασμός Resistor

Δομική οργάνωση



Εικόνα 3.9 Resistor στη Unity

Μοντέλο 3D: Το resistor αποτελείται από ένα mesh με 6 υλικά (materials)

Κύρια Μέρη:

- ResistorBaseMaterial: Για το κυρίως σώμα
- IronMaterial: Για τα μεταλλικά άκρα
- 4 Band Materials: Για τις χρωματικές ζώνες (band 1-4)

Box Collider: Προσαρμοσμένος στις διαστάσεις

Συνιστώσες (Components)

Resistor (Script): Πυρήνας λογικής με:

- Terminals = 2 (θετικός και αρνητικός ακροδέκτης)
- Resistance = 1000 Ω (βασική τιμή)

Resistor Color Code (Script): Διαχείριση χρωματικού κωδικά:

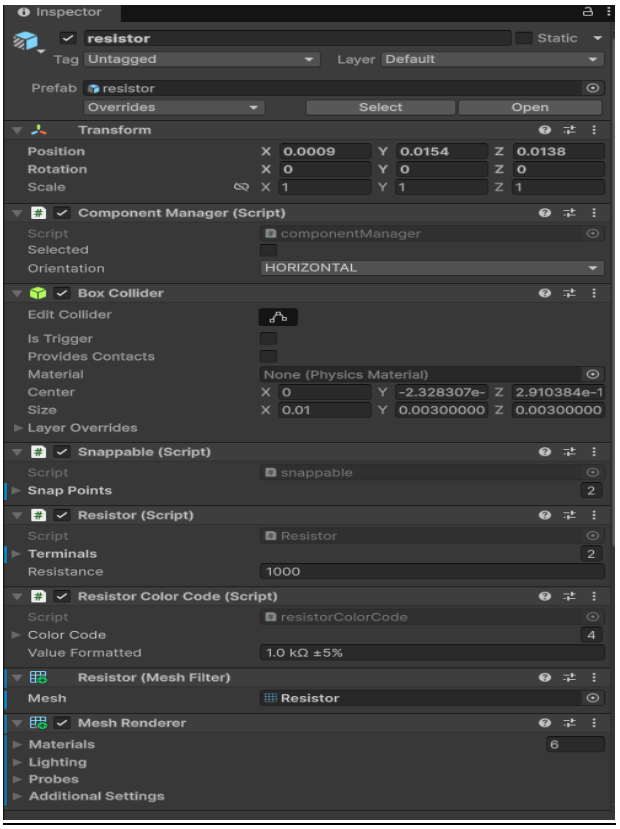
- Color Code = 4 (4-ζώνης αντίσταση)
- Value Formatted = "1.0 kΩ ±5%"

Component Manager: Διαχείριση προσανατολισμού:

- Orientation = HORIZONTAL

Snappable System: Για το snapping σε breadboard:

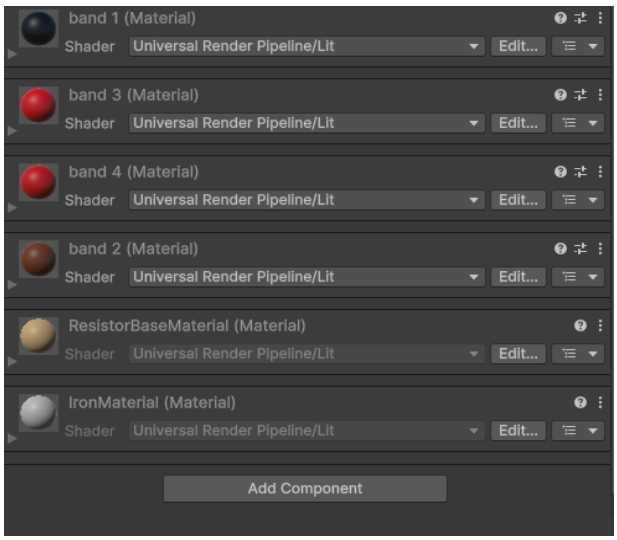
- Snap Points = 2 (αντιστοιχούν στους ακροδέκτες)



Εικόνα 3.10 Inspector του Resistor

Σχεδιασμός Χρωματικού Κωδικά

- Band 1: Πρώτο ψηφίο (χρώμα: [προσδιορίζεται από material])
- Band 2: Δεύτερο ψηφίο
- Band 3: Πολλαπλασιαστής
- Band 4: Ανοχή (±5%,±10%)



Εικόνα 3.11 Τα Bands και τα χρώματα του resistor από το Inspector

Υπολογισμός Τιμής

Η κλάση resistorColorCode υλοποιεί το σύστημα εμφάνισης του χρωματικού κωδικά μιας αντίστασης στο 3D μοντέλο. Μεταφράζει μια αριθμητική τιμή αντίστασης σε οπτική αναπαράσταση μέσω 4 χρωματικών ζωνών, ακριβώς όπως οι πραγματικές αντιστάσεις.

```
using System;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UIElements;

Unity Script (1 asset reference) | 4 references
public class resistorColorCode : MonoBehaviour
{

    private MeshRenderer renderer;
    private readonly Dictionary<int, int> _bandIndexToMaterialSlot = new Dictionary<int, int>()
    {
        // Color Code Index -> Mesh Material Slot Index
        { 0, 3 }, // Band 1 color code (index 0) goes into slot 3
        { 1, 4 }, // Band 2 color code (index 1) goes into slot 4
        { 2, 5 }, // Band 3 (Multiplier) color code (index 2) goes into slot 5
        { 3, 2 }  // Band 4 (Tolerance) color code (index 3) goes into slot 2
    };

    public int[] colorCode = new int[4];
    public string valueFormatted;


    Unity Message | 0 references
    void Start()
    {
        renderer = GetComponent<MeshRenderer>();
    }
```

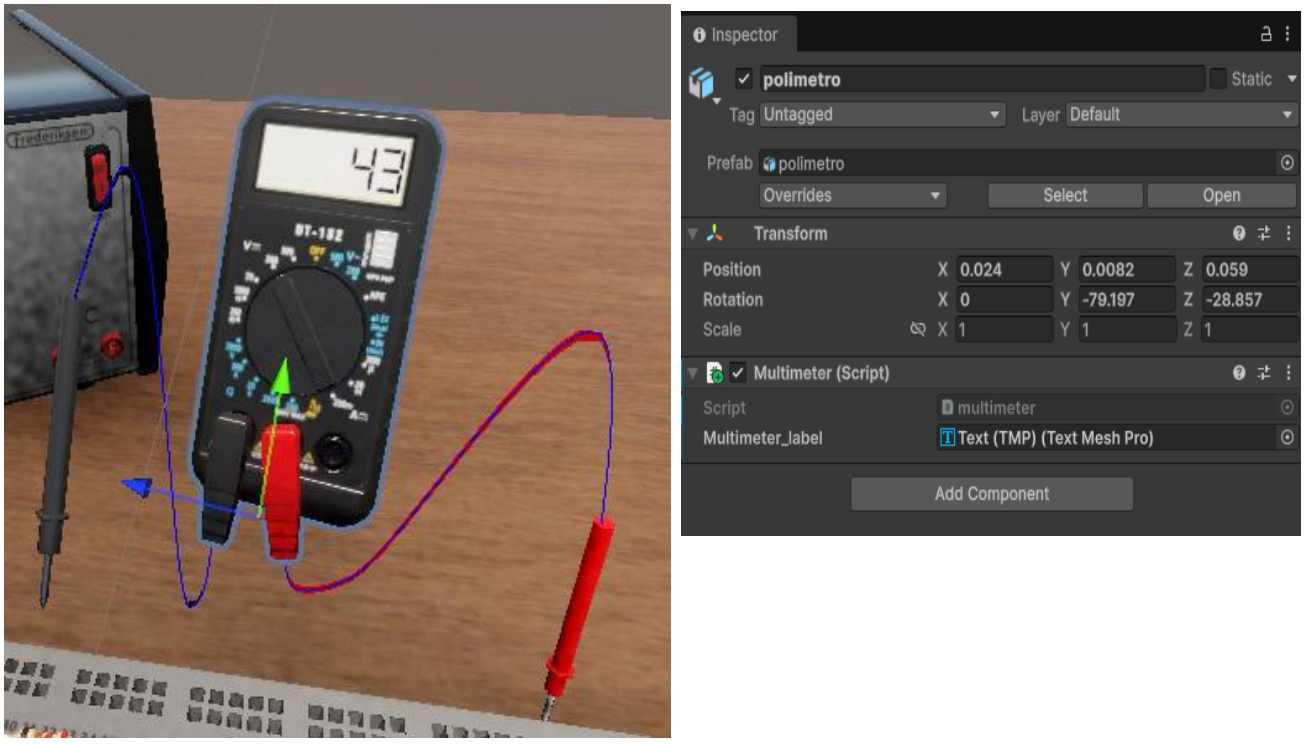
Στην πράξη, επειδή το τρισδιάστατο μοντέλο της αντίστασης έχει σχεδιαστεί με συγκεκριμένο τρόπο, οι χρωματικές ζώνες που βλέπουμε δεν ακολουθούν την ίδια σειρά με τα υλικά (materials) στο Unity. Για να λυθεί αυτό, χρησιμοποιούμε ένα λεξικό αντιστοίχισης (mapping), το οποίο λειτουργεί σαν «μεταφραστής»: λέει στο πρόγραμμα ότι η πρώτη ζώνη της αντίστασης αντιστοιχεί, για παράδειγμα, στο τέταρτο υλικό του μοντέλου. Παράλληλα, χρησιμοποιούμε έναν πίνακα δεδομένων (colorCode) που κρατάει αποθηκευμένα τα χρώματα, ενώ η μεταβλητή valueFormatted χρησιμεύει μόνο για να εμφανίζεται η τιμή της αντίστασης με κατανοητό τρόπο στην οθόνη του χρήστη (π.χ. "10kΩ"), χωρίς να επηρεάζει το πώς βάφεται το μοντέλο.

```
    Unity Message | 0 references
    void Start()
    {
        renderer = GetComponent<MeshRenderer>();
    }


    1 reference
    public void setColorCode(int[] code) // Using PascalCase for public method name
    {
        if (renderer == null || code == null || code.Length != 4)
        {
            Debug.LogError("Cannot set color code: Renderer not initialized or code array is invalid (expected 4 elements).");
            return;
        }
        Array.Copy(code, colorCode, colorCode.Length);

        // 1. Get a COPY of the materials array from the renderer
        Material[] meshMaterials = renderer.materials;

        for (int i = 0; i < code.Length; i++)
        {
            int colorCode = code[i];

            if (_bandIndexToMaterialSlot.TryGetValue(i, out int slotIndex))
            {
                // Check if the material slot exists in the mesh renderer
                if (slotIndex >= meshMaterials.Length)
                {
                    Debug.LogError($"Material slot index {slotIndex} is out of bounds for the MeshRenderer materials array.");
                    continue;
                }

                try
                {
                    // This line modifies the materials array COPY
                    meshMaterials[slotIndex] = componentsController.instance.bandMaterials[colorCode];
                }
                catch (System.Exception e)
                {
                    // Catch common errors like componentsController.instance being null or bandMaterials index being out of bounds
                    Debug.LogError($"Failed to set material for Band {i} (Slot {slotIndex}, Color Code {colorCode}). Error: {e.Message}");
                }
            }
        }
```

Η υλοποίηση βασίζεται σε μια δομημένη διαδικασία αρχικοποίησης και ασφάλειας, η οποία ξεκινά από τη μέθοδο Start() για την ορθή σύνδεση με το σύστημα γραφικών της Unity, αποφεύγοντας σφάλματα χρονισμού. Το σύστημα επιβάλλει αυστηρούς ελέγχους (όπως τον περιορισμό στις 4 χρωματικές ζώνες), διασφαλίζοντας ότι το μοντέλο της αντίστασης ακολουθεί πάντα τα διεθνή πρότυπα. Για τη διαχείριση των δεδομένων, χρησιμοποιείται η τεχνική της βελτιστοποιημένης αντιγραφής πινάκων (Array.Copy) αντί για απλή εκχώρηση, ώστε να αποφεύγονται ανεπιθύμητες ταυτόχρονες αλλαγές που θα μπέρδευαν τα χρώματα. Η τροποποίηση της εμφάνισης ακολουθεί το πρότυπο «Λήψη-Τροποποίηση-Επιστροφή», το οποίο είναι ο μόνος ασφαλής τρόπος για να αλλάξουν τα υλικά (materials) στο Unity, ενώ η χρήση λεξικού με τη μέθοδο TryGetValue εγγυάται ότι κάθε χρώμα θα τοποθετηθεί στη σωστή γεωμετρική θέση χωρίς τον κίνδυνο κρασαρίσματος. Τέλος, ένα διπλό στρώμα προστασίας με προληπτικούς ελέγχους και διαχείριση εξαιρέσεων (try-catch) θωρακίζει το πείραμα, επιτρέποντας στο σύστημα να συνεχίζει τη λειτουργία του ακόμα και αν προκύψει κάποια απρόβλεπτη τιμή, παρέχοντας ταυτόχρονα χρήσιμες διαγνωστικές πληροφορίες.

Πολύμετρο

Το πολύμετρο είναι ένα εικονικό μέσο μέτρησης που λειτουργεί ως βολτόμετρο σε ένα προσομοιωμένο ηλεκτρικό σύστημα. Η κύρια χρήση του στο πείραμα είναι να υπολογίζει και να δείχνει τη διαφορά της τάσης (σε Volts) ανάμεσα σε δύο σημεία του κυκλώματος. Για την οπτική απεικόνιση, η τιμή αυτή μεταφέρεται στο πεδίο multimeter_label, το οποίο έχει τοποθετηθεί πάνω στο 3D μοντέλο της οθόνης (screen) του πολυμέτρου εντός της Unity, προσφέροντας μια ρεαλιστική εμπειρία ανάγνωσης των μετρήσεων στον χρήστη.



Εικόνα 3.8 Η 3D μορφή και το Inpector του πολύμετρου.

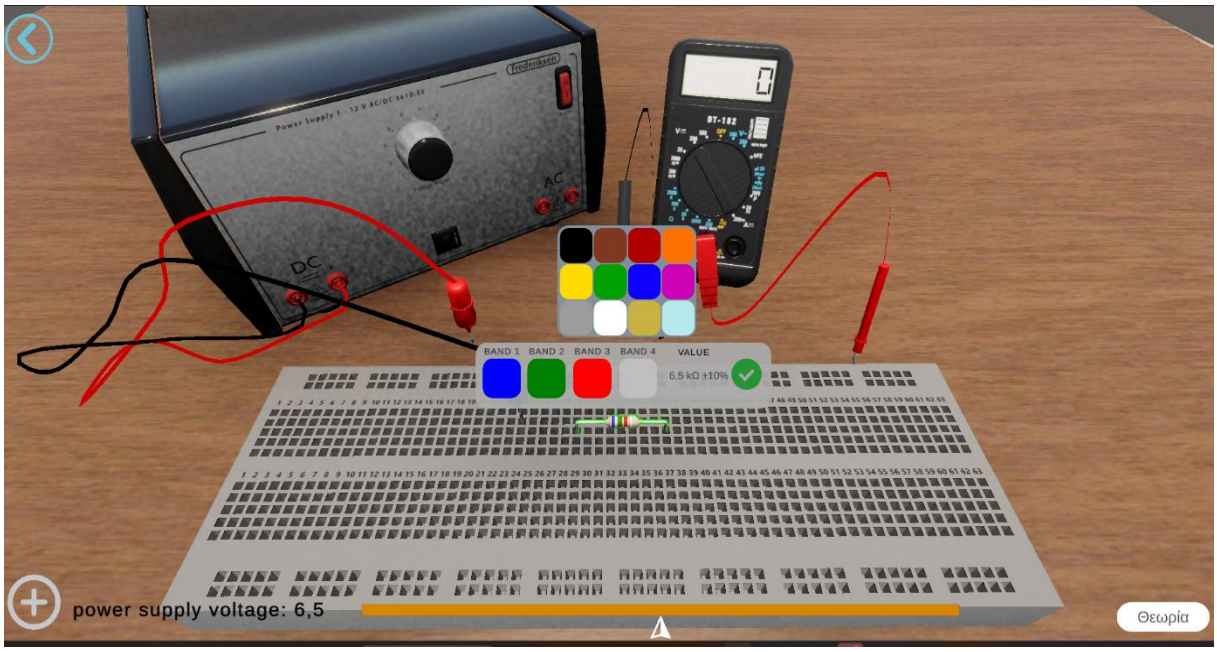
Η τεχνική εκτέλεση του multimeter.cs εστιάζει στη δυναμική σύνδεση της χωρικής αλληλεπίδρασης με την ηλεκτρική θεωρία, μετατρέποντας τις κινήσεις του χρήστη σε μαθηματικά δεδομένα. Μέσω των μεθόδων setPositive και setNegative, το script αντιστοιχίζει τους φυσικούς ακροδέκτες της συσκευής με τους συγκεκριμένους κόμβους (Nodes) του κυκλώματος που επιλέγει ο χρήστης, επιτρέποντας στο όργανο να «παρακολουθεί» την τάση σε πραγματικό χρόνο. Η αξιοπιστία της μέτρησης θωρακίζεται από έναν έλεγχο ηλεκτρικής συνέχειας (connectedTerminals.Count > 0), ο οποίος διασφαλίζει ότι το πολύμετρο θα δείξει τιμή μόνο όταν οι ακίδες εφάπτονται σε ενεργά τμήματα του κυκλώματος, αποφεύγοντας έτσι παραπλανητικές ενδείξεις σε άδεια σημεία του breadboard. Ο πυρήνας της λειτουργίας του βασίζεται στον υπολογισμό της διαφοράς δυναμικού (ΔV = V+ - V-), όπου προβάλλεται απευθείας σε ένα World Space Canvas ενσωματωμένο στο 3D μοντέλο της οθόνης.

```
using System;
using TMPro;
using UnityEngine;

Unity Script (1 asset reference) | 3 references
public class multimeter : MonoBehaviour
{
    public static multimeter instance;
    public TMP_Text multimeter_label;

    private Node negative;
    private Node positive;
    // Start is called once before the first execution of Update after the MonoBehaviour is created
    Unity Message | 0 references
    void Start()
    {
        instance = this;
    }
    1 reference
    public void setPositive(Node node)
    {
        positive = node;
    }
    1 reference
    public void setNegative(Node node)
    {
        negative = node;
    }

    Unity Message | 0 references
    private void Update()
    {
        if (positive == null || negative == null) {
            multimeter_label.text = "0";

            return;
        }
        if (positive.connectedTerminals.Count>0 && negative.connectedTerminals.Count>0)
        {
            multimeter_label.text = Math.Round(positive.voltage - negative.voltage, 2).ToString();
        }
        else
        {
            multimeter_label.text = "0";
        }
    }
```

