Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή εργασία

Τίτλος: Εφαρμογές εμβύθισης (Immersive) μέσω γυαλιών εικονικής πραγματικότητας

Μάριος Τζιρίτας (ΑΜ: 3583)

Επιβλέπων εκπαιδευτικός : Αθανάσιος Μαλάμος

ΗΡΑΚΛΕΙΟ
2019

Ευχαριστίες

Ο γράφων ευχαριστεί τον καθηγητή Αθανάσιο Μαλάμο για την καθοδήγησή του και την παροχή του εξοπλισμού του εργαστηρίου Πολυμέσων , Δικτύων και Επικοινωνιών.

Abstract

For a decade now, virtual reality has come into people's lives, at first with difficulties and skepticism, but now everyone has at least a little bit of contact with it. In the present thesis will explored and analyzed the beginning of virtual reality, today's technologies on virtual reality, and how this can be further integrated into the user's daily life through the web browsing.

The technologies that will be analyzed are initially the choices that the general public has, if they want to enter the world of virtual reality, and how they have come to be as they are today (HTC Vive, Oculus Rift, etc.). Technologies around 3D graphics and browser-level interactivity used in virtual reality applications (Web3D, VRML, etc.), as well different APIs that specialize in virtual reality web applications (ThreeJS, WebVR, etc.).

Σύνοψη

Εδώ και μια δεκαετία η εικονική πραγματικότητα έχει μπει στη ζωή των ανθρώπων, αρχικά με δυσκολίες και σκεπτικισμό, αλλά πλέον όλοι έχουν έστω και μια μικρή επαφή με αυτήν. Στην παρούσα εργασία θα μελετηθούν και θα αναλυθούν, η αρχή της εικονικής πραγματικότητας, οι τεχνολογίες του σήμερα πάνω στην εικονική πραγματικότητα, και πως αυτή μπορεί να μπεί περισσότερο στην καθημερινότητα του χρήστη μέσα από την περιήγησή του στον παγκόσμιο ιστό.

Οι τεχνολογίες που θα αναλυθούν είναι αρχικά οι επιλογές που έχει το ευρύ κοινό, αν θελήσει να μπει στον κόσμο της εικονικής πραγματικότητας, και πως αυτές έφτασαν να είναι στη μορφή που είναι σήμερα (HTC Vive, Oculus Rift, κ.α). Τεχνολογίες γύρω από τα τρισδιάστατα γραφικά και διαδραστικότητα, σε επίπεδο περιηγητών, που χρησιμοποιούνται σε εφαρμογές εικονικής πραγματικότητας (Web3D, VRML, κ.α), όπως επίσης και διαφορετικά APIs που εξειδικεύονται σε εφαρμογές εικονικής πραγματικότητας στο διαδίκτυο (ThreeJS, WebVR, κ.α).

Πίνακας Περιεχομένων

Λίστα Εικόνων

Κεφάλαιο 1: Εισαγωγή

Περίληψη

Σκοπός της παρούσας πτυχιακής είναι η ανάπτυξη μιας διαδικτυακής εφαρμογής εμβύθισης σε περιβάλλον εικονικής πραγματικότητας, σε JavaScript, με τη χρήση εργαλείων όπως το ThreeJS, εκμεταλλευόμενοι τις δυνατότης του HTC Vive και των controller που αυτό διαθέτει.

Η εφαρμογή που αναπτύχθηκε επιτρέπει στο χρήστη να δημιουργήσει ένα σχήμα της επιλογής του με χρήση του controller και να το μετακινήσει ή να το περιστρέψει με τη χρήση κουμπιών μέσα στο VR περιβάλλον.

Κίνητρο για την διεξαγωγή της εργασίας

Κίνητρο για τη διεξαγωγή της εργασίας ήταν η απόκτηση γνώσεων σχετικά με τις πρόσφατες τεχνολογίες σχετικά με την εικονική πραγματικότητα και πως αυτές εφαρμόζονται σε περιηγητές ιστού. Επίσης κίνητρο ήταν η εμβάθυνση στην JavaScript και συγκεκριμένα στο ThreeJs που αφορά σε μεγάλο βαθμό αυτή την εργασία.

Δομή Εργασίας

- Κεφάλαιο 2: Ιστορική αναδρομή και εφαρμογές - Τεχνολογίες του σήμερα.
- Κεφάλαιο 3: Πλατφόρμες τρισδιάστατων γραφικών.
- Κεφάλαιο 4: Τεχνολογίες WebVR – ThreeJs - Ammo.
- Κεφάλαιο 5: Κύριο μέρος της εργασίας - Συμπεράσματα.
- Κεφάλαιο 6: Βιβλιογραφία.

Κεφάλαιο 2: Ιστορική αναδρομή και εφαρμογές - Τεχνολογίες του σήμερα

Εικονική πραγματικότητα ορίζεται η προσομοίωση μιας τρισδιάστατης εικόνας ή περιβάλλοντος, που έχουν παραχθεί από υπολογιστή, με τα οποία μπορεί να αλληλεπιδράσει ένα πρόσωπο, με φαινομενικά αληθοφανή ή φυσικό τρόπο, χρησιμοποιώντας ειδικό ηλεκτρονικό εξοπλισμό όπως κράνη εικονικής πραγματικότητας με ενσωματωμένη οθόνη, γάντια με αισθητήρες ή χειριστήρια. Ο όρος ‘‘εικονική πραγματικότητα’’ πρωτοεμφανίστηκε στο έργο του Antoine Artaud “Le Théâtre Alchimique” το 1932 αλλά και στο έργο του “Le Théâtre et son Double” το 1938 που αναφέρει το θέατρο σαν εικονική πραγματικότητα. Ο όρος με την καθαρά τεχνολογική του έννοια εμφανίστηκε το 1989 από τον Jaron Lanier, πατέρα της εικονικής πραγματικότητας, ο οποίος μαζί με τον Thomas Zimmerman ίδρυσαν το 1984 την VPL Research, την πρώτη εταιρεία που διέθεσε προς πώληση κράνος εικονικής πραγματικότητας και γάντια.

Για να εμβυθιστεί πλήρως σε ένα κόσμο εικονικής πραγματικότητας ο χρήστης θα πρέπει να έχει απομονωθεί από τον έξω κόσμο, δεδομένου ότι θα δέχεται ερεθίσματα σε τουλάχιστον δύο αισθήσεις (όραση, ακοή) ίσως και τρεις (αφή). Σημαντικό ρόλο παίζει ότι το κράνος προβάλει στερεοσκοπικές εικόνες από διαφορετικές γωνίες στο κάθε μάτι ούτως ώστε να δημιουργηθεί η αίσθηση του βάθους, αλλά και ο στερεοσκοπικός ήχος μέσω του οποίου ο χρήστης βυθίζεται περισσότερο μέσα στον εικονικό κόσμο και παράλληλα αποκλείεται από τον πραγματικό. Η αφή επιτυγχάνεται με τη βοήθεια φορετών συσκευών (γάντια) ή με τηλεχειριστήρια με δόνηση, προκειμένου ο χρήστης να νιώθει την αντίσταση, τις υφές και το σχήμα του αντικειμένου που ακουμπάει.

Η εικονική πραγματικότητα βρίσκει εφαρμογές σε ολοένα και περισσότερους τομείς όπως στην εκπαίδευση αστροναυτών και πιλότων, στην ιατρική, στην αρχιτεκτονική, στον τουρισμό, κλπ. Στην εκπαίδευση τα οφέλη θα λέγαμε ότι είναι διπλά καθώς οι μαθητές, κυρίως σε μικρές ηλικίες, δρώντας σε ομάδες μέσα σε ένα περιβάλλον εικονικής πραγματικότητας, μαθαίνουν να συνεργάζονται αλλά παράλληλα εξοικειώνονται με τα τρισδιάστατα περιβάλλοντα.

Ένα άλλο πεδίο εφαρμογής εικονικής πραγματικότητας είναι οι ένοπλες δυνάμεις. Κυρίως σε θέματα στρατιωτικής εκπαίδευσης και ασκήσεων, μια προσομοίωση εικονικής πραγματικότητας δίνει τη δυνατότητα για την εκτέλεση δύσκολων και περίπλοκων ασκήσεων χωρίς να κινδυνεύει η σωματική ακεραιότητα των στρατιωτών και να ρισκάρεται η φθορά ή απώλεια εξοπλισμού. Το NATO εξέδωσε το 2003 μια αναφορά με τίτλο “Virtual Reality: State of Military Research and Applications in Member Countries”, η οποία δείχνει ότι οι βασικές στρατιωτικές εφαρμογές, που σχετίζονται με την εικονική πραγματικότητα, είναι αυτές που έχουν την ικανότητα να μοντελοποιήσουν ένα οπλικό σύστημα. Επίσης οι εφαρμογές αυτές μπορούν να αποδειχθούν χρήσιμες στην ανάπτυξη νέων συστημάτων στρατιωτικής εκπαίδευσης, και σχεδιασμού επιχειρήσεων. Στον τομέα της αεροπορίας η εικονική πραγματικότητα ξεφεύγει από τα όρια ενός απλού προσομοιωτή πτήσης. Η εκπαίδευση των πιλότων απαιτεί τον μεγαλύτερο δυνατό ρεαλισμό για αυτό το λόγο η χρήση της VR τεχνολογίας προσφέρεται σαν τέλεια εναλλακτική στη συμβατική εκπαίδευση, συμβάλλει παράλληλα στην ανάπτυξη τακτικών και σχεδίων, αλλά κυρίως απαλείφει τον κίνδυνο του τραυματισμού ή της απώλειας ζωής του πιλότου. Εκτός της αεροπορίας η εικονική πραγματικότητα είναι χρήσιμη στο πεζικό καθώς απαιτούνται προσομοιώσεις άκρως ρεαλιστικές για την απεικόνιση και τη λειτουργία των οχημάτων, τις αντίξοες καιρικές συνθήκες και τα διαφορετικά είδη εδάφους.

Στον τομέα της ιατρικής και συγκεκριμένα της ψυχιατρικής η εικονική πραγματικότητα είναι πολλά υποσχόμενη, καθώς έχει εφαρμοστεί σε διάφορες διαταραχές, κυρίως φοβίες. Η κλασική αντιμετώπιση κάποιας φοβίας περιλαμβάνει δύο στάδια, στο πρώτο ο ασθενής φαντάζεται να έρχεται σε επαφή με το αντικείμενο του φόβου του, ενώ στο δεύτερο έρχεται σταδιακά σε επαφή με το φόβο του. Η εικονική πραγματικότητα εισάγει ένα ενδιάμεσο τρίτο στάδιο στη θεραπεία, στο οποίο ο ασθενής έρχεται σε επαφή με το αντικείμενο του φόβου εικονικά σε προστατευμένο περιβάλλον πλήρως ελεγχόμενο. Μια δεύτερη πτυχή της θεραπείας στην οποία βοηθάει η εικονική πραγματικότητα είναι το πιθανό κόστος. Για παράδειγμα ένας ασθενής με αεροφοβία (φοβία πτήσης) πρέπει να εκτίθεται επαναληπτικά σε περιβάλλον πτήσης και αυτό σε πραγματικές συνθήκες είναι κοστοβόρο και μη ελεγχόμενο σε περίπτωση πανικού του ασθενή. Με τη χρήση εικονικού περιβάλλοντος μειώνεται το κόστος και σε περίπτωση πανικού από τον ασθενή η προσομοίωση μπορεί να σταματήσει. Έρευνες δείχνουν ότι εφαρμογές εικονικής πραγματικότητας μπορούν να συμβάλλουν στην αντιμετώπιση του πόνου σε ορισμένους ασθενείς, όταν ο πόνος αυτός εξαρτάται και από ψυχολογικούς παράγοντες. Η εικονική πραγματικότητα βοηθάει στο να αλλάξει την προσοχή του ο ασθενής με αποτέλεσμα να υπάρχει σημαντική μείωση του πόνου. Αυτό αποδείχθηκε σε πειράματα του Εργαστηρίου Τεχνολογίας Ανθρώπινων Διεπαφών του Πανεπιστημίου της Ουάσιγκτον, όταν εθελοντές υποβλήθηκαν σε fMRI ενώ τους δημιουργούνταν ερεθίσματα πόνου. Χωρίς τη χρήση εφαρμογών εικονικής πραγματικότητας κατά τη διαδικασία, φάνηκε ότι υπήρχε μεγάλη διέγερση στα κέντρα του εγκεφάλου που είναι υπεύθυνα για τα ερεθίσματα του πόνου, ενώ στην αντίθετη περίπτωση καταγράφηκε μείωση των επιπέδων διέγερσης. Οι εφαρμογές εικονικής πραγματικότητας εκτός από τους ασθενείς βοηθάνε και στην εκπαίδευση του ιατρικού προσωπικού. Εφαρμογές έχουν σχεδιαστεί για την εκπαίδευση χειρούργων, οδοντιάτρων και νοσηλευτριών. Οι ορθοπεδικοί χρησιμοποιώντας προσομοιώσεις εικονικής πραγματικότητας αναπαριστούν επεμβάσεις σε άκαμπτα οστά, όπως οστεοτομίες, αρθροπλαστικές και ακρωτηριασμούς.

HTC Vive

Το HTC Vive είναι ένα headset εικονικής πραγματικότητας που αναπτύχθηκε από την HTC και τη Valve Corporation. Το σετ μικροφώνου-ακουστικού χρησιμοποιεί την τεχνολογία παρακολούθησης "room-scale", επιτρέποντας στον χρήστη να μετακινείται σε τρισδιάστατο χώρο και να χρησιμοποιεί motion-tracked controllers για να αλληλεπιδράσουν με το περιβάλλον. Το HTC Vive παρουσιάστηκε κατά τη διάρκεια του Mobile World Congress της HTC τον Μάρτιο του 2015. Τα κιτ ανάπτυξης κυκλοφόρησαν τον Αύγουστο και τον Σεπτέμβριο του 2015 ενώ η πρώτη έκδοση για τους καταναλωτές κυκλοφόρησε στις 7 Ιουνίου 2016. Πρωτότυπα ενός συστήματος εικονικής πραγματικότητας που παράγεται από τη Valve επιδείχθηκαν κατά τη διάρκεια του 2014. Στις 23 Φεβρουαρίου 2015, η Valve ανακοίνωσε τη SteamVR και ότι θα επιδείξει ένα "σύστημα υλικού SteamVR" το 2015 στο Conference Developers Game. Η HTC παρουσίασε επίσημα τη συσκευή της, την 1η Μαρτίου 2015. Οι προπαραγγελίες άρχισαν στις 29 Φεβρουαρίου 2016 στις 10:00 πμ. Η Valve και η HTC είχαν ανακοινώσει ότι το headset θα είναι δωρεάν για επιλεγμένους προγραμματιστές. Στο Consumer Electronics Show 2016, η HTC και η Valve αποκάλυψαν μια σχεδόν τελική, αναθεωρημένη συσκευή, γνωστή ως HTC Vive Pre.

Το απλό headset της Vive έχει ρυθμό ανανέωσης 90 Hz και οπτικό πεδίο 110 μοιρών. Η συσκευή χρησιμοποιεί δύο πίνακες OLED, ένα ανά μάτι, το καθένα με ανάλυση οθόνης

1080 × 1200 (2160 × 1200 συνδυασμένα εικονοστοιχεία). Τα χαρακτηριστικά ασφαλείας περιλαμβάνουν μια κάμερα που βλέπει προς τα εμπρός και επιτρέπει στο χρήστη να παρακολουθεί το περιβάλλον του χωρίς να αφαιρεί το σετ μικροφώνου-ακουστικού. Το λογισμικό μπορεί επίσης να χρησιμοποιήσει την κάμερα για να εντοπίσει οποιαδήποτε κινούμενα ή στατικά αντικείμενα σε ένα δωμάτιο. αυτή η λειτουργία μπορεί να χρησιμοποιηθεί ως μέρος ενός συστήματος ασφαλείας "Chaperone", το οποίο θα εμφανίζει αυτόματα έναν εικονικό τοίχο ή μια τροφοδοσία από την κάμερα για την ασφαλή καθοδήγηση των χρηστών από εμπόδια ή πραγματικούς τοίχους. Μέσα από τα εξωτερικά κελύφη του ακουστικού είναι δεκάδες αισθητήρες υπερύθρων που ανιχνεύουν τους παλμούς υπέρυθρων σταθμών βάσης για να προσδιορίσουν την τρέχουσα θέση του headset στο χώρο. Άλλοι αισθητήρες που περιλαμβάνονται είναι ένας αισθητήρας G-Sensor, γυροσκόπιο και αισθητήρας εγγύτητας (proximity sensor).

Οι controllers έχουν πολλαπλές μεθόδους εισόδου, συμπεριλαμβανομένου ενός trackpad, κουμπιών πιασίματος και μιας σκανδάλης, ενώ χρειάζεται φόρτιση περίπου ανά 6 ώρες. Από το δαχτυλίδι του controller υπάρχουν 24 υπέρυθροι αισθητήρες που ανιχνεύουν τους σταθμούς βάσης για να καθορίσουν την τοποθεσία του ελεγκτή. Το σύστημα παρακολούθησης SteamVR χρησιμοποιείται για την παρακολούθηση της θέσης του ελεγκτή σε κλάσμα ενός χιλιοστού, με ρυθμούς ενημέρωσης που κυμαίνονται από 250 Hz έως 1 kHz.

Για να προσφέρει τις πιο συναρπαστικές εμπειρίες εικονικής πραγματικότητας, το HTC Vive και τα αξεσουάρ του πρέπει να προσφέρουν την καλύτερη δυνατή παρακολούθηση. Το tracking του HTC Vive εξαρτάται σε μεγάλο βαθμό από το υπέρυθρο φως που εκπέμπεται από δύο σταθερούς σταθμούς βάσης και γι 'αυτό το headset και οι controllers είναι γεμάτοι με αισθητήρες. Με τους σταθμούς αυτούς που εκπέμπουν υπέρυθρα σήματα, μια σειρά αισθητήρων σε αυτές τις συσκευές μπορεί να εντοπίσει με ακρίβεια τη θέση τους σε σχέση με τους σταθμούς. Κάθε controller έχει στην επιφάνειά του κοιλότητες ανίχνευσης υπερύθρων, με την κάθε κοιλότητα να διαθέτει ένα υπέρυθρο φίλτρο, το οποίο έχει σκοπό να αφαιρέσει το εισερχόμενο φως που εκπέμπεται από τους σταθμούς βάσης της HTC Vive.

Παρατηρώντας την κορυφή του ελεγκτή, μπορούμε να δούμε πληθώρα κουμπιών και επιφανειών. Το μεγαλύτερο κυκλικό επίπεδο είναι το touchpad (ή trackpad) του ελεγκτή, το οποίο λειτουργεί παρόμοια με αυτό του Steam Controller της Valve. Άλλες μέθοδοι εισαγωγής περιλαμβάνουν τα κουμπιά συστήματος και μενού που βρίσκονται λίγο πιο κάτω από το touchpad. Στην κάτω πλευρά της συσκευής υπάρχει ένα κουμπί σε μορφή σκανδάλης, το οποίο χρησιμοποιείται συχνά όπως το αριστερό κλικ κατά τη διάρκεια της πλοήγησης μενού και του παιχνιδιού. Στο κάτω μέρος του controller υπάρχουν δύο κουμπιά Grip, τα οποία έχουν τοποθετηθεί εκεί για εύκολη πρόσβαση και πάτημα απλά σφίγγοντας τη βάση του controller.

