

Ιόνιο Πανεπιστήμιο

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ



- Πτυχιακή Εργασία -

Σχεδίαση και Ανάπτυξη ενός Ολοκληρωμένου Τρισδιάστατου Περιβάλλοντος σε WebGL

Ρέστας Αθανάσιος Π2012050

Επιβλέπων : Μυλωνάς Φοίβος

1/2/2020

Επιβλέπων

Μυλωνάς Φοίβος, Αναπληρωτής Καθηγητής,
Ιόνιο Πανεπιστήμιο

Τριμελής Επιτροπή

Μυλωνάς Φοίβος, Αναπληρωτής Καθηγητής,
Ιόνιο Πανεπιστήμιο

Στεφανιδάκης Μιχαήλ, Επίκουρος Καθηγητής
Ιόνιο Πανεπιστήμιο

Τσουμάκος Δημήτριος, Αναπληρωτής Καθηγητής
Ιόνιο Πανεπιστήμιο

Σε όλους όσους στάθηκαν δίπλα μου

ΠΕΡΙΛΗΨΗ

Ο τομέας των γραφικών υπολογιστών τα τελευταία έτη έχει αρχίσει να κεντρίζει το ενδιαφέρον φορέων από τον ακαδημαϊκό αλλά και τον βιομηχανικό χώρο. Με τη πάροδο των χρόνων οι τεχνολογίες απόδοσης ψηφιακών γραφικών καθώς και η καθιέρωση των προσωπικών υπολογιστών, των έξυπνων κινητών και των οικιακών κονσολών ψυχαγωγίας ως απαραίτητα στοιχεία στη ζωή ενός ατόμου, έκαναν την ανάγκη για την ύπαρξη διαδραστικών εφαρμογών πιο σημαντική από ποτέ.

Όλα τα παραπάνω δημιουργούν συχνά σύγχυση σε αρχάριους προγραμματιστές σχετικά με τον τομέα του σχεδιασμού εικονικών κόσμων. Για τον λόγο αυτόν στα πλαίσια της παρούσας εργασίας θα πραγματοποιήσουμε αρχικά μία ιστορική αναδρομή στο χώρο των γραφικών υπολογιστών με έμφαση στις κομβικές στιγμές που άλλαξαν την βιομηχανία ψυχαγωγίας και έπειτα θα ερευνήσουμε τις βασικές τεχνικές ανάπτυξης μίας μηχανής γραφικών καθώς και των κύριων συστημάτων τους. Έπειτα θα εμβαθύνουμε στον τομέα της ανάπτυξης βιντεοπαιχνιδιών καθώς και στις σημαντικότερες λειτουργίες μίας μηχανής παραγωγής παιχνιδιών.

Τέλος θα παρουσιασθούν τα βήματα ανάπτυξης ενός ολοκληρωμένου τρισδιάστατου περιβάλλοντος, μέσα από την ανάλυση των σταδίων δημιουργίας ενός βιντεοπαιχνιδιού πρώτου προσώπου χρησιμοποιώντας τις δυνατότητες της τεχνολογίας WebGL μέσα από τη γλώσσα προγραμματισμού Javascript και το πλαίσιο ανάπτυξης διαδραστικών εφαρμογών BabylonJS.

Πρόλογος και Ευχαριστίες

Η παρούσα εργασία είναι το αποτέλεσμα της προσπάθειας να απομυθοποιηθεί ο τομέας των γραφικών υπολογιστών και της ανάπτυξης βιντεοπαιχνιδιών, τομέας που συχνά φαντάζει αχανής για έναν αρχάριο προγραμματιστή ο οποίος δεν είναι σε θέση να επιλέξει τα σωστά εργαλεία ώστε να καταλήξει στο σκοπό του, γεγονός που τις περισσότερες φορές τον οδηγεί στο να τα παρατήσει. Μέσα από τις σελίδες που ακολουθούν αλλά και με τη προσεκτική μελέτη του πηγαίου κώδικα που αναπτύχθηκε, θα μπορέσει κανείς να πάρει τις βάσεις με όσο πιο απλό τρόπο γίνεται ώστε να δημιουργήσει οτιδήποτε έχει στη φαντασία του, χρησιμοποιώντας εργαλεία που είναι σχεδιασμένα από προγραμματιστές για προγραμματιστές. Τα εργαλεία και οι μέθοδοι που χρησιμοποιήθηκαν για την ανάπτυξη του εκτελέσιμου της εργασίας αποτελούν τη χρυσή τομή των πολύ υψηλού αλλά και των πολύ χαμηλού επιπέδου επιλογών που είναι διαθέσιμες. Με αυτό τον τρόπο επιτεύχθηκε η απόκτηση γνώσεων που μπορούν να μεταφερθούν σε υπόλοιπους τομείς του τομέα της επιστήμης υπολογιστών, κάνοντας το όλο εγχείρημα της εργασίας μία ιδιαιτέρως διδακτική εμπειρία. Γνώσεις των δυνατοτήτων μίας γλώσσας προγραμματισμού, των μοντέλων προγραμματισμού, η ικανότητα να μελετάς το εγχειρίδιο ενός API μαζί με το πάθος για τη γνώση και τη δημιουργία είναι τα στοιχεία που θα ωθήσουν τον καθένα προς την επίτευξη των στόχων του.

Όλα τα παραπάνω συνοψίζουν το γιατί για μένα η εργασία αυτή αποτελούσε ένα ταξίδι ανάπτυξης των δεξιοτήτων μου αλλά και του χαρακτήρα μου. Ταξίδι δύσκολο που δε θα μπορούσα να κάνω ποτέ μόνος, οπότε σε αυτό το σημείο θα ήθελα να ευχαριστήσω την οικογένεια μου η οποία είναι δίπλα μου σε όλα τα στάδια της ζωής μου και με στηρίζει με το δικό της μοναδικό τρόπο απέναντι σε κάθε δυσκολία.

Όσον αφορά τον ευρύτερο μου κύκλο, επειδή θα χρειαζόμουν δεκάδες σελίδες για να αναφέρω τα ονόματα που έχω στο μυαλό μου, θα ήθελα να ευχαριστήσω όλους τους φίλους και γνωστούς για την αδιάκοπη αγάπη τους για το πρόσωπο μου και για τις στιγμές που πίστεψαν πιο πολύ σε μένα από ότι εγώ. Η αγάπη αυτή ήταν που έκανε το πληκτρολόγιο να μοιάζει γεμάτο από δεκάδες χέρια στις πιο δύσκολες στιγμές και που με ώθησε να ολοκληρώσω αυτό που είχα ξεκινήσει.

Σας ευχαριστώ όλους από τα βάθη της καρδιάς μου.

Θάνος

1 Φεβρουαρίου 2020

Περιεχόμενα

ΚΕΦΑΛΑΙΟ Α΄	1
Α.1 Εισαγωγή	1
Α.2 Σκοπός της εργασίας	2
ΚΕΦΑΛΑΙΟ Β΄	2
Β.1 Ιστορική αναδρομή	3
Β.2 Βιντεοπαιχνίδια – εικονικοί κόσμοι και τεχνικές ανάπτυξης	21
Β.2.1 Τι είναι ένα παιχνίδι	21
Β.2.2 Βιντεοπαιχνίδια ως προσομοιώσεις πραγματικού χρόνου	21
Β.2.3 Μηχανές παραγωγής παιχνιδιών	24
Β.2.4 Τύποι μηχανών παραγωγής παιχνιδιών	26
Β.2.5 Μηχανές παραγωγής παιχνιδιών	38
Β.2.6 Αρχιτεκτονική σχεδιασμού εικονικού περιβάλλοντος	43
ΚΕΦΑΛΑΙΟ Γ΄	61
Γ.1 Η γλώσσα προγραμματισμού Javascript	62
Γ.1.1 Γενικές πληροφορίες	62
Γ.1.2 Γιατί επιλέχθηκε η γλώσσα Javascript	64
Γ.2 Διεπαφή προγραμματισμού εφαρμογών WebGL	65
Γ.2.1 Γενικά χαρακτηριστικά	66
Γ.2.2 Διασωλήνωση γραφικών στη WebGL	68
Γ.2.3 Μνήμη και WebGL	69
Γ.3 Το πλαίσιο BabylonJS	69
Γ.3.1 Δυνατότητες	70
Γ.3.2 Παραδείγματα τμημάτων κώδικα	72
Γ.3.3 Αιτιολόγηση χρήσης	75
Γ.4 Το εργαλείο σχεδιασμού Blender	76
Γ.4.1 Γενικές πληροφορίες	77
Γ.4.2 Αιτιολόγηση χρήσης του Blender	77
ΚΕΦΑΛΑΙΟ Δ΄	79
Δ1 Γενικές πληροφορίες και Σκοπός Παιχνιδιού	79
Δ2 Αναλυτική παρουσίαση κώδικα	83
Δ2.1 Αρχείο index.js	84
Δ2.2 Αρχείο engine.js	84
Δ2.3 Αρχείο character.js	89
Δ2.4 Αρχείο weapon.js	93
Δ2.5 Αρχείο collectible.js	95
Δ2.6 Αρχεία κλάσεων εχθρών	97
Κεφάλαιο Ε΄	102
Ε.1 Στιγμιότυπα ανάπτυξης	102
Ε.2 Μελλοντικές βελτιώσεις	112
Ε.3 Συμπεράσματα	113
Βιβλιογραφία	114
ΠΗΓΕΣ	114
ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ	115
ΕΥΡΕΤΗΡΙΟ ΤΜΗΜΑΤΩΝ ΚΩΔΙΚΑ	116
ΕΥΡΕΤΗΡΙΟ ΣΤΙΓΜΙΟΤΥΠΩΝ ΕΚΤΕΛΕΣΙΜΟΥ	117

ΚΕΦΑΛΑΙΟ Α΄

A.1 Εισαγωγή

Η είσοδος των ηλεκτρονικών υπολογιστών ως εργαλείο στο χώρο της επιστήμης παρείχε τη δυνατότητα στους ανθρώπους να δημιουργήσουν εφαρμογές προσομοίωσης του φυσικού κόσμου σε διάφορα γραφικά περιβάλλοντα τριών διαστάσεων. Η ανάπτυξη του τομέα των Γραφικών Υπολογιστών ήταν πλέον γεγονός. Οι τομείς της πληροφορικής, της αρχιτεκτονικής, των θετικών επιστημών άλλαξαν ριζικά. Πληθώρα εργαλείων και εφαρμογών έπαιζαν καθοριστικό ρόλο στο σχεδιασμό κτιρίων, τεχνολογικών θαυμάτων, στην διεξαγωγή πειραμάτων μέσω προσομοιώσεων του φυσικού κόσμου και στην απεικόνιση δεδομένων.

Φυσικά από τα παραπάνω δε θα μπορούσε να λείπει και η αλλαγή που υπήρξε στη βιομηχανία της ψυχαγωγίας. Τηλεοράσεις και ηλεκτρονικοί υπολογιστές πήραν τη θέση τους στη ζωή κάθε καταναλωτή και σε συνδυασμό με την καθιέρωση, αρχικά, των κινητών συσκευών και έπειτα την εξέλιξη τους σε έξυπνες συσκευές η ανάγκη για την παραγωγή πολυμεσικού και διαδραστικού περιεχομένου αυξήθηκε ραγδαία. Η ζωή κάθε ατόμου πλέον κατακλύζεται από οθόνες οι οποίες είναι διαθέσιμες να προβάλουν κάθε λογής οπτικό περιεχόμενο.

Παραδείγματα μπορεί να βρει κανένας σε κάθε οθόνη που συναντάει στη καθημερινότητα του. Διαφημιστικές πινακίδες προβάλλουν τρισδιάστατες ψηφιακές αναπαραστάσεις προϊόντων, ταινίες μεγάλης διάρκειας δε περιέχουν, παρά ελάχιστα, φυσικά τοπία και αντικείμενα που σε συνδυασμό με ειδικά αναπτυγμένα οπτικά εφέ κάνουν τον καθένα να αμφισβητεί το γεγονός πως γυρίστηκαν μέσα σε μικρά στούντιο. Επιπρόσθετα οι χώροι της τέχνης και του παιχνιδιού με τη βοήθεια των νέων μεθόδων δημιουργίας τεχνητών ψηφιακών κόσμων ενώθηκαν στο όνομα της βιομηχανίας βιντεοπαιχνιδιών παρέχοντας ψυχαγωγικές, και όχι μόνο, εμπειρίες με κύριο γνώμονα τη διάδραση.

Ανθρωποι έχουν δημιουργήσει ψηφιακούς κόσμους που δεν έχουν να ζηλέψουν τίποτα από τη πραγματικότητα, οπτικά εφέ που μετατρέπουν κάθε άγγιγμα της οθόνης ενός έξυπνου κινητού σε κάτι μοναδικό και διαδραστικές εμπειρίες που μπορούν να καθηλώσουν μικρούς και μεγάλους.

Σήμερα η ανάγκη για διάδραση με τα εικονικά περιβάλλοντα που έχουν δημιουργηθεί είναι πιο αυξημένη από ποτέ. Η άνοδος των τεχνολογιών της εικονικής και της επαυξημένης πραγματικότητας καθώς και του συνδυασμούς τους, της μικτής πραγματικότητας σβήνει σιγά σιγά τη διαχωριστική γραμμή του αληθινού με το ψηφιακό.

Με αυτό τον τρόπο μπορούμε εύκολα να καταλήξουμε στο συμπέρασμα πως η ανάγκη για τη δημιουργία τρισδιάστατων εικονικών περιβαλλόντων είναι πιο αυξημένη από ποτέ με εφαρμογές αυτών να συναντώνται σε πληθώρα μέσων.

A.2 Σκοπός της εργασίας

Η παρουσίαση της πτυχιακής εργασίας που ακολουθεί αναλύει τα στάδια της δημιουργίας ενός διαδραστικού τρισδιάστατου εικονικού περιβάλλοντος μέσα από την ανάπτυξη ενός βιντεοπαιχνιδιού πρώτου προσώπου (First Person Shooter - FPS) με τη χρήση του μέσου διεπαφής WebGL.

Με κύριο γνώμονα την απλότητα και την αποφυγή πολύπλοκων μεθόδων προγραμματισμού θα παρακολουθήσει κανείς την δημιουργία ενός βιντεοπαιχνιδιού κι έτσι μετά από την εξέταση του πηγαίου κώδικα που θα παρουσιασθεί να είναι σε θέση να ακολουθήσει και ο ίδιος ένα παρόμοιο εγχείρημα. Με αυτό το τρόπο ο οποιοσδήποτε θα μάθει πως μέσα από δωρεάν εργαλεία ανοιχτού κώδικα έχει τη δυνατότητα να γίνει και αυτός μέρος της τεχνολογικής επανάστασης στο τομέα των Γραφικών Υπολογιστών και της ανάπτυξης εφαρμογών, τη στιγμή που αυτός βρίσκεται στο ζενίθ του.

Θα εξετασθούν τα βασικά εργαλεία προγραμματισμού και σχεδιασμού που μπορούν να χρησιμοποιηθούν για την ανάπτυξη του παραπάνω εγχειρήματος, οι τεχνικές που κάνουν τα βιντεοπαιχνίδια αποτελεσματικά ως μέσα για να πάρει ζωή ένα εικονικό περιβάλλον καθώς και οι προγραμματιστικές αρχιτεκτονικές και τα μοτίβα σχεδιασμού που έχουν αναπτυχθεί και χρησιμοποιούνται στη βιομηχανία από τις αρχές του τομέα ως σήμερα.

Όλα τα παραπάνω πραγματοποιούνται με απώτερο σκοπό την παρότρυνση του αναγνώστη να απομυθοποιήσει την ανάπτυξη εφαρμογών εικονικών περιβαλλόντων και βιντεοπαιχνιδιών πρόβλημα που αρκετά συχνά εντοπίζεται σε επίδοξους αρχάριους στη προσπάθεια τους να ασχοληθούν με τον τομέα.

ΚΕΦΑΛΑΙΟ Β'

Προαπαιτούμενα

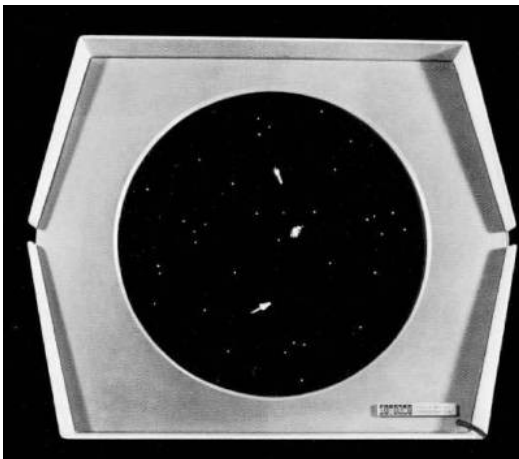
Στη παρούσα ενότητα παρουσιάζεται η βάση γνώσης πάνω στην οποία στηρίχθηκε η εκπόνηση της πτυχιακής εργασίας, αρχικά με μία σύντομη αναδρομή στο ιστορικό υπόβαθρο του τομέα των Γραφικών Υπολογιστών, των βασικών τεχνικών στην ανάπτυξη ενός διαδραστικού εικονικού

περιβάλλοντος - βιντεοπαιχνιδιού και τέλος με την παρουσίαση των εργαλείων, καθώς και τους λόγους, που χρησιμοποιήθηκαν κατά τη διαδικασία ανάπτυξης του τελικού εκτελέσιμου.

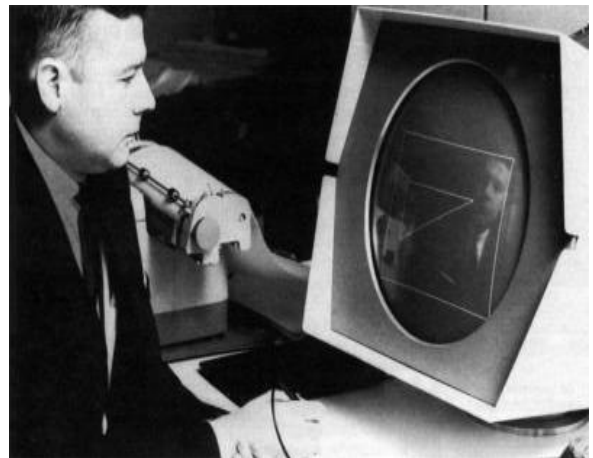
B.1 Ιστορική αναδρομή

Στο διάστημα **1940-1960** η χρήση των ηλεκτρονικών υπολογιστών επεκτάθηκε με γρήγορους ρυθμούς ξεκινώντας από την απλή επεξεργασία μεγάλου όγκου αριθμητικών πράξεων μέχρι και την αποκορύφωση τους στη διεξαγωγή ερευνητικών πειραμάτων με σκοπό την απόκτηση αποτελεσμάτων που έχουν ακρίβεια ίση με αυτών που διεξήχθησαν στο φυσικό κόσμο.