3.2.4 Λειτουργία Πειράματος στην πράξη

Βήμα 1: Ρύθμιση Αντίστασης και Τροφοδοσίας

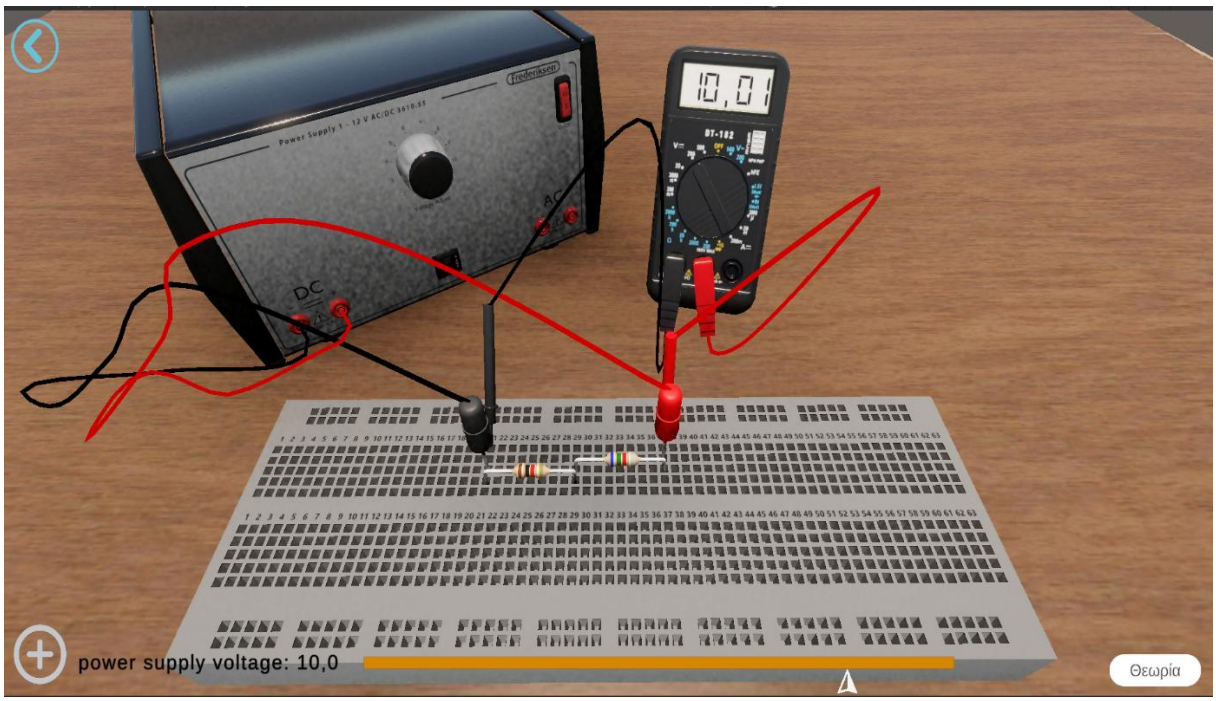
- Επιλογή Χρωμάτων: Ο χρήστης επιλέγει τα χρώματα στις ζώνες της αντίστασης για να ορίσει την τιμή της (π.χ. 6.5 kΩ).
- Οπτική Αλλαγή: Το τρισδιάστατο μοντέλο της αντίστασης «βάφεται» αυτόματα με τα επιλεγμένα χρώματα.



Εικόνα 3.12 Χρωματικός κώδικας του πειράματος.

Βήμα 2: Μέτρηση Συνολικής Τάσης Εισόδου

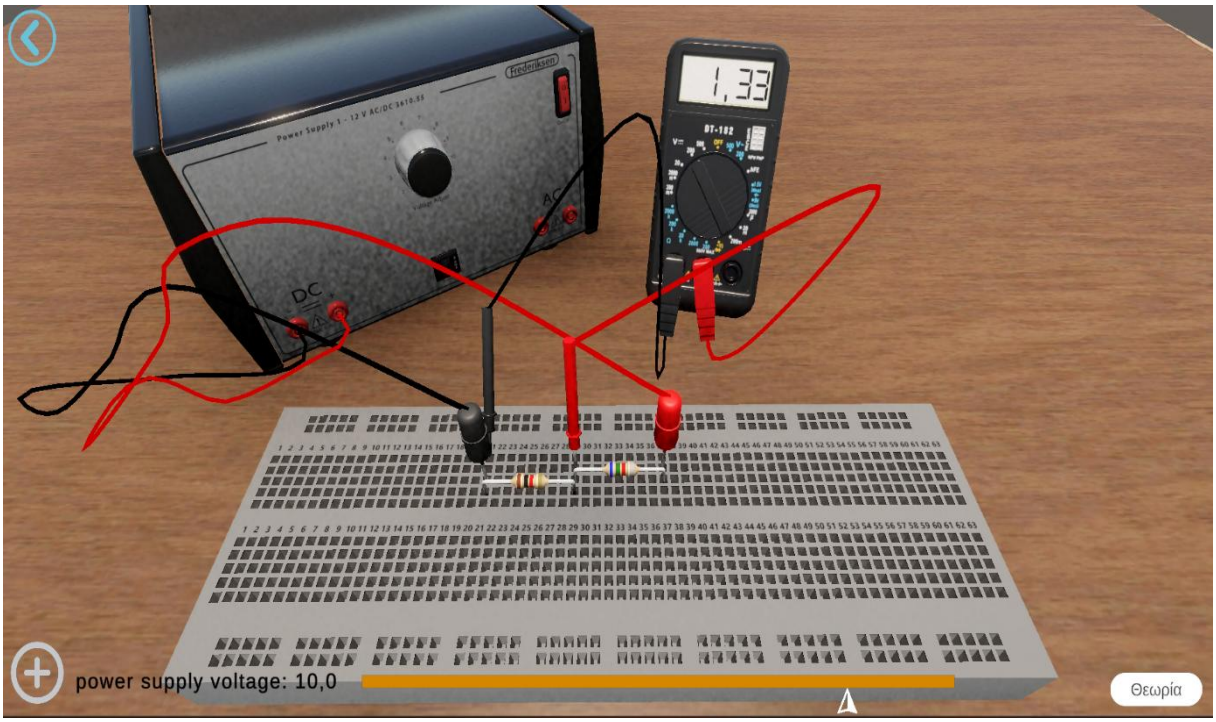
- Σύνδεση Πηγής: Τα καλώδια της πηγής συνδέονται στα άκρα της διαδρομής των δύο αντιστατών πάνω στο breadboard.
- Έλεγχος Εισόδου: Τοποθετούμε τους ακροδέκτες του πολυμέτρου κατά μήκος των αντιστάσεων για να επιβεβαιώσουμε ότι η συνολική τάση είναι 10V.
- Λειτουργία Κυκλώματος: Ο κώδικας (grid) υπολογίζει αμέσως πώς μοιράζεται η τάση σε όλο το μήκος της διαδρομής.



Εικόνα 3.13 Μέτρηση στα άκρα και των δύο αντιστάσεων.

Βήμα 3: Μέτρηση Πτώσης Τάσης στην 1η Αντίσταση

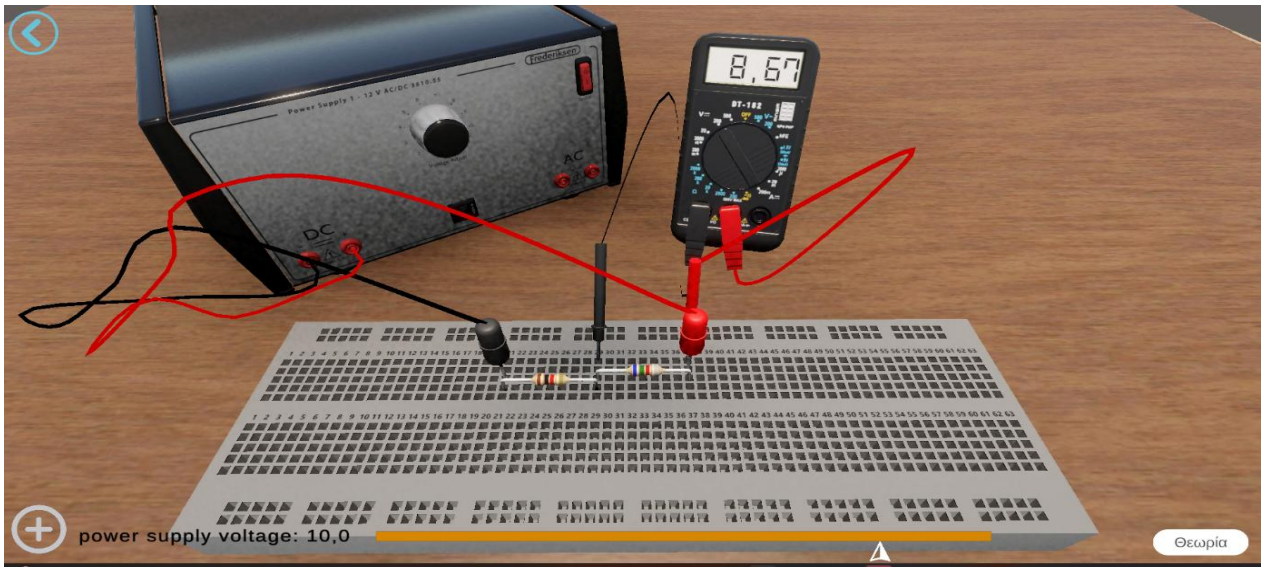
- Μετακίνηση Θετικού Ακροδέκτη: Μετακινούμε τον κόκκινο ακροδέκτη στο σημείο που ενώνονται οι δύο αντιστάσεις.
- Αποτέλεσμα Διαιρέτη: Το πολύμετρο μας δείχνει ότι ένα μέρος της τάσης (1.33V) «καταναλώνεται» από την πρώτη αντίσταση.
- Άμεση Ενημέρωση: Η οθόνη του οργάνου αλλάζει την ένδειξή της μόλις ακουμπήσουμε τον ακροδέκτη στη νέα θέση.



Εικόνα 3.14 Μέτρηση τάσης στην 1kΩ αντίσταση.

Βήμα 4: Μέτρηση Τάσης Εξόδου στη 2η Αντίσταση

- Τελική Μέτρηση: Τοποθετούμε τους ακροδέκτες στα άκρα της δεύτερης αντίστασης για να δούμε την τάση εξόδου.
- Επαλήθευση: Το άθροισμα των δύο μετρήσεων (1.33 + 8.67) μας δίνει ακριβώς τα 10V της πηγής, αποδεικνύοντας ότι το πείραμα λειτουργεί σωστά.
- Ολοκλήρωση: Η τιμή εμφανίζεται καθαρά στην οθόνη του πολυμέτρου, ολοκληρώνοντας το πείραμα με επιτυχία.



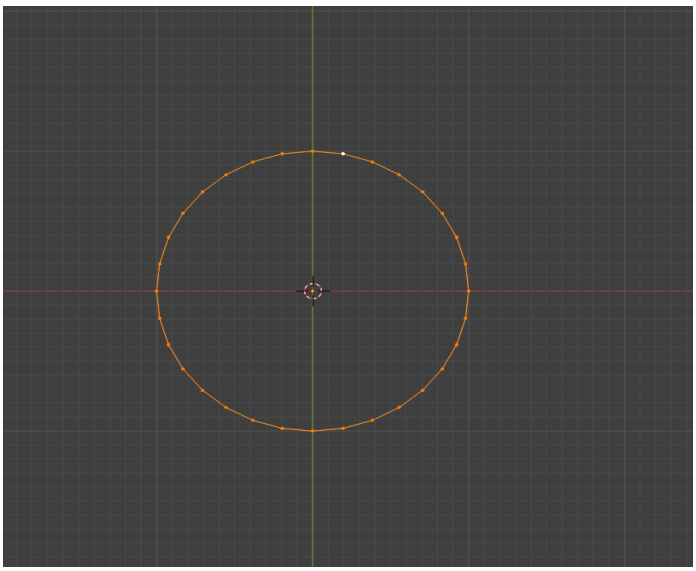
Εικόνα 3.15 Μέτρηση τάσης στην 6.5kΩ αντίσταση.

3.3 Κινητήρας Συνεχούς Ρεύματος (DC Motor)

Η λειτουργία ενός κινητήρα συνεχούς ρεύματος στηρίζεται στην ουσιαστική αλληλεπίδραση του ηλεκτρικού ρεύματος με το μαγνητικό πεδίο. Όταν ένας αγωγός που διαπερνάται από ηλεκτρικό ρεύμα τοποθετείται μέσα σε μαγνητικό πεδίο, δέχεται μηχανική δύναμη, γνωστή ως δύναμη Lorentz, η οποία είναι κάθετη στη διεύθυνση του ρεύματος και στη διεύθυνση του μαγνητικού πεδίου. Στον κινητήρα συνεχούς ρεύματος, το μαγνητικό πεδίο παράγεται από μόνιμους μαγνήτες, ενώ το ρεύμα ρέει μέσα από τα τυλίγματα του ρότορα. Η συνολική δύναμη που ασκείται στους αγωγούς του ρότορα δημιουργεί ροπή γύρω από τον άξονά του, προκαλώντας περιστροφή. Η αδιάκοπη περιστροφή εξασφαλίζεται με την κατάλληλη μεταβολή της κατεύθυνσης του ρεύματος στα πηνία, έτσι ώστε η ροπή να διατηρεί σταθερή κατεύθυνση, με αποτέλεσμα τη μετατροπή της ηλεκτρικής ενέργειας σε μηχανική.

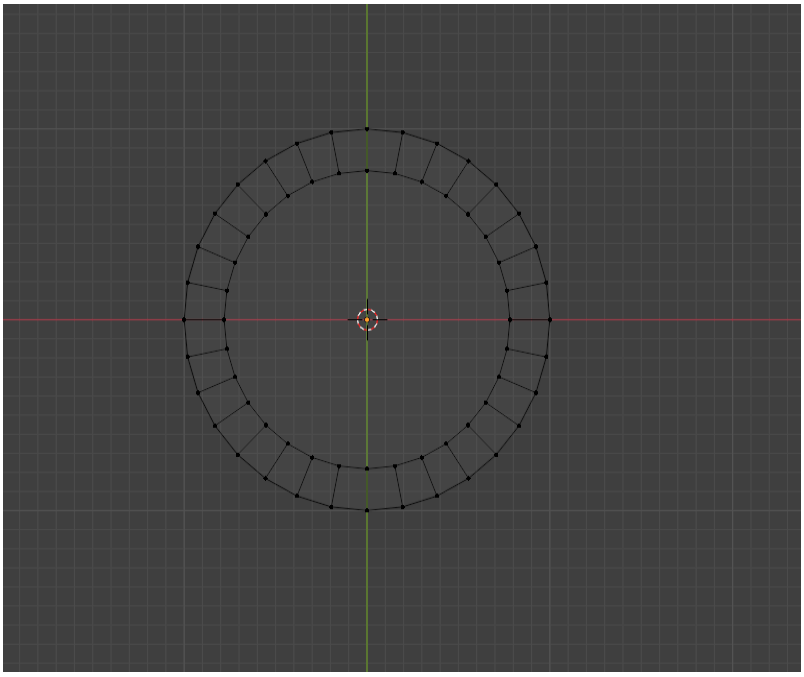
3.3.1 Σχεδίαση και Τρισδιάστατη Μοντελοποίηση στο Blender

Η διαδικασία ξεκινά στο Blender με την εισαγωγή ενός αντικειμένου τύπου Circle (Mesh), το οποίο θα αποτελέσει τη βάση.