Ανοίγοντας τον controller γίνεται σαφές ότι η HTC δεν έκανε περικοπές στην ποιότητα κατασκευής του Vive. Κάθε μία από τις πλακέτες που είναι διατεταγμένες γύρω από το δαχτυλίδι του controller φιλοξενεί έναν από τους 24 αισθητήρες. Οι αισθητήρες είναι σταθεροποιημένοι σε δύο δακτυλίους οι οποίοι ενώνονται με ένα καλώδιο. Οι αισθητήρες μοιράζονται εξίσου στο πάνω και κάτω μέρος του controller συμπεριλαμβανομένων και των δύο αισθητήρων που βρίσκονται στο κέντρο. Προχωρώντας στην κεντρική πλακέτα του controller συναντάμε έναν NXP's LPC11U3x 32-bit ARM Cortex-M0 microcontroller και ένα

Lattice's iCE40 HX FPGA. Η ανίχνευση της κίνησης γίνεται με τη βοήθεια του InvenSense's MPU-6500 Six-Axis, το οποίο περιλαμβάνει ένα επιταχυνσιόμετρο τριών αξόνων και ένα γυροσκόπιο τριών αξόνων, όλα σε μία μονάδα. Δίπλα του υπάρχει μια μονάδα Serial Flash Embedded Memory Micron M25P40 (3V, 4Mb). Παρόμοια εξαρτήματα συναντάει κανείς και στο headset. Ο controller τροφοδοτείται από μια επαναφορτιζόμενη Li-poly μπαταρία των 960 mAh, η οποία βρίσκεται στο πίσω μέρος της πλακέτας, ενώ φορτίζει μέσω θύρας Micro USB που βρίσκεται στο κάτω μέρος της συσκευής. Το touchpad ανιχνεύει τις κινήσεις του δαχτύλου του χρήστη όταν κλικάρει μέσω ενός μικροδιακόπτη που υπάρχει από κάτω του. Το σύστημα αυτό δεν είναι καινούριο καθώς έχει εφαρμοστεί ξανά στο controller του Steam, όπως προαναφέρθηκε, καθώς και οι δύο controllers (Vive και Steam) μοιράζονται τον ίδιο Cirque 1CA027 companion microcontroller.

Η κύρια ιδέα πίσω από την τεχνολογία παρακολούθησης (tracking) είναι σχετικά απλή. Ο σταθμός βάσης «βομβαρδίζει» το χώρο με μη ορατή υπέρυθρη ακτινοβολία, λειτουργώντας έτσι σαν σημείο αναφοράς για όλες τις συσκευές παρακολούθησης (headset ή controllers), για να ανιχνεύσει που βρίσκονται αυτές σε πραγματικό τρισδιάστατο χώρο. Στην ουσία ο σταθμός βάσης λειτουργεί σαν φάρος, «αφήνοντας» το headset να κάνει τους υπολογισμούς. Ο σταθμός βάσης διαθέτει ένα αριθμό από LEDs και ένα ζεύγος εκπομπών laser. Εξήντα φορές ανά δευτερόλεπτο τα LED και ένας από τους δύο πομπούς laser εκπέμπουν φως στο χώρο. Εντωμεταξύ ο αποδέκτης (headset ή controller), που διαθέτει αισθητήρες φωτός, μετράει μέχρι να εντοπίσει τον αισθητήρα που χτυπήθηκε. Ύστερα χρησιμοποιεί τη θέση του αισθητήρα στο headset και το χρόνο που χτυπήθηκε από το φως και υπολογίζει μαθηματικά την ακριβή θέση του δέκτη σε συνάρτηση με τη θέση των σταθμών βάσης στο χώρο.

Oculus Rift

Το Oculus Rift είναι μια σειρά από headset εικονικής πραγματικότητας που αναπτύχθηκαν και κατασκευάστηκαν από την Oculus VR, τμήμα της Facebook Inc. που κυκλοφόρησε στις 28 Μαρτίου 2016. Το 2012 η Oculus ξεκίνησε μια εκστρατεία kickstarter για να χρηματοδοτήσει την ανάπτυξη του Rift, αφού ιδρύθηκε ως ανεξάρτητη εταιρεία δύο μήνες πριν. Το έργο αποδείχθηκε επιτυχημένο, φθάνοντας σχεδόν τα 2,5 εκατομμύρια δολάρια από περίπου 10.000 συνεισφέροντες. Το Μάρτιο του 2014, το Oculus αγοράστηκε από το Facebook για 2 δισεκατομμύρια δολάρια. Το Rift άλλαξε διάφορα μοντέλα προ-παραγωγής μετά την εκστρατεία Kickstarter, περίπου πέντε από τα οποία προβλήθηκαν στο κοινό πριν φθάσουν στην εμπορική κυκλοφορία του. Δύο από αυτά τα μοντέλα μεταφέρθηκαν σε υποστηρικτές, που χαρακτηρίστηκαν ως κιτ ανάπτυξης, η DK1 στα μέσα του 2013 και η DK2 στα μέσα του 2014, με σκοπό να παρέχουν στους προγραμματιστές μια πλατφόρμα για την έγκαιρη ανάπτυξη του περιεχομένου για την απελευθέρωση του Rift. Ωστόσο αγοράστηκαν και οι δύο εκδόσεις από πολλούς λάτρεις που ήθελαν να πάρουν μια γεύση της τεχνολογίας αυτής στα αρχικά της στάδια. Το Rift κυκλοφόρησε στο εμπόριο το Μάρτιο του 2016 με το Rift CV1, το οποίο διακόπηκε το Μάρτιο του 2019 με την κυκλοφορία του διαδόχου του, του Oculus Rift S.

Μέσω της εικονικής πραγματικότητας του MTBS (Meant To Be Seen) και των φόρουμ συζητήσεων 3D, ο Palmer Luckey, ο ιδρυτής του Oculus και ο μακροπρόθεσμος συντονιστής της φόρουμ συζητήσεων MTBS, ανέπτυξε την ιδέα της δημιουργίας ενός νέου head-mounted display που ήταν και πιο αποτελεσματικό από ό,τι υπήρχε τότε στην αγορά, και φθηνή για τους παίκτες. Το πρώτο ακατέργαστο πρωτότυπο φτιάχτηκε το 2011 από τον Palmer Luckey (τότε 18 ετών) στο γκαράζ των γονέων του στο Long Beach της Καλιφόρνια. Συμπτωματικά, ο John Carmack έκανε τις δικές του έρευνες και συνέβη στις εξελίξεις του Luckey ως μέλος του MTBS. Μετά τη δοκιμή ενός πρώιμου πρωτοτύπου, ο Carmack τάχθηκε υπέρ της προσέγγισης του Luckey και λίγο πριν από την έκθεση Electronic Entertainment Expo του 2012, η Id Software ανακοίνωσε ότι η μελλοντική τους έκδοση του Doom 3, BFG Edition, θα ήταν συμβατή με μονάδες head-mounted display. Τον Ιούνιο του 2012, κατά τη διάρκεια της σύμβασης E3, η Carmack εισήγαγε ένα head-mounted display που βασίζεται στο πρωτότυπο Oculus Rift του Luckey, το οποίο έτρεξε το λογισμικό της Carmack. Η μονάδα διαθέτει IMU υψηλής ταχύτητας και οθόνη LCD 5,6 ιντσών (14 cm), ορατή με διπλούς φακούς, τοποθετημένα πάνω από τα μάτια, για να παρέχουν οριζόντια και 110 μοίρες κάθετη στερεοσκοπική προοπτική 3D.

Δύο μήνες μετά τη σύστασή του ως εταιρίας, η Oculus VR του Palmer ξεκίνησε μια εκστρατεία crowdfunding kickstarter την 1η Αυγούστου 2012 για το headset εικονικής πραγματικότητας που ονομάζεται Rift. Ο κύριος σκοπός του Kickstarter ήταν να δώσει ένα πρωτότυπο Oculus Rift - τώρα αναφέρεται ως DK1 (Development Kit 1) – στους προγραμματιστές για να ξεκινήσει την ενσωμάτωση της συσκευής στα παιχνίδια τους. Το Rift DK1 κυκλοφόρησε στις 29 Μαρτίου 2013 και χρησιμοποιούσε μια οθόνη 7 ιντσών (18 cm) με σημαντικά χαμηλότερο χρόνο εναλλαγής pixel από το αρχικό πρωτότυπο, μειώνοντας την καθυστέρηση και τη θαμπάδα όταν γυρνάει γρήγορα το κεφάλι του χρήστη (motion blur). Η πλήρωση των εικονοστοιχείων είναι επίσης καλύτερη, μειώνοντας το screen door effect και καθιστώντας τα επιμέρους εικονοστοιχεία λιγότερο αισθητά, ενώ η οθόνη LCD είναι φωτεινότερη και το βάθος χρώματος είναι 24 bit ανά εικονοστοιχείο.

Η οθόνη 7 ιντσών κάνει το overlapping στο στερεοσκοπικό 3D να μην είναι στο 100%, αφού το αριστερό μάτι βλέπει επιπλέον χώρο προς τα αριστερά και το δεξί μάτι βλέπει επιπλέον χώρο προς τα δεξιά, χώροι στους οποίους δεν υπάρχει αντίληψη 3D βάθους. Το οπτικό πεδίο (FOV) υπερβαίνει τις 90 μοίρες οριζόντια (διαγώνιος 110 μοίρες), το οποίο είναι περισσότερο από το διπλάσιο του FOV των προηγούμενων συσκευών VR από άλλες εταιρείες και είναι το δυνατό σημείο της συσκευής. Η ανάλυση είναι στα 1280 × 800 (λόγος διαστάσεων 16:10), η οποία οδηγεί σε ένα αποτελεσματικό 640 × 800 ανά μάτι (αναλογία διαστάσεων 4: 5). Ωστόσο, δεδομένου ότι η συσκευή δεν παρουσιάζει overlapping 100% μεταξύ των ματιών, η συνδυασμένη οριζόντια ανάλυση είναι μεγαλύτερη από 640. Η εικόνα για κάθε μάτι εμφανίζεται στην οθόνη παραμορφωμένη (barrel distortion) και στη συνέχεια διορθώνεται με pincushion effect των φακών, δημιουργώντας μια spherical-mapped εικόνα για κάθε μάτι. Τα αρχικά πρωτότυπα χρησιμοποίησαν έναν ανιχνευτή κεφαλής της Hillcrest Labs 6DoF, που κανονικά τρέχει στα 125 Hz, αλλά με ειδικό firmware, που ζήτησε ο John Carmack, το οποίο τον έκανε να τρέχει στα 250 Hz. Αυτό έγινε διότι η καθυστέρηση είναι ζωτικής σημασίας λόγω της εξάρτησης του ρεαλισμού της εικονικής πραγματικότητας από τον χρόνο απόκρισης. Η πιο πρόσφατη έκδοση περιλαμβάνει τον νέο ανιχνευτή προσέγγισης πραγματικότητας 1000 Hz της Oculus, ο οποίος στοχεύει στην παροχή πολύ χαμηλότερου

tracking latency από σχεδόν οποιοδήποτε άλλο tracker. Χρησιμοποιεί ένα συνδυασμό γυροσκοπίων τριών αξόνων, επιταχυνσιόμετρα και μαγνητόμετρα, τα οποία το καθιστούν ικανό για απόλυτη παρακολούθηση προσανατολισμού της κεφαλής (σε σχέση με τη Γη) χωρίς μετατόπιση. Το κιτ ανάπτυξης 1 περιελάμβανε επίσης ανταλλακτικούς φακούς που αποσκοπούσαν στην οπτική διόρθωση. Η ολόκληρη πηγή για το Rift DK1 κυκλοφόρησε στο κοινό το Σεπτέμβριο του 2014, συμπεριλαμβανομένου του firmware, των schematics και των μηχανικών για τη συσκευή. Το firmware εκδόθηκε με απλοποιημένη άδεια BSD, ενώ τα σχήματα και οι μηχανισμοί κυκλοφορούν με την άδεια Creative Commons Attribution 4.0 International License. Τον Ιούνιο του 2013, ένα πρωτότυπο του Rift που χρησιμοποίησε ένα πάνελ LCD 1080p παρουσιάστηκε στο Electronic Entertainment Expo. Αυτό το βήμα οδηγεί στο διπλάσιο του αριθμού των εικονοστοιχείων, καθώς η DK1 μείωσε σημαντικά το screen door effect και καθιστούσε αντικείμενα στον εικονικό κόσμο πιο ξεκάθαρα, ειδικά σε απόσταση. Η κακή ανάλυση ήταν η κύρια κριτική της DK1. Αυτό το πρωτότυπο HD είναι το μοναδικό πρωτότυπο του Rift που παρουσιάζεται στο κοινό το οποίο δεν μετατράπηκε σε κιτ ανάπτυξης λογισμικού. Τον Ιανουάριο του 2014 παρουσιάστηκε στο Consumer Electronics Show ένα ενημερωμένο πρωτότυπο με την κωδική ονομασία "Crystal Cove", το οποίο χρησιμοποίησε μια ειδική OLED low-persistence οθόνη, καθώς και ένα νέο σύστημα εντοπισμού κίνησης που χρησιμοποίησε μια εξωτερική κάμερα για την παρακολούθηση υπέρυθρων κουκκίδων πάνω στο headset. Το νέο σύστημα εντοπισμού κίνησης θα επέτρεπε στο σύστημα να ανιχνεύει πράξεις όπως η κλίση ή η σκύψιμο, η οποία υποστηρίχθηκε ότι βοηθά στην ανακούφιση από το ανακάτεμα που βιώνουν οι χρήστες όταν το λογισμικό δεν ανταποκρίνεται στις ενέργειες αυτές. Το Oculus ξεκίνησε την αποστολή του Development Kit 2 (DK2) τον Ιούλιο του 2014. Πρόκειται για μια μικρή βελτίωση του πρωτότυπου Crystal Cove, που περιλαμβάνει αρκετές βασικές βελτιώσεις σε σχέση με το πρώτο κιτ ανάπτυξης, όπως η υψηλότερη ανάλυση (960 × 1080 ανά μάτι) οθόνη low-persistence OLED, υψηλότερος ρυθμός ανανέωσης, εντοπισμός θέσης, αποσπώμενο καλώδιο και τη μη χρήση του εξωτερικού κουτιού ελέγχου. Ένα teardown της DK2 αποκάλυψε ότι ενσωματώνει μια τροποποιημένη οθόνη του smartphone Samsung Galaxy Note 3, συμπεριλαμβανομένου και του μπροστινού πάνελ από την ίδια τη συσκευή. Τον Φεβρουάριο του 2015, η Oculus ανακοίνωσε ότι πάνω από 100.000 μονάδες DK2 είχαν αποσταλεί μέχρι εκείνη τη στιγμή. Τον Σεπτέμβριο του 2014, ο Oculus παρουσίασε και πάλι μια ενημερωμένη έκδοση του Rift με κωδικό όνομα Crescent Bay. Αυτή η έκδοση έχει μεγαλύτερη ανάλυση από την DK2, χαμηλότερο βάρος, ενσωματωμένο ήχο και παρακολούθηση 360 μοιρών χάρη στην παρουσία LED που βρίσκονται στο πίσω μέρος του headset. Η Oculus έχει επίσης άδεια στη βιβλιοθήκη λογισμικού RealSpace3D, η οποία αναμένεται να δώσει στο Rift αλγόριθμους HRTF και reverb. Κατά τη διάρκεια ενός πάνελ στο SXSW 2015, με τίτλο "Εξερευνήστε το μέλλον του VR", ανακοινώθηκε δημοσίως για πρώτη φορά ότι το πρωτότυπο χρησιμοποιεί δύο οθόνες αντί για μία όπως είχε προηγουμένως προταθεί. Το Oculus Rift, μερικές φορές αναφέρεται ως "Consumer Version 1" ή "CV1". Η Oculus VR ανακοίνωσε στις 6 Μαΐου 2015 ότι η consumer version του Rift θα σταλεί το πρώτο τρίμηνο του 2016 και όντως στις 25 Μαρτίου 2016 η πρώτη παρτίδα headsets άρχισε να διανείμεται στους καταναλωτές.

Η consumer version είναι μια βελτιωμένη έκδοση του πρωτότυπου Crescent Bay, που διαθέτει οθόνες με ανάλυση ανά μάτι 1080 × 1200, συχνότητα στα 90 Hz, εντοπισμό θέσης 360 μοιρών, ενσωματωμένο ήχο, πολύ αυξημένο όγκο εντοπισμού θέσης και μεγάλη προσοχή

την εργονομία και την αισθητική του καταναλωτή. Τον Μάρτιο του 2019, κατά την ανακοίνωση του Rift S, ειπώθηκε ότι το Rift S θα αντικαταστήσει το αρχικό Rift. Ωστόσο, η Oculus VR δήλωσε ότι σχεδίαζε να υποστηρίξει το CV1 με ενημερώσεις λογισμικού για "το προβλέψιμο μέλλον". Στις 21 Μαΐου 2019 η Oculus άρχισε τη διανομή ενός νέου VR headset γνωστό ως Rift S. Το Rift S διαθέτει οθόνη LCD 2,560 × 1,440 στα 80Hz και λίγο μεγαλύτερο οπτικό πεδίο από αυτό του CV1 αλλά δεν διαθέτει μηχανική ρύθμιση IPD (μόνο μέσω software). Το Rift S παρακολουθεί τη θέση του εαυτού του και τους controllers του σε τρισδιάστατο χώρο χρησιμοποιώντας ένα σύστημα γνωστό ως Oculus Insight, το οποίο χρησιμοποιεί τις 5 κάμερες του HMD για να παρακολουθεί σημεία στο περιβάλλον και τα υπέρυθρα LED στα χειριστήρια, πληροφορίες από τα επιταχυνσιόμετρο στο HMD και στα controllers, και την όραση του υπολογιστή για να προβλέψουμε ποια είναι η πορεία που το HMD και οι controllers είναι πιο πιθανό να πάρουν. Το Rift S χρησιμοποιεί μια θύρα DisplayPort 1.2 και μία θύρα USB 3.0, σε αντίθεση με τη θύρα HDMI και USB 3.0 που χρησιμοποιείται στο Rift CV1.