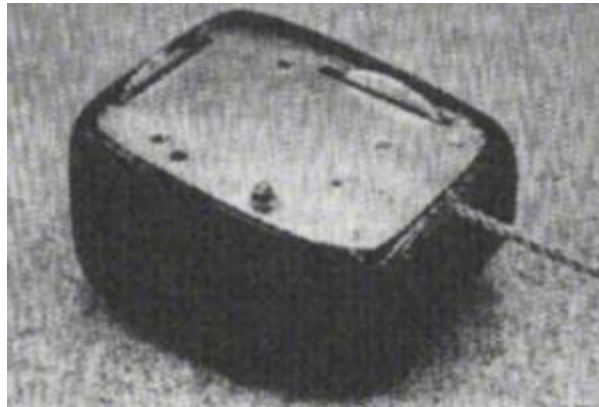
Έπειτα η χρονιά **1961** θα ήταν η πραγματική αρχή των όσων θα διαπραγματευτούμε στη παρούσα εργασία. Το πρώτο βιντεοπαιχνίδι με δημιουργό τον Steve Russel, επιστήμων υπολογιστών και δημιουργό της συναρτησιακής γλώσσας Lisp, ήταν γεγονός και ονομαζόταν Space War. Το περιβάλλον του παιχνιδιού προβαλλόταν μέσω ενός οσillosκοπίου, οθόνης που ανιχνεύει και προβάλλει τις αλλαγές στη τάση του ρεύματος, σε δύο διαστάσεις. Την ίδια χρονιά το πρώτο πρόγραμμα σχεδίασης, σε δυο διαστάσεις, με όνομα Sketchpad δημιουργείται από τον Ivan Sutherland το οποίο περιελάμβανε και στοιχεία μενού στο γραφικό του περιβάλλον. Το **1963**, δημιουργείται το ηλεκτρονικό ποντίκι.



Εικόνα 1 SpaceWar, το πρώτο βιντεοπαιχνίδι



Εικόνα 2: Sketchpad, το πρώτο πρόγραμμα σχεδίασης



Εικόνα 3: Το ηλεκτρονικό ποντίκι

Το 1965 ο Jack Bresenham επινοεί των “ιδανικό” αλγόριθμο σχεδίασης ευθειών γραμμών. Με βάση αυτόν, μία γραμμή μπορεί να αναπαρασταθεί σε ένα σύνολο από pixels με τη βοήθεια των αρχικών (x_1, y_1) και τελικών (x_2, y_2) συντεταγμένων. Ο αλγόριθμος πραγματοποιεί τον τελικό σχεδιασμό της γραμμής επιλέγοντας τα “ιδανικά” σημεία που την αποτελούν και βρίσκονται ενδιάμεσα από τις δύο προηγούμενες συντεταγμένες που αναφέρθηκαν.

The points (x_1, y_1) and (x_2, y_2) are assumed
not equal and integer valued.
 ϵ is assumed to be real.

Let $\Delta x = x_2 - x_1$

Let $\Delta y = y_2 - y_1$

Let $m = \frac{\Delta y}{\Delta x}$

Let $j = y_1$

Let $\epsilon = m - 1$

for $i = x_1$ to $x_2 - 1$

illuminate (i, j)

if $(\epsilon \geq 0)$

$j += 1$

$\epsilon -= 1.0$

end if

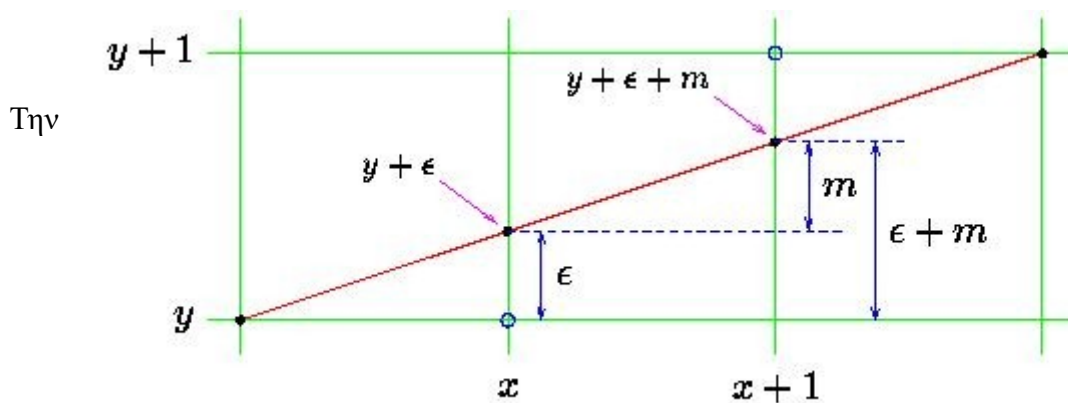
$i += 1$

$\epsilon += m$

next i

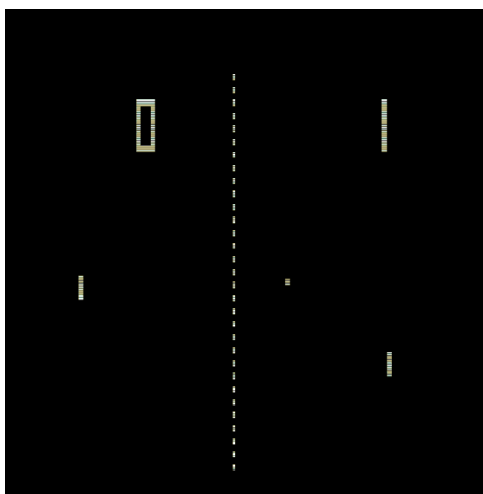
finish

Εικόνα 4: Ο αλγόριθμος σχεδίασης ευθείας γραμμής



Εικόνα 5: Παρουσίαση σχεδίασης γραμμής

περίοδο **1970 – 1981** υπήρξε μια αλληλουχία αλλαγών με πιο σημαντικές τη διεύρυνση της αγοράς προγραμμάτων σχεδίασης με ηλεκτρονικό υπολογιστή(Computer Assisted Design – CAD) τη δημιουργία του βιντεοπαιχνιδιού Pong, το οποίο για πρώτη φορά κάνει χρήση οθονών που χρησιμοποιούν pixels(Raster displays) , καθώς και η κυκλοφορία του πρώτου προσωπικού υπολογιστή που υποστήριζε γραφικό περιβάλλον για καταναλωτές, του Apple 2.



Εικόνα 6: Pong, το πρώτο βιντεοπαιχνίδι σε Raster Displays

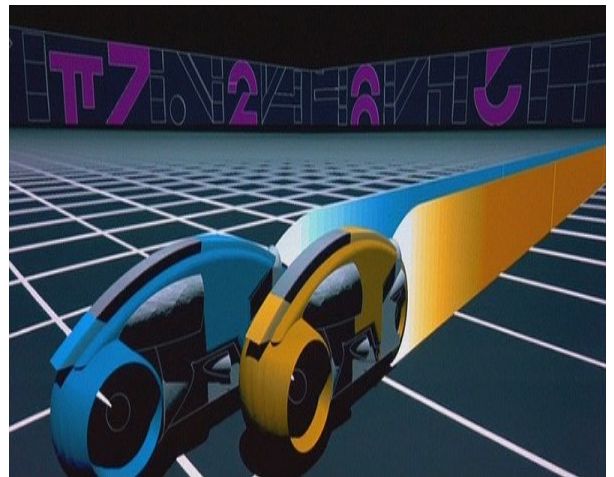


Εικόνα 7: Apple 2 : Ο πρώτος προσωπικός υπολογιστής με γραφικό περιβάλλον

Οι χρονιές **1982-1993** έφεραν στον κόσμο τη ταινία Tron, που ήταν η πρώτη η οποία έκανε εκτενέστερη χρήση γραφικών που είχαν σχεδιαστεί σε υπολογιστή καθώς και τη πρώτη έκδοση του , γνωστού πλέον σχεδιαστικού προγράμματος, AutoCAD. Το στούντιο παραγωγής ταινιών Ρίχαρ ανοίγει τις πόρτες του και εκδίδει μια ταινία 2 λεπτών υλοποιημένη με τη χρήση γραφικών υπολογιστών. Η περίοδος αυτή κλείνει με τη κυκλοφορία της πρώτης ταινίας μεγάλου budget που θα κάνει χρήση οπτικών εφέ παραγμένα από ηλεκτρονικό υπολογιστή(Computer Generated Imagery - CGI), Jurassic Park.



Εικόνα 8: *Luxor Jr.* Pixar 1986, διάρκεια 2 λεπτών



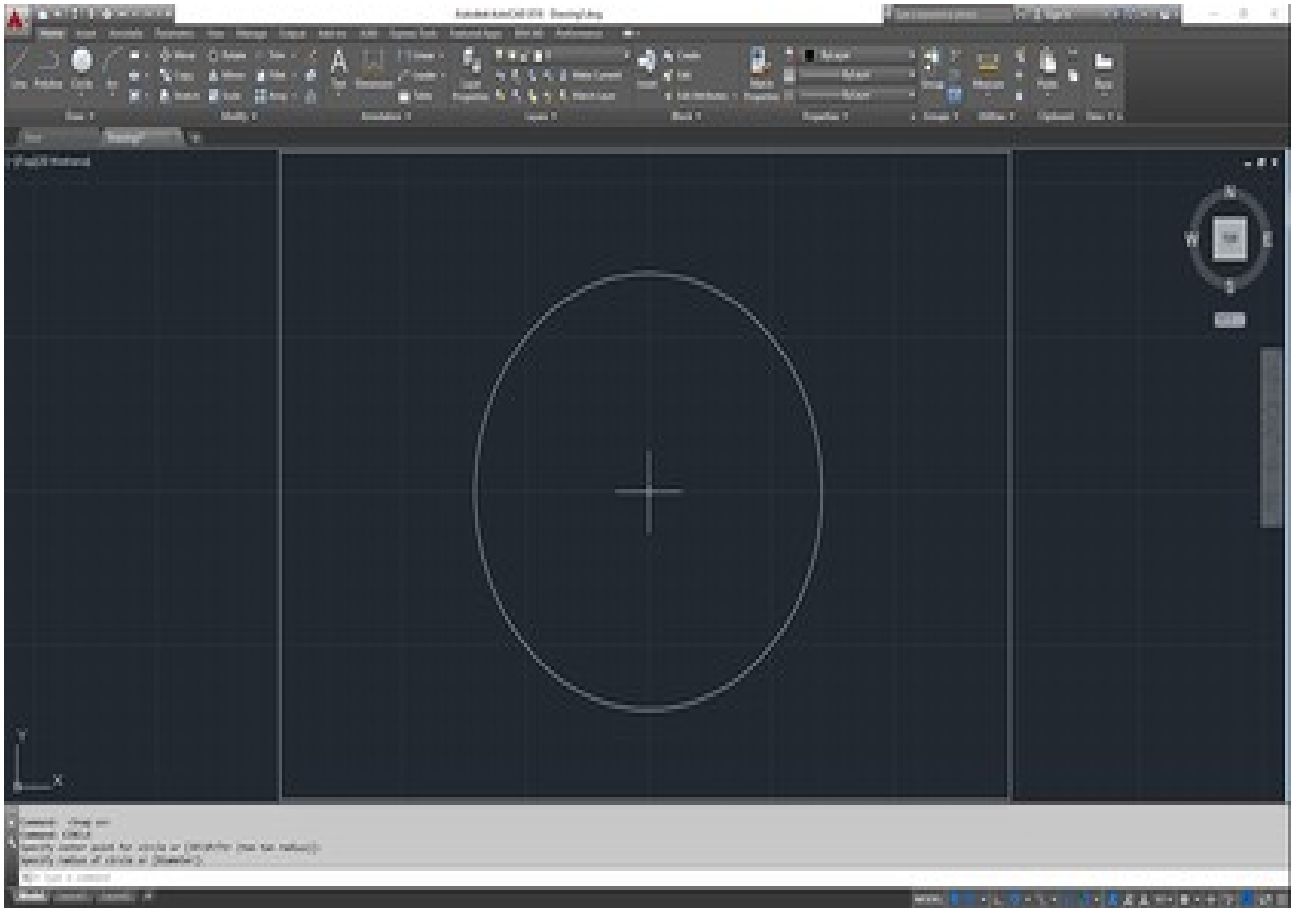
Εικόνα 9 Χρήση ψηφιακών γραφικών στη ταινία *Tron*



Εικόνα 10 *Jurassic Park* 1993, απόσπασμα που παρουσιάζει τη χρήση *Computer Generated Imagery*

Η ραγδαία εξέλιξη συνεχίστηκε τις χρονιές **1994-2006**. Αλλαγές όπως η κυκλοφορία της πρώτης τρισδιάστατης (3 Dimensional – 3D) έκδοσης του AutoCAD από την εταιρεία, με το συντριπτικά μεγαλύτερο μερίδιο της αγοράς σχεδιαστικών προγραμμάτων, Autodesk. Επίσης πλέον αποτελεί πραγματικότητα η πρώτη πτήση του Boeing 777 του οποίου οι δοκιμαστικές πτήσεις και πειράματα ασφαλείας διεξήχθησαν μέσω του προγράμματος προσομοιώσεων Analysis Systems (Ansys). Το 1995 θα φέρει τη κυκλοφορία από το στούντιο παραγωγής ταινιών Pixar της πρώτης ταινίας μεγάλου μήκους που σχεδιάστηκε εξολοκλήρου με τη χρήση Computer Generated Imagery, Toy Story, και σε συνδυασμό με τη κυκλοφορία του Solidworks, προγράμματος το οποίο θα αποτελέσει

μετέπειτα κύριο συστατικό στην εργαλειοθήκη κάθε μηχανικού, η βιομηχανία γύρω από τον κόσμο των γραφικών υπολογιστών και της παραγωγής εικονικών περιβαλλόντων είτε για χρήση στη βιομηχανία είτε για λόγους ψυχαγωγίας δεν θα είναι ποτέ ίδια.



Εικόνα 11: AutoCAD 3D, το περιβάλλον της πρώτης τρισδιάστατης έκδοσης



Εικόνα 12: Toy Story, Pixar 1995, ταινία εξ ολοκλήρου φτιαγμένη με Computer Generated Imagery

Οι οικιακές κονσόλες βιντεοπαιχνιδιών έχουν κατακλύσει την αγορά. Το παιχνίδι Star Fox στη κονσόλα Super Nintendo Entertainment System γίνεται το πρώτο που κάνει δημοφιλή τα βιντεοπαιχνίδια τριών διαστάσεων και έπειτα ακολουθεί η Sony λανσάροντας το PlayStation. Τα βιντεοπαιχνίδια πλέον αφήνουν πίσω τις δύο διαστάσεις και προσφέρουν ψυχαγωγικές εμπειρίες με τα τρισδιάστατα περιβάλλοντα στο επίκεντρο.

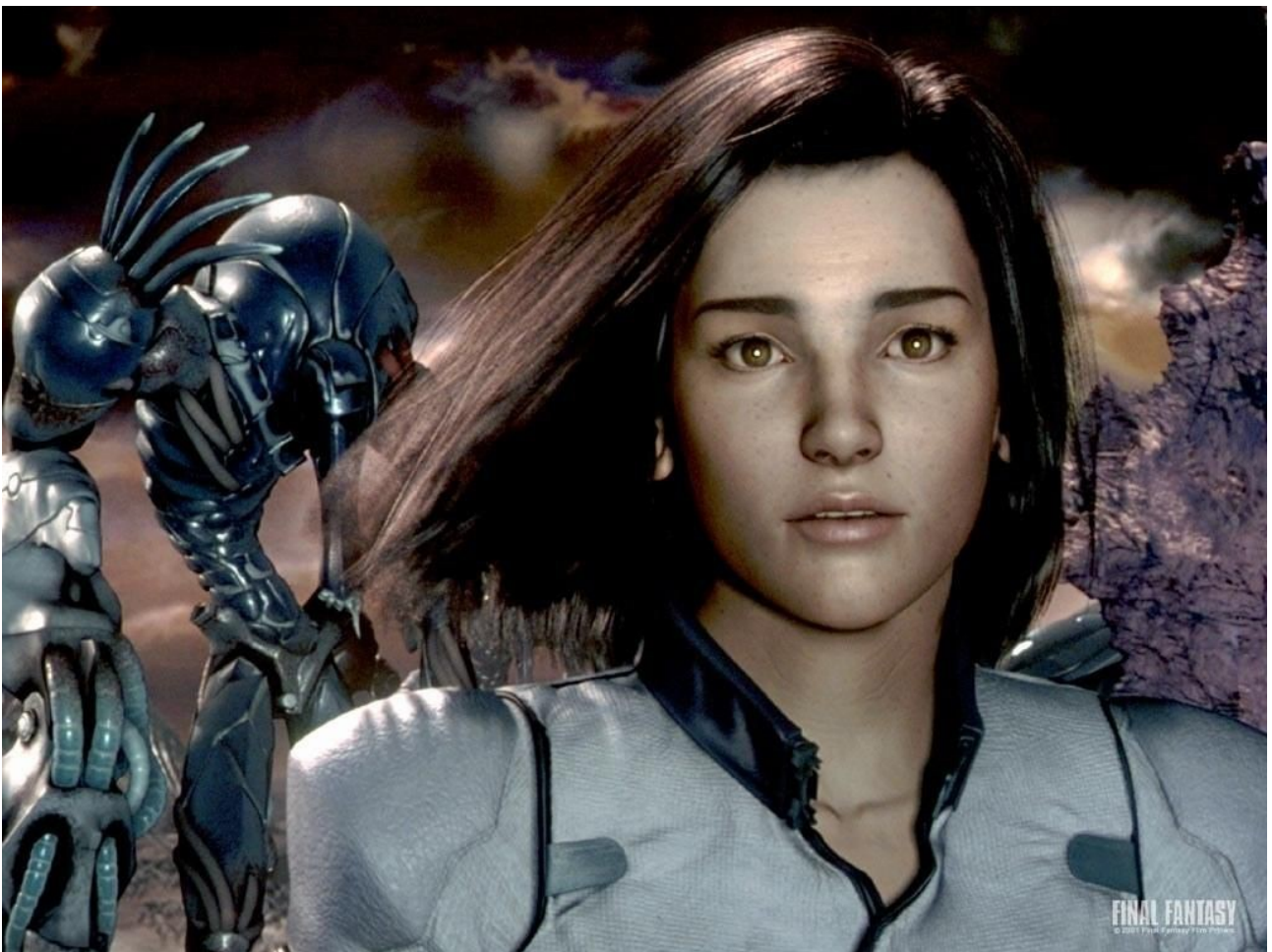


Εικόνα 13: Crash Bandicoot, Playstation 1 1996



Εικόνα 14: Star Fox, Super Nintendo Entertainment System 1993

Η κυκλοφορία του Sketchup, 2000, του πρώτου σχεδιαστικού προγράμματος έφερε νέες δυνατότητες στην βιομηχανία των γραφικών αφού παρείχε υπηρεσίες σχεδιασμού τρισδιάστατων περιβαλλόντων μέσα σε φυλλομετρητή διαδικτύου(web browser). Επιπρόσθετα την επόμενη χρονιά η ικανότητα ψηφιακής σχεδίασης υγρών γίνεται για πρώτη φορά εφικτή. Παρόλα αυτά η κίνηση των μαλλιών παραμένει αδύνατο έργο. Η ταινία Final Fantasy, βασισμένη στο ομώνυμο βιντεοπαιχνίδι ρόλων, κάνει χρήση των παραπάνω τεχνικών για την απεικόνιση δακρύων.