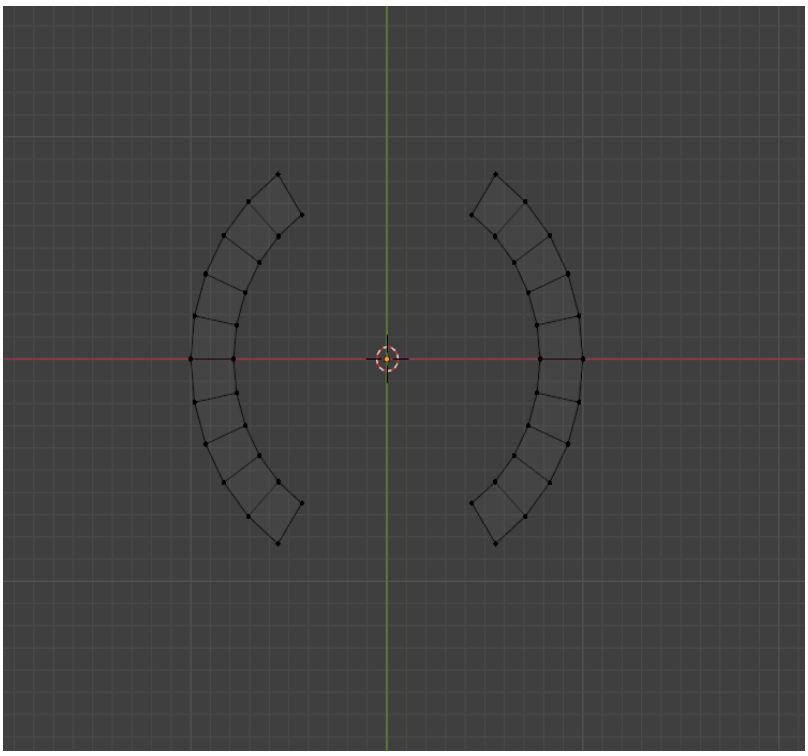
Εικόνα 3.18 Σχεδίαση κύκλου.

Με τη χρήση των εντολών Extrude και Scale, ο αρχικός κύκλος αποκτά εσωτερικό πάχος δημιουργώντας μια δισδιάστατη στεφάνη.



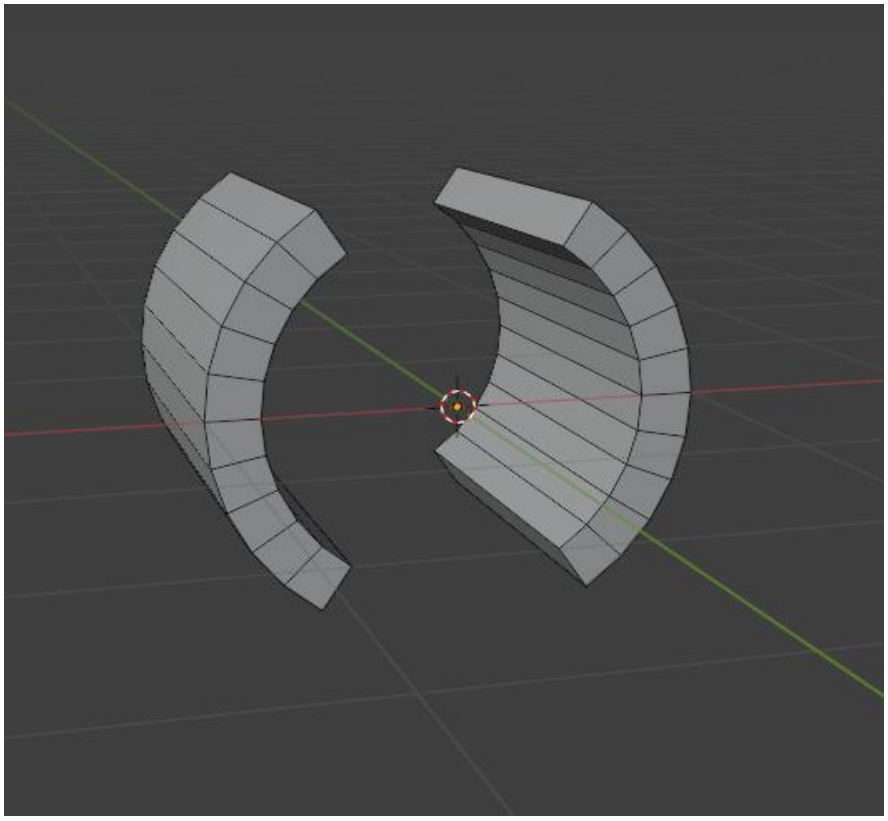
Εικόνα 3.19 Δημιουργία στεφάνης.

Επιλέγονται και διαγράφονται συγκεκριμένα faces της στεφάνης ώστε να δημιουργηθούν τα δύο κενά που χωρίζουν τους μόνιμους μαγνήτες του κινητήρα.



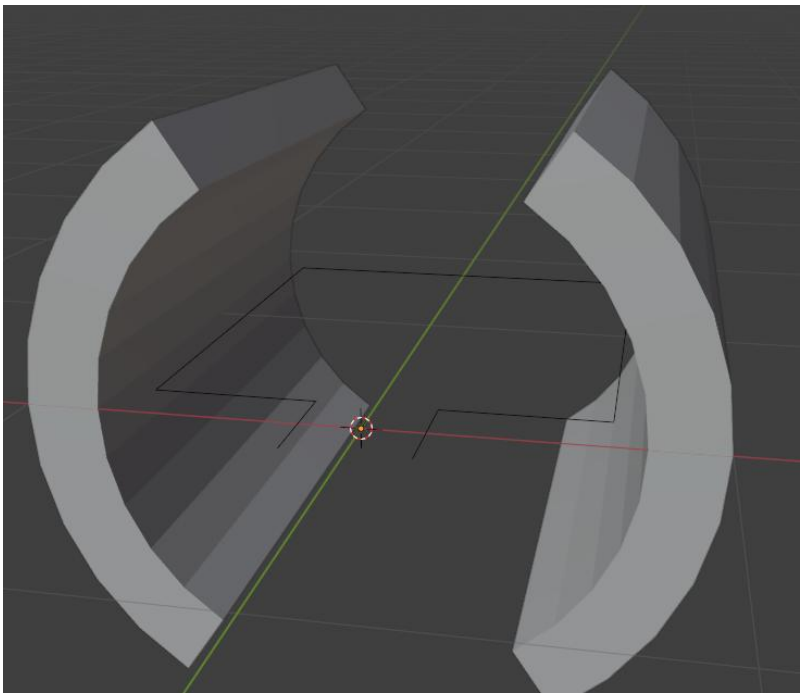
Εικόνα 3.20 Διαχωρισμός Πόλων.

Τα υπολειπόμενα ημικυκλικά τμήματα εξωθούνται (extrude) στον άξονα Z ώστε να αποκτήσουν το τελικό τους τρισδιάστατο βάθος.



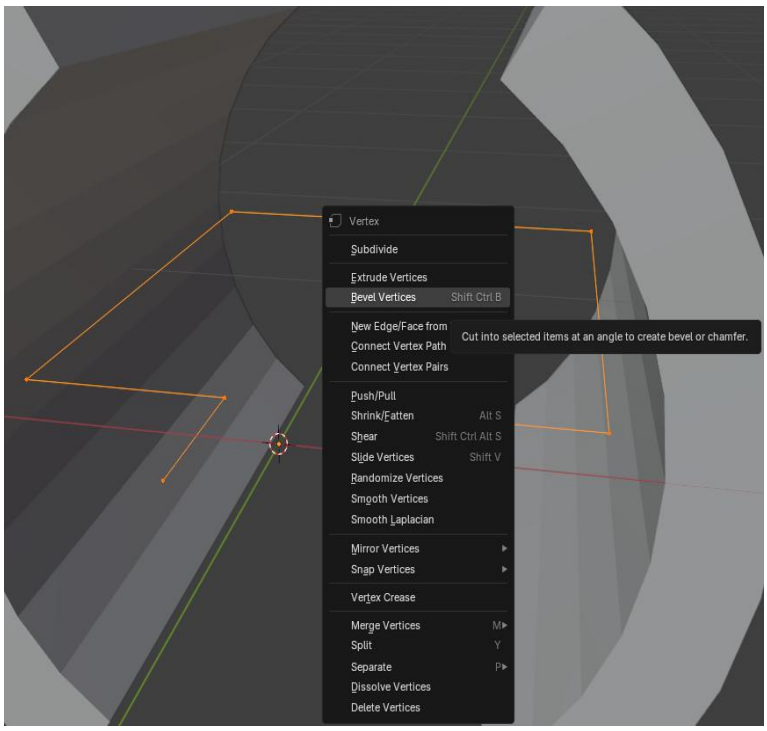
Εικόνα 3.21 Τρισδιάστατος Όγκος Μαγνητών.

Σχεδιάζεται ένα ορθογώνιο πλαίσιο με vertices που αναπαριστά τη διαδρομή του σύρματος της περιέλιξης (rotor winding) στον χώρο.



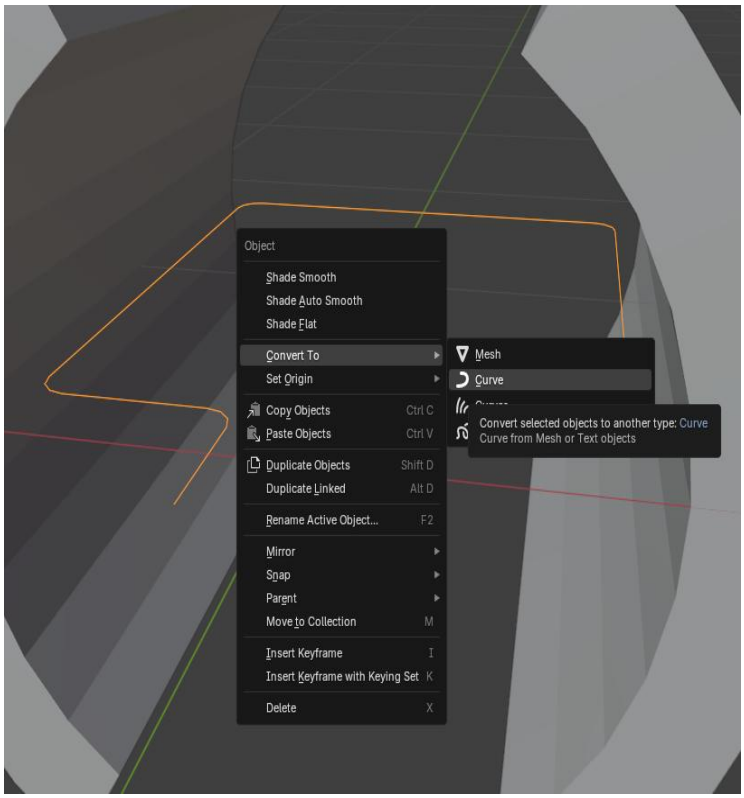
Εικόνα 3.22 Σχεδίαση Διαδρομής Πλαισίου.

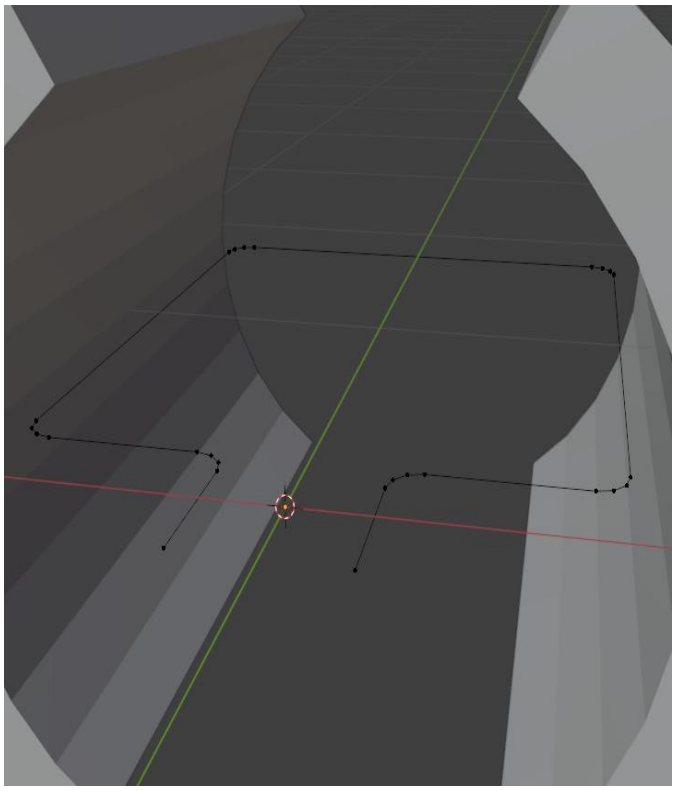
Εφαρμόζεται η εντολή Bevel Vertices (Shift+Ctrl+B) στις γωνίες του πλαισίου για να αποδοθεί η φυσική καμπυλότητα ενός πραγματικού αγωγού.



Εικόνα 3.23 Στρογγυλοποίηση Γωνιών.

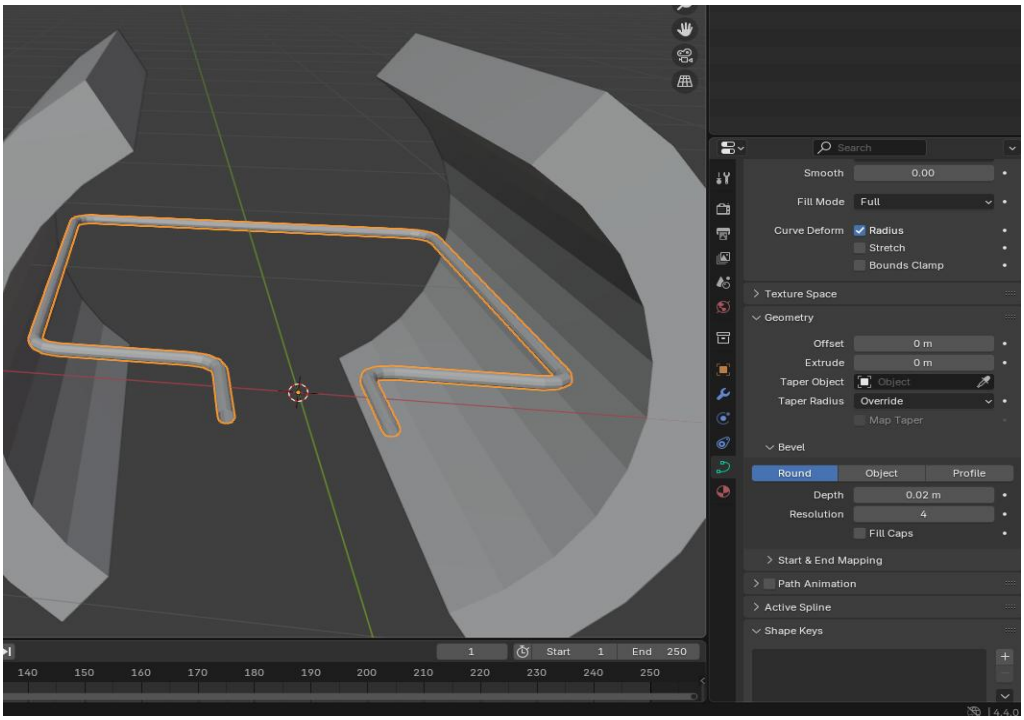
Το mesh του πλαισίου μετατρέπεται σε αντικείμενο τύπου Curve, επιτρέποντας την εφαρμογή δυναμικών παραμέτρων πάχους.