Περιεχόμενο για το Rift αναπτύσσεται χρησιμοποιώντας το Oculus PC SDK, ένα δωρεάν ιδιόκτητο SDK διαθέσιμο για τα Microsoft Windows (η υποστήριξη OSX και Linux προγραμματίζεται για το μέλλον). Πρόκειται για ένα πλήρες SDK χαρακτηριστικών που χειρίζεται για τον προγραμματιστή τις διάφορες πτυχές της δημιουργίας περιεχομένου εικονικής πραγματικότητας, όπως η οπτική παραμόρφωση και οι προηγμένες rendering techniques. Το Oculus SDK ενσωματώθηκε άμεσα στις δημοφιλείς μηχανές παιχνιδιών Unity 5, Unreal Engine 4 και Cryengine. Αυτό επιτρέπει στους προγραμματιστές που είναι ήδη εξοικειωμένοι με αυτές τις μηχανές να δημιουργούν περιεχόμενο VR με ελάχιστη έως καθόλου χρήση καινούριου κώδικα VR. Το Rift είναι μια ανοιχτή πλατφόρμα και έτσι οι προγραμματιστές δεν χρειάζονται καμία έγκριση ή επαλήθευση για να αναπτύξουν, να διανείμουν ή να πουλήσουν περιεχόμενο γι 'αυτό και δεν χρειάζεται να πληρώνουν τέλη αδειοδότησης. Το SDK, ωστόσο, δεν μπορεί να τροποποιηθεί ή να χρησιμοποιηθεί ξανά για άλλους σκοπούς ή υλικό χωρίς άδεια. Το περιεχόμενο που αναπτύχθηκε για το κιτ ανάπτυξης 2 χρησιμοποιώντας SDK έκδοση 0.8 ή παραπάνω είναι συμβατό με το Rift. Ωστόσο, το περιεχόμενο που αναπτύχθηκε για το Development Kit 1 ή με παλαιότερες εκδόσεις του SDK θα πρέπει να ανασυσταθεί χρησιμοποιώντας την τελευταία έκδοση του SDK για να είναι συμβατή. Στις 21 Δεκεμβρίου 2015, η Oculus ανακοίνωσε την κυκλοφορία του ολοκληρωμένου Rift 1.0 SDK, σε συνδυασμό με την αρχή της αποστολής της τελικής έκδοσης του ακουστικού Oculus Rift VR σε προγραμματιστές. Στο 3ο ετήσιο συνέδριο της Oculus (Oculus Connect 3) ανακοίνωσε τη νέα τεχνολογία Asynchronous Spacewarp (ASW). Αυτή η τεχνολογία επιτρέπει στο Rift να αντισταθμίσει τα πτώση των πλαισίων. Σύμφωνα με τον Oculus, η ASW μειώνει τις ελάχιστες προδιαγραφές ενός υπολογιστή για να τρέξει το Rift χωρίς να έχει τρέμουλο.

Oculus Touch είναι ένα σύστημα ελέγχου κίνησης που σχεδιάστηκε εξαρχής με την ιδέα της εικονικής πραγματικότητας (VR). Κάθε Oculus Touch αποτελείται από ένα ζευγάρι controllers, ένα για κάθε χέρι, το οποίο ουσιαστικά λειτουργεί σαν ένα ενιαίο gamepad που έχει χωριστεί στη μέση. Αυτό επιτρέπει στο Oculus Rift να παρέχει πλήρη παρακολούθηση των κινήσεων των παικτών σε VR. Οι controllers αφής Oculus Touch χρησιμοποιούνται και σαν controllers βιντεοπαιχνιδιών , με το πλήρες σύνολο αναλογικών stick, κουμπιών και σκανδάλων που απαιτούνται για να παίζουν οι χρήστες σύγχρονα παιχνίδια. Το Oculus Touch

συνδυάζει την παραδοσιακή λειτουργικότητα του game controller με την τεχνολογία εντοπισμού κίνησης που βρίσκεται στο Oculus Rift. Κάθε controller περιλαμβάνει ένα analog thumbstick παρόμοιο με εκείνο που υπάρχει σε άλλους σύγχρονους game controllers, δύο κουμπιά που μπορούν επίσης να πατηθούν με τον αντίχειρα, μια σκανδάλη που έχει σχεδιαστεί για τον δείκτη και μια δεύτερη σκανδάλη που ενεργοποιείται με συμπίεση των υπόλοιπων δακτύλων στη λαβή του controller. Εκτός από τα τυποποιημένα χειριστήρια παιχνιδιών, κάθε controller έχει επίσης έναν αριθμό χωρητικών αισθητήρων που είναι σε θέση να εντοπίζουν πού βρίσκονται τα δάχτυλα του παίκτη. Για παράδειγμα, ο ελεγκτής μπορεί να δείξει εάν ο δείκτης βρίσκεται ή όχι στη σκανδάλη και αν ο αντίχειρας βρίσκεται ή όχι σε ένα κουμπί ή σε ένα thumbstick. Αυτό επιτρέπει σε έναν παίκτη να δείξει με το εικονικό δάχτυλό του, να σφίξει χέρι του σε γροθιά και πολλά άλλα. Κάθε controller Oculus Touch είναι επίσης γεμάτος με αυτό που ο Oculus VR ονομάζει έναν «αστερισμό των LED» (a constellation of LEDs) που είναι αόρατα με γυμνό μάτι, όπως και το Oculus Rift. Αυτές οι λυχνίες LED επιτρέπουν στους αισθητήρες αστερισμού Oculus VR να παρακολουθούν τη θέση κάθε controller, η οποία επιτρέπει στον παίκτη να κινεί τα χέρια του και να τα περιστρέφει μέσα από ένα πλήρες εύρος κινήσεων.

Το Oculus Touch έχει εντοπισμό πλήρους κίνησης με έξι βαθμούς ελευθερίας (6DoF), που σημαίνει ότι μπορεί να παρακολουθεί κάθε ένα από τα χέρια σας που κινούνται προς τα εμπρός και πίσω, αριστερά και δεξιά, πάνω-κάτω και επίσης αισθάνεται περιστροφή σε κάθε έναν από αυτούς τους τρεις άξονες. Κάθε controller περιλαμβάνει επίσης χαρακτηριστικά γνώριμα σε παίχτες κονσολών, συμπεριλαμβανομένων δύο analog sticks, τεσσάρων πλήκτρων και δύο σκανδάλων. Αυτός είναι περίπου ο ίδιος αριθμός κουμπιών και σκανδάλων όπως ένας controller DualShock 4 (PS 4) ή Xbox One (Xbox). Η κύρια διαφορά μεταξύ της διαμόρφωσης του Oculus Touch και των παραδοσιακών παιχνιδιών είναι ότι δεν υπάρχει d-pad στον controller και τα κουμπιά χωρίζονται μεταξύ των δύο controllers αντί να είναι όλα προσβάσιμα από τον ίδιο αντίχειρα.

Microsoft HoloLens

Το Microsoft HoloLens, γνωστή ως Project Baraboo στα χρόνια που αναπτύσσονταν, είναι ένα ζευγάρι smartglasses που έχουν αναπτυχθεί και κατασκευαστεί από τη Microsoft. Το HoloLens ήταν η πρώτη οθόνη που έτρεχε την πλατφόρμα Windows Mixed Reality στο λειτουργικό σύστημα Windows 10. Η τεχνολογία παρακολούθησης (tracking) που χρησιμοποιείται στο HoloLens έχει την καταγωγή της στο Kinect, ένα πρόσθετο για την κονσόλα παιχνιδιών Xbox της Microsoft, το οποίο εισήχθη το 2010. Η έκδοση προ-παραγωγής του HoloLens "Development Edition" που στόχευε σε αγοραστικό κοινό προγραμματιστών, ήταν διαθέσιμη στις 30 Μαρτίου 2016, στις Ηνωμένες Πολιτείες και τον Καναδά. Η Samsung και η Asus έχουν δώσει μια προσφορά στη Microsoft για να βοηθήσουν στην παραγωγή των δικών τους προϊόντων mixed-reality, σε συνεργασία με τη Microsoft, που βασίζονται στο concept και το υλικό της HoloLens. Στις 12 Οκτωβρίου 2016, η Microsoft ανακοίνωσε την παγκόσμια επέκταση του HoloLens και ανακοίνωσε ότι θα είναι διαθέσιμο για προπαραγγελία στην Αυστραλία, την Ιρλανδία, τη Γαλλία, τη Γερμανία, τη Νέα Ζηλανδία και το Ηνωμένο Βασίλειο. Υπάρχει επίσης μια εμπορική σουίτα (παρόμοια με μια επαγγελματική έκδοση των Windows), με λειτουργίες όπως το bitlocker security.

Το HoloLens είναι ένα head-mounted display που συνδέεται με ένα ρυθμιζόμενο εσωτερικό headband, που μπορεί να στραφεί προς τα πάνω και προς τα κάτω, καθώς και προς τα εμπρός και προς τα πίσω. Για να φορέσει τη συσκευή, ο χρήστης προσαρμόζει τα HoloLens στο κεφάλι του, χρησιμοποιώντας έναν τροχό ρύθμισης στο πίσω μέρος του headset, ενώ μπορεί να ρυθμίσει την κλίση της οθόνης στη μπροστινή μεριά της συσκευής. Στο μπροστινό τμήμα της μονάδας υπάρχουν πολλοί από τους αισθητήρες και το σχετικό hardware, συμπεριλαμβανομένων των επεξεργαστών, των καμερών και των φακών προβολής. Η μάσκα είναι χρωματισμένη, ενώ ενσωματώνει ένα ζευγάρι διάφανων φακών, στους οποίους οι προβαλλόμενες εικόνες εμφανίζονται στο κάτω μισό. Τα HoloLens πρέπει να ρυθμιστούν ως προς την απόσταση των ματιών του χρήστη. Κατά μήκος των κάτω άκρων των πλευρών, κοντά στα αυτιά του χρήστη, είναι ένα ζευγάρι των μικρών, κόκκινων ηχείων 3D ήχου. Τα ηχεία, που ανταγωνίζονται τα τυπικά ηχητικά συστήματα, δεν εμποδίζουν τους εξωτερικούς ήχους, επιτρέποντας έτσι στο χρήστη να ακούει εικονικούς ήχους, μαζί με τους εξωτερικούς ήχους. Το HoloLens παράγει binaural ήχο, το οποίο συμβάλλει στην προσομοίωση του ήχου στο χώρο, που σημαίνει ότι ο χρήστης, ουσιαστικά, μπορεί να αντιληφθεί και να εντοπίσει έναν ήχο, όταν έρχεται από μια εικονική τοποθεσία.

Στην επάνω άκρη υπάρχουν δύο ζεύγη κουμπιών: κουμπιά φωτεινότητας πάνω από το αριστερό αυτί και πλήκτρα έντασης πάνω από το δεξί αυτί. Τα γειτονικά κουμπιά έχουν διαφορετικό σχήμα - ένα κοίλο, ένα κυρτό - έτσι ώστε ο χρήστης να τα διακρίνει με το άγγιγμα. Στο τέλος του αριστερού βραχίονα είναι ένα κουμπί τροφοδοσίας και μια σειρά από πέντε, μικρές μεμονωμένες λάμπες LED, που χρησιμοποιούνται για την ένδειξη της κατάστασης του συστήματος, καθώς και για τη διαχείριση ενέργειας, που δείχνει τη στάθμη της μπαταρίας και τη ρύθμιση της κατάστασης ισχύος / αναμονής. Μια υποδοχή micro-B USB 2.0 βρίσκεται κατά μήκος της κάτω πλευράς. Ένας υποδοχέας ήχου audio jack 3,5 mm βρίσκεται κατά μήκος του κάτω άκρου του δεξιού βραχίονα. Το HoloLens διαθέτει μια αδρανειακή μονάδα μέτρησης (IMU) (η οποία περιλαμβάνει επιταχυνσιόμετρο, γυροσκόπιο και μαγνητόμετρο), τέσσερις αισθητήρες "κατανόησης του περιβάλλοντος" (δύο σε κάθε πλευρά), μια ενεργειακά αποδοτική κάμερα βάθους με γωνία 120 ° επί 120 ° προβολή, camera 2,4 megapixel, συστοιχία τεσσάρων μικροφώνων και αισθητήρα περιβάλλοντος φωτός. Εκτός από το λογισμικό Intel Cherry Trail SoC που περιέχει τη CPU και τη GPU, η HoloLens διαθέτει ένα Microsoft Holographic Processing Unit της Microsoft (HPU) και έναν coprocessor ειδικά για τα HoloLens από τη Microsoft. Το SoC και το HPU έχουν το καθένα 1GB LPDDR3 και μοιράζονται 8MB SRAM, με το SoC να ελέγχει επίσης το eMMC 64GB και να εκτελεί το λειτουργικό σύστημα Windows 10. Το HPU χρησιμοποιεί 28 προσαρμοσμένους DSP από την Tensilica για να επεξεργαστεί και να ενσωματώσει δεδομένα από τους αισθητήρες, καθώς και να χειριστεί εργασίες όπως η χωρική χαρτογράφηση, η αναγνώριση χειρονομίας και η αναγνώριση φωνής και ομιλίας. Σύμφωνα με τον Alex Kipman, το HPU επεξεργάζεται "terabytes των πληροφοριών", ένας συμμετέχων εκτιμά ότι το οπτικό πεδίο προβολής των μονάδων επίδειξης ήταν 30 ° × 17,5 °. Σε συνέντευξή του στην ηλεκτρονική διασκέδαση Expo του 2015 τον Ιούνιο, ο αντιπρόεδρος της Microsoft, κ. Kudo Tsunoda, δήλωσε ότι το field of view να έχει μεγάλες διαφορές από την κυκλοφορία της τρέχουσας έκδοσης.

Το HoloLens περιέχει μια εσωτερική επαναφορτιζόμενη μπαταρία, με μέση διάρκεια 2-3 ωρών ενεργής χρήσης ή 2 εβδομάδες χρόνου αναμονής. Τα HoloLens μπορούν να

λειτουργούν κατά τη φόρτιση, ενώ διαθέτει ασύρματη συνδεσιμότητα IEEE 802.11ac Wi-Fi και Bluetooth 4.1 Low Energy (LE). Το σετ μικροφώνου-ακουστικού χρησιμοποιεί το Bluetooth LE για τη σύνδεση με το Clicker, μια συσκευή εισόδου που λειτουργεί με τον αντίχειρα και μπορεί να χρησιμοποιηθεί για selecting και scrolling. Το Clicker διαθέτει μια clickable επιφάνεια για επιλογή και έναν αισθητήρα προσανατολισμού ο οποίος παρέχει λειτουργίες κύλισης μέσω κλίσης και περιστροφής της μονάδας. Το Clicker διαθέτει ένα ελαστικό δαχτυλίδι για να στηρίζεται στο χέρι και μια υποδοχή micro-B USB 2.0 για τη φόρτιση της μπαταρίας. Το HoloLens, μέσω της χρήσης του HPU, χρησιμοποιεί εντολές μέσω της αφής και φυσικές εντολές διεπαφής - βλέμματος, χειρονομίας και φωνής - που μερικές φορές αναφέρονται ως "GGV" inputs. Οι εντολές Gaze, όπως το head-tracking, επιτρέπουν στην εφαρμογή να εστιάσει σε ό,τι αντιλαμβάνεται ο χρήστης. Ο χρήστης μπορεί να επιλέξει διάφορα εικονικά στοιχεία με τη μέθοδο "air tap", η οποία μοιάζει με το κλασικό κλικ του ποντικιού . Αυτό το νέου τύπου κλικάρισμα μπορεί να χρησιμοποιηθεί για προσομοίωση drag and drop ενός στοιχείου, επίσης γίνεται χρήση και φωνητικών εντολών για συγκεκριμένες εντολές και ενέργειες. Το shell του HoloLens διαθέτει πολλά στοιχεία από το περιβάλλον εργασίας των Windows. Μια χειρονομία για την πρόσβαση στο shell (αντίστοιχη με το πάτημα του πλήκτρου των Windows σε ένα πληκτρολόγιο ή το κουμπί Xbox σε ένα controller Xbox One) εκτελείται ανοίγοντας το χέρι του, ενώ τα δάχτυλα απλώνονται με την παλάμη στραμμένη προς τα πάνω . Τα παράθυρα μπορούν να μεταφερθούν σε μια συγκεκριμένη θέση, καθώς και να αλλάξουν το μέγεθός τους. Τον Απρίλιο του 2016 η Microsoft δημιούργησε την εφαρμογή Microsoft HoloLens App για Windows 10 PC και Windows 10 Mobile, η οποία επιτρέπει στους προγραμματιστές να εκτελούν εφαρμογές, να χρησιμοποιούν το τηλέφωνο ή το πληκτρολόγιο του υπολογιστή για να πληκτρολογούν κείμενο, να βλέπουν ένα live stream από την οπτική γωνία του χρήστη που χρησιμοποιεί τα HoloLens, και να τραβάει εξ αποστάσεως mixed reality φωτογραφίες και βίντεο.

Κεφάλαιο 3: Πλατφόρμες τρισδιάστατων γραφικών

Web3D

Το Web3D είναι μια πλατφόρμα ψηφιακών μέσων σχεδιασμένη συγκεκριμένα για τρισδιάστατο περιεχόμενο και για άλλα πλούσια (rich) διαδραστικά μέσα. Υλοποιημένο με τη χρήση παραδοσιακών διαδικτυακών τεχνολογιών, το Web3D καθιστά δυνατή την ανάπτυξη τρισδιάστατων καινοτόμων εφαρμογών όπως διαδραστικές ταινίες, στερεοσκοπικό κινηματογράφο, εικονική πραγματικότητα, παιχνίδια, εφαρμογές τηλεϊατρικής, κλπ. Η χρήση της πλατφόρμας Web3D υποστηρίζει ιδιωτικούς (proprietary) και ανοιχτούς τύπους μέσων για την ανάπτυξη rich media εφαρμογών δημιουργημένες από τεχνολογίες τύπου Flash, Shockwave, VRML, Java3D, X3D, MPEG-4 και προσφάτως JavaScript, WebGL, κλπ.

Ο Παγκόσμιος Ιστός ξεκίνησε τη ζωή του στα τέλη της δεκαετίας του 1980 ως ένα σχετικά απλό σύστημα υπερκειμένων και έχει ωριμάσει όλα αυτά τα χρόνια για να γίνει ένα ζωτικής σημασίας κατανεμημένο σύστημα πολυμέσων που καλύπτει τον πλανήτη. Καθώς το ίδιο το διαδίκτυο έχει αυξηθεί, έχει αυξηθεί και ο πλούτος αλλά και η πολυπλοκότητα των ψηφιακών μέσων που υποστηρίζει. Σήμερα, πολλά προγράμματα περιήγησης Web υποστηρίζουν εγγενώς βασικές μορφές μέσων όπως ψηφιακό ήχο (π.χ. WAV και MIDI), εικόνες bitmap (π.χ. GIF, JPEG και PNG) και ακόμη διανυσματικά γραφικά (π.χ. SVG και VML).

Οι πιο εξελιγμένες μορφές δεδομένων και οι ιδιόκτητοι τύποι μέσων (private types) σπάνια υποστηρίζονται απ 'ευθείας από το πρόγραμμα περιήγησης και απαιτούν έναν αντίστοιχο "player" που έρχεται με τη μορφή plug-in, εφαρμογής applet ή βοηθητικής εφαρμογής που μπορεί να αποκωδικοποιήσει περιεχόμενο και να αλληλεπιδρά με τον χρήστη απευθείας μέσα στο παράθυρο του προγράμματος περιήγησης. Επειδή τα προγράμματα περιήγησης ιστού μπορούν εύκολα να επεκταθούν με αυτό τον τρόπο, μια τεράστια ποικιλία νέων τύπων ψηφιακών μέσων έχει εισαχθεί κατά τη διάρκεια των ετών που δεν θα ήταν βιώσιμη αν οι προμηθευτές του προγράμματος περιήγησης ήταν υποχρεωμένοι να υποστηρίζουν αυτό το περιεχόμενο εγγενώς.