Εικόνα 15: Final Fantasy, Square Enix

Το 2003 κυκλοφόρησε το βιντεοπαιχνίδι πρώτου προσώπου DOOM 3 από την Interactive Development. Με τη χρήση της μηχανής παραγωγής γραφικών ID Tech 4 πλέον είναι δυνατή η ρεαλιστικότερη αναπαράσταση εσοχών σε τρισδιάστατα μοντέλα και περιβάλλοντα καθώς και η

αλληλεπίδραση αυτών με το φωτισμό της σκηνής – περιβάλλοντος. Οι δυνατότητες επινοήθηκαν από το ιδρυτικό μέλος της εταιρείας και καινοτόμο στον τομέα των γραφικών υπολογιστών και των βιντεοπαιχνιδιών John Carmack. Το 2006 η Google αποφασίζει να μπει και αυτή στο χώρο της σχεδίασης τρισδιάστατων γραφικών με την εξαγορά του προγράμματος σχεδίασης Sketchup.



Εικόνα 16 Doom 3, νέες ικανότητες φωτισμού με τη δύναμη της ID Tech 4

Οι παραπάνω καινοτομίες δεν έγιναν τυχαία σε τόσο μικρό χρονικό διάστημα. Υπήρχαν σημαντικές εξελίξεις στον χώρο κατασκευής υλικού(hardware) υπολογιστών οι οποίες βοήθησαν σημαντικά το έργο των προγραμματιστών που βρίσκονταν πίσω από τον σχεδιασμό και την ανάπτυξη όσων εργαλείων αναφέρθηκαν ως τώρα. Πιο συγκεκριμένα, τη περίοδο αυτή υπήρξε ίσως η μεγαλύτερη καινοτομία στο τομέα των γραφικών υπολογιστών που του έδωσε τη μορφή που τον γνωρίζουμε σήμερα, ως τον τομέα που μπορεί να παράξει εντυπωσιακά θεάματα μέσω του συνδυασμού του προγραμματισμού, των μαθηματικών αλλά και των νέων τεχνολογιών υλικού υπολογιστών(hardware). Με πρωτοπόρο των προσπαθειών που έγιναν την εταιρεία κατασκευής και διάθεσης καρτών παραγωγής γραφικών(graphics processing unit – GPU), Nvidia οι προγραμματιστές πλέον είχαν τη δυνατότητα να επέμβουν και να επηρεάσουν το κάθε pixel ενός τρισδιάστατου περιβάλλοντος ατομικά και σε πραγματικό χρόνο. Αντίστοιχες υπηρεσίες ξεκίνησαν να προσφέρουν και οι μονάδες επεξεργασίας γραφικών της εταιρείας διάθεσης υλικού ηλεκτρονικών υπολογιστών ATI(Array Technology Inc) η οποία, έπειτα από την εξαγορά της από την εταιρεία μονάδων επεξεργαστών AMD, αποτελεί τον μεγάλο ανταγωνιστή της Nvidia μέχρι και σήμερα.



Εικόνα 18: Το σύμβολο της ATI

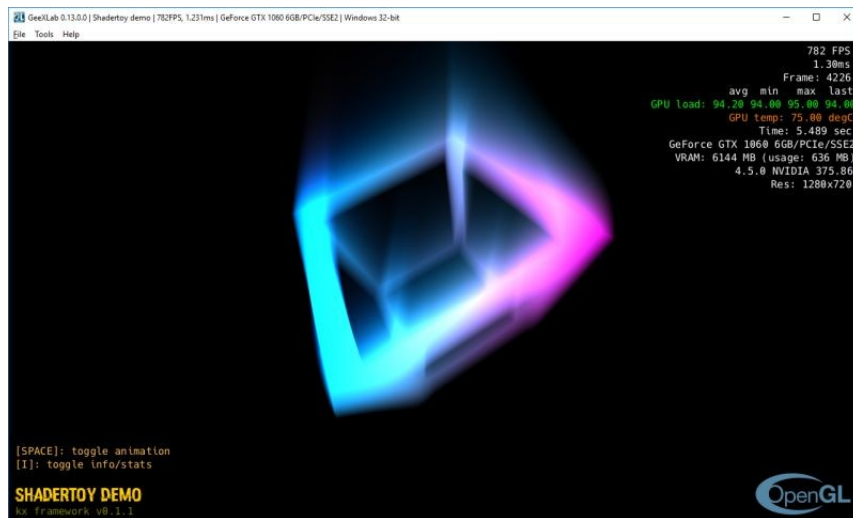


Εικόνα 17: Το σύμβολο της εταιρείας Nvidia

Πιο συγκεκριμένα το πρώτο ηλεκτρονικό τσιπ από την Nvidia που είχε αυτή τη δυνατότητα ονομαζόταν GeForce 3. Μικρά σε μέγεθος προγράμματα είχαν πρόσβαση πλέον σε κάθε pixel ενός τρισδιάστατου εικονικού περιβάλλοντος και με διάφορες προγραμματιστικές τεχνικές επηρέαζαν όχι μόνο τη σκίαση(shading), δηλαδή το χρώμα και τον τρόπο που αλληλεπιδρά με το φως της σκηνής-περιβάλλοντος το υπο-επεξεργασία pixel αλλά ακόμα και τη μορφή του ως προς το μέγεθος,τοποθεσία και κλίμακα.



Εικόνα 19: Μοντέλο της GeForce 3



Εικόνα 20: Παράδειγμα εφαρμογής προγράμματος σκίασης

Τέλος παρακάτω θα παρουσιαστεί η περίοδος **2007 – 2019** η οποία περιγράφει τις καινοτομίες στο χώρο των γραφικών υπολογιστών και ανάπτυξης εικονικών κόσμων που βλέπουμε γύρω μας. Κατά τη διάρκεια του Special Interest Group on Computer Graphics and Interactive Techniques (SIGGRAPH) του 2009 παρουσιάστηκαν σημαντικές έρευνες που διευκόλυναν ακόμα περισσότερο την πορεία ανάπτυξης τρισδιάστατων εικονικών κόσμων. Μερικές από αυτές διαπραγματεύονταν τεχνικές που επέτρεπαν πλέον τη δημιουργία ρεαλιστικής απεικόνισης ειδικών εφέ φωτιάς, τρισδιάστατων μοντέλων που είχαν τη δυνατότητα να αλλοιώνουν τη μορφή τους σε πραγματικό χρόνο σε καταστάσεις σύγκρουσης με στοιχεία του χώρου και παρακολούθηση μελών του ανθρωπίνου σώματος μέσω κάμερας και αναπαράσταση αυτών σε εικονικό τρισδιάστατο χώρο σε πραγματικό χρόνο.



Εικόνα 21: Ρεαλιστικό κινούμενο εφέ φωτιάς



Εικόνα 22: Τεχνικές αλλοίωσης μορφής μοντέλου

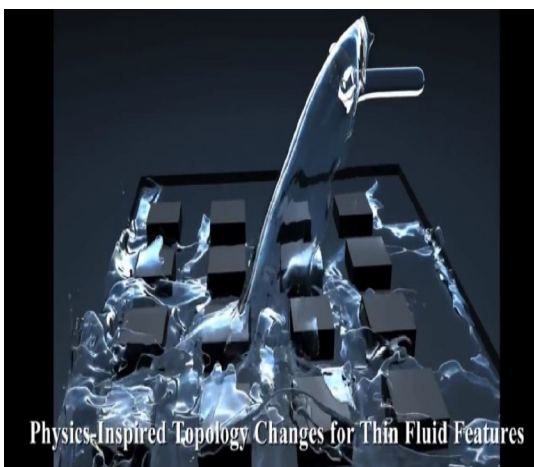
Real-Time Hand-Tracking with a Color Glove



Open / Closed Poses

Εικόνα 23: Παρακολούθηση μέλους σώματος και αναπαράσταση αυτού σε ψηφιακό περιβάλλον σε πραγματικό χρόνο

Στην έκθεση του 2010 αναπτύχθηκε η δυνατότητα ρεαλιστικής αναπαράστασης της κίνησης του τριχώματος μιας οντότητας σε ένα εικονικό περιβάλλον καθώς και η τεχνική προσομοίωσης κίνησης υγρών στοιχείων σε ένα χώρο με βάση τη τοπολογία αυτού.

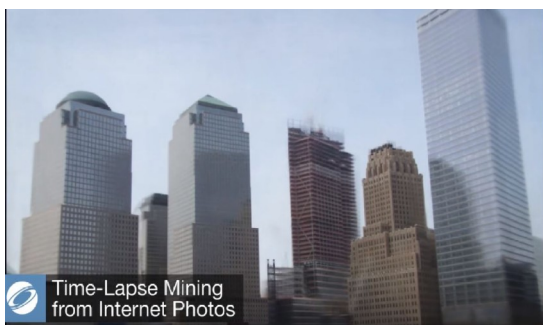


Εικόνα 24: Κίνηση νερού με βάση τη τοπολογία του χώρου

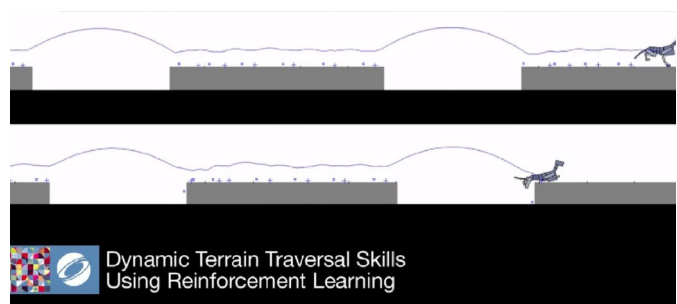


Εικόνα 25: Ρεαλιστική απεικόνιση τριχών

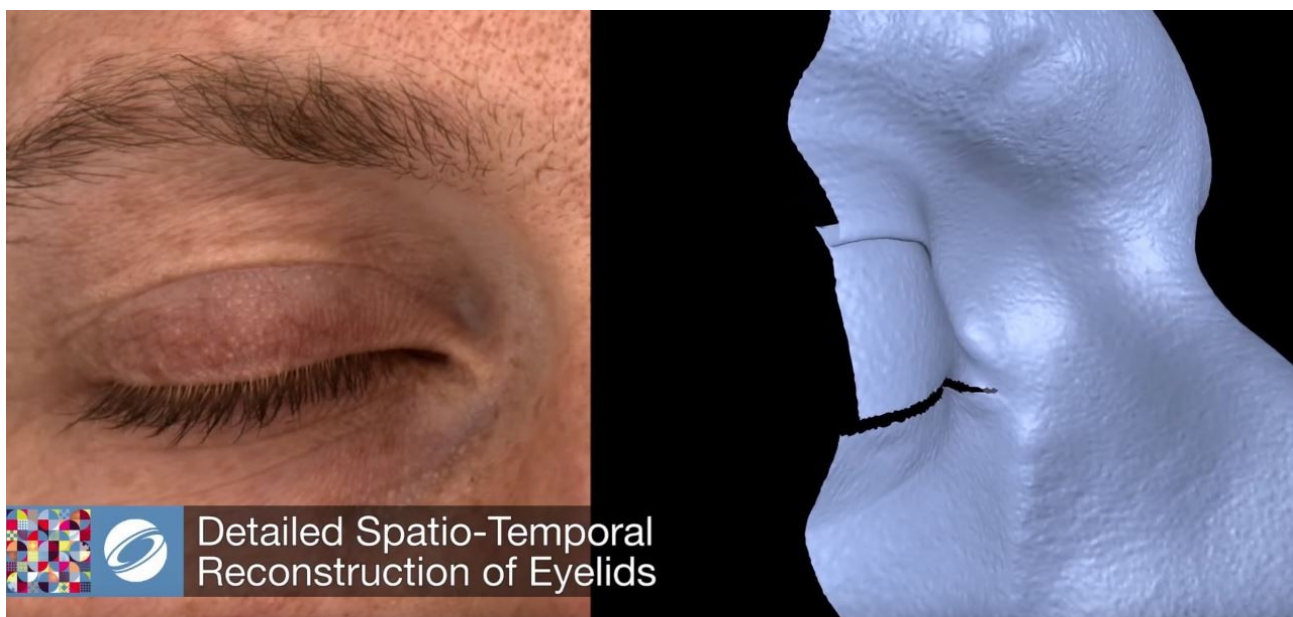
Τα επόμενα χρόνια ο τομέας της μηχανικής μάθησης άρχισε να ακμάζει και να ενισχύει πολλές τεχνικές ανάπτυξης στον τομέα των Γραφικών Υπολογιστών. Η δυνατότητα να εκπαιδεύουμε μοντέλα με δεδομένα που προέρχονται από το φυσικό περιβάλλον ώστε να έχουν αποδοτικότερη κίνηση σε ένα ψηφιακό κόσμο οδήγησε σε ρεαλιστικότερα αποτελέσματα σε διάφορων ειδών εικονικών προσομοιώσεων. Εφαρμογές μηχανικής μάθησης σε συνδυασμό με εξόρυξη δεδομένων από μία βάση δεδομένων με φωτογραφίες κατασκευής οικιακών συγκροτημάτων έδωσε τη δυνατότητα σε ερευνητές να παρουσιάσουν μία εικονική διαδικασία κατασκευής σε πραγματικό χρόνο. Αντίστοιχες τεχνικές χρησιμοποιήθηκαν για τη απεικόνιση κίνησης ζώων σε έναν εικονικό κόσμο καθώς και για την αναπαράσταση του φυσικού τρόπου κίνησης των βλεφάρων ενός ανθρώπου. Αποτελέσματα των παραπάνω ερευνητικών εγχειρημάτων ήρθαν στο φως στην έκθεση SIGGRAPH του 2015.



Εικόνα 26: Ανακατασκευή οικιακών συγκροτημάτων μέσω data mining σε πραγματικό χρόνο

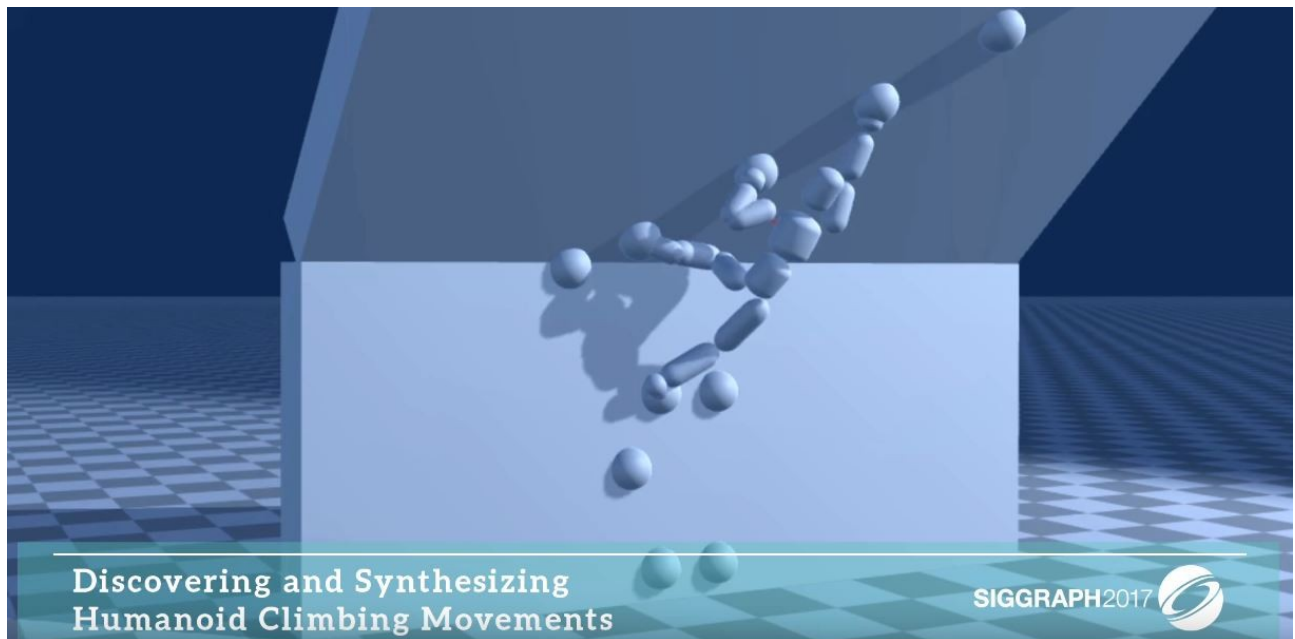


Εικόνα 27: Ρεαλιστική κίνηση ζώων σε εικονικό χώρο

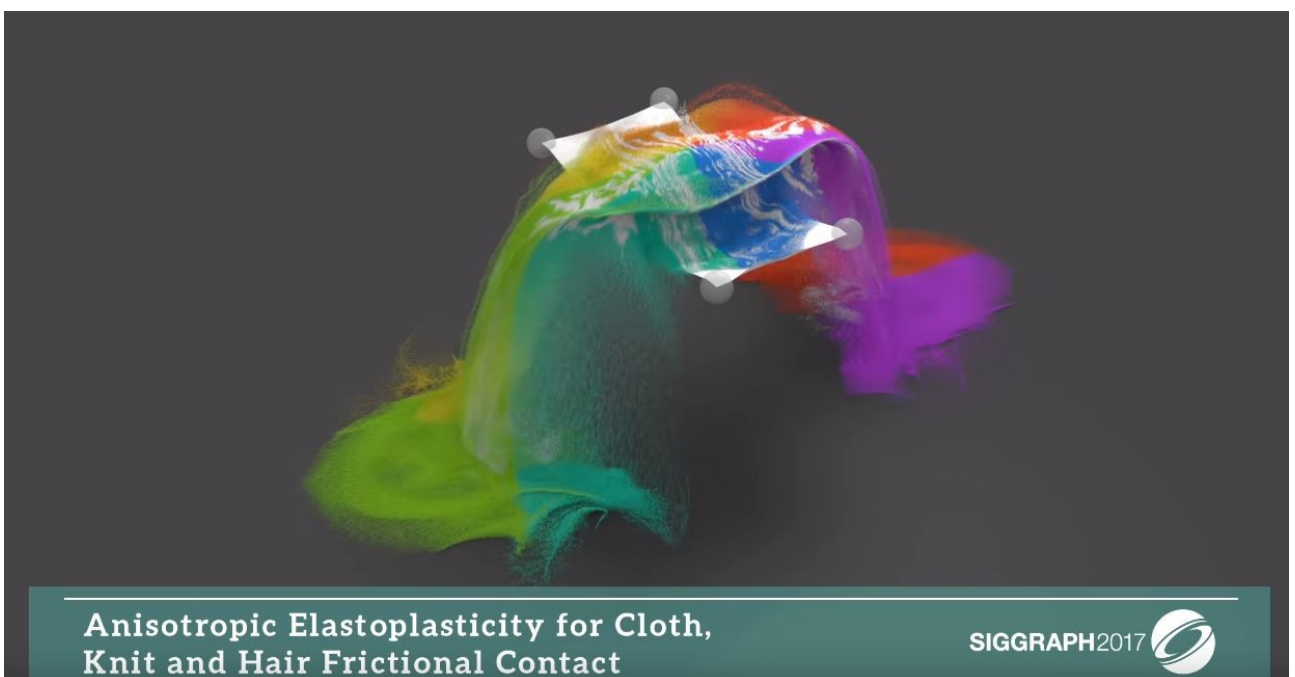


Εικόνα 28: Προσομοίωση ρεαλιστικής μορφολογίας και κίνησης βλεφάρων

Στα πλαίσια της έκθεσης του 2017 αποτελέσματα ερευνών που οδήγησαν στην κατασκευή μοντέλων που μιμούνται αποδοτικά τις αναρριχητικές ικανότητες του ανθρωπίνου σώματος καθώς και νέες εξελίξεις στην προσομοίωση της ελαστικότητας και πλαστικότητας υλικών όπως υφάσματος και τριχών είχαν ως αποτέλεσμα την άμεση χρήση τους στον χώρο της ψυχαγωγίας και της διεξαγωγής επιστημονικών πειραμάτων.