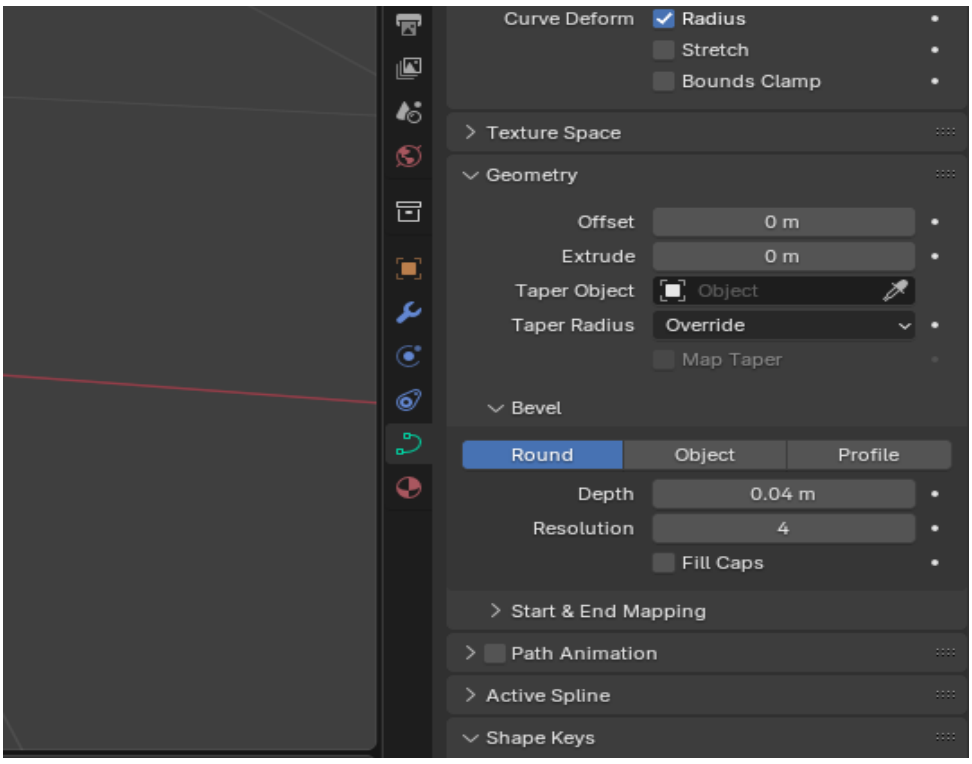
Εικόνα 3.24 Μεταρροπή σε Καμπύλη.

Μέσα από το panel ιδιοτήτων της καμπύλης, ρυθμίζεται η τιμή Bevel Depth (π.χ. 0.02m) ώστε το σύρμα να αποκτήσει κυλινδρικό όγκο.



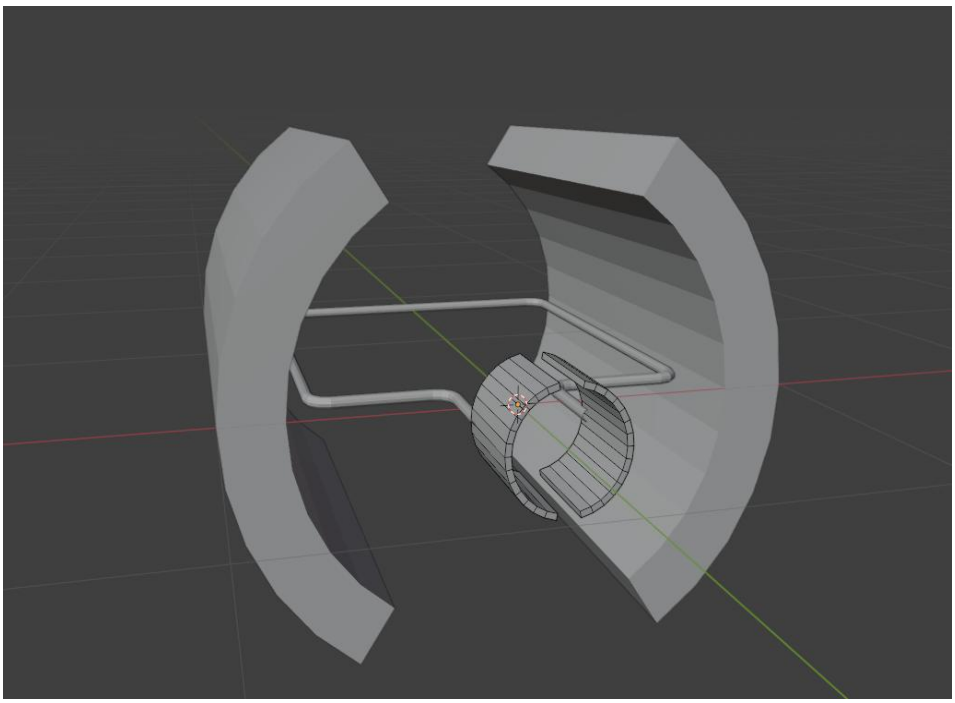
Εικόνα 3.25 Προσθήκη Γεωμετρικού Πάχους

Αυξάνεται η τιμή του βάθους (Depth) σε 0.04m για να επιτευχθεί η επιθυμητή οπτική πυκνότητα της περιέλιξης στο μοντέλο.



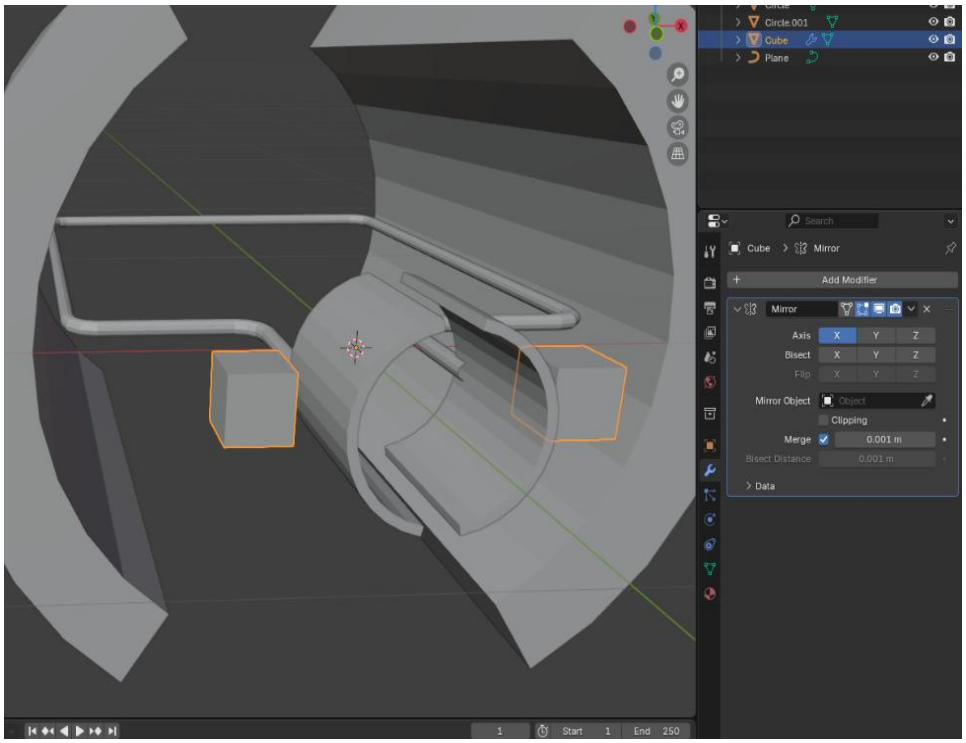
Εικόνα 3.26 Βελτιστοποίηση Διαμέτρου

Προστίθεται ένας κεντρικός κύλινδρος που λειτουργεί ως ο συλλέκτης (commutator) του κινητήρα, ο οποίος δέχεται την ηλεκτρική επαφή.



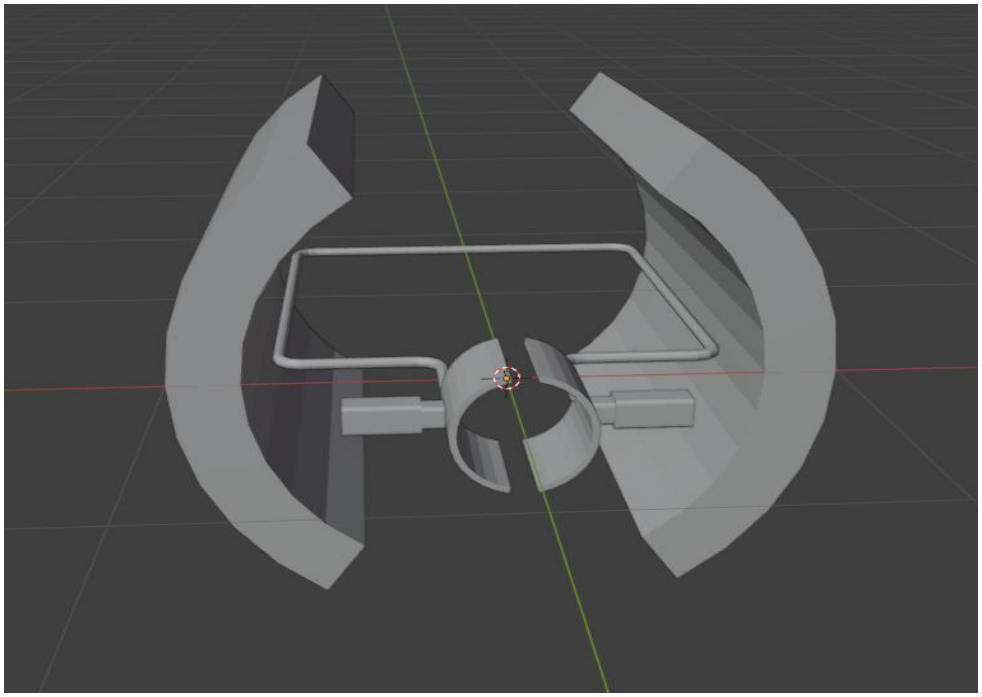
Εικόνα 3.27 Σχεδίαση Συλλέκτη

Εισάγονται δύο κύβοι που θα αποτελέσουν τις ψήκτρες (brushes) του συστήματος, τοποθετημένοι εκατέρωθεν του συλλέκτη. Χρησιμοποιείται ο Mirror Modifier για την ακριβή συμμετρική τοποθέτηση των εξαρτημάτων ως προς τον κεντρικό άξονα περιστροφής.



Εικόνα 3.28 Τοποθέτηση Ψυκτρών.

Το μοντέλο ολοκληρώνεται με τη σύνθεση όλων των τμημάτων (στάτορας, ρότορας, συλλέκτης, ψήκτρες) σε μια ενιαία γεωμετρική οντότητα έτοιμη για εισαγωγή στη Unity.

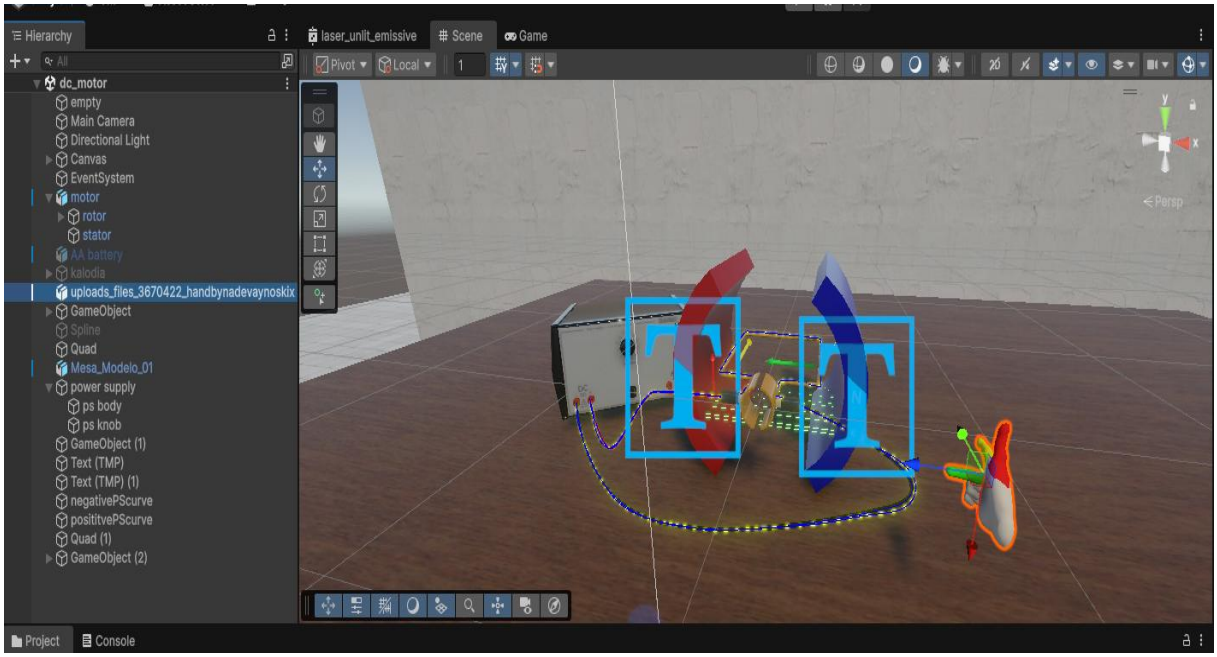


Εικόνα 3.29 Τελική Σύνθεση.

3.3.2 Δομή και Οργάνωση του Διαιρέτη Τάσης στο Unity.

Όπως αναλύθηκε λεπτομερώς στο προηγούμενο κεφάλαιο σχετικά με τον διαιρέτη τάσης, η Ιεραρχία της Unity έχει οργανωθεί ώστε οι μαθηματικοί κόμβοι να απομονώνονται από τα οπτικά αντικείμενα. Η ίδια λογική των γονικών αντικειμένων εφαρμόζεται και σε αυτή την περίπτωση, εξασφαλίζοντας τη συνοχή του κώδικα και τη διευκόλυνση στη διαχείριση των ηλεκτρικών πληροφοριών

Χαρακτηριστικά Πειράματος DC Motor και ιεραρχεία



Εικόνα 3.30 DC motor στο Unity

Οργάνωση και Δομή

Δομημένο Μοντέλο Κινητήρα:

- Κύριο GameObject: dc_motor
- Διχοτόμηση σε δύο βασικά μέρη:

  Rotor (Ρότορας): Περιστρεφόμενο μέρος

  Stator (Στάτορας): Σταθερό μέρος.

Οπτικά Στοιχεία και Διαχείριση

- Μαγνητικό Πεδίο - Οπτικοποίηση:

  Χρωματική κωδικοποίηση πόλων: Κόκκινο χρώμα: Βόρειος Πόλος (N - North) - Μπλε χρώμα: Νότιος Πόλος (S - South).

  Πράσινες φωσφορίζον γραμμές ανάμεσα απο τους πόλους απεικονίζουν το μαγνητικό πεδίο.

- Βελάκια που δείχνουν το μαγνητικό πέδιο (πράσινο), το ηλεκτρικό (κίτρινο) και τη δύναμη (κόκκινο).
- Κείμενα αναγνώρισης (Text Mesh Pro): Εικονίδιο "N" για Βόρειο Πόλο - Εικονίδιο "S" για Νότιο Πόλο.

Συστήματα Συνδεσιμότητας

- Δυναμική Καλωδίωση:

  Χρήση Spline καμπυλών για ρεαλιστική σύνδεση.

  Διαφοροποίηση χρωμάτων καλωδίων: Κόκκινο καλώδιο: Θετικό καλώδιο (+) - Μαύρο: Αρνητικό καλώδιο (-).

- Οπτική παρακολούθηση ροής ρεύματος σε πραγματικό χρόνο κίτρινη διακεκομένη γραμμή πανω απο τα καλώδια.

Σύστημα Ελέγχου και Αλληλεπίδρασης

- Πηγή Τροφοδοσίας (Power Supply):

  Κεντρική πηγή τάσης για όλο το σύστημα.

  Δυναμικός έλεγχος μέσω περιστροφικού μηχανισμού (ps knob).

- Διαδραστικότητας Χρήστη:

  Μοντέλο χεριού για φυσική αλληλεπίδραση .

  Transform Gizmos για εύκολη μετακίνηση και περιστροφή.

- Δυνατότητες χειροκίνητης προσαρμογής: Αυξομειώση τιμής ρεύματος.

Λειτουργικά Χαρακτηριστικά

- Προσομοίωση Φυσικής Συμπεριφοράς:

  Ρεαλιστική απόκριση σε αλλαγές τάσης.

  Οπτική ανατροφοδότηση περιστροφής.

  Προσομοίωση μαγνητικών δυνάμεων και ροπής.