Αν και η συντριπτική πλειοψηφία των οπτικών μέσων που ήταν διαθέσιμα στον παραδοσιακό ιστό ήταν δισδιάστατη, έχοντας μόνο τις διαστάσεις ύψους και πλάτους, το περιεχόμενο 3D που έχει την πρόσθετη διάσταση βάθους γίνεται όλο και πιο δημοφιλές. Το 3D για τον Παγκόσμιο Ιστό ή το Web3D δεν είναι κάτι νέο. Όταν η ανάπτυξη της Virtual Reality Modeling Language (VRML) ξεκίνησε το 1994 για να δημιουργήσει μια τρισδιάστατη εναλλακτική λύση στη δισδιάστατη HyperText Markup Language (HTML), πυροδότησε τη φαντασία των web developers και τεχνολόγων γύρω από το έργο, πυροδοτώντας φωτιά της διαφημιστικής εκστρατείας ότι η ελπιδοφόρα νέα τεχνολογία δεν θα μπορούσε να ανταποκριθεί. Το όνειρο ενός ιστού Web3D ήταν εκθαμβωτικό, εντυπωσιακό και μακριά. Εκείνη την εποχή, η μεγάλη πλειοψηφία των τελικών χρηστών δεν είχε την απαιτούμενη ισχύ από τον υπολογιστή και το εύρος ζώνης του δικτύου από το πιο συναρπαστικό περιεχόμενο 3D. Τα εργαλεία συγγραφής VRML της ημέρας ήταν ελλιπή και

αρκετά πρωτόγονα σε σύγκριση με τα εργαλεία συγγραφής 3D που είναι διαθέσιμα τώρα. Συνεπώς, ο μέσος χρήστης του Διαδικτύου ήταν σκληρά πιεσμένος για να βρει υψηλής ποιότητας VRML καθ 'όλη τη διάρκεια της δεκαετίας του 1990, υποβιβάζοντας την τεχνολογία σε μια εξειδικευμένη αγορά στην καλύτερη περίπτωση. Το 1997, το VRML έγινε το πρώτο Διεθνές Πρότυπο για το 3D στο Διαδίκτυο και ενέπνευσε μια σειρά από ιδιόκτητες τεχνολογίες Web3D τα τελευταία χρόνια. Παρά αυτές τις πρόωρες τεχνολογικές εξελίξεις, ο παγκόσμιος, παγκοσμίως προσβάσιμος Web3D Web παρέμεινε ένα όνειρο μέχρι τώρα.

Μετά από σχεδόν μια δεκαετία σκληρής δουλειάς από χιλιάδες αφοσιωμένους ανθρώπους και πολλές επιχειρήσεις λογισμικού και υλικού, ο Web3D Web είναι τελικά εφικτός. Τα εμπόδια που εμπόδιζαν το Web3D Web να γίνει πραγματικότητα στο παρελθόν έχουν εξαλειφθεί σταθερά με τα χρόνια. Οι σημερινοί desktop υπολογιστές είναι αρκετά γρήγοροι για να χειριστούν τους τύπους πλούσιου περιεχομένου 3D που έφεραν την προηγούμενη γενιά των υπολογιστών σε αδράνεια. Η ευρυζωνικότητα είναι υπαρκτή πλέον σε σπίτια και γραφεία, και τα εργαλεία συγγραφής περιεχομένου 3D είναι ισχυρά, άφθονα και οικονομικά προσιτά. Με την πάροδο των ετών αυτά τα κρίσιμα ζητήματα έχουν κυριολεκτικά γίνει μη-ζητήματα, ενώ ταυτόχρονα έχει προκύψει μια βασική σειρά τεχνολογικών προτύπων 3D που μαζί αποτελούν τη ραχοκοκαλιά του Web3D. Χάρη σε αυτές τις προόδους, οι σημερινοί δημιουργοί περιεχομένου μπορούν τώρα να δημιουργήσουν συναρπαστικές και συγκλονιστικές εμπειρίες 3D που ο μέσος χρήστης του Διαδικτύου μπορεί πραγματικά να απολαύσει, σηματοδοτώντας μια κρίσιμη στιγμή στην ιστορία του Web3D.

VRML

Η Virtual Reality Modeling Language (VRML) ήταν μια γλώσσα για την περιγραφή διαδραστικών προσομοιώσεων πολλαπλών συμμετεχόντων - εικονικών κόσμων που συνδέονται μέσω του παγκόσμιου Διαδικτύου με τον Παγκόσμιο Ιστό (World Wide Web). Όλες οι πτυχές της απεικόνισης, της αλληλεπίδρασης και της δικτύωσης με τον εικονικό κόσμο μπορούν να καθοριστούν χρησιμοποιώντας VRML. Ήταν η πρόθεση των σχεδιαστών της να καταστεί η VRML η τυπική γλώσσα για διαδραστική προσομοίωση στο World Wide Web.

Η πρώτη έκδοση της VRML επιτρέπει τη δημιουργία εικονικών κόσμων με περιορισμένη διαδραστική συμπεριφορά. Οι κόσμοι αυτοί μπορούν να περιέχουν αντικείμενα που έχουν υπερ-συνδέσεις με άλλους κόσμους, έγγραφα HTML ή άλλα έγκυρα MIME types. Όταν ο χρήστης επιλέξει ένα αντικείμενο με μια υπερ-σύνδεση, ανοίγει το κατάλληλο πρόγραμμα προβολής MIME. Όταν ο χρήστης επιλέξει μια σύνδεση με ένα αρχείο VRML από ένα πρόγραμμα περιήγησης (browser), ένα πρόγραμμα προβολής VRML εκκινείται. Οι VRML viewers είναι οι κατάλληλες εφαρμογές σύντροφοι σε τυπικούς browsers WWW για πλοήγηση και οπτικοποίηση του ιστού. Οι μελλοντικές εκδόσεις του VRML επιτρέπουν πλουσιότερες συμπεριφορές, συμπεριλαμβανομένων animations, φυσική κίνησης και αλληλεπίδραση πολλών χρηστών σε πραγματικό χρόνο.

Η σύλληψη της VRML έγινε την άνοιξη του 1994 στην πρώτη ετήσια διάσκεψη του World Wide Web στη Γενεύη της Ελβετίας. Ο Tim Berners-Lee και ο Dave Raggett διοργάνωσαν μια συνάντηση Birds of a Feather (BOF) για να συζητήσουν τις διεπαφές Εικονικής Πραγματικότητας με τον Παγκόσμιο Ιστό. Αρκετοί συμμετέχοντες του BOF περιέγραψαν έργα που βρίσκονται ήδη σε εξέλιξη για την κατασκευή εργαλείων γραφικής απεικόνισης τρισδιάστατων γραφικών τα οποία αλληλεπιδρούν με τον Ιστό. Οι συμμετέχοντες συμφώνησαν σχετικά με την ανάγκη τα εργαλεία αυτά να έχουν μια κοινή γλώσσα για τον προσδιορισμό της περιγραφής 3D στον κόσμο και των υπερ-συνδέσεων του WWW - ένα ανάλογο του HTML για την εικονική πραγματικότητα. Ο όρος Virtual Language

Markup Language (VRML) δημιουργήθηκε και η ομάδα αποφάσισε να ξεκινήσει να εξειδικεύεται μετά το συνέδριο. Η λέξη «Markup» μετατράπηκε αργότερα σε «Modeling» για να αντικατοπτρίζει τη γραφική φύση της VRML.

Λίγο μετά τη σύνοδο της Γενεύης BOF, δημιουργήθηκε η ομάδα αλληλογραφίας www-vrml για να συζητηθούν οι προδιαγραφές για την πρώτη έκδοση του VRML. Η ανταπόκριση στην πρόκληση ήταν τεράστια καθώς μέσα σε μια εβδομάδα, ο αριθμός των μελών ξεπέρασε τα χίλια. Μετά από μια αρχική περίοδο προσαρμογής, ο συντονιστής της ομάδας Mark Pesce του Labyrinth Group ανακοίνωσε την πρόθεσή του για το σχεδιασμό ενός προσχεδίου έκδοσης των προδιαγραφών από το συνέδριο WWW Fall 1994, με χρονοδιάγραμμα μόλις πέντε μηνών. Υπήρξε γενική συμφωνία της ομάδας ότι, ενώ αυτό το πρόγραμμα ήταν επιθετικό μεν, ήταν εφικτό δε υπό την προϋπόθεση ότι οι απαιτήσεις για την πρώτη έκδοση δεν ήταν υπερβολικά φιλόδοξες και ότι η VRML θα μπορούσε να προσαρμοστεί από μια υπάρχουσα λύση. Η ομάδα γρήγορα συμφώνησε σε ένα σύνολο απαιτήσεων για την πρώτη έκδοση, και άρχισε η έρευνα για τεχνολογίες που θα μπορούσαν να προσαρμοστούν ώστε να ταιριάξουν στις ανάγκες του VRML. Μετά από διαβούλευση κατέληξαν στο Open Inventor ASCII File Format της Silicon Graphics Inc.

Στα βασικά χαρακτηριστικά της γλώσσας τα αντικείμενα μπορεί να περιέχουν οτιδήποτε, 3D geometries, MIDI data, εικόνες JPEG, κλπ. Η VRML ορίζει μια ομάδα αντικειμένων ικανών να υλοποιήσουν τρισδιάστατα γραφικά, τους Κόμβους (Nodes). Οι Κόμβοι είναι διατεταγμένοι σε ιεραρχικές δομές
τους γράφους σκηνών (scene graphs). Τα scene graphs δεν είναι απλά μια ομάδα από κόμβους, αλλά ορίζουν τη διάταξη των κόμβων. Τα scene graphs έχουν την έννοια της κατάστασης, πρωτύτεροι κόμβοι του εικονικού κόσμου επηρεάζουν τη συμπεριφορά επόμενων κόμβων που εισάγονται στον κόσμο αργότερα. Οι κόμβοι έχουν τα κάτωθι χαρακτηριστικά, είδος αντικειμένου, πεδία (fields), όνομα, κόμβοι-παιδιά. Ένας κόμβος μπορεί να είναι κύβος, σφαίρα, χάρτης υφής (texture map), κλπ. Τα πεδία είναι παράμετροι για να ξεχωρίζουν οι κόμβοι ίδιου τύπου μεταξύ τους. Για παράδειγμα κάθε κόμβος με σχήμα σφαίρας έχει διαφορετική διάμετρο και texture. Ένας κόμβος μπορεί να μην έχει κανένα πεδίο. Κάθε κόμβος έχει ένα μόνο, αλλά όχι μοναδικό, όνομα που επιτρέπει στον προγραμματιστή να τον χρησιμοποιήσει.
Πολλοί κόμβοι μπορούν να έχουν το ίδιο όνομα.

X3D

Το X3D (extensible 3D) είναι ο επίσημος διάδοχος της VRML στο οποίο είχε εστιάσει την προσοχή του της κοινοπραξίας Web3D τα προηγούμενα χρόνια. Το X3D βελτιώνει τη VRML προσθέτοντάς της νέα γνωρίσματα, εξελιγμένα προγραμματιστικά περιβάλλοντα, επιπρόσθετες δομές κωδικοποίησης, τμηματοποιημένη αρχιτεκτονική που επιτρέπει σε μια πιο τμηματοποιημένη προσέγγιση για την υποστήριξη του προτύπου. Πλέον το X3D κυρώθηκε με τα στάνταρντ ISO, και προσφέρει αποθήκευση, ανάκτηση και αναπαραγωγή περιεχομένου τρισδιάστατων γραφικών σε πλατφόρμες και παίχτες, εφαρμογών PC, mobile εφαρμογών, browser pulgins, και άμεσο rendering σε σελίδες HTML5 χρησιμοποιώντας βιβλιοθήκες JavaScript. Οι γράφοι σκηνών (scene graphs) του X3D μπορούν να αποθηκευτούν και να μεταδοθούν σε αρκετές κανονικοποιημένες κωδικοποιήσεις, συμπεριλαμβανομένων της κλασικής VRML, XML, και Binary format. Καθορίζεται ένα κανονικοποιημένο API, επιτρέποντας στο γράφημα σκηνής (scene graph) να το χειριστούν πολλές γλώσσες προγραμματισμού.

Το μοντέλο αντικειμένου και κόμβου του X3D θέτει όλα τα χαρακτηριστικά υποστηρικτικών γραφικών πού οι σύγχρονες διαδραστικές εφαρμογές τρισδιάστατων

γραφικών χρειάζονται, από ανταλλαγή αντικειμένων μέχρι animation και αισθητήρες σε συστήματα εμβύθισης. Το πρότυπο έχει οργανωθεί σε λειτουργικά στοιχεία τα οποία παρέχουν εκτεταμένη λειτουργικότητα περιλαμβάνοντας πολλαπλούς τύπους γεωμετρίας, εμφανίσεις υλικών (materials), textures, φωτισμούς, σκιάσεις, animation, διάδραση του χρήστη βασισμένη στην αφή, scripting, rigid body physics, μεθόδους volume rendering, metadata και CAD assembly structure. Η κανονικοποίηση-τυποποίηση αυτή επιτρέπει σε συγγραφείς και προγραμματιστές να συνεργαστούν έχοντας την απαιτούμενη λειτουργικότητα και υποστήριξη, επιτρέποντας τη χρήση του X3D σε ένα φάσμα από απλούς μέχρι πολύ απαιτητικούς πελάτες-χρήστες. Οι developers μπορούν μέσω scripting να επεκτείνουν τη λειτουργικότητα των κόμβων και των στοιχείων του X3D.

Η αξία του X3D έγκειται στις εξής παραμέτρους. Επιτρέπει στις εφαρμογές να επικοινωνούν μέσω του web χρησιμοποιώντας πιστοποιημένα με ISO μοντέλα γραφήματος σκηνής (scene graph model), κωδικοποιημένα σε πολλαπλά formats (XML, VRML, JSON, Binary, κλπ) και προγραμματιζόμενα σε διάφορες γλώσσες (JavaScript, Java, κλπ). Είναι τμηματοποιήσιμο (modular) και επεκτάσιμο, κερδίζοντας χρόνο και χρήμα, προσδίδοντας αξία στον πωλητή και τον καταναλωτή και με τα μοντέλα να παραμένουν ορατά και εκτελέσιμα. Είναι δωρεάν, ανοιχτού κώδικα και σταθερό, κάτι που ενθαρρύνει την ανάπτυξη νέων εργαλείων προς ικανοποίηση των, συνεχώς μεταβαλλόμενων, αναγκών των συγγραφέων-developers. Επειδή έχει αναπτυχθεί μέσω του community και της βιομηχανίας προσφέρει ταυτόχρονα εμπορική και open-source υποστήριξη.

Το X3D με την μεγάλη γκάμα από λειτουργίες που διαθέτει, μπορεί να εξατομικευθεί για χρήση σε πολλές και διαφορετικές πλατφόρμες. Ενδεικτικά χρησιμοποιείται από κυβερνήσεις της Ευρωπαϊκής Ένωσης, τις Ηνωμένες Πολιτείες της Αμερικής, την Αυστραλία, από οργανισμούς όπως η NASA, το Πολεμικό Ναυτικό των Ηνωμένων Πολιτειών, πανεπιστήμια σε όλη την υφήλιο και από κατασκευαστικές εταιρίες και ενεργειακούς κολοσσούς μέχρι εταιρίες πληροφορικής. Αποτελεί τον μεγαλύτερο κοινό παρονομαστή διαφόρων τομέων καθώς χρησιμοποιείται στην αρχιτεκτονική, στην ιατρική, στην τοπογραφική απεικόνιση, κλπ, ενώ πρόσφατα άρχισε να χρησιμοποιείται στα κινητά, σε πλατφόρμες εικονικής πραγματικότητας και σε τρισδιάστατους εκτυπωτές και εφαρμογές τρισδιάστατης σάρωσης.

WebGL

Το WebGL είναι ένα JavaScript Api που έχει σχεδιαστεί και συντηρείται από το μη-κερδοσκοπικό οργανισμό Khronos Group. Χρησιμοποιείται για αναπαράσταση και απόδοση διαδραστικών 3D και 2D γραφικών σε οποιοδήποτε συμβατό web browser χωρίς την χρήση plugins. Έχει ενσωματωθεί πλήρως σε όλα τα web πρότυπα του προγράμματος περιήγησης, επιτρέποντας στη GPU να κάνει χρήση της φυσικής, της επεξεργασίας εικόνας και κάποιων εφέ ως μέρος του καμβά ιστοσελίδας. Τα στοιχεία του WebGL μπορούν να αναμιχθούν με άλλα στοιχεία HTML και με άλλα τμήματα της σελίδας ή του φόντου της σελίδας. Τα WebGL προγράμματα αποτελούνται από κώδικα γραμμένο σε JavaScript και από κώδικα (shader) που εκτελείται στην μονάδα επεξεργασίας γραφικών του υπολογιστή (GPU).

Το WebGL εξελίχθηκε από τα πειράματα 3D Canvas που ξεκίνησε ο Vladimir Vukićević στο Mozilla . Ο Vukićević παρουσίασε για πρώτη φορά το πρωτότυπο Canvas 3D το 2006. Μέχρι τα τέλη του 2007, τόσο τα Mozilla όσο και η Opera είχαν κάνει ξεχωριστές εφαρμογές. Στις αρχές του 2009, η μη κερδοσκοπική τεχνολογική κοινοπραξία Khronos Group ξεκίνησε την ομάδα εργασίας WebGL, με αρχική συμμετοχή από την Apple , Google , Mozilla, Opera και άλλους. Η έκδοση 1.0 της προδιαγραφής WebGL κυκλοφόρησε τον Μάρτιο του 2011. Από τον Μάρτιο του 2012, ο πρόεδρος της ομάδας εργασίας είναι ο Ken Russell. Οι πρώτες εφαρμογές του WebGL περιλαμβάνουν το Zygote Body . Το Νοέμβριο

του 2012, η Autodesk ανακοίνωσε ότι μεταφέρει τις περισσότερες από τις εφαρμογές τους στο cloud που εκτελείται σε τοπικούς πελάτες WebGL. Αυτές οι εφαρμογές περιελάμβαναν το Fusion 360 και το AutoCAD 360. Η ανάπτυξη της προδιαγραφής WebGL 2 ξεκίνησε το 2013 με τελικό τον Ιανουάριο του 2017. Αυτή η προδιαγραφή βασίζεται στο OpenGL ES 3.0. Οι πρώτες υλοποιήσεις είναι στο Firefox 51, Chrome 56 και Opera 43.

Το WebGL 1.0 βασίζεται στο OpenGL ES 2.0 και παρέχει ένα API για γραφικά 3D. Χρησιμοποιεί το στοιχείο καμβά HTML5 και προσπελάζεται χρησιμοποιώντας διεπαφές μοντέλου αντικειμένων εγγράφου (DOM Object Model). Το WebGL 2.0 βασίζεται στο OpenGL ES 3.0 και παρέχει εγγυημένη διαθεσιμότητα πολλών προαιρετικών επεκτάσεων του WebGL 1.0 και εκθέτει νέα API. Όπως το OpenGL ES 2.0, το WebGL δεν διαθέτει τα API σταθερής λειτουργίας που έχουν εισαχθεί στο OpenGL 1.0 και έχουν καταργηθεί στο OpenGL 3.0. Αυτή η λειτουργικότητα, εάν απαιτείται, πρέπει να εφαρμοστεί από τον ίδιο τον τελικό προγραμματιστή, παρέχοντας τον κώδικα shader και τη ρύθμιση των δεσμεύσεων δεδομένων στη γλώσσα JavaScript. Οι Shaders στο WebGL εκφράζονται απευθείας στο GLSL και μεταβιβάζονται στο API WebGL ως textual strings. Το WebGL μεταφράζει αυτές τις οδηγίες shader σε κώδικα GPU. Αυτός ο κώδικας εκτελείται για κάθε κορυφή που στέλνεται μέσω του API και για κάθε εικονοστοιχείο που ραστεροποιείται στην οθόνη. Το WebGL υποστηρίζεται ευρέως στους περισσότερους σύγχρονους browsers στους υπολογιστές στον Chrome, Firefox, Safari, Opera, Internet Explorer και Microsoft Edge.