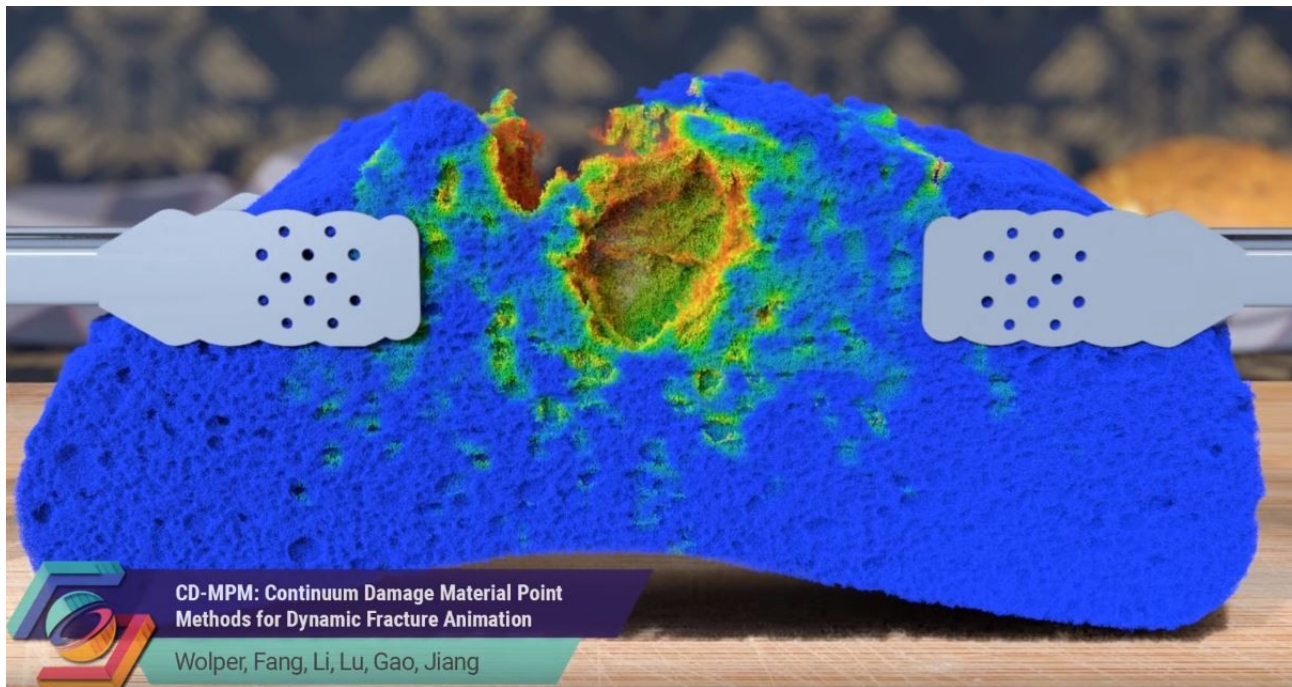


Εικόνα 29: Μιμητικό μοντέλο αναρριχητικών ικανοτήτων ανθρωπίνου σώματος



Εικόνα 30: Προσομοίωση ελαστικών ιδιοτήτων διαφόρων υλικών

Ιδιότητες ρεαλιστικής δυναμικής διάσπασης υλικών σε πραγματικό χρόνο παρουσιάστηκαν στην έκθεση SIGGRAPH 2019. Επιπρόσθετα, νέες τεχνικές παραγωγής κόσμων με έμφαση στην ομαλή κατανομή πανίδας στο χώρο δίνουν πλέον τη δυνατότητα για δημιουργία ακόμα περισσότερων αληθοφανών ψηφιακών περιβαλλόντων.



Εικόνα 31: Δυναμική διάσπαση μοντέλων διάφορων υλικών



Εικόνα 32: Παραγωγή κόσμων με έμφαση στη ρεαλιστική και προοδευτική απεικόνιση και εξάπλωση ενός οικοσυστήματος

Σήμερα βρισκόμαστε στην περίοδο όπου οι προσπάθειες ώστε η διαχωριστική γραμμή μεταξύ της αληθινής ζωής και των εικονικών κόσμων να εξαλειφθεί και να έχουμε για πρώτη φορά την ένωση των τους. Με αυτό το τρόπο οδηγούμαστε στην εποχή της Εικονικής Πραγματικότητας (Virtual Reality).

Ως Εικονική Πραγματικότητα μπορεί να ορισθεί η χρήση τεχνολογικών μέσων για τη δημιουργία ενός τεχνητού, διαδραστικού περιβάλλοντος το οποίο στη βέλτιστη μορφή του ο άνθρωπος-χρήστης το αντιλαμβάνεται ως πραγματικό. Το σημαντικό στοιχείο είναι η δυνατότητα διάδρασης του χρήστη, η οποία τροποποιεί σε πραγματικό χρόνο το περιβάλλον, όπως θα συνέβαινε και στο πραγματικό κόσμο. Το Εικονικό Περιβάλλον (Virtual Environment) μπορεί να δημιουργείται σε αντιστοιχία με έναν υφιστάμενο ή έναν ιδεατό κόσμο. Χαρακτηριστικό παράδειγμα για κάθε περίπτωση αποτελούν οι περιπτώσεις εικονικού περιβάλλοντος που παρουσιάζουν την ψηφιακή απεικόνιση ενός αρχαιολογικού χώρου, με δυνατότητες εικονικής περιήγησης κάποιου επισκέπτη, ή την ψηφιακή ενσωμάτωση ενός χρήστη-παίκτη στο εικονικό περιβάλλον ενός βιντεοπαιχνιδιού. Είναι ξεκάθαρο πλέον πως η χρήση ειδικά κατασκευασμένου εξοπλισμού με ενσωμάτωση σε περιβάλλοντα εικονικής πραγματικότητας μπορεί να χρησιμοποιηθεί όχι μόνο για λόγους ψυχαγωγίας αλλά και σε πληθώρα ενεργειών όπως εκπαιδευτικές δραστηριότητες, εκπαίδευση προσωπικού και επιστημονικά πειράματα.



Εικόνα 33: Περιήγηση σε αρχαιολογικό χώρο με τη χρήση εξοπλισμού εικονικής πραγματικότητας



Εικόνα 34: Βιντεοπαιχνίδι πρώτο προσώπου με τρόπο παιχνιδιού εικονικής πραγματικότητας

Επιπρόσθετα ο κλάδος εικονική πραγματικότητα ακολουθήθηκε από τις ακόλουθες κατά περίπτωση εφαρμογές :

- **Επαυξημένη πραγματικότητα (Augmented Reality)** : η οποία αναφέρεται στην επαύξηση του πραγματικού κόσμου με εικονικά αντικείμενα (συνήθως με τη μορφή ήχου, εικόνας ή γραφικών υπολογιστών.
- **Επαυξημένη εικονικότητα (Augmented Virtuality)** : η οποία αναφέρεται στην επαύξηση του εικονικού κόσμου με πραγματικά αντικείμενα.
- **Μεικτή πραγματικότητα** : η οποία αναφέρεται στο αποτέλεσμα του συνδυασμού διάφορων μεθόδων που λαμβάνουν χώρα σε έναν έναν εικονικό και μη – εικονικό κόσμο.



Εικόνα 35: Το συνεχές της μεικτής πραγματικότητας

Τα χαρακτηριστικά της εικονικής πραγματικότητας και των εικονικών περιβαλλόντων συνοψίζονται στους ακόλουθους βασικούς πυλώνες :

1. **Εμβύθιση (Immersion)** : μπορεί να οριστεί ως ο βαθμός στον οποίο ο χρήστης αισθάνεται ότι βρίσκεται πραγματικά μέσα στο εικονικό περιβάλλον και όχι στο πραγματικό. Από αυτή την ανάγκη προκύπτει η ραγδαία εξέλιξη των τεχνικών που προσφέρουν ρεαλιστική αναπαράσταση του φυσικού κόσμου.
2. **Επικοινωνία – Διάδραση (Interaction)** : σχετίζεται με την δυνατότητα του υπολογιστή να διαμορφώνει άμεσα το συνθετικό κόσμο ανάλογα με τις κινήσεις του ανθρώπου-χρήστη. Έτσι η ανάπτυξη εύχρηστων και με μεγάλη ανταπόκριση εξοπλισμού και χειριστηρίων με δέκτες μεγάλης ακρίβειας κρίνεται απαραίτητη.
3. **Ένταση Πληροφορίας (Information Intensity)** : σχετίζεται όχι μόνο με τη πληθώρα της πληροφορίας αλλά και την ποικιλία διαύλων επικοινωνίας μέσω των οποίων προσφέρεται η πληροφορία στο χρήστη (οπτικός, ακουστικός, απτικός).
4. **Τηλε-παρουσία (telepresence)** : ο βαθμός στον οποίο ο χρήστης αισθάνεται παρών σε ένα συνθετικό, εικονικό κόσμο που βρίσκεται μακριά από τον πραγματικό ή τον βαθμό στον οποίο άλλοι χρήστες θεωρούν ότι ο χρήστης είναι παρών στο δικό τους εικονικό κόσμο.

Η εικονική πραγματικότητα δέχεται μαζικές επιρροές από ένα σύνολο διαφορετικών γνώσεων και τεχνικών που προέρχονται από διαφορετικές επιστημονικές περιοχές ώστε η δημιουργία εικονικών περιβαλλόντων να συνεχίζει να εξελίσσεται με ραγδαίους ρυθμούς. Ακολουθούν οι κυριότερες :

- **Γραφικά Υπολογιστών (Computer Graphics)** : Η πηγή των αλγορίθμων και των τεχνικών για τη δημιουργία γραφικών σε πραγματικό χρόνο.
- **Διάδραση Ανθρώπου Υπολογιστή (Human Computer Interaction)** : Προσφέρει μεθόδους και οδηγίες σχετικές με τη διάδραση του χρήστη μέσα σε ένα εικονικό περιβάλλον.
- **Εργονομία (Ergonomics)** : Μέσω της εξέτασης γύρω από την ανθρώπινη φυσιολογία μπορούμε να κατασκευάσουμε εξοπλισμό και εικονικά περιβάλλοντα με γνώμονα την ευχρηστία και την ασφάλεια.
- **Τεχνητή Νοημοσύνη (Artificial Intelligence)** : Περιλαμβάνει την προσομοίωση έξυπνης συμπεριφοράς για χρήση σε ανθρώπινα ή μη εικονικών χαρακτηριστικών.
- **Κοινωνιολογία (Sociology)** : Προσφέρει γνώσεις γύρω από τον σωστό τρόπο αξιολόγησης εμπειρίας ενός χρήστη σε ένα εικονικό περιβάλλον αλλά και την αξιολόγηση των αποτελεσμάτων του.

- **Γνωσιακή Επιστήμη (Cognitive Science)** : Ιδέες που κινούνται γύρω από την ανθρώπινη ευφυΐα και πως αυτή συμπεριφέρεται κάτω από την ύπαρξη της σε ένα ψηφιακό περιβάλλον.
- **Ψυχολογία (Psychology)** : Σχετίζεται με την μελέτη της ανθρώπινης συμπεριφοράς και έτσι μπορούμε να προβλέψουμε το πως θα επηρεαστεί η ψυχοσύνθεση ενός χρήστη σε ένα εικονικό περιβάλλον.

Ο εξοπλισμός εικονικής πραγματικότητας συχνά περιλαμβάνει συσκευές που τοποθετούνται στο κεφάλι του χρήστη, ειδικά χειριστήρια με δυνατότητες ανίχνευσης αλλαγής ταχύτητας και δόνησης αλλά και ειδικούς προβολείς που τοποθετούνται μέσα στο φυσικό χώρο.



Εικόνα 36: Χειριστήριο για χρήση σε περιβάλλοντα εικονικής πραγματικότητας



Εικόνα 37: Συσκευή κεφαλής Headset



Εικόνα 39: Συσκευή κεφαλής μικτής πραγματικότητας



Εικόνα 38: Προβολείς μικτής πραγματικότητας

B.2 Βιντεοπαιχνίδια – εικονικοί κόσμοι και τεχνικές ανάπτυξης

B.2.1 Τι είναι ένα παιχνίδι;

Οι περισσότεροι από εμάς έχουμε ήδη μια αυθαίρετη και έμφυτη ιδέα για το τι είναι ένα παιχνίδι. Ο γενικός όρος “παιχνίδι” περιλαμβάνει δραστηριότητες όπως επιτραπέζια παιχνίδια δηλαδή μία παρτίδα σκάκι ή Monopoly, παιχνίδια που κάνουν χρήση καρτών όπως το πόκερ ή το black jack, παιχνίδια τζόγου όπως η ρουλέτα και τα φρουτάκια στο καζίνο ή ακόμα παιχνίδια πολέμου, όπως risk και βιντεοπαιχνίδια στρατηγικής. Στην ακαδημαϊκή κοινότητα χρησιμοποιείται συχνά ο όρος “Θεωρία παιγνίων” (Game Theory) κατά την οποία πολλαπλοί “πράκτορες”-παίκτες επιλέγουν στρατηγικές και τακτικές για να μεγιστοποιήσουν τα κέρδη τους μέσα από μια αλληλουχία κινήσεων που υπακούουν σε ένα σύνολο αυστηρά καθορισμένων κανόνων.

Όταν ο όρος “παιχνίδι” χρησιμοποιείται εντός του χώρου των οικιακών κονσολών ψυχαγωγίας ή των ηλεκτρονικών υπολογιστών λαμβάνει σημασία που φέρνει στο νου κάποιου σκηνές τρισδιάστατων εικονικών περιβαλλόντων με κύρια συστατικά στοιχεία κάποιον κεντρικό χαρακτήρα, με ανθρώπινα χαρακτηριστικά ή όχι, ή ακόμα και ένα όχημα το οποίο εμείς χειριζόμαστε με κάποιο συνδυασμό ποντικιού και πληκτρολογίου ή ειδικά κατασκευασμένου χειριστηρίου. Συνήθως ο σκοπός τέτοιων παιχνιδιών είναι να ψυχαγωγήσουν τον χρήστη και να έχει μία ευχάριστη εμπειρία. Αυτή η ψυχαγωγία έρχεται συχνά μέσα από την αναγνώριση των διάφορων μοτίβων προβλημάτων που θα αντιμετωπίσει ο παίκτης κατά τη διάρκεια του παιχνιδιού και των λύσεων που θα επινοήσει. Με κάθε επόμενη επιτυχία του παίκτη που του παρέχει τη δυνατότητα να συνεχίσει να εξελίσσεται στα πλαίσια του εικονικού χώρου του εκάστοτε παιχνιδιού έρχεται και το κομμάτι της προσωπικής ευχαρίστησης.

B.2.2 Βιντεοπαιχνίδια ως προσομοιώσεις πραγματικού χρόνου

Τα περισσότερα παιχνίδια δύο ή τριών διαστάσεων αποτελούν παραδείγματα αυτού που θα έλεγαν οι ερευνητές ως ευέλικτες διαδραστικές προσομοιώσεις πραγματικού χρόνου βασισμένες σε πράκτορες (*soft real-time interactive agent-based simulations*). Στη συνέχεια θα προσπαθήσουμε να

δούμε τμηματικά την έννοια του όρου αυτού ώστε να δούμε την σχέση μεταξύ παιχνιδιού και εικονικού περιβάλλοντος.

Αρχικά σε πληθώρα βιντεοπαιχνιδιών ένα υποσύνολο του πραγματικού, ή ακόμα και ένας ολοκληρωμένος φανταστικός κόσμος, μορφοποιούνται σε ψηφιακή απεικόνιση με τη χρήση μαθηματικών εξισώσεων ώστε να μπορούν να εισαχθούν και να επεξεργαστούν από τον ηλεκτρονική υπολογιστή. Το μοντέλο που προκύπτει από αυτή τη διαδικασία αποτελεί μία κατά προσέγγιση απλοποίηση της πραγματικότητας γιατί αποτελεί έργο δύσκολο και μη πρακτικό το να ψηφιοποιήσει κανείς μέχρι και τη παραμικρή λεπτομέρεια του φυσικού κόσμου γύρω μας. Οι τεχνικές της προσέγγισης και της απλοποίησης είναι το κεντρικό στοιχείο της εργαλειοθήκης του προγραμματιστή βιντεοπαιχνιδιών καθώς με τους σωστούς χειρισμούς ακόμα και η πιο απλή μοντελοποίηση μπορεί να ξεπεράσει το αληθινό.

Ο χαρακτηρισμός “βασισμένο σε πράκτορες” (agent-based) προσδίδεται σε προσομοιώσεις όπου το εικονικό περιβάλλον τους απαρτίζεται από διακριτές οντότητες που χαρακτηρίζονται ως “πράκτορες” οι οποίοι αλληλεπιδρούν μεταξύ τους. Η παραπάνω κατηγορία προσομοιώσεων σκιαγραφεί τα περισσότερα βιντεοπαιχνίδια με μεγάλη ακρίβεια με τον ρόλων των πρακτόρων να αναλαμβάνουν τρισδιάστατα μοντέλα που αναπαριστούν οχήματα, χαρακτήρες, σφαίρες και όπλα. Το γεγονός ότι τα περισσότερα παιχνίδια απαρτίζονται από πράκτορες στις μέρες οδηγεί εύκολα στο συμπέρασμα πως τα καταλληλότερα μοτίβα προγραμματισμού είναι αυτά του αντικειμενοστραφή προγραμματισμού (object oriented programming).

Όλα τα διαδραστικά βιντεοπαιχνίδια μπορούν να χαρακτηριστούν και ως χρονικές προσομοιώσεις (temporal simulations) γεγονός το οποίο υποδεικνύει ότι το εικονικό περιβάλλον του παιχνιδιού είναι δυναμικό (dynamic), στοιχείο το οποίο δείχνει πως αυτό αλλάζει με τη πάροδο του χρόνου και την εξέλιξη της ιστορίας και της επίτευξης στόχων στα πλαίσια του. Επιπρόσθετα αξίζει να σημειωθεί πως η συγκεκριμένη εικονική εμπειρία αυτή θα πρέπει να παρέχει τη δυνατότητα υποδοχής και επεξεργασίας τυχαίων πληροφοριών εισόδου (inputs) από τον άνθρωπο-χρήστη/παίκτη γεγονός που τη κάνουν διαδραστική χρονική προσομοίωση (interactive temporal simulation). Επιπρόσθετα η δυνατότητα να αποτελούν μία αλληλουχία εργασιών και κινήσεων που πραγματοποιούνται σε πραγματικό χρόνο, προσδίδει στα βιντεοπαιχνίδια την έννοια του real-time χαρακτηριστικού. Από τα παραπάνω εξαιρούνται περιπτώσεις εικονικών κόσμων μη πραγματικού χρόνου όπως ένα βιντεοπαιχνίδι σκάκι ή άλλο μη πραγματικού χρόνου παιχνίδι στρατηγικής οι οποίες παρόλα αυτά θα έχουν σίγουρα και κάποια στοιχεία πραγματικού χρόνου, παράδειγμα αποτελεί η δυνατότητα ενός παίκτη να πληκτρολογεί τα απαραίτητα κουμπιά τα οποία του παρουσιάζουν στατιστικά για τη πορεία του στο παιχνίδι καθώς περιμένει τη σειρά του να

παίζει και χωρίς να διακόπτει τις εργασίες που πραγματοποιεί ο αντίπαλος του εκείνη τη δεδομένη στιγμή.