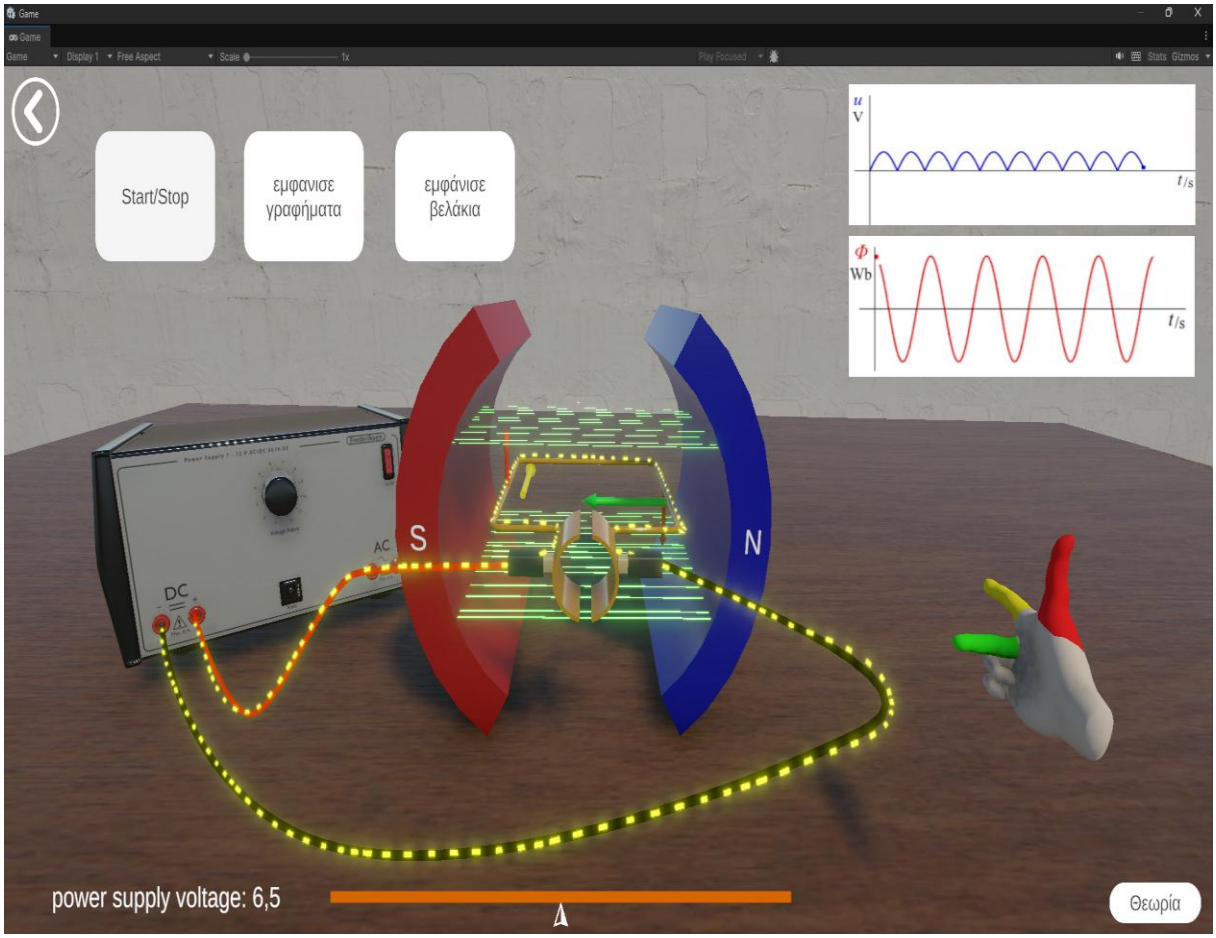
- Εκπαιδευτικά Στοιχεία:

  Οπτική αναπαράσταση μαγνητικού πεδίου.

  Διάγραμμα ροής ενέργειας.

  Πραγματικού χρόνου μετρήσεις και ανατροφοδότηση.

  Ανάγνωση θεωρίας.



Εικόνα 3.31 Το game πειβαλλον του DC motor.

3.3.3 Σκοπός του Script (motor) C#.

1. Περιστρέφεται ανάλογα με την εφαρμοζόμενη τάση.
2. Εμφανίζει οπτικά στοιχεία (βέλη, χέρι, γραμμές) για εκπαιδευτικούς σκοπούς.
3. Προσομοιώνει τη λειτουργία του commutator (αντιστροφή κατεύθυνσης ρεύματος).
4. Συγχρονίζει βίντεο με την ταχύτητα περιστροφής.

Συνάρτηση Update()

```
// Update is called once per frame
Unity Message | 0 references
void Update()
{
    if (isRotating)
    {
        float deltaRot = vSource.voltage* rotationSpeed * Time.deltaTime;

        // Rotate the rotor
        rotor.transform.Rotate(Vector3.forward * deltaRot);

        // Track rotation
        accumulatedRotation += Mathf.Abs(deltaRot);

        // Check for 180° turn
        if (accumulatedRotation >= 180f)
        {
            accumulatedRotation -= 180f;   // reset for next half turn
            changeArrowOrientation();      // call your method
        }
    }
```

Αυτό το κομμάτι είναι ο καρδιά της προσομοίωσης. Ο κινητήρας περιστρέφεται με ρυθμό ανάλογο της τάσης. Κάθε φορά που ολοκληρώνει 180 μοίρες (μισή περιστροφή), αλλάζει η κατεύθυνση των βελών, προσομοιώνοντας την εναλλαγή πολικότητας στον κινητήρα DC.

Ενημέρωση Θέσεων και Κίνησης και Χρονικός Έλεγχος Ταχύτητας Βίντεο

```
    arrow1.transform.position = arrow1pos.transform.position;
    arrow2.transform.position = arrow2pos.transform.position;
    animateArrows();


    timer += Time.deltaTime;

    if (timer >= interval)
    {
        timer = 0f;
        player1.playbackSpeed = vSource.voltage * conversionFactor;
        player2.playbackSpeed = vSource.voltage * conversionFactor;
    }
}
```

Τα βέλη τοποθετούνται σε προκαθορισμένες θέσεις και κινούνται με ταλαντευτικό τρόπο (animateArrows()) για να δείξουν τη ροή του ρεύματος. Κάθε 0.2 δευτερόλεπτα ενημερώνει την ταχύτητα αναπαραγωγής των βίντεο των γραφιμμάτων να είναι ανάλογη της τάσης. Όσο μεγαλύτερη τάση, τόσο πιο γρήγορα παίζουν τα βίντεο.

Συνάρτηση rotate()

```
0 references
public void rotate() {
    if (isRotating)
    {
        dottedLine.SetActive(true);
        rotor.transform.rotation = Quaternion.identity;

        isRotating = false;
    }
    else
    {
        dottedLine.SetActive(false);

        isRotating = true;
    }
}
```

Αυτή η συνάρτηση λειτουργεί ως ON/OFF διακόπτης. Όταν ο κινητήρας λειτουργεί και πατηθεί, τον σταματάει και επαναφέρει τον ρότορα στην αρχική του θέση. Όταν είναι σβηστός και πατηθεί, τον ξεκινάει. Η διακεκομμένη γραμμή εμφανίζεται μόνο όταν ο κινητήρας είναι απενεργοποιημένος.

Συνρτήσεις showLines() και showHand()

```
0 references
public void showLines() {
    if (grafima1.gameObject.activeInHierarchy)
    {
        grafima1.gameObject.SetActive(false);
        grafima2.gameObject.SetActive(false);
    }
    else
    {
        grafima1.gameObject.SetActive(true);
        grafima2.gameObject.SetActive(true);
    }
}
0 references
public void showHand()
{
    if (hand.gameObject.activeInHierarchy)
    {
        hand.gameObject.SetActive(false);
        arrows.gameObject.SetActive(false);
    }
    else
    {
        hand.gameObject.SetActive(true);
        arrows.gameObject.SetActive(true);
    }
}
```

Αυτές οι συναρτήσεις είναι ένας απλός εναλλαγέας ορατότητας. Ελέγχουν αν τα διαγράμματα, το μοντέλο χεριού και τα βοηθητικά βελάκια είναι ορατά.

Συναρτήσεις changeArrowOrientation() και animateArrows()

```
1 reference
public void changeArrowOrientation() {
    arrow1.transform.Rotate(Vector3.forward * 180);
    arrow2.transform.Rotate(Vector3.forward * 180);
}
1 reference
public void animateArrows()
{
    timeCounter += Time.deltaTime * moveSpeed;

    float offset = Mathf.Sin(timeCounter) * moveDistance;

    // Yellow (forward/back = Z axis)
    if (arrowYellow != null)
    {
        arrowYellow.transform.localPosition =
            yellowStartPos + new Vector3(0f, 0f, offset);
    }

    // Green (left/right = X axis)
    if (arrowGreen != null)
    {
        arrowGreen.transform.localPosition =
            greenStartPos + new Vector3(offset, 0f, 0f);
    }
}
}
```

Η changeArrowOrientation() περιστρέφει τα βέλη κατά 180 μοίρες. Χρησιμοποιείται όταν ο κινητήρας ολοκληρώνει μισή περιστροφή, για να δείξει την αλλαγή πολικότητας και κατεύθυνσης ρεύματος. Ενώ η animateArrows() δημιουργεί ημιτονοειδή κίνηση για τα βέλη.

- Το κίτρινο βέλος κινείται στον άξονα Z (μπροστά-πίσω)
- Το πράσινο βέλος κινείται στον άξονα X (αριστερά-δεξιά)
- Η ταχύτητα κίνησης ελέγχεται από το moveSpeed
- Η μέγιστη μετατόπιση ελέγχεται από το moveDistance

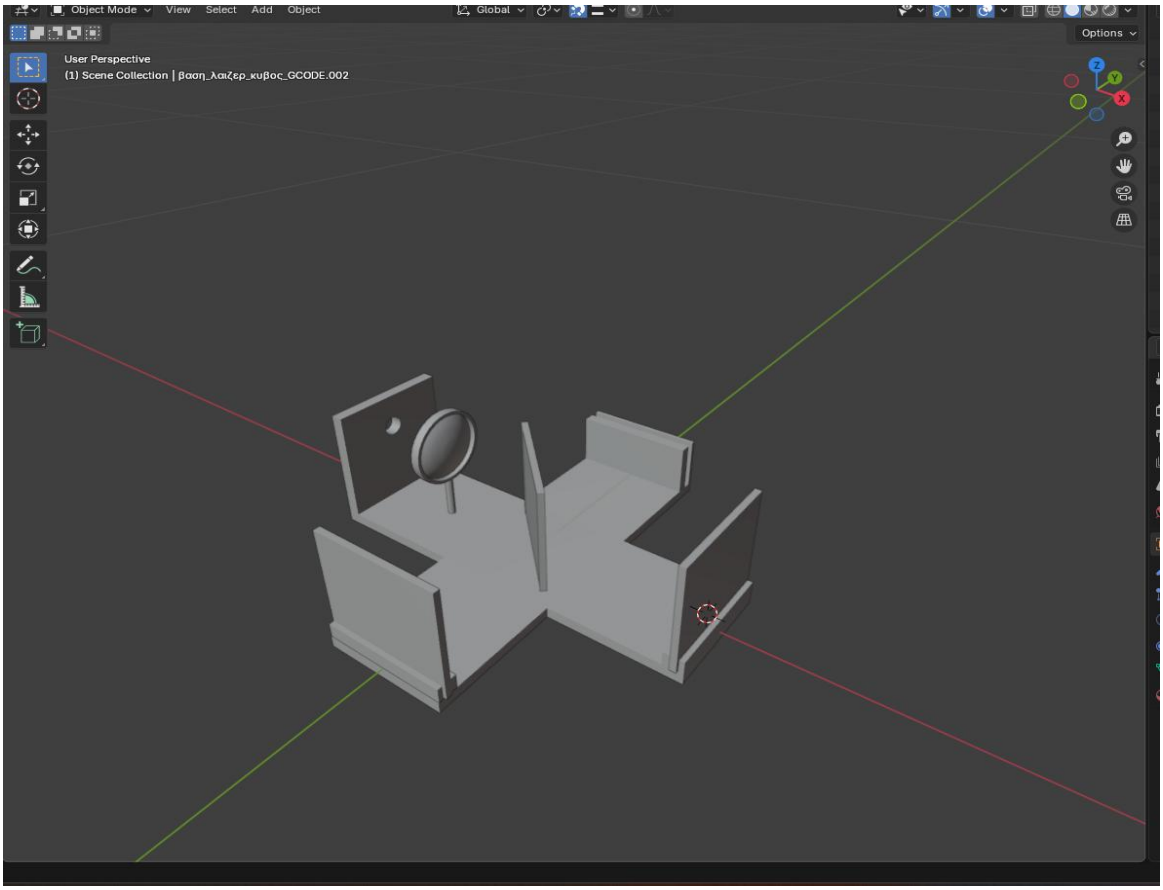
Το εικονικό εργαστήριο ενός DC κινητήρα επιτρέπει στους φοιτητές να παρατηρήσουν άμεσα, πώς η αλλαγή της τάσης επηρεάζει την ταχύτητα του κινητήρα, να δοκιμάσουν διάφορες ρυθμίσεις χωρίς κίνδυνο, να κατανοήσουν βασικές αρχές των κινητήρων και να συνδυάσουν τη θεωρία με την πρακτική μέσα από μια διαδραστική και ασφαλή προσομοίωση.

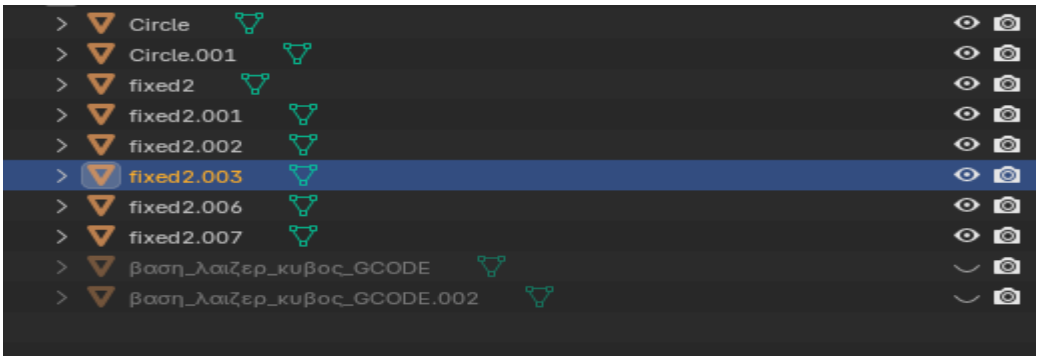
3.4 Συμβολόμετρο

Το φαινόμενο της συμβολής, το οποίο εξετάζεται μέσω του συμβολόμετρου (interferometer), είναι ένα θεμελιώδες φαινόμενο της κυματικής οπτικής με εκτενή χρήση σε σύγχρονες τεχνολογίες. Μέσα στο περιβάλλον του εικονικού εργαστηρίου, δημιουργήθηκε στο Unity ένα διαδραστικό μοντέλο interferometer που προσφέρει στους φοιτητές τη δυνατότητα να πειραματιστούν με το φαινόμενο της συμβολής, να μετακινήσουν οπτικά στοιχεία και να παρακολουθήσουν σε πραγματικό χρόνο τις μεταβολές. Αυτή η εκτέλεση, όπως και τα άλλα πειράματα του εργαστηρίου, συνδυάζει την προσομοίωση με διαισθητική οπτική αναπαράσταση, προσφέροντας μια ασφαλή και αποδοτική εκπαιδευτική εμπειρία.

3.4.1 Σχεδίαση και Τρισδιάστατη Μοντελοποίηση στο Blender

Η βάση του interferometer δόθηκε από τον καθηγητή σε μορφή G-code, η οποία είχε αρχικά σχεδιαστεί για την εκτέλεση άλλου πειράματος και προοριζόταν για εκτύπωση μέσω 3D εκτυπωτή τύπου Pronter. Για να χρησιμοποιηθεί το συγκεκριμένο γεωμετρικό μοντέλο στην παρούσα εργασία, το αρχείο G-code μετατράπηκε μέσω διαδικτυακού εργαλείου σε τρισδιάστατη μορφή .fbx. Το αρχείο στη συνέχεια εισήχθη στο Blender, όπου έγινε επιπλέον επεξεργασία του μοντέλου, με την προσθήκη οπτικών στοιχείων, όπως ο φακός (Cirlce), καθρέφτες και ο διαχωριστής δέσμης (beam splitter). Με αυτόν τον τρόπο, το αρχικό μοντέλο προσαρμόστηκε στις ανάγκες του συγκεκριμένου πειράματος, διατηρώντας ταυτόχρονα τη γεωμετρική ακρίβεια της αρχικής κατασκευής.





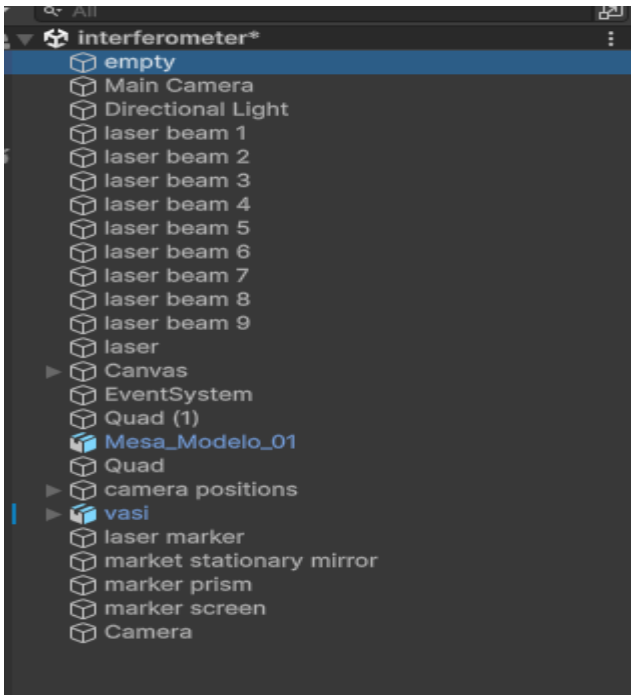
Εικόνα 3.32 Το μοντέλο του interferometer στο blender.