Flash 10 – Stage3D

Η Adobe Flash είναι μία παρωχημένη πλατφόρμα πολυμέσων που χρησιμοποιούνταν για την παραγωγή animations, rich internet applications, εφαρμογών σε υπολογιστές, εφαρμογών σε κινητά, παιχνίδια και ήταν ενσωματωμένη σε browser video players. Το Flash για να παράσχει animations, παιχνίδια και εφαρμογές χρησιμοποιεί κείμενο, διανυσματικά γραφικά (vector graphics) και raster graphics (bitmap). Έχει τη δυνατότητα για streaming video και ήχου ενώ δέχεται σαν συσκευές εισόδου το ποντίκι, το πληκτρολόγιο, μικρόφωνο και κάμερα. Στις αρχές του 2000 το Flash χρησιμοποιούνταν ευρέως παγκοσμίως, κυρίως για να εμφανίζει διαδραστικές σελίδες, διαδικτυακά παιχνίδια, καθώς και να αναπαράγει οπτικό και ακουστικό υλικό (audio και video). Το 2005 ιδρύεται το YouTube το οποίο , εκείνη την εποχή, χρησιμοποιούσε αποκλειστικά τον Flash Player σαν μέσω παρουσίασης συμπιεσμένου video στον παγκόσμιο ιστό. Στην περίοδο 2000-2010 μεγάλος αριθμός επιχειρήσεων χρησιμοποιούσαν ιστοσελίδες βασισμένες σε Flash για να προωθήσουν προϊόντα ή για να δημιουργήσουν διαδραστικές πύλες μεταξύ εταιριών για επικοινωνία. Η έκδοση του Flash Player 11 εισήγαγε, το 2011 ένα 3D shader API, το Stage3D, το οποίο είναι αρκετά όμοιο με το WebGL. Το Stage3D, επιτρέπει το, επιταχυνόμενο από την κάρτα γραφικών, rendering τρισδιάστατων γραφικών σε παιχνίδια και εφαρμογές Flash, ενώ χρησιμοποιήθηκε για την παραγωγή του Angry Birds και άλλων αξιοσημείωτων παιχνιδιών. Αξίζει να σημειωθεί ότι στις προηγούμενες εκδόσεις του Flash οι τρισδιάστατες εφαρμογές Flash χρησιμοποιούσαν για rendering μόνο τον επεξεργαστή. Οι GPU Shaders στο Stage3D είναι υλοποιημένοι στην Adobe Graphics Assembly Language (AGAL).

O3D

Το O3D είναι ένα API JavaScript JavaScript ανοικτού κώδικα (license BSD) το οποίο δημιουργήθηκε από τη Google για τη δημιουργία διαδραστικών εφαρμογών 3D γραφικών που εκτελούνται σε ένα παράθυρο προγράμματος περιήγησης στο Web ή σε μια εφαρμογή επιφάνειας εργασίας XUL. Το O3D μπορεί να σχεδιαστεί για χρήση σε οποιοδήποτε πεδίο εφαρμογής, ωστόσο απευθύνεται σε παιχνίδια, διαφημίσεις, θεατές μοντέλων 3D, επιδείξεις

προϊόντων, προσομοιώσεις, εφαρμογές μηχανικής, συστήματα ελέγχου και παρακολούθησης ή μαζικούς ηλεκτρονικούς εικονικούς κόσμους. Το O3D βρίσκεται σήμερα στο εργαστήριο ανάπτυξης της Google και αρχικά κατασκευάστηκε ως plugin για προγράμματα περιήγησης στο Web. Η νέα εφαρμογή του O3D είναι μια βιβλιοθήκη JavaScript που υλοποιείται πάνω από το WebGL. Το O3D θεωρείται ότι γεφυρώνει το χάσμα μεταξύ εφαρμογών επιταχυνόμενων γραφικών 3D που βασίζονται σε PC και προγραμμάτων περιήγησης web που βασίζονται σε HTML. Οι υποστηρικτές ισχυρίζονται ότι η δημιουργία μιας πλήρους τρισδιάστατης μηχανής γραφικών που μπορεί να μεταφορτωθεί και να τρέξει μέσα από browsers, ενδέχεται να εξαλείψει την ανάγκη εγκατάστασης μεγάλων εφαρμογών σε έναν υπολογιστή. Αυτό επιτρέπει στο O3D να μεγιστοποιεί την επαναχρησιμοποίηση μεταξύ των πόρων της εφαρμογής, παρέχοντας παράλληλα μια ισχυρή διασύνδεση με την CPU και τη GPU του χρήστη χρησιμοποιώντας JavaScript. Αρχικά, το O3D χρησιμοποίησε μια αρχιτεκτονική βασισμένη σε plug-in, η οποία επέτρεψε σε τρίτους προγραμματιστές να ενσωματώσουν προσαρμοσμένες λειτουργίες όπως τα αποτελέσματα πριν και μετά την εμφάνιση, τα συστήματα σωματιδίων ή και οι μηχανές φυσικής (physics engines), για παράδειγμα. Είναι σημαντικό να σημειωθεί ότι το plugin ήταν γραμμένο σε C που επικοινωνούσε απευθείας με το υλικό, οπότε η ταχύτητα του rendering της σκηνής εξαρτιόταν σε μεγάλο βαθμό από την κάρτα γραφικών του υπολογιστή που την καθιστούσε. Τώρα, μεγάλο μέρος της ίδιας λειτουργικότητας ενσωματώνεται στο WebGL.

Το βασικό πλεονέκτημα που προσφέρει το O3D είναι εναλλακτικές μηχανές 3D rendering βασισμένα σε επιτραπέζιους υπολογιστές ή κονσόλα είναι ότι το O3D μπορεί να φορτώνει, να αποδίδει και να μετατρέπει τα μοντέλα και τις αντίστοιχες υφές τους δυναμικά χρησιμοποιώντας AJAX ή COMET σε πραγματικό χρόνο. Η παραδοσιακή συλλογή πηγαίου κώδικα, πόρων εφαρμογών και βιβλιοθηκών αντικειμένων δεν είναι πλέον απαραίτητη, αφού όλες αυτές οι πτυχές φορτώνονται σε πραγματικό χρόνο. Αυτοί οι απομακρυσμένοι πόροι μπορούν να σχεδιαστούν, να αναπτυχθούν και να διατηρηθούν εκτός της βασικής εφαρμογής απεικόνισης ή προβολής σε μια τυπική εφαρμογή MVC με αντικείμενο. Το άμεσο αποτέλεσμα αυτού καθιστά ευκολότερη την ανάπτυξη rich 3D εφαρμογής, καθώς δεν χρειάζεται να ξαναγίνει compile η εφαρμογή O3D. Αυτό επιτρέπει μια πιο ισχυρή και επιμεριστική προσέγγιση κατά το σχεδιασμό 3D εφαρμογών. Στις 7 Μαΐου του 2010, η Google ανακοίνωσε ότι το O3D θα αλλάξει από plugin σε βιβλιοθήκη JavaScript που τρέχει πάνω από το WebGL.

X3DOM

Το X3DOM είναι ένα JavaScript Framework ανοιχτού κώδικα, που χρησιμοποιείται για τη δημιουργία δηλωτικών τρισδιάστατων σκηνών (declarative 3D scenes) σε web pages, οι οποίες τρέχουν χωρίς κάποιο plugin, γιατί το X3DOM βασίζεται σε ήδη υπάρχουσες συμβατικές τεχνολογίες περιηγητών. Ο όρος “declarative 3D scene” σημαίνει ότι επιτρέπεται η δημιουργία διαδραστικών τρισδιάστατων σκηνών, χρησιμοποιώντας δομημένη κειμενική αναπαράσταση (structutred textual representation), στη θέση του γραπτού κώδικα. Στην περίπτωση του X3DOM αυτή η κειμενική αναπαράσταση είναι κομμάτι του κώδικα HTML του αρχείου της σελίδας. Το X3DOM αποτελείται από δύο γνώριμα συστατικά, το πρώτο είναι το X3D (βλέπε παράγραφο X3D), και το DOM. Τα στοιχεία X3D κατευθύνονται από DOM operations όπως τα υπόλοιπα στοιχεία HTML. Για παράδειγμα, μπορούμε να αλλάξουμε δυναμικά το χρώμα ενός αντικειμένου 3D με μια κλήση JavaScript setAttribute (...) στο αντίστοιχο στοιχείο DOM, ακριβώς όπως θα αλλάζαμε δυναμικά το κείμενο μιας ετικέτας μέσα σε μια κοινή ιστοσελίδα.

Τα πλεονεκτήματα του X3DOM σε σχέση με άλλες βιβλιοθήκες είναι ότι βασίζεται σε τεχνολογίες HTML5 και WebGL. Ένα μεγάλο κομμάτι του X3DOM είναι συμβατό με το

X3D ISO. Αυτό διευκολύνει πολύ όχι μόνο την εκμάθηση του X3DOM, αλλά και την ανταλλαγή περιεχομένου X3DOM. Από την έναρξη της ανάπτυξης το 2009, το community του X3DOM διευρύνεται και σε χρήστες και σε προγραμματιστές. Δε χρειάζεται εξολοκλήρου εκμάθηση από την αρχή αν ήδη γνωρίζει κάποιος σχετικά με το HTML και DOM.

HTML5

Το HTML5 είναι η πέμπτη και τρέχουσα κύρια έκδοση της HTML και περιλαμβάνει το XHTML. Το τρέχον πρότυπο, το HTML Living Standard, αναπτύσσεται από το WHATWG, το οποίο αποτελείται από τους μεγαλύτερους προμηθευτές προγραμμάτων περιήγησης (Apple, Google, Mozilla και Microsoft), με το Living Standard να υπάρχει και σε συντομευμένη έκδοση. Το HTML5 κυκλοφόρησε για πρώτη φορά σε δημόσια μορφή στις 22 Ιανουαρίου 2008, με σημαντική ενημέρωση και κατάσταση "σύστασης του W3C" τον Οκτώβριο του 2014. Στόχοι του ήταν η βελτίωση της γλώσσας με υποστήριξη για τα τελευταία multimedia και άλλα νέα χαρακτηριστικά. να διατηρούν τη γλώσσα τόσο εύκολα κατανοητή από τους ανθρώπους όσο και να είναι κατανοητή από τους υπολογιστές και τις συσκευές, όπως οι browsers, οι parsers κλπ., χωρίς την ακαμψία του XHTML. και να παραμείνουν συμβατά προς τα πίσω με παλαιότερο λογισμικό. Το HTML5 προορίζεται να περιλαμβάνει όχι μόνο HTML 4, αλλά και HTML HTML XHTML 1 και DOM 2. Το HTML5 περιλαμβάνει λεπτομερή μοντέλα επεξεργασίας για την ενθάρρυνση περισσότερων διαλειτουργικών υλοποιήσεων. επεκτείνει, βελτιώνει και εξορθολογίζει τη διαθέσιμη σήμανση για έγγραφα και εισάγει διεπαφές προγραμματισμού σήμανσης και εφαρμογών (API) για σύνθετες εφαρμογές ιστού. Για τους ίδιους λόγους, το HTML5 είναι επίσης υποψήφιο για κινητές εφαρμογές πολλαπλών πλατφορμών, καθώς περιλαμβάνει χαρακτηριστικά σχεδιασμένα με συσκευές χαμηλής κατανάλωσης ενέργειας. Περιλαμβάνονται πολλά νέα συντακτικά χαρακτηριστικά. Για να συμπεριληφθεί και να είναι διαχειρίσιμο περιεχόμενο πολυμέσων και γραφικών, προστέθηκαν τα νέα στοιχεία <video>, <audio> και <canvas> και υποστήριξη για περιεχόμενο SVG και MathML για μαθηματικούς τύπους. Για να εμπλουτιστεί το σημασιολογικό περιεχόμενο των εγγράφων, έχουν προστεθεί νέα στοιχεία δομής σελίδας όπως <main>, <section>, <article>, <header>, <footer>, <aside>, <nav> και <figure>. Εισάγονται νέα χαρακτηριστικά, ορισμένα στοιχεία και ιδιότητες έχουν καταργηθεί και άλλα όπως τα <a>, <cite> και <menu> έχουν αλλάξει ή επαναπροσδιοριστεί. Τα API και το μοντέλο αντικειμένων εγγράφου (DOM) είναι πλέον θεμελιώδη μέρη της προδιαγραφής HTML5 και η HTML5 ορίζει επίσης καλύτερα την επεξεργασία για τυχόν μη έγκυρα έγγραφα.

Το Web Hypertext Application Technology Working Group (WHATWG) ξεκίνησε να δουλεύει πάνω στο νέο πρότυπο το 2004. Εκείνη την εποχή το HTML 4.0 είχε να ανανεωθεί από το 2000, και το World Wide Web Consortium (W3C) είχε επικεντρώσει την προσοχή του στην μελλοντική ανάπτυξη του XHTML 2.0, ιδέα την οποία εγκατέλειψε το 2009. Η Mozilla Foundation και Opera Software παρουσίασαν τις θέσεις τους στο σεμινάριο του W3C τον Ιούνιο του 2004, επικεντρώνοντας την προσοχή τους πάνω στην ανάπτυξη τεχνολογιών που θα έχουν backwards compatibility με τους υπάρχοντες browsers. Μετά από ψηφοφορία καταψηφίστηκε η πρόταση για να συνεχιστούν οι εργασίες πάνω στην HTML, και αμέσως ιδρύθηκε το WHATWG που άρχισε να δουλεύει πάνω σε αυτή την ιδέα. Τελικά στο συνέδριο του W3C το 2007 η HTML5 υιοθετήθηκε. Οι Ian Hickson και David Hyatt παρήγαγαν το πρώτο δημόσιο προσχέδιο του W3C, της HTML5, στις 22 Ιανουαρίου του 2008.

Στις 14 Φεβρουαρίου 2011, το W3C επέκτεινε τον χάρτη της ομάδας εργασίας HTML με σαφή ορόσημα για HTML5. Τον Μάιο του 2011, η ομάδα εργασίας προώθησε το HTML5

στο “Last Call”, μια πρόσκληση σε κοινότητες μέσα και έξω από το W3C για να επιβεβαιώσει την τεχνική ευρωστία των προδιαγραφών. Το W3C ανέπτυξε μια ολοκληρωμένη σουίτα δοκιμών για να επιτύχει ευρεία διαλειτουργικότητα για την πλήρη προδιαγραφή μέχρι το 2014, η οποία ήταν η ημερομηνία στόχος για σύσταση. Τον Ιανουάριο του 2011, το WHATWG μετονόμασε σε HTML5 την HTML Living Standard. Ωστόσο, το W3C συνέχισε το σχέδιό του για την απελευθέρωση του HTML5. Τον Ιούλιο του 2012, το WHATWG και το W3C αποφάσισαν να διαχωρίσουν. Το W3C θα συνεχίσει τις εργασίες προδιαγραφής HTML5, εστιάζοντας σε ένα ενιαίο οριστικό πρότυπο, το οποίο θεωρείται ως "στιγμιότυπο" (σταθμός) από το WHATWG. Ο οργανισμός WHATWG συνεχίζει τη δουλειά του με το HTML5 ως “Living standard”. Η έννοια του living standard είναι ότι δεν είναι ποτέ ολοκληρωμένο και πάντα ενημερώνεται και βελτιώνεται. Μπορούν να προστεθούν νέες λειτουργίες, αλλά η λειτουργία δεν θα καταργηθεί. Τον Δεκέμβριο του 2012, το W3C χαρακτήρισε HTML5 ως υποψήφια σύσταση. Το κριτήριο για την πρόοδο στη σύσταση του W3C είναι "δύο 100% ολοκληρωμένες και πλήρως διαλειτουργικές υλοποιήσεις". Στις 16 Σεπτεμβρίου 2014, το W3C μετέφερε το HTML5 σε πρόταση σύστασης. Στις 28 Οκτωβρίου 2014, το HTML5 κυκλοφόρησε ως σύσταση του W3C, ολοκληρώνοντας τη διαδικασία προδιαγραφής. Την 1η Νοεμβρίου 2016, το HTML 5.1 κυκλοφόρησε ως σύσταση του W3C, και στις 14 Δεκεμβρίου 2017, το HTML 5.2.

Το HTML 5 εισάγει στοιχεία και χαρακτηριστικά που παρουσιάζουν οι τυπικές σύγχρονες ιστοσελίδες. Μερικές από αυτές είναι οι σημαντικές αντικαταστάσεις των generic block ( ) και inline () στοιχείων, για παράδειγμα τα <nav> (block navigation site), <footer> (συνήθως αναφέρεται στο κάτω μέρος της ιστοσελίδας ή τελευταίες γραμμές κώδικα HTML) ή <audio> και <video> αντί για <object>. Ορισμένα στοιχεία του HTML 4.01 έχουν αποσυρθεί, συμπεριλαμβανομένων στοιχείων με καθαρά παρουσίαση όπως <font> και , τα αποτελέσματα των οποίων έχουν αντικατασταθεί από τα CSS. Επίσης έχει δοθεί έμφαση στη σημασία του DOM scripting. Η σύνταξη HTML 5 δεν βασίζεται πλέον στο SGML, παρά την ομοιότητα της σήμανσής της. Έχει σχεδιαστεί ώστε να είναι συμβατό με το τις παλαιότερων εκδόσεων HTML. Από τις 5 Ιανουαρίου 2009, η HTML 5 περιλαμβάνει επίσης το Web 2.0, μια προηγουμένως ξεχωριστή προδιαγραφή WHATWG.

Collada

Η COLLADA (COLLAborative Design Activity) είναι μια μορφή αρχείου για διαδραστικές εφαρμογές 3D. Διαχειρίζεται από την κοινοπραξία τεχνολογιών μη κερδοσκοπικού χαρακτήρα, Khronos Group και έχει υιοθετηθεί από το ISO ως δημόσια διαθέσιμη προδιαγραφή ISO / PAS 17506. Η COLLADA ορίζει ένα ανοικτό πρότυπο σχήμα XML για την ανταλλαγή ψηφιακών στοιχείων μεταξύ διαφόρων εφαρμογών λογισμικού γραφικών που διαφορετικά θα αποθηκεύονταν τα στοιχεία τους σε ασύμβατες μορφές αρχείων. Τα έγγραφα COLLADA που περιγράφουν τα ψηφιακά στοιχεία είναι αρχεία XML, τα οποία συνήθως αναγνωρίζονται με επέκταση αρχείου .dae (digital asset exchange).