Ακόμη στο επίκεντρο κάθε βιντεοπαιχνιδιού πραγματικού χρόνου βρίσκεται η ιδέα του χρόνου ζωής (deadline). Αρχικό παράδειγμα στο χώρο των βιντεοπαιχνιδιών αποτελεί η απαίτηση, η οθόνη να ανανεώνεται 24 φορές κάθε δευτερόλεπτο ώστε να προσφέρει την ελάχιστη ομαλή οπτική απάτη της κίνησης. Άλλου είδους παρόμοιοι χρόνοι ζωής αποτελούν οι ανάγκες των συστημάτων φυσικής προσομοίωσης (physics systems) να ανανεώνονται 120 φορές ανά δευτερόλεπτο ώστε να παράγουν ένα πιστό στη πραγματικότητα και το φυσικό κόσμο που ξέρουμε εικονικό αποτέλεσμα. Στο τομέα των δυνατοτήτων τεχνητής νοημοσύνης των πρακτόρων του κόσμου μας είναι πιθανό να απαιτείται μία σκέψη – επεξεργασία των πιθανών σεναρίων που θα εμπλακεί και αλληλουχιών κινήσεων που θα ακολουθήσει, μία φορά το δευτερόλεπτο και στη περίπτωση του ήχου να χρειαζόμαστε ανανεώσεις των εισόδων ηχητικών σημάτων 60 φορές ανά δευτερόλεπτο. Από τα παραπάνω οδηγούμαστε στο συμπέρασμα πως η προσεκτική ρύθμιση των ιδιοτήτων του εικονικού μας περιβάλλοντος μπορεί να οδηγήσει σε εμπειρίες ρεαλιστικές, φανταστικές ή ακόμα και σε ευχάριστες ή μη ανάλογα με την εμπειρία που προσφέρουν έμμεσα αυτές στον παίκτη.

Παρ' όλες τις σημειώσεις σχετικά με τον χρόνο ζωής διαφόρων ιδιοτήτων συναντήσαμε και τον όρο ευέλικτο – soft στον αρχικό ορισμό που δώσαμε στα βιντεοπαιχνίδια από τη σκοπιά των εικονικών προσομοιώσεων. Πιο συγκεκριμένα ένα βιντεοπαιχνίδι χαρακτηρίζεται ευέλικτο επειδή όταν χρόνοι ζωής που αναφέρθηκαν παραπάνω δεν τηρούνται ή αποτυγχάνουν να περατωθούν τη χρονική στιγμή που ορίσαμε, το αποτέλεσμα αποτελεί καταστροφικό για το εικονικό περιβάλλον μας και την ποιότητα της εμπειρίας του χρήστη. Για παράδειγμα η πτώση του ρυθμού ανανέωσης της οθόνης θα οδηγήσει σε χειρότερη εμπειρία, λόγω μικρότερης ευκρίνειας της κίνησης των σωμάτων στον εικονικό μας χώρο, αλλά δεν θα έχει ως αποτέλεσμα ο παίκτης να “πεθάνει” ή εφαρμογή μας να τερματίσει απρόοπτα. Σε αντίθεση σε ένα μη ευέλικτο(hard) σύστημα εικονικών προσομοιώσεων όπως είναι αυτό που χρησιμοποιείται στα όργανα αεροπλάνων για τη πρόβλεψη της κίνησης και λοιπών παραμέτρων που θα επηρεάσουν τη πορεία του αεροπλάνου, κάθε λάθος μπορεί να οδηγήσει σε καταστροφή του οχήματος και φυσικά σε κίνδυνο της ασφάλειας των επιβατών.

Τα μαθηματικά μοντέλα χωρίζονται σε αναλυτικά και αριθμητικά. Ένα αναλυτικό μοντέλο μπορεί να επαληθευθεί για κάθε τιμή των ανεξάρτητων μεταβλητών του. Κάποια προβλήματα μπορούν να λυθούν με αυτόν τον τρόπο αλλά δεν μπορούν να λυθούν όλα τα προβλήματα πάντα με αυτόν τη μέθοδο. Στα βιντεοπαιχνίδια συγκεκριμένα η είσοδος των πληροφοριών που δίνει ο χρήστης-παίκτης κατά τη διάδραση του με το εικονικό μας περιβάλλον είναι απρόβλεπτη και δε μπορούμε να την αντιμετωπίσουμε ως ένα αναλυτικό πρόβλημα. Από την άλλη ένα αριθμητικό μοντέλο συνήθως υλοποιούνται με τη διεξαγωγή αλληπάλληλων μαθηματικών υπολογισμών στη προσπάθεια να εξακριβωθεί η κατάσταση του μαθηματικού συστήματος μας σε κάθε διακριτή

χρονική στιγμή. Εφαρμογή αυτής της μεθόδου προσδιορισμού των διάφορων καταστάσεων συναντάται στα βιντεοπαιχνίδια.

Κύριο χαρακτηριστικό κατά την ανάπτυξη ενός παιχνιδιού είναι αυτό που ονομάζουμε κύκλος παιχνιδιού (game loop). Το συγκεκριμένο στοιχείο εκτελείται συνεχώς και με κάθε νέα επανάληψη του, διάφορα συστατικά στοιχεία του συστήματος, παιχνιδιού στη περίπτωση μας για τα πλαίσια της παρούσας εργασίας, όπως είναι οι λειτουργίες της τεχνητής νοημοσύνης των πρακτόρων, των προσομοιώσεων φυσικής κίνησης των αντικειμένων στο χώρο καθώς και ο ήχος έχουν την ευκαιρία να ανανεώσουν τις τιμές τους ώστε να προετοιμαστεί το επόμενο διακριτό βήμα. Τα αποτελέσματα αυτά αποδίδονται (rendered) στην αρχή του επόμενου κύκλου και έτσι γίνονται ορατά στον παίκτη αλλά και μέσα από τις συνέπειες αυτών στο περιβάλλον του παιχνιδιού.

B.2.3 Μηχανές παραγωγής παιχνιδιών

Ο όρος μηχανή παιχνιδιών (game engine) έκανε την εμφάνισή του στα μέσα της δεκαετίας του 1990 λόγω της απήχισής των παιχνιδιών πρώτου προσώπου (first person shooter – FPS) όπως το Doom της id Software. Κατά τα στάδια ανάπτυξης της αρχιτεκτονικής του Doom δόθηκε ιδιαίτερη σημασία στο διαχωρισμό του πυρήνα των συστημάτων λογισμικού, όπως είναι η τρισδιάστατη μηχανή απόδοσης γραφικών (three-dimensional graphics rendering system), η μηχανή εντοπισμού συγκρούσεων (collision detection system), το σύστημα ήχου (audio system) με τον χώρο που καταλαμβάνουν οι καλλιτεχνικοί πόροι του παιχνιδιού, το σύστημα δημιουργίας επιπέδων-κόσμων (world/map generation system) και τέλος του συστήματος που ορίζει τους κανόνες παιχνιδιού. Η αξία του επιτυχούς αυτού διαχωρισμού συστημάτων φαίνεται από το γεγονός ότι οι προγραμματιστές μπόρεσαν να δημιουργούν νέα παιχνίδια με τη δημιουργία νέων καλλιτεχνικών πόρων, εικόνες και μοντέλα πρακτόρων, νέων κόσμων και επιπέδων καθώς και κανόνων παιχνιδιού με μηδαμινές αλλαγές πάνω στη μηχανή λογισμικού. Με αυτό το τρόπο οδηγηθήκαμε στην δημιουργία της κοινότητας μετατροπών (modding community), μίας ομάδας από παίκτες που δεν είχαν ιδιαίτερες γνώσεις προγραμματισμού οι οποίοι δημιουργούσαν νέα σενάρια παιχνιδιού ή ακόμα και ολόκληρα παιχνίδια πραγματοποιώντας μικρές μετατροπές σε ήδη υπάρχουσα παιχνίδια και βάσεις κώδικα συχνά με τη χρήση εργαλείων τα οποία αναπτύχθηκαν από τους αρχικούς προγραμματιστές των παιχνιδιών.

Μέχρι το τέλος της δεκαετίας του 1990 όλο και περισσότερα παιχνίδια κυκλοφορούσαν εξ αρχής με τις δυνατότητες επέκτασης που αναφέρθηκαν παραπάνω να έχουν ληφθεί υπόψη. Οι διάφορες μηχανές παραγωγής παιχνιδιών πλέον διευκόλυναν πιθανές αλλαγές με τη δημιουργία γλωσσών προγραμματισμού τύπου scripting όπως η Quake C και ταυτόχρονα η δυνατότητα δανεισμού

ολόκληρου του πηγαίου κώδικα μιας μηχανής προς χρήση τρίτων για εμπορικούς σκοπούς ξεκίνησε να γίνεται πάγια τακτική των ομάδων ανάπτυξης βιντεοπαιχνιδιών γεγονός το οποίο οδηγούσε σε ραγδαία αύξηση των κερδών τους.

Η διαχωριστική γραμμή μεταξύ παιχνιδιού και μηχανής παραγωγής παιχνιδιών είναι συχνά δύσκολο να τοποθετηθεί επαρκώς. Κάποιες μηχανές παραγωγής κάνουν αρκετά σαφές το γεγονός αυτό ενώ άλλες δεν κάνουν καν κινήσεις προς τη κατεύθυνση αυτή. Σε ένα παιχνίδι ο κώδικας απόδοσης(rendering code) είναι πιθανό να έχει τη δυνατότητα να γνωρίζει με μεγάλη ακρίβεια το τρόπο σχεδίασης και τα χαρακτηριστικά ενός πράκτορα. Από την άλλη σε κάποιο άλλο παιχνίδι η μηχανή απόδοσης μπορεί περιλαμβάνει μία αυθαίρετη έννοια του ίδιου τύπου πράκτορα, με τη προηγούμενη περίπτωση, παρέχοντας μόνο τα ειδικά εργαλεία και στοιχεία που θα κάνει χρήση η υλοποίηση αυτού με αποτέλεσμα να επιτάσσεται η ανάγκη για τροφοδότησή αυτών με δεδομένα από τον προγραμματιστή.

Ομολογουμένως η τεχνική της δεδομενο-κεντρικής αρχιτεκτονικής αποτελεί το χαρακτηριστικό που διαχωρίζει μία μηχανή παραγωγής παιχνιδιών από ένα απλό τμήμα λογισμικού που χαρακτηρίζεται ως παιχνίδι και όχι ως μηχανή παραγωγής παιχνιδιών. Όταν ένα παιχνίδι περιέχει τμήματα προδιαγεγραμμένης λογικής(hard-coded logic), κανόνων ή επιστρατεύει ειδικά σχεδιασμένα τμήματα κώδικα ώστε να αποδώσει συγκεκριμένα ήδη αντικειμένων στον χώρο, τότε αυτόματα καθίσταται ακατάλληλο για επαναχρησιμοποίηση στη διαδικασία ανάπτυξη κάποιου άλλου παιχνιδιού. Με αυτό το τρόπο καταλήγουμε στο συμπέρασμα πως πρέπει να χρησιμοποιούμε τον όρο “μηχανή παραγωγής” παιχνιδιών για λογισμικό το οποίο έχει τη δυνατότητα να επεκταθεί ώστε να αποτελέσει τη βάση ανάπτυξης και δημιουργίας πολλών διαφορετικών παιχνιδιών χωρίς υπερβολικές τροποποιήσεις.

Σε αυτό το σημείο θα πρέπει να τονιστεί πως μία μηχανή παραγωγής γραφικών απέχει αρκετά από ένα γενικής χρήσης λογισμικό όπως μία εφαρμογή αναπαραγωγής μουσικής ή ταινιών τα οποία έχουν τη δυνατότητα να αναπαράγουν κάθε είδος αρχείου μουσικής ή ταινίας. Πιο συγκεκριμένα καμία μηχανή παραγωγής παιχνιδιών δεν είναι κατάλληλη για ανάπτυξη κάθε είδους παιχνιδιού. Οι περισσότερες μηχανές δημιουργήθηκε με ένα ή έστω μία μικρή ομάδα ειδών παιχνιδιών ως στόχο εξόδου έπειτα από τη διαδικασία χρήσης τους. Είμαστε βέβαιοι για το γεγονός πως κάθε μηχανή που έχει ως στόχο την υποστήριξη ανάπτυξης πολλών ειδών παιχνιδιών είναι εύκολο να ξεκινήσει να υστερεί στο τομέα της βελτιστοποίησης κάποιων λειτουργιών τους.

Το παραπάνω φαινόμενο παρατηρείται διότι ο σχεδιασμός ενός μεγάλου μεγέθους αποδοτικού λογισμικού περιλαμβάνει το στάδιο των επιλογών που θα έχουν αντίκτυπο ως προς τις πιθανές χρήσεις του στο μέλλον. Επιλογές όπως το τι υλικό(hardware) θα υποστηρίξει το λογισμικό-μηχανή μέχρι το τι παιχνίδια θα είναι εύκολο να αναπτυχθούν κάνοντας χρήση των λειτουργιών του είναι

κύριες που επηρεάζουν τη διαδικασία ανάπτυξης. Για παράδειγμα μία μηχανή που έχει σχεδιαστεί με γνώμονα τις ανάγκες δημιουργίας μικρών εσωτερικών χώρων θα σταθεί εμπόδιο σε επίδοξους προγραμματιστές που στόχο έχουν τη δημιουργία μεγάλου μεγέθους φανταστικών τρισδιάστατων εικονικών περιβαλλόντων. Από την άλλη μία μηχανή με δυνατότητες να αποκρύπτει λεπτομέρειες μεγάλων τμημάτων εδάφους(terrain) ώστε να επιτυγχάνονται σταθερές επιδόσεις όσον αφορά τον ρυθμό ανανέωσης της οθόνης, είναι σίγουρο πως θα αποτελέσει πρόβλημα στη διαδικασία ανάπτυξης ενός εικονικού περιβάλλοντος με πρωταγωνιστές μικρής κλίμακας εσωτερικού χώρου. Η τεχνικές απόκρυψης λεπτομερειών που θα βοηθούσαν τη περιήγηση σε έναν μεγάλης έκτασης χώρο θα επηρεάσουν αρνητικά την εμπειρία σε ένα εσωτερικό περιβάλλον στο οποίο η έλλειψη λεπτομέρειας θα είναι ιδιαίτερα ορατή.

Σήμερα με τη κατασκευή ολοένα και πιο γρήγορου υλικού υπολογιστών και εξειδικευμένων καρτών γραφικών, μαζί με το σχεδιασμό βελτιστοποιημένων αλγορίθμων απόδοσης(rendering algorithms) και δομών δεδομένων το χάσμα μεταξύ διαφορετικών τύπου μηχανών παραγωγής παιχνιδιών έχει ξεκινήσει να μικραίνει. Είναι πλέον εφικτό το να χρησιμοποιηθεί μία μηχανή παραγωγής παιχνιδιών πρώτου προσώπου ώστε να παραχθεί ένα βιντεοπαιχνίδι στρατηγικής, αλλά ακόμα και σε αυτή τη περίπτωση πρέπει να γίνουν ακόμα επιλογές ανάμεσα στην αποδοτικότητα μίας μηχανής και στο πόσο γενικής χρήσης είναι.

B.2.4 Τύποι μηχανών παραγωγής παιχνιδιών

Όπως αναφέρθηκε στη προηγούμενη υπο-ενότητα οι μηχανές παραγωγής παιχνιδιών συχνά προορίζονται για τη παραγωγή ενός είδους παιχνιδιού. Μία μηχανή σχεδιασμένη για ένα παιχνίδι πάλης (fighting game) δύο ατόμων σε μία παλαίστρα πυγμαχίας θα είναι πολύ διαφορετική από μία η οποία προορίζεται για την ανάπτυξη μαζικών διαδικτυακών βιντεοπαιχνιδιών (massively multiplayer online game – MMOG) ή μία μηχανή για την ανάπτυξη παιχνιδιών πρώτου προσώπου (First Person Shooter – FPS) ή μία για παραγωγή παιχνιδιών στρατηγικής πραγματικού χρόνου (real time strategy – RTS). Παρ' όλες τις διαφορές όμως κάθε τρισδιάστατο βιντεοπαιχνίδι, ανεξαρτήτως είδους, μοιράζεται κάποια κοινά στοιχεία με τα υπόλοιπα. Απαιτήσεις για την είσοδο χαμηλού επιπέδου δεδομένων μέσω χειριστηρίου ή πληκτρολογίου (user input), δυνατότητες απόδοσης στην οθόνη τρισδιάστατων μοντέλων, προβολή στην οθόνη πληροφοριών σχετικές με τις ιδιότητες και τα χαρακτηριστικά του παίκτη κάθε δεδομένη στιγμή (Heads Up Display – HUD), δυνατότητες απόδοσης χαρακτήρων κειμένου (text rendering) στην οθόνη καθώς και συστήματα παραγωγής ήχου (audio systems) συναντώνται σε κάθε είδος βιντεοπαιχνιδιού.

Στη συνέχεια θα έχουμε την ευκαιρία να εξετάσουμε τα πιο βασικά είδη βιντεοπαιχνιδιών και θα εξερευνήσουμε κάποια παραδείγματα ιδιοτήτων και τεχνολογιών που αφορά αποκλειστικά την ανάπτυξη τους.

Παιχνίδια πρώτου προσώπου

Τα παιχνίδια πρώτου προσώπου καθιερώθηκαν μέσα από τίτλους όπως τα Quake, Unreal Tournament, Half – Life και Battlefield. Παιχνίδια αυτού του τύπου στη πλειοψηφία τους περιλαμβάνουν παραδοσιακά, σχετικά αργή κίνηση σε χώρους μεγάλης κλίμακας αλλά στα πλαίσια μικρών εσωτερικών χώρων. Στη μοντέρνα εποχή παρόλα αυτά τα παιχνίδια πρώτου προσώπου έχουν αρχίσει να λαμβάνουν χώρα και σε μεγάλης κλίμακας επίπεδα και τρισδιάστατα περιβάλλοντα. Συχνά πλέον τα μέσα με τα οποία κινείται ο χαρακτήρας σε ένα τέτοιο παιχνίδι πέρα από την αργή κίνηση με οπτική πρώτου προσώπου περιλαμβάνουν διαφόρων ειδών οχήματα και εργαλεία που αυξάνουν τη ταχύτητα μετατόπισης στο εικονικό περιβάλλον.