3.4.2 Δομή και Οργάνωση του Συμβοόμετρου στο Unity

Το φαινόμενο της συμβολής, το οποίο εξετάζεται μέσω του συμβολόμετρου (interferometer), είναι ένα θεμελιώδες φαινόμενο της κυματικής οπτικής με εκτενή χρήση σε σύγχρονες τεχνολογίες. Μέσα στο περιβάλλον του εικονικού εργαστηρίου, δημιουργήθηκε στο Unity ένα διαδραστικό μοντέλο interferometer που προσφέρει στους φοιτητές τη δυνατότητα να πειραματιστούν με το φαινόμενο της συμβολής, να μετακινήσουν οπτικά στοιχεία και να παρακολουθήσουν σε πραγματικό χρόνο τις μεταβολές. Αυτή η εκτέλεση, όπως και τα άλλα πειράματα του εργαστηρίου, συνδυάζει την προσομοίωση με διαισθητική οπτική αναπαράσταση, προσφέροντας μια ασφαλή και αποδοτική εκπαιδευτική εμπειρία.

Ιεραρχία

Η οργάνωση της δομής του πειράματος είναι ιεραρχική γύρω από ένα κεντρικό GameObject (interferometer), το οποίο λειτουργεί σαν εννοιολογικό δοχείο για όλα τα επιμέρους στοιχεία

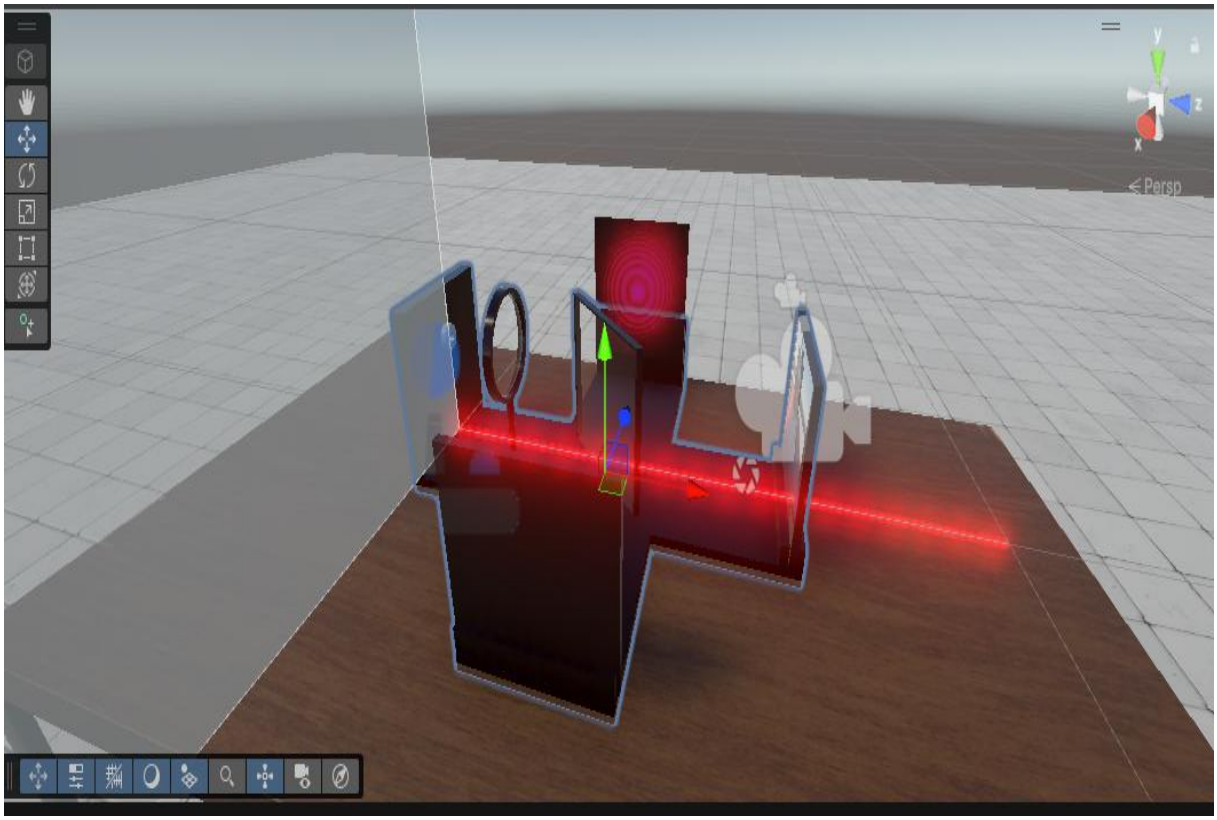


Εικόνα 3.33 Ιεραρχία του Indterferometer στο Unity.

Αναλυτίκα περιλαμβάνονται διακριτές κατηγορίες αντικειμένων:

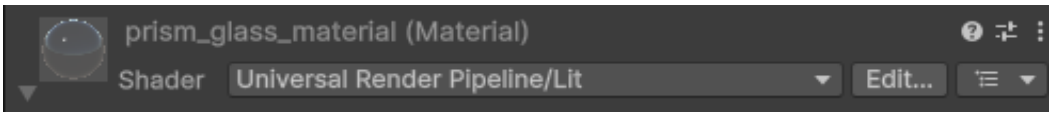
- Πηγές φωτός (laser beams): Η στρατηγική σχεδίασης που περιλαμβάνει την τμηματοποίηση της δέσμης σε διάφορα αυτόνομα αντικείμενα (laser beams 1-9) διευκολύνει την ακριβή απεικόνιση της πορείας του φωτός, επιτρέποντας την ξεχωριστή διαχείριση των ανακλάσεων και διαθλάσεων σε κάθε οπτικό στοιχείο της διάταξης.
- Οπτικά στοιχεία: Η διάταξη του συμβολόμετρου Michelson αποτελείται από μια σειρά αυτόνομων τρισδιάστατων στοιχείων που περιλαμβάνουν τον κεντρικό διαχωριστή δέσμης (Beam Splitter), τον σταθερό και τον κινούμενο καθρέφτη για την απομόνωση των δύο οπτικών βραχιόνων, καθώς και μία οθόνη παρατήρησης με ειδικό υλικό (material) για την τελική απεικόνιση των κροσσών συμβολής.
- Δείκτες/markers: Αποτελούν τα σημεία στα οποία θα ξεκινήσουν η θα χτυπήσουν τα laser.
- Κάμερες και στοιχεία διεπαφής: Η ενσωμάτωση πολλαπλών θέσεων κάμερας ακολουθώντας την πορεία των laser διευκολύνει στην πλήρη κατανοήση του πειράματος καθώς προσομοιάζουν τη φυσική κίνηση του παρατηρητή γύρω από τη διάταξη.

Στη διαμόρφωση της βάσης του interferometer, κάθε ξεχωριστό στοιχείο συνοδεύεται από το αντίστοιχο υλικό, το οποίο επιλέχθηκε με τρόπο που αποτυπώνει λειτουργικά και εννοιολογικά τον ρόλο του στην πειραματική διαδικασία.



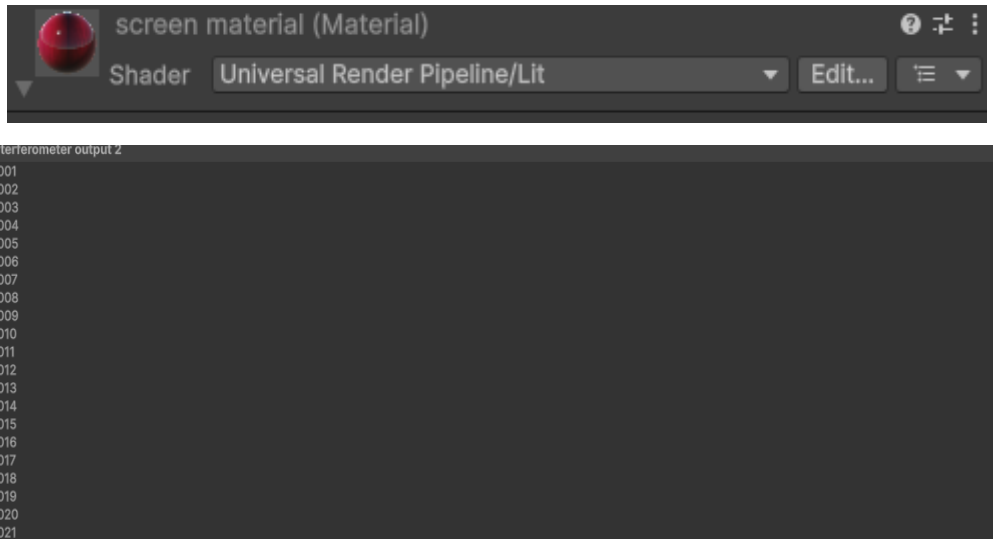
Εικόνα 3.34 Σύνθεση του συμβολόμετρου στο Unity.

- Συγκεκριμένα, το πρίσμα και ο φακός διαθέτουν ειδικό υλικό τύπου πρίσματος, το οποίο χρησιμοποιείται για να δείξει οπτικά τη διαφάνεια και τη διάθλαση του στοιχείου.



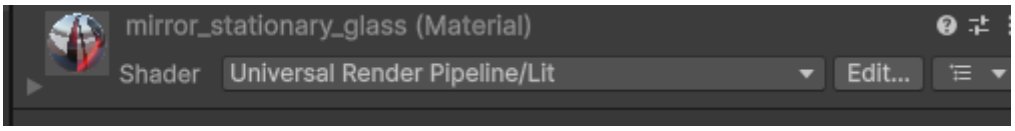
Εικόνα 3.35 Glass Material για το πρίσμα και τον φακό.

- Η οθόνη παρατήρησης περιέχει υλικό που στηρίζεται σε καρέ βίντεο, μέσω των οποίων προβάλλεται το μοτίβο συμβολής[34].



Εικόνα 3.36 Screen material και τα καρέ από του βίντεο που το αποτελούν.

- Ο σταθερός και ο κινούμενος καθρέφτης διαθέτουν αντίστοιχα material τύπου mirror, το οποίο επιλέχθηκε για να αποτυπώνει την ανακλαστική ικανότητα των επιφανειών.



Εικόνα 3.37 Mirror material για τους καθρέφτες.

3.4.3 Ανάπτυξη κώδικα και λειτουργία συστήματος (C#).

Σκοπός του κώδκα είναι η ψηφιακή προσομοίωση ενός συμβολομέτρου Michelson, συνδυάζοντας τον δυναμικό έλεγχο της κίνησης του κατόπτρου και των οκτώ διαδρομών του laser με τη διαχείριση διαφορετικών οπτικών γωνιών της κάμερας, έτσι ώστε να απεικονίζονται με ακρίβεια τα παραγόμενα σχέδια συμβολής (κρόσοι) στην οθόνη παρατήρησης.

Μέθοδοι Update(), sliderUpdated() και updateImage():

```
}
Unity Message | 0 references
private void Update()
{
    if (animationFinished)
    {
        beamSecondary_toMovingMirror.pointB = movingMirrorMarker.position;
        beamTertiary_fromMovingMirror.pointA = movingMirrorMarker.position+ new Vector3(0,0, beamOffsetAmount);
       // beamTertiary_fromMovingMirror.pointA = movingMirrorMarker.position + new Vector3(0, 0, beamOffsetAmount);
    }
}
1 reference
public void sliderUpdated()
{
    if (mirrorTransform == null || mirrorSlider == null)
        return;

    // Calculate new x-position based on slider value (0 → initial, 1 → max offset)
    float newX = initialMirrorPosition.x - (mirrorSlider.value * maxOffset);

    // Apply the new position while keeping y and z the same
    mirrorTransform.position = new Vector3(newX, initialMirrorPosition.y, initialMirrorPosition.z);
    UpdateEmissionIntensity();
    updateImage();
}

2 references
void updateImage()
{
    if (!animationFinished)
        return;


    int n = frames.Length;
    int index = Mathf.FloorToInt(mirrorSlider.value * (n - 1));
    index = Mathf.Clamp(index, 0, n - 1);
    screenMat.SetColor("_BaseColor", Color.white);

    screenMat.SetTexture("_BaseMap", frames[index]);
}
```

Η μέθοδος updateImage() εμφανίζει στην οθόνη ένα από τα προκατασκευασμένα σχέδια παρεμβολής, ανάλογα με τη θέση που έχει ορίσει ο χρήστης στο slider.

Διαδικασία Εκτέλεσης Animation:

```
0 references
public void animate()
{
    //  screenMat.SetTexture("_BaseMap", Color.black);
    screenMat.SetColor("_BaseColor", Color.black);

    endGraphic.SetActive(false);
    animationFinished = false;
    resetBeamPoints();
    cameraPositionCounter = 0;
    StartCoroutine(animateInterferometer());
}
1 reference
public IEnumerator animateInterferometer()
{
    // --- Stage 1: Primary beam from laser to prism ---
    beamPrimary.pointA = laserMarker.position;
    beamPrimary.pointB = laserMarker.position;
    transition(cameraSpeed);

    float t1 = 0f;
    while (t1 < 1f)
    {
        t1 += Time.deltaTime * primarySpeed;
        beamPrimary.pointB = Vector3.Lerp(laserMarker.position, prismMarker.position, t1);
        yield return null;
    }

    // --- Stage 2: Secondary beams from prism to mirrors ---
    beamSecondary_toMovingMirror.pointA = prismMarker.position;
    beamSecondary_toStationaryMirror.pointA = prismMarker.position;

    beamSecondary_toMovingMirror.pointB = prismMarker.position;
    beamSecondary_toStationaryMirror.pointB = prismMarker.position;
    transition(cameraSpeed);


    float t2 = 0f;
    while (t2 < 1f)
    {
        t2 += Time.deltaTime * secondarySpeed;
        beamSecondary_toMovingMirror.pointB = Vector3.Lerp(prismMarker.position, movingMirrorMarker.position, t2);
        beamSecondary_toStationaryMirror.pointB = Vector3.Lerp(prismMarker.position, stationaryMirrorMarker.position, t2);

        yield return null;
    }
```

Αυτές οι τεχνικές ενεργοποιούν και πραγματοποιούν ένα βηματικό animation της πορείας του φωτός στο παρεμβολόμετρο, μετατοπίζοντας τις φωτεινές δέσμες και την κάμερα σε συγκεκριμένα σημεία.

Μέθοδος resetBeamPoints():

```
1 reference
private void resetBeamPoints()
{
    if (beamPrimary != null)
    {
        beamPrimary.pointA = Vector3.zero;
        beamPrimary.pointB = Vector3.zero;
    }

    if (beamSecondary_toMovingMirror != null)
    {
        beamSecondary_toMovingMirror.pointA = Vector3.zero;
        beamSecondary_toMovingMirror.pointB = Vector3.zero;
    }

    if (beamSecondary_toStationaryMirror != null)
    {
        beamSecondary_toStationaryMirror.pointA = Vector3.zero;
        beamSecondary_toStationaryMirror.pointB = Vector3.zero;
    }

    if (beamTertiary_fromMovingMirror != null)
    {
        beamTertiary_fromMovingMirror.pointA = Vector3.zero;
        beamTertiary_fromMovingMirror.pointB = Vector3.zero;
    }

    if (beamTertiary_fromStationaryMirror != null)
    {
        beamTertiary_fromStationaryMirror.pointA = Vector3.zero;
        beamTertiary_fromStationaryMirror.pointB = Vector3.zero;
    }
    if (beamQuaternary != null)
    {
        beamQuaternary.gameObject.SetActive(true);
        beamQuaternary.pointA = Vector3.zero;
        beamQuaternary.pointB = Vector3.zero;
    }
    if (beam_5 != null)
    {
        beam_5.pointA = Vector3.zero;
        beam_5.pointB = Vector3.zero;
    }
    if (beam_6_toLaser != null)
    {
        beam_6_toLaser.pointA = Vector3.zero;
        beam_6_toLaser.pointB = beam_6_toLaser.pointA;
    }
    if (beam_6_toScreen != null)
    {
        beam_6_toScreen.gameObject.SetActive(true);
        beam_6_toScreen.pointA = Vector3.zero;
        beam_6_toScreen.pointB = beam_6_toScreen.pointA;
    }
```

Η μέθοδος resetBeamPoints() μηδενίζει και επαναφέρει όλες τις φωτεινές δέσμες στην αρχή τους, προετοιμάζοντας το σύστημα για την έναρξη νέου animation.