Δημιουργήθηκε αρχικά στη Sony Computer Entertainment από τον Rémi Arnaud και τον Mark C. Barnes, και έχει περάσει στην κατοχή του Khronos Group, μιας κοινοπραξίας που χρηματοδοτείται από μέλη, η οποία τώρα μοιράζεται τα πνευματικά δικαιώματα με τη Sony. Το σχήμα και οι προδιαγραφές COLLADA είναι ελεύθερα διαθέσιμα από τον όμιλο Khronos. Το COLLADA DOM χρησιμοποιεί την Share Source License 1.0 της SCEA. Πολλές εταιρίες γραφικών συνεργάστηκαν με τη Sony από τις αρχές της COLLADA για να δημιουργήσουν ένα εργαλείο που θα ήταν χρήσιμο στο ευρύ κοινό με αποτέλεσμα το COLLADA να συνεχίζει να εξελίσσεται με τις προσπάθειες των συνεργατών του Khronos. Οι πρώτοι συνεργάτες περιελάμβαναν το Alias Systems Corporation, το Criterion Software, το Autodesk, Inc. και την Avid Technology. Δεκάδες στούντιο παραγωγής παιχνιδιών και game

engines έχουν υιοθετήσει το πρότυπο. Τον Μάρτιο του 2011, ο Khronos κυκλοφόρησε τη δοκιμαστική σουίτα συμμόρφωσης COLLADA (CTS). Η σουίτα επιτρέπει σε εφαρμογές που εισάγουν και εξάγουν το COLLADA να δοκιμάσουν μια μεγάλη σειρά παραδειγμάτων, εξασφαλίζοντας ότι συμμορφώνονται με τις προδιαγραφές. Τον Ιούλιο του 2012, το λογισμικό CTS κυκλοφόρησε στο GitHub, επιτρέποντας τις κοινοτικές συνεισφορές. ISO / PAS 17506: 2012 Βιομηχανικά συστήματα αυτοματισμού και ενσωμάτωση - Προδιαγραφή του ψηφιακού περιουσιακού στοιχείου COLLADA για 3D απεικόνιση βιομηχανικών δεδομένων δημοσιεύθηκε τον Ιούλιο του 2012.

Από την έκδοση 1.4, προστέθηκε στο πρότυπο COLLADA υποστήριξη φυσικής. Ο στόχος είναι να επιτρέπεται στους δημιουργούς περιεχομένου να καθορίζουν διάφορα φυσικά χαρακτηριστικά σε οπτικές σκηνές. Για παράδειγμα, μπορεί κανείς να ορίσει ιδιότητες επιφάνειας υλικού όπως τριβή. Επιπλέον, οι δημιουργοί περιεχομένου μπορούν να ορίσουν τις φυσικές ιδιότητες των αντικειμένων στη σκηνή. Αυτό γίνεται με τον καθορισμό των άκαμπτων σωμάτων που πρέπει να συνδέονται με τις οπτικές αναπαραστάσεις. Άλλα χαρακτηριστικά περιλαμβάνουν υποστήριξη για ragdolls, όγκους σύγκρουσης, φυσικούς περιορισμούς μεταξύ φυσικών αντικειμένων και παγκόσμιες φυσικές ιδιότητες όπως είναι η βαρύτητα. Τα προϊόντα μεσαίας σειράς Physics που υποστηρίζουν αυτό το πρότυπο περιλαμβάνουν τη βιβλιοθήκη φυσικής Bullet, το Open Dynamics Engine, το PAL και το PhysX της NVIDIA. Τα προϊόντα αυτά υποστηρίζουν την ανάγνωση της περίληψης που βρέθηκε στο αρχείο COLLADA και τη μεταφορά της σε μια μορφή που το middleware μπορεί να υποστηρίξει και να αντιπροσωπεύσει σε μια φυσική προσομοίωση. Αυτό επιτρέπει επίσης διαφορετικά μέσα και εργαλεία για την ανταλλαγή δεδομένων φυσικής με τυποποιημένο τρόπο. Το Layer Abstraction Physics παρέχει υποστήριξη για την COLLADA Physics σε πολλές μηχανές φυσικής που δεν παρέχουν εγγενώς υποστήριξη COLLADA, συμπεριλαμβανομένων των μηχανών JigLib, OpenTissue, Physics Tokamak και True Axis. Το PAL παρέχει επίσης υποστήριξη για COLLADA σε μηχανές φυσικής που διαθέτουν επίσης μια εγγενή διεπαφή.

Κεφάλαιο 4: Τεχνολογίες WebVR – ThreeJs - Ammo

WebVR

Το WebVR είναι ένα πειραματικό JavaScript Application Programming Interface (API) που παρέχει υποστήριξη σε συσκευές εικονικής πραγματικότητας, όπως το HTC Vive, το Oculus Rift, το Google Cardboard ή το OSVR σε ένα web browser. Το API αυτό σχεδιάστηκε με γνώμονα τους εξής στόχους, την ανίχνευση διαθέσιμων συσκευών εικονικής πραγματικότητας, την αναζήτηση των δυνατοτήτων των συσκευών, τη διερεύνηση της θέσης και του προσανατολισμού της συσκευής και την προβολή εικόνων στη συσκευή με τον κατάλληλο ρυθμό καρέ. Το υλικό που επιτρέπει εφαρμογές εικονικής πραγματικότητας απαιτεί διεπαφές υψηλής ακρίβειας και χαμηλής καθυστέρησης για την επίτευξη αποδεκτής ποιότητας εμπειρίας. Το WebVR παρέχει διεπαφές ειδικά κατασκευασμένες για τα VR hardware για να επιτρέπουν στους προγραμματιστές να κατασκευάσουν συναρπαστικές, άνετες εμπειρίες VR.

Το WebVR σχεδιάστηκε για πρώτη φορά την άνοιξη του 2014 από τον Vladimir Vukićević από το Mozilla. Οι συνεισφέροντες του API περιλαμβάνουν τον Brandon Jones, τον Boris Smus και άλλους από την ομάδα Mozilla. Την 1η Μαρτίου 2016, η ομάδα VR του Mozilla και η ομάδα του Google Chrome ανακοίνωσαν την έκδοση 1.0 της πρότασης API WebVR. Το αποτέλεσμα refactoring API που προέκυψε έφερε πολλές βελτιώσεις στο WebVR. Η τελευταία έκδοση με ετικέτα είναι 1.1, η οποία επεξεργάστηκε τελευταία στις 5 Απριλίου 2017. Οι συντάκτες του εγγράφου περιλαμβάνουν μέλη από τις ομάδες Mozilla και Google. Ωστόσο, ορισμένα μέλη της Microsoft έχουν προσχωρήσει και συνεργάζονται ενεργά στη διαδικασία σύνταξης της έκδοσης 2.0 για το API WebVR. Το API WebVR εκθέτει μερικές νέες διεπαφές (όπως VR Display, VR pose) που επιτρέπουν στις εφαρμογές web να παρουσιάζουν περιεχόμενο στην εικονική πραγματικότητα, χρησιμοποιώντας το WebGL με τις απαραίτητες ρυθμίσεις της κάμερας και τις αλληλεπιδράσεις συσκευών (όπως ελεγκτές ή άποψη). Το API έχει σχεδιαστεί για να ακολουθήσει μια συγκεκριμένη διαδρομή, η οποία είναι πολύ παρόμοια με άλλα παρεμβατικά API Web όπως το API Geolocation. Το WebVR 1.0 υποστηρίζεται αυτή τη στιγμή στην έκδοση έκδοσης του Firefox 55+ για Windows (έκδοση 64 bit μόνο) και στο Chrome για Android ως πείραμα δοκιμής προέλευσης, πράγμα που σημαίνει ότι οι προγραμματιστές μπορούν να ζητήσουν ένα διακριτικό για να προσθέσουν στον ιστότοπό τους το οποίο θα ενεργοποιήσει άψογα το WebVR . Το WebVR 1.1 υποστηρίζεται από το Microsoft Edge από το build 15002+ και από το Samsung Internet, το Chromium, το Servo και το Oculus Carmel. Το 2018, το API WebXR αντικατέστησε το WebVR, σχεδιασμένο τόσο για συσκευές επαυξημένης πραγματικότητας όσο και για συσκευές εικονικής πραγματικότητας.

WebXR

Το WebXR είναι ένα πειραματικό JavaScript API που παρέχει διεπαφές με το VR hardware, επιτρέποντας στους προγραμματιστές να δημιουργούν εφαρμογές εικονικής πραγματικότητας σε browser. Το web μπορεί να γίνει ο προτιμώμενος τρόπος διανομής περιεχομένου VR, με τον ίδιο τρόπο που έγινε ο προτιμώμενος τρόπος διανομής video περιεχομένου. Είναι συμβατό με πολλές συσκευές, πλατφόρμες και οι βασικές τεχνολογίες του υποστηρίζουν τη διαδραστικότητα και την επικοινωνία με πρωτοφανείς τρόπους για το VR. Το ακρωνύμιο XR (Extended Reality) χρησιμοποιείται για να αναφερθεί στο φάσμα του υλικού, των εφαρμογών και των τεχνικών που χρησιμοποιούνται για την Εικονική Πραγματικότητα (VR), την Αυξημένη Πραγματικότητα (AR) και άλλες σχετικές τεχνολογίες όπως η Μικτή Πραγματικότητα (Mixed Reality).

ThreeJS

Το Three.js είναι μια cross-browser βιβλιοθήκη JavaScript και API που χρησιμοποιείται για τη δημιουργία και εμφάνιση κινούμενων γραφικών 3D σε ένα internet browser. Το Three.js επιτρέπει τη δημιουργία τρισδιάστατων γραφικών, από την κάρτα γραφικών, με τη χρήση της γλώσσας JavaScript, ως μέρος μιας ιστοσελίδας χωρίς να εξαρτάται από plug-ins του προγράμματος περιήγησης. Αυτό είναι δυνατό λόγω της εμφάνισης του WebGL. Το Tri.js εκδόθηκε για πρώτη φορά από τον Ricardo Cabello στο GitHub τον Απρίλιο του 2010. Οι απαρχές της βιβλιοθήκης εντοπίζονται στην ενασχόληση του Cabello με τη demoscene στις αρχές της δεκαετίας του 2000. Ο κώδικας αναπτύχθηκε για πρώτη φορά σε ActionScript, στη συνέχεια μεταφέρθηκε το 2009 σε JavaScript. Σύμφωνα με τον Cabello, τα δύο πλεονεκτήματα για τη μεταφορά σε JavaScript είναι ότι πρώτον, ο κώδικας δε θα γινόταν compile πριν από κάθε εκτέλεση και δεύτερον ότι η JavaScript είναι platform independent . Με την εμφάνιση του WebGL, ο Paul Brunt μπόρεσε να προσθέσει τον Renderer. Οι συνεισφορές του Cabello περιλαμβάνουν το σχεδιασμό των API, CanvasRenderer, SVGRenderer όντας υπεύθυνος για το συντονισμό των διαφόρων που συμμετείχαν στο έργο. Ο δεύτερος συνεργάτης, ο Branislav Ulicny, ξεκίνησε με το Three.js το 2010, αφού δημοσίευσε μια σειρά από demo WebGL στον δικό του ιστότοπο. Ήθελε τις δυνατότητες rendering του WebGL στο Three.js να ξεπερνούν τις δυνατότητες του CanvasRenderer ή του SVGRenderer. Η συνεισφορά του στο Three.js περιλαμβάνει materials, Shaders και post-processing. Το Three.js περιλαμβάνει χαρακτηριστικά, όπως Scenes, Cameras, Animation, Materials, Lights, Geometries, Loaders, Exporters, Textures, Virtual Reality κλπ.

Για να μπορεί να εμφανιστεί οτιδήποτε στην οθόνη χρειάζονται τρεις βασικές μεταβλητές, μια σκηνή (scene), μια camera και ένας Renderer. Παρακάτω θα δοθεί ένα παράδειγμα του κώδικα για τη δημιουργία μιας απλής σκηνής, ένα μαύρο background και ένα πράσινο κύβο να περιστρέφεται στη μέση. Πρωτίστως «στήνουμε» τη σκηνή, την camera και τον Renderer στις δηλώσεις μεταβλητών. Το Three.js έχει διάφορες επιλογές καμερών (Orthographic, Prespective, κλπ), στο παράδειγμα θα χρησιμοποιηθεί η Prespective. Η πρώτη τιμή μέσα στην παρένθεση είναι το οπτικό πεδίο της κάμερας και μετριέται σε μοίρες, ενώ η δεύτερη τιμή είναι το aspect ratio. Οι επόμενες δύο τιμές είναι το μακρινό και κοντινό

clipping plane (near and far clipping plane). Τα αντικείμενα που είναι από την πρώτη τιμή και κάτω, δηλαδή πιο κοντά στην κάμερα, δε θα γίνουν Render, αντίστοιχα και τα αντικείμενα που είναι μακρύτερα από τη δεύτερη τιμή δε θα γίνουν Render. Το επόμενο είναι ο Renderer. Εκτός από το WebGLRenderer που χρησιμοποιείται εδώ, το Three.js συνοδεύεται από μερικούς άλλους, που συχνά χρησιμοποιούνται ως εναλλακτικές λύσεις για χρήστες με παλαιότερα προγράμματα περιήγησης ή για όσους δεν έχουν υποστήριξη WebGL. Εκτός από τη δημιουργία του στιγμιότυπου εμφάνισης, πρέπει επίσης να οριστεί το μέγεθος στο οποίο επιθυμούμε να προβληθεί η εφαρμογή μας. Είναι καλή ιδέα να χρησιμοποιηθεί το πλάτος και το ύψος της περιοχής που θέλουμε να γεμίσουμε με την εφαρμογή, σε αυτή την περίπτωση, το πλάτος και το ύψος του παραθύρου του προγράμματος περιήγησης. Για απαιτητικές εφαρμογές, μπορούν επίσης να χρησιμοποιηθούν μικρότερες τιμές, όπως window.innerWidth / 2 και window.innerHeight / 2, οι οποίες θα κάνουν την εφαρμογή να εμφανίζεται στο μισό μέγεθος.

```
var scene = new THREE.Scene();
var camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight,
0.1, 1000 );

var renderer = new THREE.WebGLRenderer();
renderer.setSize( window.innerWidth, window.innerHeight );
document.body.appendChild( renderer.domElement );
```

Εικόνα 4.1 - Scene,Camera,Render

Για να δημιουργηθεί ένας κύβος, θα χρησιμοποιηθεί μια BoxGeometry. Η BoxGeometry είναι ένα αντικείμενο που περιέχει όλα τα σημεία (vertices) και faces του κύβου. Εκτός από τη γεωμετρία, χρειαζόμαστε ένα material για να το χρωματίσουμε. Το Three.js έχει διάφορα materials, αλλά εδώ θα χρησιμοποιηθεί το MeshBasicMaterial. Όλα τα υλικά απαρτίζονται από ένα σύνολο ιδιοτήτων που εφαρμόζονται σε αυτά. Επειδή το παράδειγμα είναι απλό, χρησιμοποιείται μόνο μια ιδιότητα, αυτή του χρώματος 0x00ff00, το οποίο είναι πράσινο. Η γεωμετρία και το material εφαρμόζονται σε ένα τρίτο αντικείμενο, το mesh, το οποίο και εισάγεται μέσα στη σκηνή. Αν δεν οριστούν συντεταγμένες τοποθεσίας του mesh, αυτό εμφανίζεται στη θέση (0,0,0). Επειδή η κάμερα βρίσκεται στο (0,0,0) και αν μείνει ως έχει δε θα εμφανίζεται ο κύβος, τη μετακινούμε στο σημείο (0,0,5).

Ο παραπάνω κώδικας μόνος του δε φτάνει, αν τον τρέξει κάποιος δε θα εμφανιστεί τίποτα στον browser, καθώς δε γίνεται rendering. Για αυτό το λόγο χρησιμοποιείται μια Render function, στην προκειμένη περίπτωση η animate. Σε αυτή την επανάληψη ο Renderer δημιουργεί τη σκηνή από την αρχή κάθε φορά που γίνεται ανανέωση στην οθόνη (συνήθως είναι 60 Hz ο ρυθμός ανανέωσης). Η μέθοδος requestAnimationFrame προτιμάται από άλλες, κυρίως γιατί παγώνει όταν ο χρήστης δε βρίσκεται στην καρτέλα της εφαρμογής και έτσι κερδίζεται υπολογιστική ισχύς και μειώνεται η κατανάλωση ρεύματος.

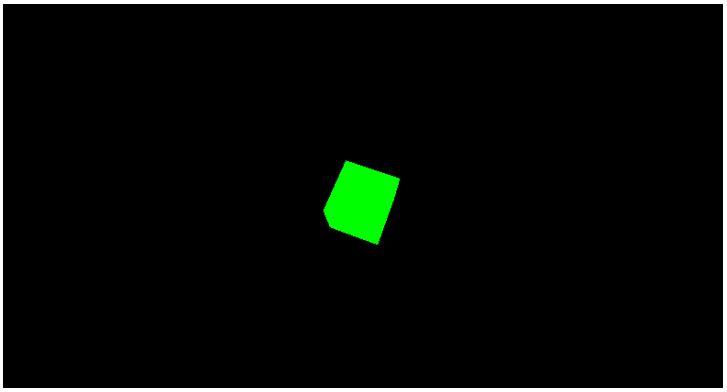
```
function animate() {
        requestAnimationFrame( animate );
        renderer.render( scene, camera );
}
animate();
```

Εικόνα 4.2 - Function animate

Επιπροσθέτως στον κώδικα προστίθενται δύο rotation μέσα στην animate και έτσι ο κύβος περιστρέφεται κάθε φορά που γίνεται Render η σκηνή.

```
<html>
	<head>
		<title>My first three.js app</title>
		<style>
			body { margin: 0; }
			canvas { width: 100%; height: 100% }
		</style>
	</head>
	<body>
		<script src="js/three.js"></script>
		<script>
			var scene = new THREE.Scene();
			var camera = new THREE.PerspectiveCamera( 75,
window.innerWidth/window.innerHeight, 0.1, 1000 );

			var renderer = new THREE.WebGLRenderer();
			renderer.setSize( window.innerWidth, window.innerHeight );
			document.body.appendChild( renderer.domElement );

			var geometry = new THREE.BoxGeometry( 1, 1, 1 );
			var material = new THREE.MeshBasicMaterial( { color: 0x00ff00
} );

			var cube = new THREE.Mesh( geometry, material );
			scene.add( cube );

			camera.position.z = 5;

			var animate = function () {
				requestAnimationFrame( animate );

				cube.rotation.x += 0.01;
				cube.rotation.y += 0.01;

				renderer.render( scene, camera );
			};

			animate();
		</script>
	</body>
</html>
```

Εικόνα 4.3 - Τελικό παράδειγμα



Εικόνα 4.4 - Αποτέλεσμα

Εισαγωγή 3D μοντέλων

Τα τρισδιάστατα μοντέλα διατίθενται σε εκατοντάδες μορφές αρχείων. Το Three.js παρέχει πολλούς loaders, αλλά η επιλογή του σωστού loader θα εξοικονομήσει χρόνο και θα γλυτώσει τον προγραμματιστή από ανεπιθύμητα errors. Ορισμένα format είναι δύσκολα στη διαχείριση, δεν είναι αποτελεσματικά για εμπειρίες σε πραγματικό χρόνο, ή απλά δεν υποστηρίζονται πλήρως μια δεδομένη στιγμή. Όπου είναι δυνατόν, συνίσταται η χρήση του glTF (GL Transmission Format), ενώ αμφότερες οι εκδόσεις της μορφής GLB και GLTF υποστηρίζονται επαρκώς. Το glTF περιλαμβάνει χαρακτηριστικά όπως meshes, materials, textures, skins, animations, φώτα και κάμερες. Όταν το glTF δε μπορεί να χρησιμοποιηθεί, χρησιμοποιούνται άλλες μορφές, όπως οι FBX, OBJ ή COLLADA, που είναι επίσης διαθέσιμες και συντηρούνται τακτικά. Το Three.js περιλαμβάνει ορισμένο αριθμό loaders (π.χ. ObjectLoader) οι υπόλοιποι θα πρέπει να προστεθούν ξεχωριστά. Όταν γίνει η δήλωση και δημιουργηθεί ο loader, μπορεί να εισαχθεί και το 3D μοντέλο. Στις περιπτώσεις χρήσης glTF καλό θα ήταν να χρησιμοποιούνται global scripts, όπως στο παρακάτω παράδειγμα.

```
// global script
<script src="GLTFLoader.js"></script>

// commonjs
var THREE = window.THREE = require('three');
require('three/examples/js/loaders/GLTFLoader');

// ES modules
import { GLTFLoader } from 'three/examples/jsm/loaders/GLTFLoader';
```

Εικόνα 4.5 - Imports

```
var loader = new THREE.GLTFLoader();

loader.load( 'path/to/model.glb', function ( gltf ) {

        scene.add( gltf.scene );

}, undefined, function ( error ) {

        console.error( error );

} );
```

Εικόνα 4.6 - Loader

Animation στο Three.js

Το animation system του Three.js επιτρέπει πλέον το animation διάφορων παραμέτρων των μοντέλων, τις υφές, το visibility, την κίνηση στο χώρο, κλπ. Παράμετροι όπως το βάρος, ο χρόνος του animation, ενός ή περισσοτέρων αντικειμένων μπορεί να αλλάξει σε καθένα ξεχωριστά, όπως επίσης διαφορετικά animations μπορούν να συγχρονιστούν. Όλα αυτά δεν ήταν αυτονόητα πριν το 2015, οπότε και το animation system του Three.js άλλαξε ριζικά και πλέον προσομοιάζει αυτά των Unity και Unreal Engine.