Τα παιχνίδια πρώτου προσώπου είναι συνήθως τα πιο απαιτητικά ως προς τον σχεδιασμό τους και τις τεχνολογίες που πρέπει να ληφθούν υπόψη σε κάθε στάδιο ανάπτυξης. Το γεγονός αυτό αποδίδεται στο ότι είναι απαραίτητο για τα παιχνίδια πρώτου προσώπου να προσφέρουν στον χρήστη-παίκτη μία καθηλωτική και υπερρεαλιστική εμπειρία. Για τους παραπάνω λόγους διακρίνεται πως δεν είναι τυχαίο ότι οι περισσότερες καινοτομίες στο χώρο της βιομηχανίας ανάπτυξης και σχεδιασμού εικονικών περιβαλλόντων και βιντεοπαιχνιδιών προήρθαν από παιχνίδια του συγκεκριμένου είδους.

Σε αυτό το είδος θα συναντήσει κανείς ιδιότητες ανάπτυξης και σχεδιασμού όπως :

- Βέλτιστες τεχνικές απόδοσης μεγάλης κλίμακας τρισδιάστατων κόσμων
- Λειτουργίες κάμερας και στόχευσης υψηλής ανταπόκρισης
- Υψηλής ευκρίνειας κινήσεις και μοντέλα για τα άκρα και τον εξοπλισμό των χαρακτήρων
- Μεγάλος αριθμός περίτεχνου εξοπλισμού προς χρήση
- Ευέλικτη, μη άκρως ρεαλιστική και απαιτητική προσομοίωση της κίνησης των χαρακτήρων, γεγονός που δίνει πολλές φορές την αίσθηση πως ο χαρακτήρας μας επιπλέει σε νερό καθώς μετατοπίζεται στο χώρο
- Ιδιαίτερες ιδιότητες τεχνητής νοημοσύνης για τους μη χειριζόμενους χαρακτήρες του ψηφιακού περιβάλλοντος μας (Non player characters – NPCs)
- Μικρής κλίμακας διαδικτυακού τρόπου παιχνιδιού υποστηρίζοντας περιβάλλοντα 10-100 παικτών

Η τεχνολογία απόδοσης στα παιχνίδια πρώτου προσώπου είναι σχεδιασμένη να είναι πάντα υψηλά αποδοτική και προσεκτικά ρυθμισμένη ανάλογα με το περιβάλλον που καλείται να απεικονίσει στην οθόνη. Για παράδειγμα παιχνίδια που διαδραματίζονται σε μικρούς χώρους όπως σπηλιές και μπουντρούμια (dungeon crawl) συχνά επιστρατεύουν τεχνικές όπως τα (binary space partitioning trees). Κατά τη συγκεκριμένη τεχνική ο χώρος του παιχνιδιού χωρίζεται σε τμήματα μέσω της αναδρομής κι έτσι το αποτέλεσμα είναι να δημιουργείται σιγά σιγά ένα δυαδικό δέντρο όπως ονομάζεται στον τομέα των δομών δεδομένων. Παιχνίδια πρώτου προσώπου που λαμβάνουν χώρα σε εξωτερικούς χώρους μεγάλης κλίμακας από την άλλη χρησιμοποιούν τεχνικές απόκρυψης λεπτομερειών που δε βρίσκονται στο φάσμα όρασης του παίκτη (occlusion culling) με μία αυτοματοποιημένη μέθοδο τμηματοποίησης του περιβάλλοντα χώρου σε ορατό και μη ορατό με ακτίνες (rays) που προέρχονται από τη τοποθεσία του παίκτη. Με αυτόν τρόπο αυξάνονται οι πιθανότητες ο ρυθμός ανανέωσης της οθόνης να μην επηρεαστεί και κατ'επέκταση η διαδραστική εμπειρία του χρήστη.

Φυσικά το έργο δημιουργίας μίας καθηλωτικής εμπειρίας για τον χρήστη δεν σταματάει στις τεχνολογίες απόδοσης γραφικών. Οι κινήσεις των μοντέλων (animations), τα ηχητικά εφέ και η μουσική, η προσομοίωση ευχάριστης στον χρήστη κίνησης των σωμάτων του χώρου καθώς και οι μη διαδραστικές σκηνές του παιχνιδιού (in game cutscenes), που συχνά στόχο έχουν την εγκαθίδρυση της ιστορίας του παιχνιδιού, είναι μερικά ακόμα στοιχεία που θεωρούνται εξίσου σημαντικά.

Παιχνίδια πλατφόρμας και παιχνίδια τρίτου προσώπου

Τα παιχνίδια πλατφόρμας (platformer) είναι όρος που αποδίδεται στα βιντεοπαιχνίδια τα οποία παρουσιάζουν ένα χαρακτήρα μέσω οπτική τρίτου προσώπου με κύριο συστατικό τη μεταπήδηση από πλατφόρμα σε πλατφόρμα ως βασικού τρόπου μετατόπισης του παίκτη στον τρισδιάστατο ή δισδιάστατο περιβάλλοντα χώρο του παιχνιδιού. Παραδείγματα αποτελούν βιντεοπαιχνίδια δύο διαστάσεων όπως το Super Mario και το Donkey Kong καθώς και βιντεοπαιχνίδια τριών διαστάσεων όπως το Crash Bandicoot και το Sonic the Hedgehog.

Ως προς τις τεχνολογικές απαιτήσεις τα παιχνίδια πλατφόρμας μπορούν να συγκριθούν με τα παιχνίδια τρίτου προσώπου γενικότερα. Παραδείγματα αποτελούν οι σειρές Resident Evil και Gears of War. Τα παιχνίδια τρίτου προσώπου που βασίζονται σε χαρακτήρες, σε αντίθεση με κάποιο παιχνίδι puzzle κατά το οποίο ο χρήστης βλέπει το ταμπλό του παιχνιδιού σε τρίτο πρόσωπο κατά τη διάρκεια του σχεδιασμού των κινήσεων του, μοιράζονται κοινά στοιχεία με αυτά που

συναντήσαμε στη κατηγορία των παιχνιδιών πρώτου προσώπου. Η χαρακτηριστική διαφορά τους είναι πως τα παιχνίδια τρίτου προσώπου δίνουν αρκετά μεγαλύτερη έμφαση στις ικανότητες τις οποίες κατέχει ο παίκτης στο ρεπερτόριο του καθώς και στις επιλογές που του παρέχονται σχετικά με τον τρόπο με τον οποίο θα διασχίσει ένα περιβάλλον. Επιπρόσθετα περισσότερες λεπτομέρειες απαιτούνται για το σχεδιασμό του τρισδιάστατου μοντέλου του κεντρικού χαρακτήρα μιας και λόγω της οπτικής γωνίας είναι ορατό στον χρήστη ολόκληρο το “σώμα” του εκάστοτε ήρωα, ανάγκη που δε συναντάται στα παιχνίδια πρώτου προσώπου επειδή το μόνο ορατό τμήμα στην οθόνη είναι συνήθως τα χέρια του χαρακτήρα τα οποία τοποθετούνται μπροστά ακριβώς από το οπτικό πεδίο του χρήστη. Σε αυτό το σημείο είναι καλό να σημειωθεί πως σχεδόν όλα τα παιχνίδια πρώτου προσώπου πλέον υποστηρίζουν και τρόπους παιχνιδιού με πολλούς παίκτες οπότε κατά περίπτωση είναι δυνατό να υπάρχει και μία ολοκληρωμένη απόδοση του σώματος – τρισδιάστατου μοντέλου. Παρ’ όλα αυτά η ευκρίνεια ακόμα και σε αυτή τη περίπτωση των μοντέλων σε ένα παιχνίδι πρώτου προσώπου δε πλησιάζει την ποιότητα των μοντέλων των υπόλοιπων οντοτήτων του περιβάλλοντος πόσο μάλλον τη ποιότητα των μοντέλων σε ένα τρίτου προσώπου παιχνίδι.

Σε ένα παιχνίδι τύπου πλατφόρμας ο κεντρικός χαρακτήρας μοιάζει συχνά με καρικατούρα και δεν αποτελεί ρεαλιστική ή υψηλών λεπτομερειών αναπαράσταση. Παρ’ όλα αυτά στη γενικότερη περίπτωση των παιχνιδιών τρίτου προσώπου συναντούμε αρκετά συχνά περιπτώσεις λεπτομερών ανθρωποειδών μοντέλων χαρακτήρων. Και στις δύο περιπτώσεις ο χαρακτήρας του παίκτη έχει παραδοσιακά μία πλούσια εργαλειοθήκη από πιθανές δυνατότητες και εύρος κινήσεων.

Μερικά απαραίτητα στοιχεία για τη δημιουργία ενός διαδραστικού περιβάλλοντος τρίτου προσώπου γενικότερα είναι τα εξής:

- Κινούμενες πλατφόρμες, σκάλες, σχοινιά και κάθε είδος αντικείμενο προς ανάβαση από το κεντρικό χαρακτήρα.
- Στοιχεία γρίφων που καλούν τον χρήστη να επινοήσει λύσεις ώστε να εξελιχθεί η ιστορία του παιχνιδιού
- Μία κάμερα τρίτου προσώπου η οποία είναι προσκολλημένη πίσω από το μοντέλο του κεντρικού χαρακτήρα και τον ακολουθεί καθ όλη τη διάρκεια των κινήσεων του μέσα στο περιβάλλον του παιχνιδιού. Οι περιστροφικές κινήσεις της κάμερας αυτής ελέγχονται, στη συντριπτική πλειοψηφία των περιπτώσεων με εντολές του χρήστη του ποντικιού ή του δεξιού μοχλού του χειριστηρίου τα οποία δρουν ως δείκτες για το που θα στρέψει τη προσοχή της η κάμερα.
- Ένα πολύπλοκο σύστημα συγκρούσεων κάμερας (camera collision system) το οποίο εγγυάται πως η οπτική θέση της κάμερας δεν θα ξεπεράσει ποτέ τα όρια κάποιου αντικειμένου του περιβάλλοντα χώρου ή τα όρια του ίδιου του χώρου. Το πρόβλημα αυτό συναντάται συχνά και ονομάζεται clipping.

Παιχνίδια πάλης

Τα παιχνίδια πάλης είναι συχνά δύο παικτών και περιλαμβάνουν τις περισσότερες φορές μάχες με ρεαλιστικά ή φανταστικά στοιχεία σε μία παλαίστρα. Τα πιο γνωστά παιχνίδια που έδωσαν το έναυσμα για τη καθιέρωση του τομέα των παιχνιδιών πάλης είναι τα Street Fighter και Tekken.

Τα κύρια παραδοσιακά χαρακτηριστικά που περιέχουν τα παιχνίδια πάλης και κατ' επέκταση οι μηχανές που έχουν σχεδιαστεί και αναπτυχθεί για τη παραγωγή τους είναι τα εξής :

- Πολλαπλές ιδιότητες κινήσεων
- Αναγνώριση χτυπημάτων που δέχεται ένα μοντέλο με υψηλή ακρίβεια
- Ένα σύστημα υποδοχής και επεξεργασίας δεδομένων εισόδου που προέρχονται από το χειριστήριο ή πληκτρολόγιο του χρήστη, το οποίο να είναι ικανό να απορροφήσει υψηλό αριθμό εισόδων αλλά και πολύπλοκους συνδυασμούς αυτών. Η συγκεκριμένη λειτουργία βοηθάει στην εκτέλεση μεγάλων αλυσιδωτών πατημάτων κουμπιών (combo) για την εκτέλεση πολύπλοκων κινήσεων ενός χαρακτήρα.
- Απλά περιβάλλοντα τα οποία είναι χαμηλής ποιότητας χωρίς κίνηση τα οποία λειτουργούν ως υπόβαθρο της πίστας-παλαίστρας.
- Κοινό το οποίο είναι και αυτό χαμηλής ποιότητας συχνά δισδιάστατης αναπαράστασης.

Από τη στιγμή που ο τρισδιάστατος κόσμος των παιχνιδιών αυτών είναι μικρός και η κάμερα έχει συνεχώς στο κέντρο του οπτικού της πεδίου τη δράση ανάμεσα στους δύο χαρακτήρες της παλαίστρας, τα παιχνίδια τέτοιου είδους δε χρησιμοποιούν τεχνικές τμηματοποίησης του κόσμου (world subdivision) ή απόκρυψης λεπτομερειών.

Τα μοντέρνα παιχνίδια πάλης παρ' όλα αυτά όπως το Injustice και Mortal Kombat έχουν ανεβάσει τον πήχη επιστρατεύοντας τα εξής σύγχρονα στοιχεία :

- Τρισδιάστατα μοντέλα χαρακτήρων υψηλής ευκρίνειας
- Ρεαλιστικές σκιάσεις του δέρματος των χαρακτήρων και οπτικά εφέ όπως ιδρώτας
- Φωτό-ρεαλιστικό φωτισμό και εφέ σωματιδίων με πιθανές χρήσεις σε εφέ αίματος ή βροχής
- Πολύπλοκες κινήσεις μοντέλων
- Φυσικές προσομοιώσεις της κίνησης υφασμάτων και τριχών

Τέλος μοντέρνα εγχειρήματα όπως το βιντεοπαιχνίδι πάλης For Honor θέτουν αντιμέτωπους μεγάλο αριθμό παικτών με τη παλαίστρα να αποτελεί πλέον ένα μεγάλων διαστάσεων εικονικό περιβάλλον. Συχνά παιχνίδια τέτοιου τύπου ανήκουν σε μία υποκατηγορία που χαρακτηρίζεται ως brawler.

Παιχνίδια οδήγησης

Η συγκεκριμένη κατηγορία περιλαμβάνει όλα τα παιχνίδια που έχουν ως κεντρικό τρόπο παιχνιδιού την οδήγηση κάποιου οχήματος όπως ένα αμάξι ή ακόμα και ένα φανταστικό όχημα σε μία πίστα ταχύτητας. Το είδος αυτό και οι μηχανές παραγωγής τους παρουσιάζει ιδιαίτερο ενδιαφέρον λόγω των υποκατηγοριών τους ανάλογα με το επίπεδο ρεαλισμού. Παιχνίδια τα οποία είναι βασισμένα σε πιστές προσομοιώσεις της οδήγησης όπως συμβαίνει στη πραγματική ζωή (simulation based) όπως το Gran Turismo και παιχνίδια που έχουν ως στόχο τη παροχή διασκέδασης και όχι ρεαλιστικών εμπειριών όπως το Forza Horizon θέτουν τα όρια για τις δύο μεγάλες υποκατηγορίες. Έπειτα επιπρόσθετες κατηγορίες μπορούν να δημιουργηθούν ανάλογα με το ύφος του παιχνιδιού. Διαδραστικές εμπειρίες όπως το Need For Speed με τις αγωνιστικές πίστες να αποτελούνται από τους δρόμους πόλεων (street racing), παιχνίδια με αγώνες μικρών αμαξιδίων (kart racing) ή ακόμα και παιχνίδια με αγώνες ταχύτητας αεροπλάνων.

Ένα παιχνίδι αγώνων ταχύτητας είναι συχνά αρκετά γραμμικό ως προς τον τρόπο που ο παίκτης κινείται κατά την εξέλιξη των αγώνων ή της ιστορίας γενικότερα χαρακτηριστικό που τα κάνει να μοιάζουν με τα παιχνίδια πρώτου προσώπου που έχουμε αναφέρει. Με αυτό το τρόπο έμφαση δίνεται σε μεσαίου μεγέθους αγωνιστικές πίστες που ίσως εμφανίζουν και στοιχεία παράκαμψης κατά περίπτωση.

Τα παιχνίδια οδήγησης στη πλειοψηφία των περιπτώσεων εστιάζουν όλη τους την προσοχή στις λεπτομέρειες των οχημάτων και των κοντινών στοιχείων τους. Γι αυτό το λόγο κάποιες λειτουργίες που συναντάμε σε μία μηχανή παραγωγής παιχνιδιών οδήγησης είναι οι εξής :

- Διάφορες τεχνικές-τρικ χρησιμοποιούνται στην απόδοση αντικειμένων που βρίσκονται σε μεγάλη απόσταση από το όχημα, όπως δέντρα. Πιο συγκεκριμένα η αντικατάσταση τρισδιάστατων αντικειμένων με δισδιάστατα “χαρτόνια” έπειτα από το πέρας μίας απόστασης βοηθάει στην συντήρηση σταθερών επιδόσεων του παιχνιδιού χωρίς να καταλαβαίνει τη διαφορά ο παίκτης.

- Κάθε πίστα χωρίζεται σε τμήματα τα οποία τροφοδοτούν τα συστήματα τεχνητής νοημοσύνης ώστε να λειτουργούν με ακρίβεια καλά σχεδιασμένοι αλγόριθμοι εύρεσης μονοπατιού (path finding).
- Η κάμερα κατά κύριο λόγο όπως και στα παιχνίδια τρίτου προσώπου ακολουθάει το όχημα, κεντρικό χαρακτήρα στη παρούσα περίπτωση, από το πίσω μέρος. Επιπρόσθετα παρέχονται λειτουργίες πρώτου προσώπου που θέτουν τη ματιά του παίκτη στο εσωτερικό της καμπίνας του οχήματος.
- Σε περίπτωση που μία πίστα έχει ως τμήματα της τούνελ ή άλλα ημι-εσωτερικά περιβάλλοντα τότε δίνεται ιδιαίτερη προσοχή από τους προγραμματιστές και χρήστες της μηχανής ώστε να μην υπάρχει η δυνατότητα η κάμερα να συγκρουστεί με κάποιο από τα όρια προς κάθε κατεύθυνση.

Παιχνίδια στρατηγικής

Το μοντέρνο είδος παιχνιδιών στρατηγικής έγινε γνωστό με τα παιχνίδια Warcraft, Command and Conquer, Age of Empires και Starcraft. Σε αυτό το είδος ο παίκτης τοποθετεί τα πιόνια του στο χώρο του παιχνιδιού με σκοπό να κατατροπώσει τον αντίπαλο του. Ο χώρος του παιχνιδιού αυτός συχνά αποτελείται από τη κάτοψη ενός τρισδιάστατου, ή δισδιάστατου παλαιότερα χάρτη. Επιπρόσθετα συχνά πραγματοποιείται ο διαχωρισμός ανάμεσα στα παιχνίδια πραγματικού και μη χρόνου, με τα δεύτερα να βλέπουν μια αλληλουχία γύρων που ο μόνο ο ένας παίκτης μπορεί να πραγματοποιήσει κινήσεις. Παιχνίδια αυτού τύπου περιλαμβάνονται στη σειρά Civilization. Ο παίκτης τις περισσότερες φορές είναι περιορισμένος αρκετά στο τρόπο που μπορεί να αλλάξει τη γωνία θέασης του περιβάλλοντα χώρου και τον μονάδων του, γεγονός που οφείλεται σε λόγους σχετικά με το ότι οι προγραμματιστές των μηχανών σε αυτά τα παιχνίδια έχουν πραγματοποιήσει τεχνικές βελτιστοποίησης που θα αλλοιώνονταν σε διαφορετική περίπτωση.

Παλαιότερα παιχνίδια του είδους συχνά χρησιμοποιούσαν ένα δισδιάστατο χώρο όπου ο παίκτης βρισκόταν στην οπτική γωνία της κάτοψης. Από την άλλη τα σύγχρονα παιχνίδια στρατηγικής προσφέρουν εναλλακτικές γωνίες θέασης του εικονικού περιβάλλοντος όπως επίσης και τρισδιάστατα μοντέλα. Δημοφιλές παράδειγμα αποτελεί η σειρά Total War.