Camera transition:

```
359     public void transition( float cameraSpeed_)
360     {
361
362         // Ensure the array is valid and has at least two transforms
363         if (cameraTransforms == null || cameraTransforms.Length < 2)
364         {
365             Debug.LogWarning("Camera transition failed: cameraTransforms is null or has fewer than 2 elements.");
366             return;
367         }
368
369         // Ensure the current index is within bounds
370         if (cameraPositionCounter < 0 || cameraPositionCounter + 1 >= cameraTransforms.Length)
371         {
372             Debug.LogWarning($"Camera transition failed: cameraPositionCounter ({cameraPositionCounter}) is out of range.");
373             return;
374         }
375
376         // Get the start and end transforms
377         Transform from = cameraTransforms[cameraPositionCounter];
378         Transform to = cameraTransforms[cameraPositionCounter+1];
379
380         // Ensure both transforms are valid
381         if (from == null || to == null)
382         {
383             Debug.LogWarning("Camera transition failed: one or both target transforms are null.");
384             return;
385         }
386
387         // Start the transition coroutine
388         cameraPositionCounter++;
389         StartCoroutine(TransitionBetweenTransforms(from, to, cameraSpeed_));
390
391     }
```

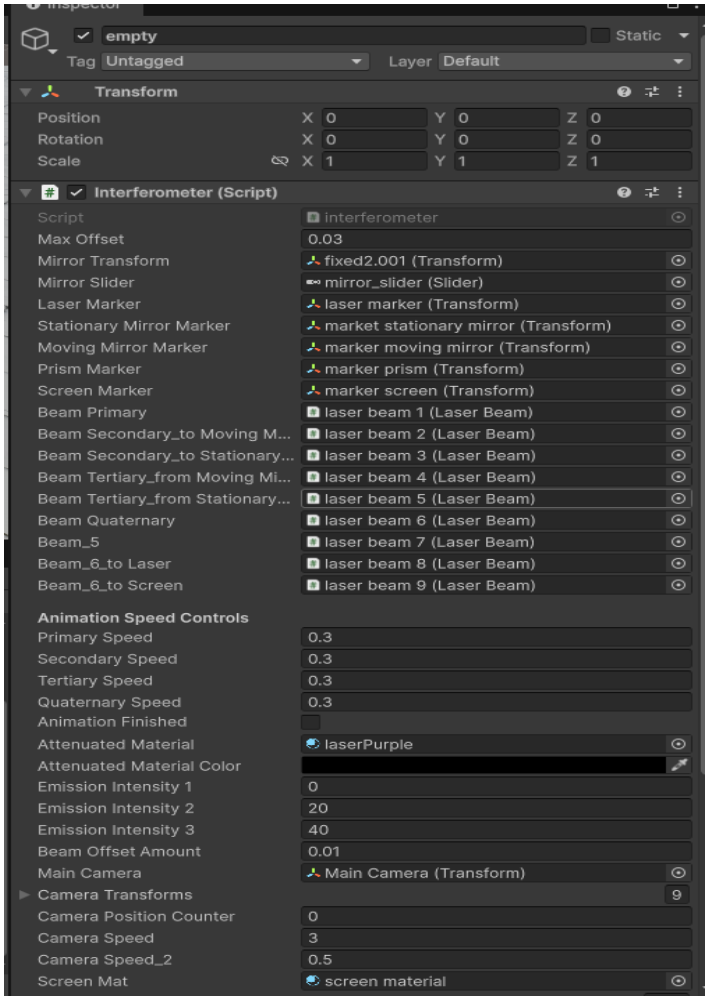
Η μέθοδος transition() μετακινεί ομαλά την κάμερα από την τρέχουσα προκαθορισμένη θέση στην επόμενη θέση της λίστας, ελέγχοντας με debug πρώτα την ορθότητα των δεδομένων για να αποφύγει σφάλματα.

Μέθοδος TransitionBetweenTransforms():

```
390
391     }
392
393     1 reference
        public IEnumerator TransitionBetweenTransforms(Transform transformA, Transform transformB, float time)
394     {
395         // Get reference to the main camera's transform
396         Transform cameraTransform = Camera.main?.transform;
397         if (cameraTransform == null)
398         {
399             Debug.LogWarning("Transition failed: Main Camera not found.");
400             yield break;
401         }
402
403         // Immediately snap to transformA's world position and rotation
404         cameraTransform.position = transformA.position;
405         cameraTransform.rotation = transformA.rotation;
406
407         // Capture world-space start and end points
408         Vector3 startPosition = transformA.position;
409         Quaternion startRotation = transformA.rotation;
410         Vector3 endPosition = transformB.position;
411         Quaternion endRotation = transformB.rotation;
412
413         float elapsed = 0f;
414
415         // Smoothly move from A to B in world space
416         while (elapsed < time)
417         {
418             elapsed += Time.deltaTime;
419             float t = Mathf.Clamp01(elapsed / time);
420
421             // Interpolate in world space
422             cameraTransform.position = Vector3.Lerp(startPosition, endPosition, t);
423             cameraTransform.rotation = Quaternion.Slerp(startRotation, endRotation, t);
424
425             yield return null;
426         }
427
428         // Snap exactly to the end transform
429         cameraTransform.position = endPosition;
430         cameraTransform.rotation = endRotation;
431     }
432
433
434
435 }
436
437
```

Η μέθοδος TransitionBetweenTransforms() μεταφέρει ομαλά την κύρια κάμερα από μία θέση και προσανατολισμό (transformA) σε μία νέα (transformB) σε προκαθορισμένο χρόνο, χρησιμοποιώντας γραμμικές παρεμβολές για τη θέση και την περιστροφή.

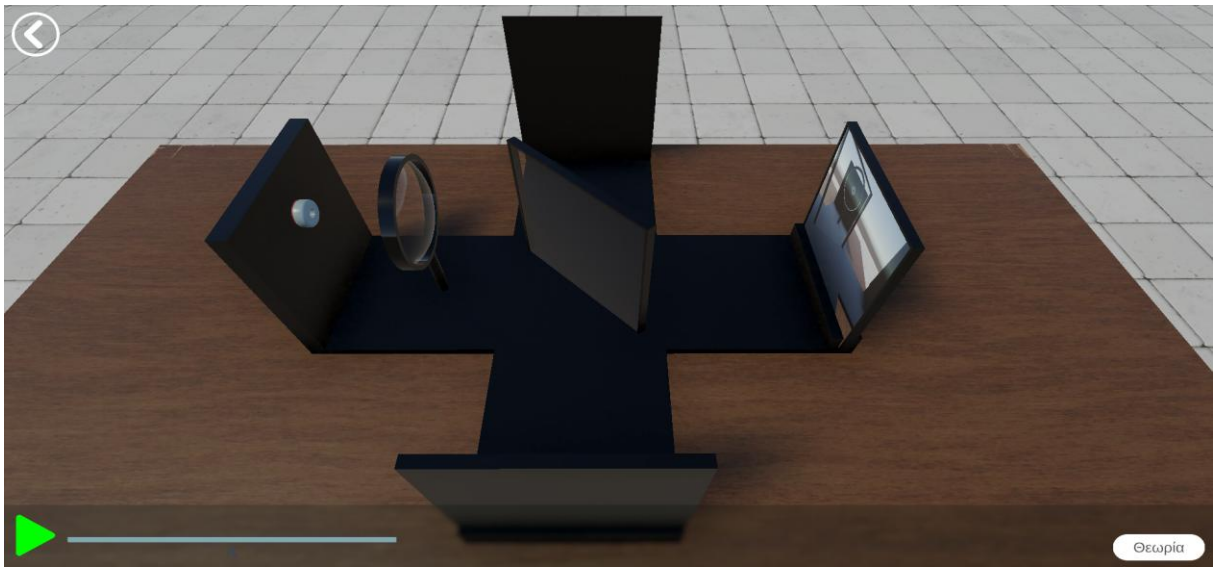
Το τελικό αποτέλεσμα του script «Interferometer» εμφανίζεται ως ένα πλήρως παραμετροποιημένο σύστημα ελέγχου, όπου είναι εμφανής η σύνδεση όλων των οπτικών στοιχείων (beams 1-9), των δεικτών θέσης (markers) και των εννέα θέσεων της κάμερας, διευκολύνοντας την ομαλή αλληλεπίδραση του χρήστη με το πείραμα.



Εικόνα 3.38 Inspector του συμβολόμετρου.

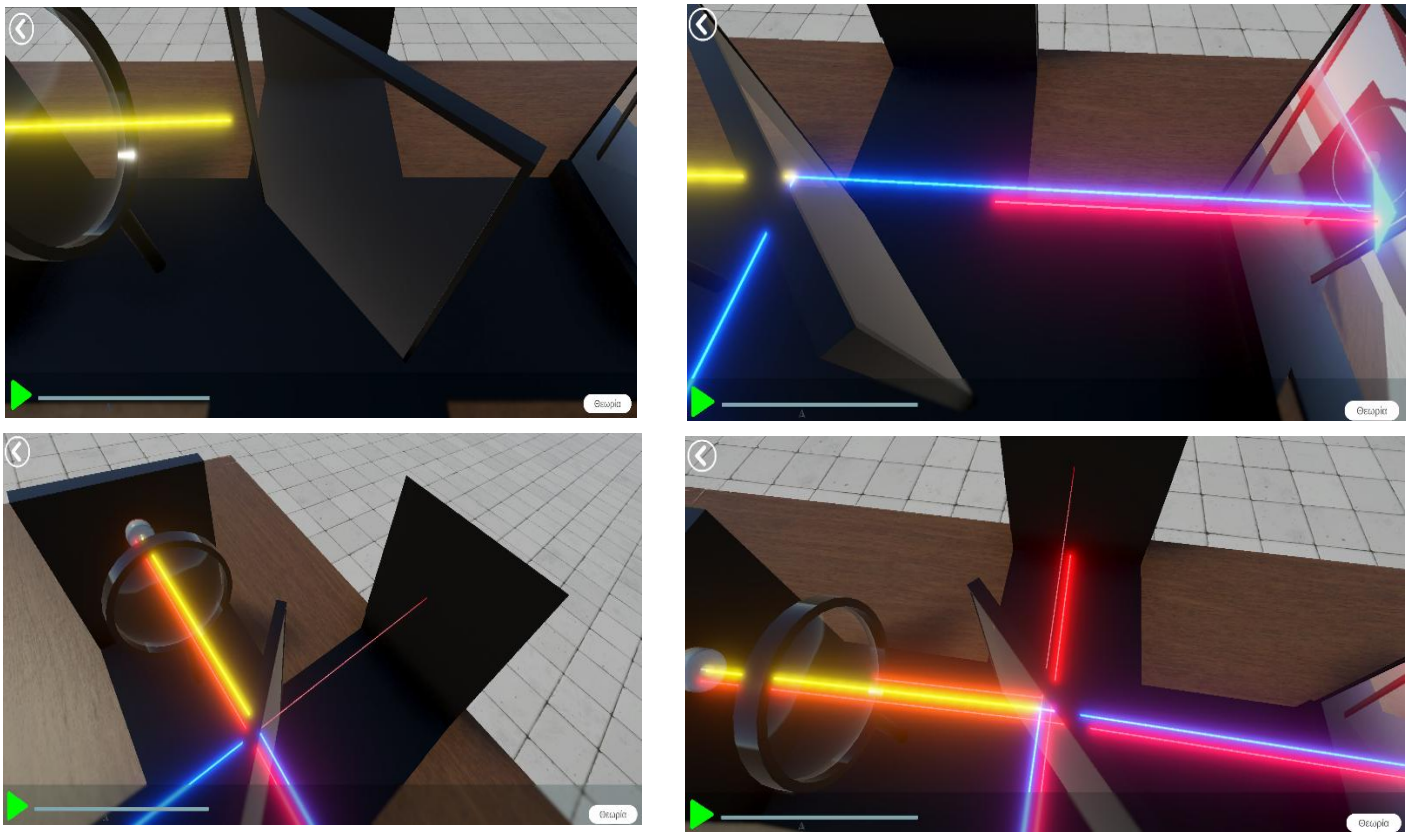
3.4.4 Λειτουργία Πειράματος στην πράξη

Βήμα 1: Ανοίγοντας την εφαρμογή ο χρήστης παρατηρεί την εργαστηριακή διάταξη ολοκληρωμένη πάνω στον πάγκο, με όλα τα οπτικά στοιχεία στη σωστή τους θέση.



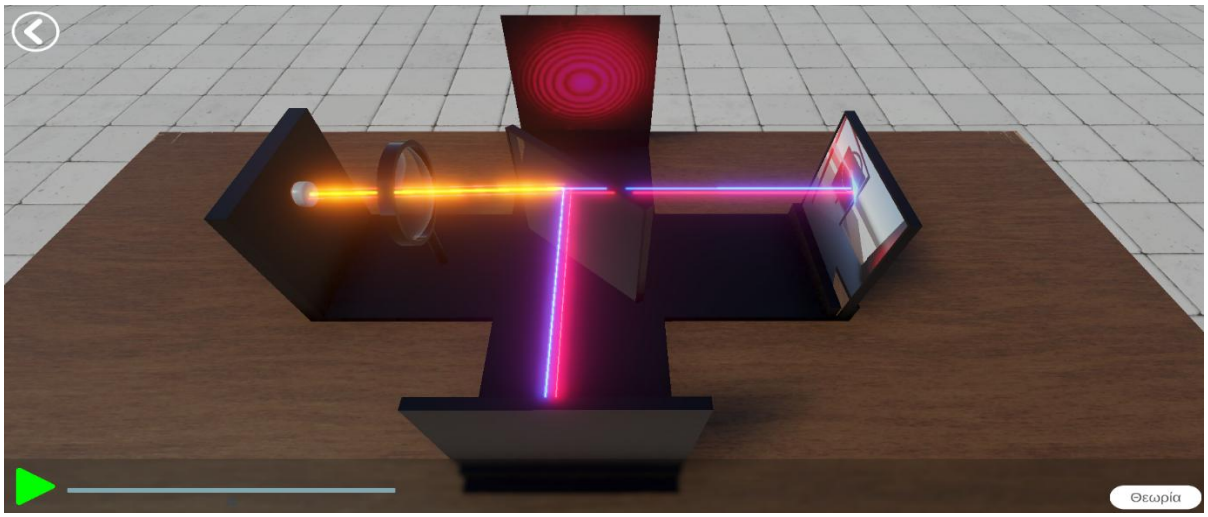
Εικόνα 3.39 Game του Interferometer.

Βήμα 2: Μόλις αρχίσει το πείραμα, η δέσμη του laser εκκινεί από την πηγή, διέρχεται από τον φακό και διαχωρίζεται σε δύο μέρη. Οι δύο καινούριες δέσμες κινούνται σε διαφορετικές κατευθύνσεις, προσκρούουν στους καθρέφτες και επιστρέφουν για να συναντηθούν πάλι, σχηματίζοντας μια φωτεινή διαδρομή καταλήγοντας στην οθόνη.

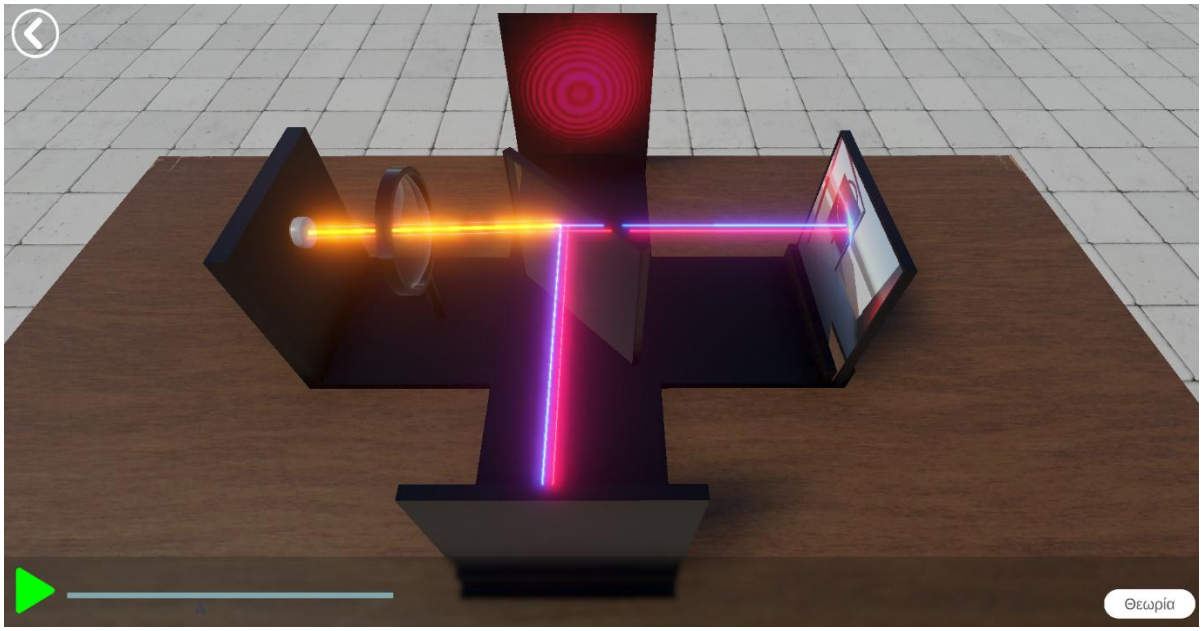


Εικόνα 3.40 Οι διαδρομές του laser .