Εισάγοντας ένα 3D αντικείμενο, παραδείγματος χάριν ένα glTF που έγινε export από το Blender, ένα από τα response fields θα λέγεται «animations», και θα περιλαμβάνει τα AnimationClips του μοντέλου. Ένα AnimationClip είναι ένα σύνολο από keyframes που μαζί συνιστούν ένα animation. Κάθε AnimationClip περιέχει δεδομένα για συγκεκριμένες ενέργειες του αντικειμένου. Για παράδειγμα αν το αντικείμενο είναι κάποιος χαρακτήρας ενός παιχνιδιού, θα έχει ένα AnimationClip για το περπάτημα, ένα δεύτερο για το τρέξιμο, ένα τρίτο για κάθισμα, κλπ. Μέσα σε ένα AnimationClip τα δεδομένα για κάθε κίνηση αποθηκεύονται σε ξεχωριστά KeyframeTracks. Έστω ότι ένας χαρακτήρας έχει έναν σκελετό, ένα KeyframeTrack θα αποθηκεύσει τα δεδομένα για τις αλλαγές θέσης του κατώτερου οστού του βραχίονα , ένα δεύτερο KeyframeTrack τις αλλαγές περιστροφής του ίδιου οστού, ένα τρίτο την περιστροφή ενός άλλου οστού και ούτω καθεξής. Είναι σύνηθες ένα AnimationClip να αποτελείται από πολλά τέτοια κομμάτια. Αν υποθέσουμε ότι το μοντέλο έχει morph targets (για παράδειγμα, ένα morph target που δείχνει ένα φιλικό πρόσωπο και ένα άλλο που δείχνει ένα θυμωμένο πρόσωπο), κάθε KeyframeTrack περιέχει πληροφορίες σχετικά με τον τρόπο αλλαγής της επίδρασης ενός συγκεκριμένου morph target κατά τη διάρκεια του κλιπ. Τα αποθηκευμένα δεδομένα αποτελούν μόνο τη βάση για τα animations, η πραγματική αναπαραγωγή ελέγχεται από το AnimationMixer. Το ίδιο το AnimationMixer έχει πολύ λίγες

ιδιότητες και μεθόδους, επειδή μπορεί να ελεγχθεί από τα AnimationActions. Με τη ρύθμιση ενός AnimationAction μπορεί να καθοριστεί πότε ένα συγκεκριμένο AnimationClip θα παιχτεί, θα σταματήσει, εάν και πόσο συχνά πρέπει να επαναληφθεί το κλιπ, αν πρέπει να γίνει με ξεθώριασμα, κλπ. Να σημειωθεί ότι δεν περιλαμβάνουν όλες οι μορφές μοντέλων animations, το OBJ δεν περιλαμβάνει για παράδειγμα, και ότι μόνο μερικοί loaders του Three.js υποστηρίζουν AnimationClips. Κάποιοι εξ αυτών είναι οι ακόλουθοι, THREE.ObjectLoader, THREE.BVHLoader, THREE.ColladaLoader, THREE.FBXLoader, THREE.GLTFLoader, THREE.MMDLoader, THREE.SEA3DLoader.

Δημιουργία VR περιεχομένου με Three.js

Για τη δημιουργία VR σκηνών με Three.js ισχύουν όλα τα βασικά που αναφέρθηκαν παραπάνω δηλαδή μια κάμερα, μια σκηνή και ένας Renderer. Για να γίνει η σκηνή VR compatible πρέπει να συμπεριληφθεί το VRButton.js. στο σημείο του κώδικα που γίνονται τα imports.

```
import { VRButton } from 'three/examples/jsm/webxr/VRButton.js';

document.body.appendChild( VRButton.createButton( renderer ) );
```

Εικόνα 4.7 - VR Button

Επιπλέον πρέπει να γίνουν αλλαγές στη συνάρτηση Render για να ενεργοποιηθεί το rendering για VR. Αντί για τη requestAnimationFrame() που χρησιμοποιήθηκε στο παραπάνω παράδειγμα με τον κύβο στα VR projects χρησιμοποιείται η setAnimationLoop(). Ο ελάχιστος κώδικας που απαιτείται είναι ο ακόλουθος

```
renderer.vr.enabled = true;

renderer.setAnimationLoop( function () {

        renderer.render( scene, camera );

} );
```
Εικόνα 4.8 - Renderer

Το παρακάτω παράδειγμα, που θα το βρείτε στο https://threejs.org/examples ως vr/panorama/depth, δείχνει ένα πανόραμα 360 μοιρών με φαινομενική κίνηση και περιστροφή της κάμερας γύρω από τον εαυτό της.


Εικόνα 4.9 - Landscape 360°

Ο κώδικας περιλαμβάνει σε γενικές γραμμές γνωστά στοιχεία, μια σκηνή, ένα VR Button το οποίο γίνεται import στην αρχή του script, μια κάμερα, φώτα και μια σφαίρα μέσα στην οποία τοποθετείται η κάμερα. Η σφαίρα παίζει το ρόλο του κόσμου, γίνεται load ένα texture, μια jpeg εικόνα 360 μοιρών στην προκειμένη περίπτωση, με έναν TextureLoader, το οποίο texture τοποθετείται στην εσωτερική επιφάνεια της σφαίρας δίνοντας την ψευδαίσθηση του πραγματικού κόσμου. Στη συνέχεια στη συνάρτηση Render η σφαίρα κινείται στους άξονες x και z συναρτήσει του χρόνου, ενώ περιστρέφεται στον y, δημιουργώντας έτσι την ψευδαίσθηση ότι κινείται η κάμερα ενώ στην ουσία κινείται ο κόσμος.

```
// Create the panoramic sphere mesh
sphere = new THREE.Mesh( panoSphereGeo, panoSphereMat );

// Load and assign the texture and depth map
var manager = new THREE.LoadingManager();
var loader = new THREE.TextureLoader( manager );

loader.load( './textures/kandao3.jpg', function ( texture ) {

        texture.minFilter = THREE.NearestFilter;
        texture.format = THREE.RGBFormat;
        texture.generateMipmaps = false;
        sphere.material.map = texture;

} );

loader.load( './textures/kandao3_depthmap.jpg', function ( depth ) {

        depth.minFilter = THREE.NearestFilter;
        depth.format = THREE.RGBFormat;
        depth.generateMipmaps = false;
        sphere.material.displacementMap = depth;

} );

// On load complete add the panoramic sphere to the scene
manager.onLoad = function () {

        scene.add( sphere );

};
function render() {

        // If we are not presenting move the camera a little so the effect is visible

        if ( renderer.vr.isPresenting() === false ) {

                var time = clock.getElapsedTime();

                sphere.rotation.y += 0.001;
                sphere.position.x = Math.sin( time ) * 0.2;
                sphere.position.z = Math.cos( time ) * 0.2;

        }

        renderer.render( scene, camera );

}
```

Εικόνα 4.10 - Απόσπασμα κώδικα

Ammo.js

Το Ammo.js είναι μια απευθείας μεταφορά της μηχανής φυσικής Bullet, η οποία είναι γραμμένη σε C++, σε JavaScript με τη χρήση του Emscripten. Το Emscripten είναι ένα εργαλείο που χρησιμοποιείται για τη μετατροπή C και C++ κώδικα σε asm.js (υποκατηγορία της JavaScript) και WebAssembly. Ο πηγαίος κώδικας της Bullet μεταφράζεται απευθείας σε JavaScript, χωρίς ανθρώπινη παρέμβαση, οπότε θεωρητικά είναι λειτουργικός όπως ο αρχικός.

Για να γίνει αντιληπτή η χρήση του Ammo.js πρέπει να γίνουν κατανοητές ορισμένες παράμετροι που χρησιμοποιεί η συγκεκριμένη μηχανή φυσικής.

- Physics World: Είναι ο κόσμος μέσα στον οποίο αναπαρίστανται οι νόμοι της φυσικής του πραγματικού κόσμου ή του φανταστικού κόσμου ενός δημιουργού. Στο Ammo.js αυτός ο κόσμος ονομάζεται Collision World με παράγωγό του τον Dynamic World. Ένας Physics World πρέπει να δίνει τη δυνατότητα να ορίζεται ή να ρυθμίζεται κάποια βαρύτητα.

- Rigid Body and Collision Shape: Για να υπάρξει οποιαδήποτε μορφή φυσικής αλληλεπίδρασης πρέπει να υπάρχει ένα σώμα. Στο Ammo.js αυτό το σώμα ονομάζεται collision object ή rigid body. Ένα rigid body έχει τις εξής ιδιότητες κινείται, συγκρούεται, έχει μάζα και μπορεί να του ασκηθεί κάποια δύναμη. Από μόνο του όμως, όντας άμορφο, δε μπορεί να έχει κάποια αλληλεπίδραση, για αυτό το λόγο έρχονται στο προσκήνιο τα Collision Shapes. Το Bullet wiki αναφέρει αυτούσια «...Κάθε rigid body πρέπει να έχει σημείο αναφοράς ένα Collision Shape. Το Collision Shape είναι μόνο για συγκρούσεις και συνεπώς δεν εμπεριέχει έννοιες όπως μάζα, αδράνεια, επαναφορά, κλπ. Αν έχετε πολλά σώματα που χρησιμοποιούν το ίδιο Collision Shape (π.χ. κάθε διαστημόπλοιο στη προσομοίωση σας να είναι μια σφαίρα ακτίνας 5 μονάδων) είναι καλή πρακτική να υπάρχει μόνο ένα Collision Shape Bullet (Ammo.js) το οποίο να χρησιμοποιείται σε όλα αυτά τα σώματα. Το Bullet (Ammo.js) υποστηρίζει μια μεγάλη ποικιλία από διαφορετικά Collision Shapes και δύναται ο προγραμματιστής να προσθέσει το δικό του...». Ορισμένα από τα Collision Shapes που υποστηρίζονται είναι primitive shapes (π.χ. κύβος, σφαίρα, κύλινδρος, κάψουλα, κώνος, κλπ), σύνθετα σχήματα, mesh τριγώνου, στατικό επίπεδο κ.α. Επιπλέον να σημειωθεί ότι όταν ένα rigid body έχει μάζα μηδέν σημαίνει ότι το σώμα έχει άπειρη μάζα ως εκ τούτου είναι στατικό.

- Rigid Body Dynamics: Έτσι ορίζονται μεταβλητές όπως η δύναμη, η μάζα, η τριβή, η ταχύτητα και διάφοροι περιορισμοί του κόσμου. Παραδείγματος χάριν σε ένα παιχνίδι ποδοσφαίρου όταν ο παίχτης κλωτσάει τη μπάλα αυτή φεύγει με μια δύναμη και την κίνησή της επηρεάζουν η μάζα της, η βαρύτητα, η ενδεχόμενη τριβή της με το έδαφος, κλπ, δηλαδή μεταβλητές που έχουν οριστεί από τον προγραμματιστή.

- Collision Filtering and Collision Detection: Η παράμετρος Collision Filtering καθορίζει ποια αντικείμενα μπορούν να κάνουν κρούση και ποια όχι. Για παράδειγμα σε ένα shooting game συνήθως οι σφαίρες δεν χτυπάνε τους συμπαίκτες αλλά μόνο ορισμένα αντικείμενα και αντιπάλους. Αντίστοιχα σε ένα MOBA κάνει κάποιος

χαρακτήρας ένα barrier spell, θέλουμε αυτό το barrier να μην διαπερνάται από spells αντιπάλων αλλά μόνο από κάποια spells συμπαικτών, πχ healing spells. Στο Ammo.js υπάρχουν τρεις τρόποι για να επιτευχθεί collision filtering, οι μάσκες (masks) και διάφορα callbacks (boardphase, nearcallback). Το Collision Detection από την άλλη αφορά την ανίχνευση της κρούσης. Στο αντίστοιχο shooting game του προηγούμενου παραδείγματος το Collision Detection θα χρησιμοποιηθεί για να ανιχνευθεί η κρούση μεταξύ εχθρού και σφαίρας προκειμένου να γίνει κάτι, όπως μείωση ζωής ή θάνατος του εχθρού.

Κεφάλαιο 5 Κεφάλαιο 5: Κύριο μέρος της εργασίας – Συμπεράσματα

Ανάλυση Προβλήματος

Προϋπόθεση για την ενασχόληση με κάποια εργασία πάνω στην ανάπτυξη VR περιεχομένου είναι η γνώση μιας ή περισσοτέρων γλωσσών προγραμματισμού. Είτε χρησιμοποιηθεί Unity (C#), είτε Unreal Engine (C++), είτε κάποιο άλλο εργαλείο, ο δημιουργός θα χρειαστεί γνώσεις προγραμματισμού σε μικρό ή μεγάλο βαθμό. Στην περίπτωση της παρούσης εργασίας, επειδή τρέχει σε browser, απαιτούνται γνώσεις JavaScript και συναφών της API.

Απαιτήσεις Συστήματος

Θα χρειαστεί κάποιο VR Headset με controllers (HTC Vive, Oculus Rift, κλπ), πάνω στο οποίο θα υλοποιηθεί η εργασία. Απαραίτητη θα πρέπει να θεωρείται και η κατοχή υπολογιστή με δυνατή κάρτα γραφικών για να υποστηρίζονται οι αναλύσεις και τα fps, ώστε η εφαρμογή να τρέχει σε ομαλά πλαίσια. Επίσης χρειάζεται οπωσδήποτε να είναι εγκατεστημένο το xampp και κάποιος κειμενογράφος (sublime text ή notepad++). Τέλος η εγκατάσταση ενός browser πέραν του Edge (κατά προτίμηση Mozilla Firefox).

Σχεδιασμός για την υλοποίηση με Three.js

Κατ’ αρχάς απαιτείται εξοικείωση με παρεμφερή project εικονικής πραγματικότητας που υπάρχουν στο internet και εκτενής μελέτη του κώδικα αυτών. Χαρακτηριστικό παράδειγμα τα παραδείγματα στη σελίδα threejs.org αλλά και στο medium.com. Μελέτη για τη δημιουργία αρχικά απλού VR περιεχομένου χωρίς τη χρήση controllers, κάποιο πανόραμα ή κάποιο video 360 μοιρών για παράδειγμα. Στη συνέχεια θα πρέπει να μελετηθούν πιο σύγχρονες εφαρμογές με ενδεχόμενη κίνηση κάμερας, animations, εισαγωγή 3D models και φυσικά χρήση των controllers για να γίνει η εφαρμογή διαδραστική.

Υλοποίηση

Παρακάτω θα παρουσιαστούν εκτενώς οι λεπτομέρειες της υλοποίησης της εφαρμογής. Θα γίνει διεξοδική ανάλυση του κώδικα σε JavaScript, με χρήση της βιβλιοθήκης Three.js, καθώς και αναφορά στον εξοπλισμό εικονικής πραγματικότητας που χρησιμοποιήθηκε. Η εφαρμογή αφορά ένα περιβάλλον εικονικής πραγματικότητας, με κάποια σχετικά απλά γραφικά, στο οποίο ο χρήστης με τη χρήση του ενός controller θα μπορεί να σχεδιάζει τρισδιάστατα αντικείμενα και στη συνέχεια να τα μετακινεί ή να τα περιστρέφει με τη χρήση του δεύτερου controller.

HTC Vive-SteamVR

Για την υλοποίηση της εφαρμογής χρησιμοποιήθηκε ο εξοπλισμός του εργαστηρίου ο οποίος αποτελείται, εκτός από ένα υπολογιστή με GPU GTX 1050 TI, από ένα headset HTC Vive με δύο controllers και δύο σταθμούς βάσης. Η λειτουργία του Vive αναλύεται σε προηγούμενο κεφάλαιο. Το SteamVR είναι μια πλατφόρμα που αναπτύχθηκε από τη Valve για το Steam και στην παρούσα εργασία χρησιμοποιήθηκε για να επικοινωνήσει το υλικό (VR Headset) με το software.

Κώδικας

Το HTML αρχείο τρέχει σε ένα τοπικό apache HTTP server που δημιουργείται με τη βοήθεια του xampp. Ο HTML κώδικας περιλαμβάνει το head container με τα meta tags και ακολουθεί το body tag. Μέσα στο body tag υπάρχουν ένα style tag, με τον απαραίτητο ελάχιστο κώδικα και τέσσερα script tags εκ των οποίων τα τρία χρησιμοποιούνται για να «τραβήξουν» js αρχεία και το ένα για τη δημιουργία VR περιεχομένου (type=module). Μέσα στο τελευταίο script tag υπάρχει ένα import της βιβλιοθήκης Three.js, το οποίο πλέον ίσως και να είναι περιττό, αφού το ίδιο αρχείο καλείται σε ένα από τα προηγούμενα script tags. Ακολουθούν οι δηλώσεις μεταβλητών, οι κλήσεις των τριών βασικών συναρτήσεων και εν τέλει η υλοποίηση τους.