Μερικές από τις σημαντικότερες και συνήθεις τεχνικές και βελτιστοποιήσεις που συναντάει κανείς στο είδος των παιχνιδιών αυτών είναι οι εξής :

- Κάθε μονάδα ή αντικείμενο που έχει υπό τον έλεγχο του ο παίκτης προβάλλεται σε σχετικά χαμηλή ανάλυση ώστε να έχει τη δυνατότητα η μηχανή να παρουσιάζει στην οθόνη μεγάλο αριθμό μονάδων (large scale).
- Το έδαφος του χώρου πάνω στον οποίο τοποθετούνται και κινούνται οι διάφοροι πράκτορες του παιχνιδιού είναι χωρίζεται σε τμήματα ποικίλου ύψους (height – field terrain).
- Ο παίκτης έχει τη δυνατότητα να επιλέξει το σημείο στο οποίο θα δημιουργήσει κτίρια.
- Η διάδραση με τον κόσμο του παιχνιδιού από πλευράς χρήστη γίνεται μέσω κλικ του ποντικιού και επιλογές μέσω διαφόρων διαθέσιμων μενού για τη χρήση του κάθε πράκτορα ή κάθε κτιρίου.

Παιχνίδια πολλών παικτών μεγάλης κλίμακας

Οι γνωστότερες σειρές παιχνιδιών αυτού του είδους περιλαμβάνουν τα World of Warcraft, Everquest και Guild Wars. Ένα παιχνίδι πολλών παικτών μεγάλης κλίμακας είναι (MMO) ορίζεται ως οποιοδήποτε παιχνίδι υποστηρίζει μεγάλο αριθμό παικτών που δρουν ταυτόχρονα. Συχνά αυτός ο αριθμός βρίσκεται μεταξύ των χιλιάδων και εκατοντάδων χιλιάδων. Επιπρόσθετα οι χρήστες αυτοί δρουν σε ένα περιβάλλον χώρο όπου οι πράξεις τους διατηρούνται για μεγάλο χρονικό διάστημα κι έτσι αυτός παραμένει σταθερός κάτι που δημιουργεί τη ψευδαίσθηση ότι βρίσκεται σε μια παράλληλη με τη πραγματική ζωή. Τα υπόλοιπα τμήματα των παιχνιδιών αυτού του είδους μοιάζουν με όλα τα υπόλοιπα διαδικτυακά τμήματα άλλων ειδών παιχνιδιών. Υποκατηγορίες αυτών είναι τα MMORPG, παιχνίδια όπου ο παίκτης δημιουργεί ένα δικό του χαρακτήρα και αναλαμβάνει ρόλους όπως μάγος και τοξότης, MMOFPS και MMORTS.

Οι παραπάνω διαδικτυακές υπηρεσίες μεγάλης κλίμακας που προσφέρουν τα παιχνίδια αυτού του είδους παρέχονται από το υλικό ηλεκτρονικών υπολογιστών που λειτουργούν ως διακομιστές (servers) και αποθηκεύουν τη κατάσταση του εικονικού περιβάλλοντος κάθε δεδομένη στιγμή μαζί με τις πράξεις που πραγματοποιούν οι παίκτες. Επιπρόσθετα λόγω της ύπαρξης αυτών των διακομιστών, που επιβαρύνουν την εταιρεία παραγωγής του εκάστοτε παιχνιδιού με μακροχρόνια έξοδα συντήρησης τους, οι παίκτες καλούνται να πληρώνουν κάθε μήνα κάποιο είδος συνδρομής. Άλλες μέθοδοι αύξησης των εσόδων είναι η παροχή περιορισμένου χρόνου διάθεσης αντικειμένων προς χρήση από τους παίκτες, εξοπλισμός ή εικονικά κατοικίδια, μέσω μικρο-συναλλαγών (micro transactions). Με αυτό το τρόπο οδηγούμαστε στο συμπέρασμα πως ίσως το πιο σημαντικό έργο που καλούνται να διεκπεραιώσουν οι διακομιστές κάποιου παιχνιδιού είναι η ομαλή λειτουργία των υπηρεσιών χρηματικών συναλλαγών.

Η ευκρίνεια των γραφικών στα παιχνίδια αυτού του είδους είναι σε κάθε περίπτωση χαμηλότερη συγκριτικά με τα υπόλοιπα είδη που αναφέραμε λόγω της μεγάλης κλίμακας του εικονικού περιβάλλοντος και του μεγάλου αριθμού παιχτών.

Σήμερα έχουν αναπτυχθεί παιχνίδια όπως το Destiny τα οποία ενώ το κύριο χαρακτηριστικό τους είναι αποστολές τεσσάρων ατόμων σε οπτική πρώτου προσώπου, υπάρχουν ειδικά σχεδιασμένες περιοχές εντός του παιχνιδιού που δανειζόμενες στοιχεία από το τον χώρο των MMOG, επιτρέπουν τη δυνατότητα αλληλεπίδρασης με παραπάνω από 4 παίκτες. Επιπλέον παράδειγμα παιχνιδιού το οποίο ενώνει τον κόσμο παιχνιδιών πρώτου και τρίτου προσώπου με αυτόν των MMOG είναι το Player's Unknown's Battlegrounds (PUBG). Κάνοντας δημοφιλές το νέο είδος battle royale, επιτρέπει την αναμέτρηση 100 παικτών σε ένα περιβάλλον την ίδια στιγμή.

Παιχνίδια με περιεχόμενο σχεδιασμένο από χρήστες

Με την ραγδαία αύξηση χρήσης των κοινωνικών μέσων (social media) τα παιχνίδια έχουν ξεκινήσει να αποκτάνε ολοένα και περισσότερα στοιχεία συν εργατικότητας. Μια πρόσφατη τάση στο τομέα της σχεδίασης και ανάπτυξης εικονικών περιβαλλόντων και διαδραστικών εμπειριών είναι η υποστήριξη του περιεχομένου σχεδιασμένο από παίκτες (player authored content). Παραδείγματα αυτής της στάσης εντοπίζονται στη σειρά βιντεοπαιχνιδιών Little Big Planet. Το κύριο είδος στο οποίο ανήκουν είναι τα παιχνίδια γρίφων πλατφόρμας (puzzle platformer) αλλά τα πιο γνωστά και αξιοπρόσεκτα στοιχεία τους έχουν να κάνουν με το γεγονός ότι το ίδιο το παιχνίδι ενθαρρύνει τους παίκτες να δημιουργήσουν και να εκδώσουν τους δικούς τους γρίφους και κόσμους ώστε να παιχθούν από άλλους παίκτες.

Παρ' όλα αυτά το πιο γνωστό παιχνίδι στο χώρο αυτόν είναι το Minecraft. Το συγκεκριμένο παιχνίδι φαίνεται ιδιοφυές στη σύλληψη του μιας και ο βασικός τρόπος σχεδιασμού του περιβάλλοντος του έχει ως κεντρικό στοιχείο κύβους (voxels) οι οποίοι συνθέτουν τμηματικά το έδαφος και κάθε άλλο αντικείμενο του εικονικού περιβάλλοντος που σε συνδυασμό με χαμηλής ανάλυσης μοτίβα που τοποθετούνται πάνω τους, προσομοιώνουν διάφορα υλικά όπως χώμα, ξύλο και μάρμαρο. Ο χώρος του περιβάλλοντος επίσης απαρτίζεται από πράκτορες τεχνητής νοημοσύνης που λειτουργούν είτε ως σύμμαχοι, ανθρωποειδή και κάτοικοι ψηφιακών χωριών, είτε ως εχθροί, βρικόλακες και λύκοι.

Οι παίκτες μπορούν να δημιουργήσουν ένα μοναδικό κόσμο ο οποίος δημιουργήθηκε με βάση τυχαία δεδομένα και έπειτα έχουν τη δυνατότητα να αλλοιώσουν των περιβάλλοντα χώρο του παιχνιδιού σκάβοντας τούνελ ή ακόμα και με το να τοποθετούν διάφορους κύβους υλικών ώστε να

δημιουργήσουν κατασκευές όπως κτίρια και φράγματα. Η μεγαλύτερη ίσως καινοτομία που προσφέρει η εργαλειοθήκη του Minecraft είναι η παροχή ενός ειδικού υλικού που ονομάζεται redstone. Το redstone λειτουργεί ως σύρμα ηλεκτρικής συνδεσμολογίας (wiring) επιτρέποντας στους παίκτες να δημιουργήσουν ολοκληρωμένα ηλεκτρικά κυκλώματα, αυτόματους μοχλούς και συστήματα σιδηροδρόμων μεταξύ άλλων δυναμικών εργαλείων. Τα παραπάνω έχουν ως αποτέλεσμα οι παίκτες να είναι σε θέση να κατασκευάσουν οτιδήποτε επινοήσουν με τη φαντασία τους και ταυτόχρονα να είναι και διαθέσιμα προς χρήση σε άλλους παίκτες.

Παιχνίδια εικονικής πραγματικότητας

Η εικονική, η επαυξημένη και η μεικτή πραγματικότητα είναι ενδιαφέρουσες νέες τεχνολογίες οι οποίες στόχο έχουν να καθηλώσουν των θεατή-χρήστη σε τρισδιάστατα περιβάλλοντα τα οποία είναι εξολοκλήρου προϊόντα ηλεκτρονικών υπολογιστών ή είναι επαυξημένα με τη χρήση ψηφιακών εικόνων. Οι τεχνολογίες αυτές έχουν εφαρμογές σε πληθώρα τομέων σχετικά με τα βιντεοπαιχνίδια, τις οποίες θα εξετάσουμε παρακάτω.

Εικονική πραγματικότητα

Η εικονική πραγματικότητα (Virtual Reality-VR) μπορεί να οριστεί ως μία εμπειρία πολυμέσων ή ως μία υπολογιστή προσομοίωση η οποία αποδίδει την εμπειρία ενός χρήστη σε ένα περιβάλλον υπαρκτό ή φανταστικό. Η υπολογιστικά παραγόμενη εικονική πραγματικότητα αποτελεί ένα υποσύνολο της τεχνολογίας κατά την οποία ο εικονικός κόσμος είναι εξολοκλήρου προϊόν γραφικών υπολογιστών. Στη περίπτωση αυτή ο χρήστης τοποθετεί τον εαυτό του στο ψηφιακό περιβάλλον με τη χρήση συσκευών κεφαλής. Οι συσκευές αυτές προβάλλουν οποιοδήποτε περιεχόμενο απευθείας στα μάτια του χρήστη, ταυτόχρονα υπάρχουν δυνατότητες παρακολούθησης και καταγραφής της κίνησης της συσκευής ώστε να αντικατοπτρίζεται σε αντίστοιχη κίνηση στο ψηφιακό περιβάλλον. Επιπρόσθετα ο χρήστης κρατάει στα χέρια του συσκευές οι οποίες επιτρέπουν στην εκάστοτε μηχανή γραφικών να καταγράφει και τη κίνηση των άκρων του. Με αυτό τον τρόπο αντικείμενα όπως κουτιά μπορούν να σπρωχτούν ή να κρατηθούν από τα εικονικά άκρα του χρήστη.

Εμπειρίες επαυξημένης και μεικτής πραγματικότητας

Οι όροι επαυξημένη πραγματικότητα (augmented reality) και μεικτή πραγματικότητα (mixed reality) συχνά χρησιμοποιούνται με την ίδια σημασία, παρ' όλα αυτά λανθασμένα. Κατά τις δύο αυτές τεχνολογίες παρουσιάζονται στο χρήστη εικόνες από τον πραγματικό κόσμο αλλά με τη χρήση υπολογιστών προστίθενται κάποια επιπρόσθετα εικονικά στοιχεία. Επίσης και στις δύο κατηγορίες συναντάμε εξωτερικές συσκευές να χρησιμοποιούνται αντί των συσκευών κεφαλής που αναλύσαμε στη προηγούμενη κατηγορία. Αυτές μπορεί να είναι ένα κινητό, ένα τάμπλετ ή ακόμα και ένα ειδικά κατασκευασμένο ζευγάρι γυαλιών και προβάλλουν μέσω των οθονών τους κάνοντας χρήση κάμερας μία στατική απεικόνιση του αληθινού περιβάλλοντος στην οποία έχουν “προσκολληθεί” ψηφιακά αντικείμενα. Στα πραγματικού χρόνου συστήματα επαυξημένης και μεικτής πραγματικότητας χρησιμοποιούνται οι διάφοροι δέκτες της συσκευής όπως ο μετρητής αλλαγής ταχύτητας σε ένα κινητό κι έτσι δίνεται η δυνατότητα στη κάμερα του εικονικού περιβάλλοντος να καταγράψει τις κινήσεις της κάμερας της φυσικής συσκευής. Με αυτό τον τρόπο δημιουργείται η ψευδαίσθηση στο χρήστη πως η οθόνη της συσκευής που χρησιμοποιεί είναι ένα παράθυρο στον εικονικό κόσμο, ολοκληρώνοντας έτσι την ιδέα των προσκολλημένων στην επιφάνεια του ψηφιακών στοιχείων.

Αρκετοί έχουν επιχειρήσει να διακρίνουν τις δύο αυτές κατηγορίες με το να χρησιμοποιούν τον όρο επαυξημένη πραγματικότητα για τεχνολογίες όπου γραφικά υπολογιστών τοποθετούνται στην επιφάνεια μίας άμεσης ή έμμεσης απεικόνισης της πραγματικότητας αλλά είναι ξεκάθαρο πως δεν υπάρχουν σε αυτόν. Από την άλλη ο όρος μεικτή πραγματικότητα εφαρμόζεται συχνότερα σε τεχνολογίες κατά τις οποίες γραφικά υπολογιστών αποδίδουν εικονικά αντικείμενα τα οποία προσομοιώνουν την ύπαρξή τους στο πραγματικό.

Μερικά παραδείγματα επαυξημένης πραγματικότητας είναι τα εξής :

- Ο αμερικάνικος στρατός παρέχει στους στρατιώτες συσκευές οι οποίες μέσω επαυξημένης πραγματικότητας δίνουν τη δυνατότητα ενός συστήματος παρόμοιου με τα παιχνίδια το οποίο έχει τη δυνατότητα να προβάλλει στα μάτια του χρήστη τον χάρτη της τοποθεσίας της οποίας βρίσκεται καθώς και στοιχεία όπως θερμοκρασία και ηλιοφάνεια.
- Κάμερες μιας συσκευής μπορούν να ανιχνεύσουν κάποιο σχέδιο στην επιφάνεια ενός χαρτιού και να προβάλλουν στην οθόνη της συσκευής αυτής ένα τρισδιάστατο μοντέλο το οποίο είναι συσχετιζόμενο. Η Disney είχε παρουσιάσει αντίστοιχες λειτουργίες χρησιμοποιώντας τους χαρακτήρες της.
- Η εταιρεία Pepsi είχε δημιουργήσει μία διαφημιστική καμπάνια κατά την οποία περαστική υπόκεινται σε φάρσες μέσω προβολών εικόνων επαυξημένης πραγματικότητας άγριων ζώων.

Μερικά παραδείγματα μεικτής πραγματικότητας είναι τα εξής :

- Με την τελευταία έκδοση του λογισμικού κινητών συσκευών Android από την εταιρεία Google οι χρήστες μπορούν να χρησιμοποιήσουν αυτοκόλλητα μεικτής πραγματικότητας που τοποθετούνται σε φωτογραφίες και βίντεο.
- Η συσκευή Hololens της εταιρείας Microsoft μπορεί να προβάλει καρφίτσωμένα αντικείμενα σε οντότητες του πραγματικού κόσμου. Συχνά χρήση του γίνεται για εκπαιδευτικούς σκοπούς στη μηχανική και ιατρική.

Αυτή τη στιγμή η βιομηχανία βιντεοπαιχνιδιών βρίσκεται σε στάδιο πειραματισμού όσον αφορά τις τεχνολογίες εικονικής πραγματικότητας. Οι πρώτες προσπάθειες έγιναν με τη μεταφορά ήδη αναπτυγμένων συμβατικών παιχνιδιών σε συσκευές εικονικής πραγματικότητας.

Πιο εντυπωσιακές προσπάθειες παρουσιάζονται στο βιντεοπαιχνίδι Job Simulator κατά το οποίο ο παίκτης τοποθετείται σε ένα εικονικό εργασιακό περιβάλλον μουσείο. Σε αυτή τη περίπτωση η εργοδότης του, εξωγήινο με σκοπό να μελετήσουν το ανθρώπινο είδος, των αναγκάζουν να πραγματοποιήσει αναπαραστάσεις εργασιακών δεξιοτήτων που συναντάει κανείς στη πραγματική ζωή. Το επόμενο παιχνίδι της σειράς Vacation Simulator θέτει στον παίκτη μία λίστα από κινήσεις που πρέπει να εκτελέσει για να χαλαρώσει και να απολαύσει τις διακοπές του. Ο τόνος και τον δύο παιχνιδιών αυτών είναι χιουμοριστικός με στοιχεία καρτούν.

Οι τεχνολογίες των μηχανών παιχνιδιών εικονικής πραγματικότητας δε διαφέρει και πολύ με αυτές των παιχνιδιών πρώτου προσώπου. Στη πραγματικότητα πολλές μηχανές παραγωγής παιχνιδιών πρώτου προσώπου προσφέρουν πλέον ενσωματωμένες δυνατότητες μεταφοράς κάθε υπό σχεδίασης παιχνιδιού σε περιβάλλοντα εικονικής πραγματικότητας. Συγκεκριμένα κάποιες τεχνολογίες μηχανών που προορίζονται για χρήση σε εικονικά περιβάλλοντα είναι οι εξής :

- Στερεοσκοπικές τεχνικές απόδοσης γραφικών (stereoscopic rendering). Ένα παιχνίδι εικονικής πραγματικότητας πρέπει να αποδίδει δύο φορές την ίδια εικόνα, μία φορά για κάθε μάτι. Το γεγονός αυτό διπλασιάζει τις απαιτήσεις όπως οι απόδοση των στοιχειωδών οντοτήτων του χώρου. Έτσι καταλαβαίνουμε πως τα παιχνίδια εικονικών κόσμων έχουν συχνά περισσότερους περιορισμούς στην ανάπτυξη της λόγω των περισσότερων πόρων που απαιτούνται για την ομαλή λειτουργία των τεχνολογιών απόδοσης.
- Τα παιχνίδια αυτού του είδους απαιτούν υψηλές απόδοσης που συνοδεύονται και από ανάγκες για υψηλούς ρυθμούς ανανέωσης της οθόνης. Έρευνες έχουν δείξει πως ρυθμοί ανανέωσης χαμηλότεροι από τα 90 καρτέ ανά δευτερόλεπτο προκαλούν ναυτία και αποπροσανατολισμό στον χρήστη. Το γεγονός αυτό σε συνδυασμό με το προηγούμενο επιβαρύνει ακόμα περισσότερο τους επεξεργαστές και τις κάρτες γραφικών των ηλεκτρονικών υπολογιστών.