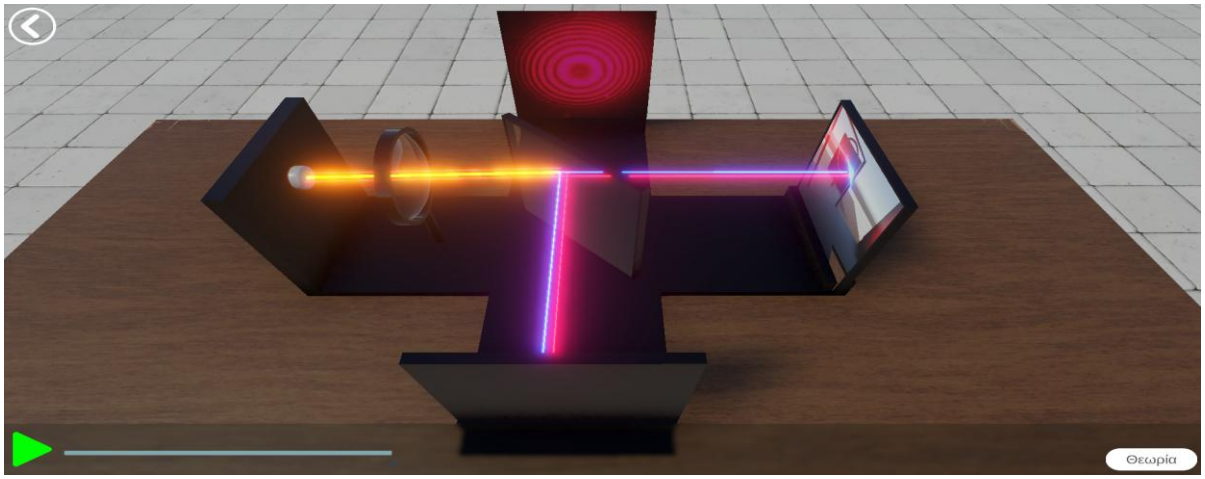
Βήμα 3: Αφού οι δύο δέσμες φτάσουν στην οθόνη, ο χρήστης αλλάζει θέση στο slider που μετακινεί λίγο τον έναν καθρέφτη. Αυτή η απλή κίνηση μεταβάλλει τον τρόπο σύνδεσης των δύο δέσμεων δείχνοντας το φαινόμενο της συμβολής ως φωτεινές και σκοτεινές γραμμές (κροσσούς), οι οποίες αναμορφώνονται σε πραγματικό χρόνο



Εικόνα 3.41 Το αποτέλεσμα στην οθόνη του συμβολόμετρου.



Εικόνα 3.42 Πρώτη μεταβολή θέσης του κινούμενου καθρέφτη.



Εικόνα 3.43 Μεταβολή του κινούμενου καθρέφτη σε άλλη θέση με το αποτέλεσμα να αλλάζει στην οθόνη.

Κεφάλαιο 4ο: Περιορισμοί, συμπεράσματα και προτάσεις βελτίωσης

4.1 Περιορισμοί

Η κατασκευή του εικονικού εργαστηρίου που περιγράφεται σε αυτή την εργασία έγινε αποκλειστικά στο περιβάλλον της μηχανής γραφικών Unity, με κύριο σκοπό την ανάπτυξη ενός διαδραστικού και εκπαιδευτικού εργαλείου. Ενώ η Unity παρέχει ευρείες επιλογές οπτικοποίησης και αλληλεπίδρασης, η προσομοίωση φυσικών φαινομένων περιορίζεται από την ίδια τη φύση της μηχανής, η οποία δεν είναι εξειδικευμένο επιστημονικό λογισμικό προσομοίωσης. Ως αποτέλεσμα, κάποια φαινόμενα, όπως η διάδοση του φωτός στο interferometer, η λειτούργία του κινητήρα και η ηλεκτρική συμπεριφορά των κυκλωμάτων στον διαιρέτη τάσης, αναπαρίστανται με προσεγγιστικό και όχι απόλυτα φυσικό ακριβή τρόπο, δίνοντας έμφαση κυρίως στην εννοιολογική κατανόηση και όχι στην αναλυτική επιστημονική ανάλυση.Επιπλέον, η δομή του συστήματος στηρίζεται σε αυτοσχέδια scripts, που παρακάμπτουν τη χρήση εξωτερικών βιβλιοθηκών για να ενισχύσουν μια πιο διαχειρίσιμη μαθηματική μοντελοποίηση, ειδικά σε περίπλοκες διατάξεις. Ταυτόχρονα, ο καθορισμένος τρόπος αλληλεπίδρασης (drag-and-drop, προκαθορισμένες ενέργειες) περιορίζει την ελευθερία του πειραματισμού σε σύγκριση με ένα φυσικό περιβάλλον. Τέλος, η απαίτηση για θεμελιώδεις ψηφιακές ικανότητες και γνώση της πλατφόρμας Unity μπορεί να συνιστά εμπόδιο πρόσβασης για ορισμένες ομάδες χρηστών.

4.2 Προοπτικές για το μέλλον

Οι βασικές δυνατότητες του εικονικού εργαστηρίου για την ανάπτυξη και διεύρυνσή του περιλαμβάνουν:

1. Αναβάθμιση της Φυσικής Προσομοίωσης: Μια σημαντική βελτίωση θα αποτελούσε η ενσωμάτωση εξελιγμένων μοντέλων υπολογιστικής φυσικής. Αυτό θα μπορούσε να επιτευχθεί μέσω διασύνδεσης με ειδικευμένες επιστημονικές βιβλιοθήκες ή με την εφαρμογή υβριδικών αρχιτεκτονικών που συνδυάζουν την πλατφόρμα Unity με εργαλεία υψηλών υπολογιστικών δυνατοτήτων (High-Performance Computing). Έτσι, από ένα βασικό εκπαιδευτικό παιχνίδι, η εφαρμογή θα εξελιχθεί σε ένα σοβαρό μέσο για επιστημονική έρευνα.

2. Χρήση VR και AR (Εικονική και Επαυξημένη Πραγματικότητα): Μια εναλλακτική θα ήταν η εφαρμογή να είναι συμβατή με γυαλιά εικονικής πραγματικότητας (VR) ή τεχνολογίες AR. Κατ'αυτόν τον τρόπο, ο χρήστης θα αισθάνεται ότι είναι «μέσα» στο εργαστήριο. Αντί να χρησιμοποιεί μόνο το ποντίκι, θα έχει τη δυνατότητα να πιάνει και να μετακινεί τα αντικείμενα με φυσικές κινήσεις, καθιστώντας το μάθημα πολύ πιο ζωντανό και συναρπαστικό.

3. Καλύτερη Εκπαιδευτική Εμπειρία: Για να εξελιχθεί η εφαρμογή σε ένα πλήρες εκπαιδευτικό εργαλείο, θα πρέπει να ενισχυθεί με εργαλεία που προάγουν τη μάθηση. Αρχικά, θα πρέπει να περιλαμβάνει έτοιμα πειράματα με συγκεκριμένα βήματα που θα καθοδηγούν τον μαθητή στην ανακάλυψη θεμελιωδών αρχών. Δεύτερον, αξιολόγησημε αυτόματα τεστ και ερωτήσεις για να βλέπει ο μαθητής τι έμαθε. Τέλος, χρειάζεται την ανάπτυξη ενός συστήματος παρακολούθησης της προόδου που θα καταγράφει την εξέλιξη του χρήστη και θα επισημαίνει τα σημεία που απαιτούν περισσότερη εξάσκηση.

4. Δυνατότητα Εξέλιξης και Νέες Ενότητες: Το σύστημα πρέπει να είναι φτιαγμένο έτσι ώστε να μπορεί να επεκτείνεται εύκολα. Πρέπει επομένως, να ενσωματώνονται και να προστίθενται

νέα πειράματα και θεματικές ενότητες χωρίς να χρειάζεται να ανακατασκευαστεί η εφαρμογή από την αρχή. Έτσι, το ψηφιακό εργαστήριο θα παραμένει χρήσιμο για πολλά χρόνια και για περισσότερα αντικείμενα σπουδών.

4.3 Συνεισφορά στην επιστημονική και εκπαιδευτική κοινότητα

Η συγκεκριμένη εργασία προσφέρει μία σημαντική συμβολή επιστημονική και εκπαιδευτική κοινότητα παρουσιάζοντας ένα πρακτικό, μηδενικού κόστους, ολοκληρωμένο και επεκτάσιμο εικονικό εργαστήριο ηλεκτρονικής, που έχει δημιουργηθεί αποκλειστικά στο περιβάλλον της Unity. Αυτή η μέθοδος δείχνει πώς οι σύγχρονες τεχνολογίες σχεδίασης παιχνιδιών μπορούν να χρησιμοποιηθούν αποτελεσματικά για εκπαιδευτικούς σκοπούς, κάνοντάς τη μάθηση πιο προσιτή και διαδραστική. Παράλληλα, η εργασία προσφέρει ένα ανοιχτό και ευέλικτο πλαίσιο ανάπτυξης, το οποίο μπορεί να χρησιμοποιηθεί ως βάση για μελλοντικές ακαδημαϊκές ή ερευνητικές δραστηριότητες. Με τον τρόπο αυτό, η παρούσα εργασία λειτουργεί ως γέφυρα μεταξύ θεωρητικής γνώσης, τεχνολογικής υλοποίησης και εκπαιδευτικής εφαρμογής, ενισχύοντας την καινοτομία στον τομέα των εικονικών εργαστηρίων και συνεισφέροντας στην βαθύτερη κατανόηση καταστάσεων που συχνά απασχολούν τους μαθητές σε κλασικά εργαστήρια.

4.4 Συμπέρασμα

Συμπερασματικά, η παρούσα διπλωματική εργασία δείχνει ότι η κατασκευή ενός εικονικού εργαστηρίου φυσικής στον κόσμο της Unity είναι όχι μόνο εφικτή, αλλά και εξαιρετικά αποτελεσματική ως μέσο εκπαίδευσης. Με τον συνδυασμό της 3D απεικόνισης, της διαδραστικότητας και των προσομοιώσεων, το αναπτυγμένο σύστημα παρέχει στους χρήστες έναν εναλλακτικό και εντυπωσιακό τρόπο να πειραματιστούν, να εξερευνήσουν και να κατανοήσουν θεμελιώδεις αρχές της ηλεκτρονικής και της μηχανικής. Οι χρήστες δεν είναι πλέον απλοί θεατές, αλλά ενεργοί πειραματιστές σε ένα ασφαλές ψηφιακό περιβάλλον, όπου μπορούν να δοκιμάσουν ιδέες, να παρατηρήσουν και να εμπεδώσουν γνώσεις μέσα από την άμεση επίδραση των πράξεών τους.

Παρά την ανάγκη για εξοπλισμό ή τη διαφορά μεταξύ προσομοίωσης και πραγματικότητας, σκοπός αυτού του εικονικού εργαστηρίου είναι να λειτουργήσει συμπληρωματικά αλλά και εναλλάξ με τα παραδοσιακά φυσικά εργαστήρια, παρέχοντας πρόσβαση σε πειράματα που θα ήταν δύσκολα ή κοστοβόρα να γίνουν στο εργαστήριο.

Τέλος, αυτή η εργασία δεν αποτελεί απλώς μια τεχνική εφαρμογή, αλλά μια απόδειξη ότι τα εργαλεία που χρησιμοποιούνται στη δημιουργία ψηφιακών παιχνιδιών μπορούν να γίνουν αποτελεσματικά και σύγχρονα εκπαιδευτικά μέσα. Σε βάθος χρόνου, η ενσωμάτωση τέτοιων ψηφιακών λύσεων στις εκπαιδευτικές υποδομές ενισχύει τον εκσυγχρονισμό της μάθησης. Προσφέρει ένα ασφαλές και σύγχρονο περιβάλλον με αποτέλεσμα να δίνει στους φοιτητές τη δυνατότητα να εξερευνούν και να κατανοούν τον κόσμο των θετικών επιστήμων με όρους του μέλλοντος και έναν νέο, ψηφιακό τρόπο.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Βιβλία

[1] M. A. Alqahtani και B. Mohammad, «Hands-on and virtual laboratories in electronic circuits learning,» Information, τόμ. 15, αρ. 11, σ. 672, 2023.

[2] A. Hasni, E. Mavrikaki και Z. Ben Lakhdar, «Effectiveness of virtual laboratories in engineering education: A systematic review and meta-analysis,» PLOS ONE, τόμ. 19, αρ. 5, e0316269, 2024.

[3] N. D. Kashaka, «Virtual laboratories in science education: Benefits and challenges,» ResearchGate Preprint, 2024.

[4] M. Majumder, «Role of simulation software in enhancing the quality of electronics teaching,» IOSR Journal of Engineering, τόμ. 3, αρ. 8, σσ. 28–32, 2013.

[5] L. Najjar και M. Osman, «Exploring users' acceptance of electronic circuits simulation,» Journal of Computer Science Applications, τόμ. 12, αρ. 2, σσ. 45–58, 2024.

[6] C. Smaill, G. B. Rowe, E. Godfrey και R. Paton, «The effectiveness of learning simulations for electronic laboratories,» στα Proceedings of the 2011 IEEE Global Engineering Education Conference (EDUCON), σσ. 161–166, IEEE, 2011.

[7] "Milestones: SPICE (Simulation Program with Integrated Circuit Emphasis), 1969–1970," IEEE Engineering and Technology History Wiki

[8] A. Vladimirescu, *The SPICE Book*, IEEE Press, τόμ. 1, σσ. 1–540, 1994.

[9] What virtual laboratory usage tells us about laboratory skill education pre- and post-COVID-19: Focus on usage, behavior, intention and adoption https://pmc.ncbi.nlm.nih.gov/articles/PMC8188155

[10] Teaching Digital Electronics during the COVID-19 Pandemic via a Remote Lab
https://pmc.ncbi.nlm.nih.gov/articles/PMC9501403/

[11] Exploring the use of virtual laboratory simulations before, during, and post COVID-19 recovery phase: An Animal Biotechnology case study
https://pubmed.ncbi.nlm.nih.gov/34291546

[12] S. Aliane, "Enhancing repeatability and experimentation through virtual laboratories," IEEE Transactions on Learning Technologies, 2020.

[13] A. Bilgin & M. S. Dikmen, «A learner-centered virtual laboratory for electronics engineering education,» *Computer Applications in Engineering Education*, vol. 26, no. 6, pp. 2207–2220, 2018.

[14] V. Potkonjak et al., «Virtual laboratories for education in science, technology, and engineering: A review,» Computers & Education, vol. 95, pp. 309–327, 2016.

[15] J. Self & R. Azevedo, "Supporting self-regulated learning in virtual engineering environments," IEEE Transactions on Learning Technologies, 2022.

[16] Y. A. Wahid, "Remote labs as a tool for innovation in engineering education," 2021.

[17] A. Almarzoug, «Using virtual labs to teach advanced electronics in engineering courses,, vol. 9, pp. 116214–116225, 2021

[18] A. Butz, «3D simulation technologies for engineering education,» Computer Graphics and Applications, τόμ. 37, αρ. 4, σσ. 15–23, 2017

[19] Sedra, A., & Smith, K., Microelectronic Circuits, Oxford University Press, 2020, σσ. 125–134.

[20] Ibrahim, D., Microcontroller-Based Practical Design, Elsevier, 2011, σσ. 45–60.

[21] Blum, J., Exploring Arduino: Tools and Techniques for Engineering Wizardry, Wiley, 2019, σσ. 45–58.

[22] Allaboutcircuits.com, Circuit Simulation Tools Review, 2022, σσ. 5–12.

[23] Shinnu Jangra, Gurjinder Singh, Interactivity Development Using Unity 3D Software and C# Programming. Unity Media, 2023.

[24] Z. Lai et al., "Design of Three-Dimensional Virtual Simulation Experiment Platform for Integrated Circuit Course," Electronics, 2022.

[25] C. Tang et al., "Open Framework Design for Electronic Virtual Simulation Experiments With Unity3D," 2024 IEEE International Conference on Consumer Electronics (ICCE), 2024.

[26] Z. Hao, "Experiment Information System based on an Online Virtual Reality Environment," Journal of Educational Technology Systems, 2023.

[27] K. Safina, "C# Development in Unity Logic Game Editor – First Person," Proceedings of the International Conference on Game Development, 2022.

[28] A. Oshima, "Development of Student Experiment on High-frequency Circuit using Virtual Reality," IEEE Transactions on Education, 2023.

[29] R. Smith (Ed.), Learning Game Development with Unity3D Engine and Arduino Microcontroller, Springer, 2023.

[30] T. Jiang et al., "A Multi-person collaborative Simulation System For Circuit Experiment System Base on Virtual Reality," in 2021 International Conference on Digital Society and Artificial Intelligence (DSAI), 2021, pp. 142-146.

3D Models

[31] Resistor, Lowpoly resistor - Download Free 3D model by Jiří Kuba (@kuba.jirka) [9fb4a7c]

[32]Multimeter,https://sketchfab.com/3d-models/compact-multimeter-lowpoly-b294ff85d4a44cc69cd01bdd7b6ea40f

[33] Power Supply, Electrical Power Supply - Download Free 3D model by kerouac9 (@kerouac9) [a8e6c84]

Youtube Links

[34] Michelson Interferometer Fringes - YouTube