Οι δηλώσεις αρχίζουν με μερικές global μεταβλητές (line, vector 1,2,3, κλπ), ενός αντικειμένου (shapes), δυο group και τριών πινάκων (raycaster, intersected, tempMatrix). Στη συνέχεια ακολουθεί η δήλωση της σκηνής (scene), η δήλωση και ο ορισμός ενός mesh που περιέχει ένα geometry και ένα material, και η κλήση των τριών βασικών συναρτήσεων. Όλες αυτές οι μεταβλητές δηλώθηκαν globally γιατί χρησιμοποιούνται σε παραπάνω από μια συναρτήσεις.

```
<script type="module">
    import * as THREE from '../three.js_master/build/three.module.js';


    var container;
    var camera, scene, renderer;
    var controller1, controller2;


    var line;
    var shapes = {};

    var up = new THREE.Vector3( 0, 1, 0 );

    var vector1 = new THREE.Vector3();
    var vector2 = new THREE.Vector3();
    var vector3 = new THREE.Vector3();
    var vector4 = new THREE.Vector3();

    var group;
    var group_mesh = new THREE.Group();

    var raycaster, intersected = [];
    var tempMatrix = new THREE.Matrix4();

    scene = new THREE.Scene();

    var geometry = new THREE.SphereGeometry(1000, 400, 300);
    var material = new THREE.MeshPhongMaterial();
    var tatooineMesh = new THREE.Mesh(geometry, material);



    init();
    initGeometry();
    animate();
```

Εικόνα 5.1 - Δηλώσεις

Η πρώτη βασική συνάρτηση της εφαρμογής είναι η συνάρτηση init. Η init περιέχει όλα τα αντικείμενα της σκηνής και υποσυναρτήσεις που αφορούν κάποια από αυτά. Αρχικά βλέπει κανείς τη δήλωση ενός Three.Group, μέσα στο οποίο θα μπουν κάποια αντικείμενα (κουμπιά) προκειμένου να έχουν τις ίδιες ιδιότητες και να λειτουργούν σαν ομάδα αντικειμένων. Παρακάτω υπάρχει η υλοποίηση της κάμερας. Επιλέχθηκε η prespective camera καθώς είναι η πιο κατάλληλη για first person θέαση. Στα χαρακτηριστικά της παρατηρεί κανείς ότι το field of view είναι στις 75 μοίρες ενώ ο άνθρωπος έχει 120. Αυτό συνέβη διότι κατά τη δοκιμή με τις 120 μοίρες υπήρχε μια παραμόρφωση στην εικόνα η οποία κούραζε. Η επόμενη τιμή είναι το aspect ratio, ενώ η μεθεπόμενη τιμή αφορά το πόσο κοντά θα βλέπει η κάμερα. Η τελευταία τιμή, η οποία αφορά το πόσο μακριά βλέπει η κάμερα, είναι τόσο μεγάλη γιατί ο πλανήτης που θα τοποθετηθεί αργότερα θα είναι αρκετά μακριά. Στη συνέχεια υπάρχει η δημιουργία του «ουρανού». Είναι ένα SphereGeometry με μεγάλη διάμετρο, με material πάνω στο οποίο υπάρχει ένα texture. Το texture γίνεται load με ένα texture loader της βιβλιοθήκης Three.js, και στη συνέχεια «κολλάει» πάνω στο material ({ map: texture}). Geometry και material προστίθενται σε ένα Mesh, αλλά με το material να είναι από τη μέσα μεριά (THREE.BackSide), και το sky προστίθεται στη σκηνή. Στην ουσία ο ουρανός είναι μια σφαίρα που μέσα υπάρχουν όλα τα αντικείμενα. Ακολουθεί το έδαφος στο οποίο υποτίθεται πατάει ο χρήστης. Είναι ένα PlaneBufferGeometry με ένα texture γυρισμένο 90 μοίρες. Πάλι όπως και πριν το Mesh έχει το geometry και το material με το texture, αλλά εδώ το texture επαναλαμβάνεται οριζόντια και κάθετα στο επίπεδο (texture.wrapS και texture.wrapT). Η περιστροφή 90 μοιρών του επιπέδου είναι γιατί όταν δημιουργείται το επίπεδο είναι σε κάθετη θέση. Ακολουθεί το mesh του πλανήτη με παρόμοια γνωρίσματα και τεχνικές που χρησιμοποιήθηκαν και στα παραπάνω (geometry, materials, textures).

```
camera = new THREE.PerspectiveCamera( 75, window.innerWidth / window.innerHeight, 0.01, 10000000000 );

var skyGeometry = new THREE.SphereGeometry(10000, 25, 25);
var loader  = new THREE.TextureLoader(),
texture = loader.load( "textures/universe.jpg" );
var material = new THREE.MeshPhongMaterial({ map: texture,});
var sky = new THREE.Mesh(skyGeometry, material);
sky.material.side = THREE.BackSide;
scene.add(sky);

var geometry = new THREE.PlaneBufferGeometry( 50, 50 );
var material = new THREE.MeshStandardMaterial( {
    color: 0xfffff0,
    roughness: 1.0,
    metalness: 0.0
} );
var loader  = new THREE.TextureLoader(),
texture = loader.load( "textures/moon.jpg" );
texture.wrapS = THREE.RepeatWrapping;
texture.wrapT = THREE.RepeatWrapping;
texture.repeat.set( 20, 20 );
var material = new THREE.MeshPhongMaterial({ map: texture,});
var floor = new THREE.Mesh( geometry, material );
floor.rotation.x = - Math.PI / 2;
scene.add( floor );


tatooineMesh.position.set(600, 1000, -1600);
tatooineMesh.material.map = THREE.ImageUtils.loadTexture('textures/tatooine.png');
tatooineMesh.material.bumpMap = THREE.ImageUtils.loadTexture('textures/tatooine_bump.png');
tatooineMesh.material.bumpScale = 2;
tatooineMesh.material.specularMap = THREE.ImageUtils.loadTexture('textures/tatooine_specular.png');
tatooineMesh.material.specular = new THREE.Color('grey');
tatooineMesh.receiveShadow = true;
tatooineMesh.castShadow = true;

scene.add( tatooineMesh );
```

Εικόνα 5.2 - Camera, Floor, Planet

Τα επόμενα αντικείμενα είναι τέσσερα κουμπιά (PlaneBufferGeometry), που θα χρησιμεύουν για την κίνηση και περιστροφή του παραγόμενου από τον χρήστη αντικείμενου, ενώ πιο μετά προστίθενται κάποια φώτα (Ambient, Hemisphere και Directional). Ως Renderer χρησιμοποιείται ο WebGLRenderer, με επιλογή για anti-aliasing και aspect ratio ίδιο με αυτό της κάμερας, ενεργοποιημένη τη λειτουργία VR (renderer.vr.enabled = true;), ενώ στη συνέχεια το αποτέλεσμα αποτυπώνεται στον canvas (container.appendChild(renderer.domElement);). Στη συνέχεια προστίθεται το κουμπί για να μετατρέψει το περιεχόμενο σε VR εφόσον υπάρχει συνδεδεμένη συσκευή που υποστηρίζει VR.

Στη συνέχεια ακολουθούν οι υποσυναρτήσεις της init που έχουν να κάνουν με τις λειτουργίες των κουμπιών-σκανδαλών των controller. Οι onSelectStart1 και onSelectEnd1, δεν έχουν όρισμα και αφορούν το χειριστήριο 1, που κάνει μόνο sculpt και χρειάζεται μόνο την εντολή this.userData.isSelecting με true όταν πατιέται το κουμπί και false όταν απελευθερώνεται. Οι onSelectStart2 και onSelectEnd2 αφορούν το χειριστήριο 2 που χρησιμοποιείται ως δείκτης και για να «πατάει» ο χρήστης τα κουμπιά της κίνησης-περιστροφής. Παίρνουν όρισμα (event) το οποίο αναφέρεται στον controller και ανάλογα με το id του κάθε κουμπιού, και αν ο raycaster ανιχνέυσει ότι υπάρχει σύγκρουση, με πάτημα σκανδάλης ταυτόχρονα, κινεί ή περιστρέφει το mesh που έφτιαξε ο παίχτης.

```
function onSelectStart2( event ) {

    var controller = event.target;

    var intersections = getIntersections( controller );


    if ( intersections.length > 0 ) {


    var intersection = intersections[ 0 ];

    tempMatrix.getInverse( controller.matrixWorld );

    var object = intersection.object;

    if (object.userData.id == "1") {
        group_mesh.position.z -= 0.1;
    }

    if (object.userData.id == "2") {
        group_mesh.position.z += 0.1;
    }

    if (object.userData.id == "3") {
        group_mesh.rotation.y += 0.2;
    }

    if (object.userData.id == "4") {
        group_mesh.rotation.y -= 0.2;
    }


    }



}

function onSelectEnd2( event ) {

    var controller = event.target;

    if ( controller.userData.selected !== undefined ) {

        var object = controller.userData.selected;

        group.add( object );

        controller.userData.selected = undefined;

    }

}
```

Εικόνα 5.3 - Function OnSelectStart2, OnSelectEnd2

Οι controller1 και controller2 ορίζονται αμέσως μετά με τους αντίστοιχους event listeners για το πάτημα των σκανδάλων. Τα userData.points και userData.matrices

χρησιμεύουν για να αποθηκεύεται η θέση του controller1. Στη συνέχεια δηλώνονται οι γεωμετρίες, τα material και τα mesh των grips των δυο controller που θα εμφανίζονται στην οθόνη του headset, που είναι στην ουσία ένας κύλινδρος, ενώ δημιουργούνται και η κεφαλή του controller1, του controller που κάνει sculpt, και η γραμμή του controller2 για να φαίνεται που δείχνει. Το controller_head είναι ένα mesh με μια γεωμετρία εικοσαέδρου με πάρα πολλές γωνίες (vertices) για να φαίνεται λείο, ενώ η γραμμή είναι ένα mesh με μια line γεωμετρία. Στη συνέχεια αυτά τα δύο meshes προστίθενται στα ήδη υπάρχοντα των controller1 και controller 2 αντίστοιχα, ενώ δηλώνεται και ο raycaster.

```
controller1 = renderer.vr.getController( 0 );
controller1.addEventListener( 'selectstart', onSelectStart1 );
controller1.addEventListener( 'selectend', onSelectEnd1 );
controller1.userData.points = [ new THREE.Vector3(), new THREE.Vector3() ];
controller1.userData.matrices = [ new THREE.Matrix4(), new THREE.Matrix4() ];
scene.add( controller1 );

controller2 = renderer.vr.getController( 1 );
controller2.addEventListener( 'selectstart', onSelectStart2 );
controller2.addEventListener( 'selectend', onSelectEnd2 );
scene.add( controller2 );

var geometry = new THREE.CylinderBufferGeometry( 0.01, 0.02, 0.08, 5 );
geometry.rotateX( - Math.PI / 2 );
var material = new THREE.MeshStandardMaterial( { flatShading: true } );
var mesh1 = new THREE.Mesh( geometry, material );
var mesh2 = new THREE.Mesh( geometry, material );

var controller_head = new THREE.Mesh( new THREE.IcosahedronBufferGeometry( 0.01, 2 ), new THREE.MeshLambertMaterial( { color: 0xd10000 } ) );
controller_head.name = 'controller_head';
controller_head.position.z = - 0.05;
mesh1.add( controller_head );


var geometry = new THREE.BufferGeometry().setFromPoints( [ new THREE.Vector3( 0, 0, 0 ), new THREE.Vector3( 0, 0, - 0.05 ) ] );

var line2 = new THREE.Line( geometry );
line2.name = 'line2';
line2.scale.z = 5;
mesh2.add( line2 );




controller1.add( mesh1.clone() );
controller2.add( mesh2.clone() );

raycaster = new THREE.Raycaster();
```

Εικόνα 5.4 - Controller1, Controller2, Raycaster

Η initGeometry είναι η δεύτερη βασική συνάρτηση της εφαρμογής η οποία είναι υπεύθυνη για να αρχικοποιήσει το παραγόμενο «υλικό» που βγαίνει από το head του controller1. Η μεταβλητή «geometry» είναι τύπου BufferGeometry ακριβώς επειδή θα της αποδοθούν τυχαίες τιμές (positions, normals, κλπ) από το χρήστη ανάλογα με την κίνηση του controller. Η «geometry.drawRange.count» ισούται με μηδέν για να μη γίνει Render κανένα vertex, ενώ το «geometry.position» ορίζεται να είναι ίδιο με το position του controller1. Ακολουθεί ένα material και ένα mesh (line) με attributes την παραπάνω γεωμετρία και material. Η Boolean τιμή του line.frustumCulled τίθεται «false» για να εξασφαλίζεται ότι το 3D object θα γίνεται Render σε κάθε frame ακόμα και αν δεν το «βλέπει» η κάμερα. Το mesh προστίθεται στην ομάδα «group_mesh» και η «group_mesh» προστίθεται στη σκηνή. Στη συνέχεια η scene.updateMatrix() κάνει update τη σκηνή.

```
var geometry = new THREE.BufferGeometry();

var positions = new THREE.BufferAttribute( new Float32Array( 1000000 * 3 ), 3 );
positions.dynamic = true;
geometry.addAttribute( 'position', positions );

var normals = new THREE.BufferAttribute( new Float32Array( 1000000 * 3 ), 3 );
normals.dynamic = true;
geometry.addAttribute( 'normal', normals );
```

Εικόνα 5.5 - Geometry, position, normals

```
geometry.drawRange.count = 0;
geometry.position = controller1.position;

var material = new THREE.MeshStandardMaterial( {
    roughness: 0.9,
    metalness: 0.0,
    vertexColors: THREE.VertexColors
} );

line = new THREE.Mesh( geometry, material );
line.frustumCulled = false;
//scene.add( line );
group_mesh.add( line );
scene.add( group_mesh );
scene.updateMatrix();
```

Εικόνα 5.6 - group_mesh

Η συνάρτηση «updateGeometry» χρησιμοποιείται χρησιμοποιείται για να γίνουν update όλα τα attributes της «initGeometry». Η συνάρτηση «stroke» είναι υπεύθυνη για να υπολογίζει και να αποθηκεύει τις θέσεις στο χώρο που κινείται ο controller1. Παίρνει σαν ορίσματα τον controller1, και τέσσερις πίνακες, τους point1, point2, matrix1, matrix2. Στη συνέχεια διανύσματα που έχουν οριστεί στην αρχή ως global μεταβλητές παίρνουν τις τιμές των κινήσεων του χρήστη. Η «stroke» καλείται στη συνάρτηση «handleController1» όταν ο χρήστης πατάει τη σκανδάλη. Αφού υπάρχουν και τα διανύσματα και τα σημεία της διαδρομής του controller, η συνάρτηση «render» αναλαμβάνει να τα αποδώσει στην οθόνη ή στο headset του χρήστη. Η συνάρτηση «render» καλείται μέσα στην «animate» η οποία είναι η Τρίτη βασική συνάρτηση που καλείται στην αρχή. Βέβαια η «render» δεν αναλαμβάνει μόνο να σχεδιάζει την κίνηση του controller, αλλά ορίζεται σε αυτήν και το rotation του πλανήτη, ενώ καλείται και η συνάρτηση «intersectObjects».

```
function render() {

    tatooineMesh.rotation.y += 0.0005;

    cleanIntersected();

    var count = line.geometry.drawRange.count;


    handleController1( controller1 );
    intersectObjects( controller2 );

    updateGeometry( count, line.geometry.drawRange.count );



    renderer.render( scene, camera );

}
```

Εικόνα 5.7- Render function

Η «intersectObjects» με τη χρήση της «getIntersections» και του raycaster που αυτή περιλαμβάνει, αναλαμβάνει να βρίσκει πότε ο δείκτης του controller2 έρχεται σε επαφή με κάποιο αντικείμενο, στην προκειμένη περίπτωση τα κουμπιά κίνησης-περιστροφής.

Συμπεράσματα

Οι εφαρμογές εικονικής πραγματικότητας με την πάροδο του χρόνου εξελίσσονται. Όσον αφορά τους browsers προσαρμόζονται στις απαιτήσεις, αλλάζουν και ενσωματώνουν νέα στοιχεία ικανά να εμπλουτίσουν την εμπειρία του χρήστη σε περιβάλλοντα εικονικής πραγματικότητας. Βέβαια ακόμα πρέπει να γίνει πολλή δουλειά όσον αφορά τη διαχείριση μεγάλων τρισδιάστατων μοντέλων καθώς η πληθώρα μοντέλων σε μια σκηνή μπορεί να προκαλέσει δυσάρεστα κολλήματα της κίνησης και να μειώσει την εμπειρία στο χρήστη, κυρίως σε εφαρμογές gaming.

Τεράστια είναι η συνεισφορά του ThreeJS, το οποίο συνεχώς ανανεώνεται, προσφέροντας καινούριες δυνατότητες στους προγραμματιστές. Χαρακτηριστικό παράδειγμα είναι ότι, τη στιγμή που γράφεται αυτή η παράγραφος, στη σελίδα του ThreeJs τα παραδείγματα που μέχρι πρότινος ήταν «webvr» εδώ και μερικές μέρες είναι «webxr» καθώς έχουν προστεθεί δυνατότητες επαυξημένης πραγματικότητας στη βιβλιοθήκη.

Σχετικά με την εργασία θα μπορούσε να χρησιμοποιηθεί από κάποιον φοιτητή που ενδιαφέρεται να αναπτύξει τη δική του εργασία πάνω σε VR εφαρμογές. Επιπλέον θα μπορούσε, με τις κατάλληλες τροποποιήσεις και προσθήκες επιλογών, να διατεθεί στο εμπόριο.

Κεφάλαιο 6: Βιβλιογραφία

Βιβλιογραφία

- https://threejsfundamentals.org/threejs/lessons/threejs-webvr.html
- https://threejsfundamentals.org/threejs/lessons/threejs-webvr-point-to-select.html
- https://www.lexico.com/en/definition/virtual_reality
- http://edutechwiki.unige.ch/en/Web_3D_technology
- https://medium.com/@bluemagnificent/intro-to-javascript-3d-physics-using-ammo-js-and-three-js-dd48df81f591
- https://medium.com/whitestormjs-framework/porting-bullet-physics-into-ammo-js-and-improving-physi-js-f0130c372f91
- https://github.com/kripken/ammo.js/
- http://xml.coverpages.org/vrml-X3D.html
- https://en.wikipedia.org/wiki/WebGL
- https://www.web3d.org/x3d-vrml-most-widely-used-3d-formats
- https://www.khronos.org/registry/webgl/specs/latest/1.0/
- https://www.khronos.org/registry/webgl/specs/latest/2.0/
- http://paulbourke.net/dataformats/vrml1/
- http://edutechwiki.unige.ch/en/X3DV#VRML_Definition
- https://www.web3d.org/standards
- https://en.wikipedia.org/wiki/HTC_Vive
- https://www.vrheads.com/exposing-magic-behind-htc-vive-controller
- https://www.web3d.org/wiki/index.php/X3D_and_HTML5
- https://gizmodo.com/this-is-how-valve-s-amazing-lighthouse-tracking-technol-1705356768
- https://w3c.github.io/gamepad/
- https://pybullet.org/Bullet/BulletFull/
- https://pybullet.org/Bullet/BulletFull/annotated.html
- https://pybullet.org/Bullet/BulletFull/namespaceBullet.html
- «Endoscopic surgery training using virtual reality and deformable tissue simulation» - U. KuK hnapfel*, H.K. C7 akmak, H. Maass
- «Virtual Reality in Medicine» - K.-F. Kaltenborn, O. Rienhoff
- «Virtual Reality: Definitions, History and Applications» - Michael A. Gigante
- «Virtual Reality History, Applications, Technology and Future» - Tomasz Mazuryk and Michael Gervautz
- «Virtual reality and its military utility» - Ajey Lele
- «X3D: Extensible 3D Graphics Standard» - Leonard Daly and Don Brutzman