- Περιορισμοί στο τρόπο μετατόπισης του παίκτη στον εικονικό κόσμο σε σχέση με τη κίνηση αυτού στον πραγματικό. Συνήθως τα τρισδιάστατα περιβάλλοντα των βιντεοπαιχνιδιών είναι μεγαλύτερου μεγέθους από ότι το πραγματικό περιβάλλον στο οποίο κινείται ο παίκτης. Γρήγορες μετατοπίσεις του χαρακτήρα στο κόσμο του παιχνιδιού καθώς και ιδιότητες πτήσεις είναι απαγορευτικές. Γι αυτό το λόγο χρησιμοποιούνται συχνότερα τεχνικές κατά τις οποίες ο παίκτης “δείχνει” το σημείο στο οποίο θέλει να μεταβεί με αποτέλεσμα να μεταφέρεται σε αυτό. Συσκευές που επιτρέπουν στον χρήστη οι οποίες επιτρέπουν να περπατήσουν επί τόπου ώστε να μετατοπίσουν το χαρακτήρα τους στο εικονικό περιβάλλον είναι υπό κατασκευή.

Φυσικά πέρα από τους παραπάνω περιορισμούς και δυσκολίες οι τεχνικές ανάπτυξης παιχνιδιών εικονικής πραγματικότητας έχουν προσφέρει και νέες συναρπαστικές δυνατότητες όπως :

- οι χρήστες μπορούν να αγγίζουν και να πετάζουν αντικείμενα του εικονικού κόσμου, απλώνοντας τα άκρα τους στον φυσικό κόσμο
- ένας παίκτης μπορεί να αποφύγει μία επίθεση προς το μέρος του στον εικονικό κόσμο με το να σκύβει στη πραγματική ζωή
- νέες δυνατότητες παρουσίασης στοιχείων πληροφοριών των ζωτικών σημείων του εικονικού χαρακτήρα ή στοιχεία για τον εξοπλισμό του, τα οποία είναι προσκολλημένα στα άκρα του χαρακτήρα.

B.2.5 Μηχανές παραγωγής παιχνιδιών

Η οικογένεια μηχανών Quake

Το πρώτο τρισδιάστατο παιχνίδι πρώτου προσώπου θεωρείται από όλους το Castle Wolfenstein 3D (1992). Αναπτυγμένο από την ID Software και προοριζόταν για τη πλατφόρμα ηλεκτρονικών υπολογιστών και αποτελούσε την αρχή μίας μεγάλης σειράς παιχνιδιών, οικογένειας μηχανών παραγωγής παιχνιδιών και τεχνολογιών που άλλαξαν τη βιομηχανία για πάντα. Οι επόμενοι τίτλοι που αναπτύχθηκαν από την ίδια εταιρεία περιλαμβάνουν τίτλους όπως τα Doom και τη σειρά Quake. Κάθε μηχανή που χρησιμοποιήθηκε για την ανάπτυξη των μετέπειτα τίτλων έχει αρκετές ομοιότητες με τη προηγούμενη.

Πολλά παιχνίδια δανείστηκαν τη μηχανή και δημιουργήθηκαν μέσα από της τεχνολογίες της οικογένειας Quake. Γνωστή περίπτωση αποτελεί η μηχανή Source της Valve η οποία έδωσε ζωή στη μεγάλη σειρά βιντεοπαιχνιδιών πρώτου προσώπου Half Life της οποίας η μηχανή ακόμα και σήμερα έχει στοιχεία από τη πρώτη μηχανή Quake.

Επιπρόσθετα ο πηγαίος κώδικας των τίτλων Quake 1 και Quake 2 διατίθεται από την εταιρεία ανοιχτός για όλους και αποτελούν υποδείγματα καθαρής και επεκτάσιμης αρχιτεκτονικής προγραμματισμού ακόμα και αν σε μερικά σημεία γίνεται χρήση απαρχαιωμένων τεχνικών μιας και έχει αναπτυχθεί με χρήση της γλώσσας προγραμματισμού C. Για τους λόγους αυτούς θεωρείται μεγάλης εκπαιδευτικής αξίας το να κατεβάσει κάποιος τον κώδικα και να τον μελετήσει ακόμα και σήμερα. Μεγάλο ενδιαφέρον παρουσιάζεται στο τρόπο που λειτουργεί κάθε τμήμα του κώδικα-μηχανής γεγονός που τον καθιστά χρήσιμο για επίδοξους προγραμματιστές βιντεοπαιχνιδιών και όχι μόνο.

Μηχανή Unreal

Η εταιρεία Epic μπήκε στο χώρο των βιντεοπαιχνιδιών πρώτου προσώπου το 1998 με το θρυλικό τίτλο Unreal. Από τότε η μηχανή του τίτλου (Unreal Engine) αποτέλεσε τον κύριο ανταγωνιστή των τεχνολογιών Quake για την ανάπτυξη παιχνιδιών πρώτου προσώπου. Με την έκδοση της Unreal Engine 2 κυκλοφόρησε και ο τίτλος Unreal Tournament 2004 (UT2004), λογισμικό το οποίο αποτέλεσε τη βάση για τον σχεδιασμό και τη κυκλοφορία πολλαπλών μεταλλάξεων (mutators) από προγραμματιστές – χρήστες, πανεπιστημιακά δημιουργήματα και εμπορικά παιχνίδια. Η πιο πρόσφατη έκδοση της μηχανής αυτής ονομάζεται Unreal Engine 4 και προσφέρει τη πλουσιότερη ως τώρα εργαλειοθήκη λειτουργιών. Οι δυνατότητες της χρησιμοποιούνται μέσα από τη γλώσσα προγραμματισμού C++ αλλά και οπτικά μέσα προγραμματισμού τύπου visual programming τα οποία ονομάζονται Blueprints και δεν απαιτούν γνώσεις προγραμματισμού.

Τα εργαλεία που την έκαναν να ξεχωρίζουν έχουν να κάνουν σχέση με τις δυνατότητες επεκτασιμότητας που προσφέρει μιας και εφόσον ο χρήστης είναι διατεθειμένος να πληρώσει μία χρηματική συνδρομή μπορεί να έχει πρόσβαση στον πηγαίο κώδικα της μηχανής. Αυτό οδηγεί τους προγραμματιστές να προβαίνουν σε διάφορες αλλαγές ώστε να βελτιστοποιήσουν το τρόπο λειτουργίας της μηχανής για τους δικούς τους σκοπούς. Μέσω αυτής μπορεί κανείς να σχεδιάσει και να αναπτύξει οποιοδήποτε παιχνίδι επιθυμεί του είδους πρώτου και τρίτου προσώπου.

Τέλος η ίδια εταιρεία παρέχει δωρεάν ολόκληρο το εγχειρίδιο χρήσης της μηχανής καθώς και επί πληρωμής θεματικές ενότητες με σκοπό την εκπαίδευση επίδοξων προγραμματιστών βιντεοπαιχνιδιών.

Μηχανή source

Η μηχανή Golden Source και η εξέλιξη της Source 1 και Source 2 οδήγησαν την ανάπτυξη των γνωστών τίτλων της σειράς Half Life 2, Left 4 Dead και DOTA2. Κύρια χαρακτηριστικά της είναι οι δυνατότητες επεκτασιμότητας τις οποίες προσφέρει καθώς και οι χαμηλές απαιτήσεις των παιχνιδιών που παράγονται από αυτή χωρίς να θυσιάζεται σε υπερβολικό βαθμό η ευκρίνεια των γραφικών που είναι σε θέση να αποδώσει. Έτσι μπορεί να καλύψει κάποιος μεγαλύτερο μερίδιο αγοράς μιας και δε χρειάζεται ο χρήστης να αποκτήσει νεότερο υλικό για τον ηλεκτρονικό του υπολογιστή. Η μηχανή αυτή βρίσκεται διαθέσιμη για εμπορικούς σκοπούς έπειτα από συνεννόηση με την εταιρεία Valve.

Μηχανή Frostbite

Η μηχανή Frostbite του στούντιο ανάπτυξης βιντεοπαιχνιδιών DICE είναι το προϊόν προσπαθειών αναζήτησης νέων τεχνολογιών απόδοσης γραφικών κατά την διαδικασία σχεδιασμού του παιχνιδιού Battlefield Bad Company κατά τη περίοδο του 2006. Από τότε το σύνολο των προσπαθειών αυτό απέκτησε όνομα και αποτέλεσε τη κύρια μηχανή παραγωγής εικονικών εμπειριών που χρησιμοποιεί πλέον η εταιρεία Electronics Arts (EA) και τροφοδοτεί σύγχρονους τίτλους όπως το Mass Effect και το Star Wars Battlefront 2.

Τα κύρια χαρακτηριστικά της που τη κάνουν να ξεχωρίζει μεταξύ άλλων είναι τα εξής :

- Σύνολο εργαλείων με ονομασία FrostED και παρέχει δυνατότητες εύκολης και γρήγορης κατασκευής οντοτήτων προς χρήση σε κάποιο βιντεοπαιχνίδι
- Διαδικτυακές υπηρεσίες οι οποίες παρέχουν δυνατότητες σύνδεσης με πολλούς παίκτες σε παιχνίδια μεγάλης κλίμακας, με όνομα Frostbite Backend Services
- Εργαλεία ανάπτυξης στοιχείων μικρο-συναλλαγών ενσωματωμένα

Παρ' όλες τις εκπληκτικές δυνατότητες της μηχανής αυτής, δεν υπάρχουν τρόποι απόκτησης και χρήσης της εκτός των ορίων της εταιρείας Electronics Arts και είναι κλειστού κώδικα.

Μηχανή Rockstar Advanced Game Engine (RAGE)

Η μηχανή RAGE τροφοδοτεί την διαδικασία ανάπτυξης βιντεοπαιχνιδιών της εταιρείας Rockstar και κρύβεται πίσω από τίτλους όπως Grand Theft Auto V και Read Dead Redemption. Τα χαρακτηριστικά της που τη κάνουν να διαφέρουν, σχετίζονται με τις δυνατότητες απόδοσης τρισδιάστατων εικονικών περιβαλλόντων μεγάλης κλίμακας ανοικτού κόσμου. Έτσι ο παίκτης έχει μία αδιάκοπη ψηφιακή εμπειρία που προσομοιώνει πολύ καλά αυτή της πραγματικής ζωής στοιχεία που είναι ορατά στη σειρά παιχνιδιών Grand Theft Auto όπου ο κεντρικός χαρακτήρας μπορεί να κινηθεί ελεύθερα σε μεγάλες πόλεις και να έχει τυπικές συναναστροφές με τους πράκτορες που κινούνται ελεύθερα στο χώρο, δημιουργώντας έτσι τη ψευδαίσθηση ενός ζωντανού κόσμου.

Μηχανή CRYENGINE

Η εταιρεία Crytek αρχικά ανέπτυξε την πανίσχυρη μηχανή γραφικών η οποία είναι γνωστή ως CRYENGINE σαν ένα προωθητικό μέσο για τις δυνατότητες των καρτών γραφικών της εταιρείας παραγωγής υλικού υπολογιστών NVIDIA. Με την αναγνώριση των δυνατοτήτων όμως από τον χώρο των γραφικών υπολογιστών η Crytek πήρε τη πρωτοβουλία να αναπτύξει ένα ολόκληρο παιχνίδι με το όνομα Far Cry το οποίο έκανε τεράστια επιτυχία. Η μηχανή αυτή από τότε αποτέλεσε το εργαλείο παραγωγής πολλαπλών τίτλων όπως είναι τα βιντεοπαιχνίδια Crysis και Ryse : Son of Rome. Με τη πάροδο των χρόνων η μηχανή γνώρισε πολλαπλές αλλαγές και προσθήκες ώστε σήμερα να κυκλοφορεί η πιο πρόσφατη έκδοση αυτής με την ονομασία CRYENGINE 5. Η χρήση της μηχανής προσφέρει δυνατότητες κυκλοφορίας τίτλων για διάφορες πλατφόρμες και κινητές συσκευές. Χρήση της μηχανής για παραγωγή εμποριών τίτλων μπορεί να γίνει από οποιονδήποτε έπειτα από συνεννόηση με την εταιρεία.

Μηχανή Unity

Η Unity είναι μία πανίσχυρη μηχανή παραγωγής παιχνιδιών η οποία προσφέρει δυνατότητες ανάπτυξης τίτλων για διάφορες πλατφόρμες, κινητές συσκευές, έξυπνες τηλεοράσεις και συσκευές εικονικής πραγματικότητας.

Πιο συγκεκριμένα οι κύριοι πυλώνες γύρω από τους οποίους κινείται ο σχεδιασμός και η ανάπτυξη της μηχανής Unity όπως και οι λειτουργίες που παρέχει είναι η ευχρηστία και η εύκολη επέκταση του κάθε τίτλου σε διάφορες πλατφόρμες. Τα παραπάνω εντοπίζονται μέσα από το ενσωματωμένο περιβάλλον επεξεργασίας (in engine editor) το οποίο προσφέρει δυνατότητες πραγματοποίησης αλλαγών στο υπό σχεδίαση εικονικό περιβάλλον όπως επίσης και τη δυνατότητα ζωντανών δοκιμών κάθε στιγμή μέσω προεπισκόπησης των διαφόρων αλλαγών στον ίδιο τον editor αλλά και στην πλατφόρμα που στοχεύει ο προγραμματιστής. Επιπρόσθετα παρέχει μια μεγάλη λίστα από εργαλεία διαθέσιμα για προγραμματιστές και σχεδιαστές σχετικά με την ανάλυση του κώδικα του παιχνιδιού αλλά και την ευκαιρία να δημιουργηθούν τεχνικές σκίασης χωρίς τη χρήση προγραμματισμού στο όνομα της λειτουργίας VFX Graph και Shader Graph. Οι παραπάνω λειτουργίες καταφέρνουν να ενώσουν τον κόσμο των προγραμματιστών και των καλλιτεχνών που δουλεύουν πάνω σε έναν τίτλο παιχνιδιού. Η γλώσσα που χρησιμοποιεί η μηχανή αυτή είναι η C# και ο πηγαίος κώδικας της παρέχεται έπειτα από τη πληρωμή μηνιαίας συνδρομής παρέχοντας έτσι τη δυνατότητα σε όποιον το επιθυμεί να πραγματοποιήσει αλλαγές μέσω της γλώσσας C++ ώστε να πετύχει το καλύτερο αποτέλεσμα με αυτό.

Η μηχανή Unity είναι ιδιαίτερα αγαπητή στην ανεξάρτητη κοινότητα προγραμματιστών και σχεδιαστών (indie developers) κυρίως λόγω της ευχρηστίας της και της πολύ καλής υποστήριξης που προσφέρεται από την εταιρεία. Μερικοί τίτλοι που χρησιμοποιούν τη μηχανή αυτή είναι το Deus Ex : The Fall, το Hollow Knight και Cup head.

Τέλος προσπάθειες έχουν γίνει για τη προώθηση της μηχανής ως μέσω παραγωγής μη διαδραστικών εμπειριών όπως ταινίες και γι αυτό έχει δοθεί μεγάλη έμφαση από την εταιρεία στην ανάπτυξη εργαλείων animation και λειτουργιών σχετικών με την απόδοση γραφικών υψηλής ποιότητας. Παραδείγματα αποτελούν η ψηφιακές ταινίες μικρού μήκους Adam και The Heretic.

Μηχανές ανοιχτού κώδικα

Οι μηχανές ανοιχτού κώδικα συχνά σχεδιάζονται από επαγγελματίες αλλά και ερασιτέχνες προγραμματιστές. Ο όρος ανοιχτού κώδικα τους αποδίδεται διότι τυπικά υπονοείται η παροχή ολόκληρου του πηγαίου κώδικα χωρίς αντίτιμο και σε συνδυασμό με πλατφόρμες που προάγουν τη συνεργασία πολλών ατόμων ώστε ο καθένας να μπορεί να υλοποιήσει αλλαγές και να προτείνει να

γίνουν αποδεκτές από τον πυρήνα της μηχανής ώστε να υιοθετηθούν ως μόνιμες από την επόμενη έκδοση αυτής. Επιπρόσθετα όπως κάθε λογισμικό ανοιχτού κώδικα έτσι και οι μηχανές παραγωγής παιχνιδιών εντάσσονται υπό τη προστασία κάποιας άδειας κυκλοφορίας (license), οι οποίες καθορίζουν τους επιτρεπόμενους τρόπους χρήσης μιας μηχανής καθώς και τις δυνατότητες αποκόμισης κερδών μέσω εμπορευματοποίησης αυτής.

Παραδείγματα μηχανών παραγωγής παιχνιδιών ανοιχτού κώδικα παρουσιάζονται παρακάτω :

- OGRE3D, θεωρείται ως μηχανή απόδοσης τρισδιάστατων γραφικών και όχι ολοκληρωμένη μηχανή παραγωγής παιχνιδιών από τους επίσημους προγραμματιστές της. Παρ' όλα αυτά με τη πάροδο των χρόνων η κοινότητα έχει αναπτύξει μεγάλο αριθμό εργαλείων που σε συνδυασμό με την αρχική μηχανή απόδοσης γραφικών που προσφέρει η OGRE3D μπορεί κανείς να κατασκευάσει ένα ολόκληρο βιντεοπαιχνίδι. Χρησιμοποιείται η γλώσσα C++ και αποτελεί από τις πρώτες στο χώρο.
- Torque και Irrlicht υπό την άδεια MIT.
- Lumberyard, μηχανή κατασκευασμένη από την Amazon η οποία περιέχει τμήματα της αρχιτεκτονικής της CRYENGINE.

B.2.6 Αρχιτεκτονική σχεδιασμού εικονικού περιβάλλοντος

Η αρχιτεκτονική σχεδιασμού μιας μηχανής παραγωγής εικονικών περιβαλλόντων και βιντεοπαιχνιδιών αποτελείται από πολλά τμήματα τα οποία θα εξεταστούν σε αυτήν την υπο-ενότητα.

Όπως όλες οι βάσεις κώδικα λογισμικού έτσι και οι μηχανές παραγωγής παιχνιδιών και εικονικών περιβαλλόντων αναπτύσσονται αν στρώσεις. Συνήθως τα ανώτερα στρώματα εξαρτώνται από την ομαλή και βέλτιστη λειτουργία των κατώτερων αλλά όχι το αντίθετο. Όταν ένα ένα κατώτερο στρώμα εξαρτάται από τη λειτουργία ενός ανώτερου, συναντάμε το πρόβλημα της κυκλικής σχέσης εξάρτησης (circular dependency). Δίνεται μεγάλη βάση στο να αποφευχθούν περιπτώσεις κατά της οποίες προκύπτει το πρόβλημα αυτό γιατί οι βάσεις κώδικα που επηρεάζει μετατρέπονται σε ασταθής και χαμηλής απόδοσης.

Παρακάτω θα εξετάσουμε τα βασικά τμήματα ενός τέτοιου συστήματος παραγωγής διαδραστικών εμπειριών καθώς και τους γενικότερους παράγοντες που το επηρεάζουν και πρέπει να ληφθούν υπόψη κατά τον σχεδιασμό.

Πλατφόρμα λογισμικού

Η πλατφόρμα που στοχεύουν ως μέσω προσπέλασης του υπό κατασκευή λογισμικού διαφέρει από περίπτωση σε περίπτωση. Τυπικά στο χώρο των βιντεοπαιχνιδιών και των γραφικών υπολογιστών περιλαμβάνει το υλικό ενός ηλεκτρονικού υπολογιστή ή μιας κονσόλας. Οι πιο γνωστές πλατφόρμες προσωπικών υπολογιστών είναι τα Windows, MacOS και Linux, κινητών συσκευών Android και iOS και κονσολών XBOX και Playstation. Οι περισσότερες τεχνικές ανάπτυξης σχεδιάζονται και χρησιμοποιούνται ανεξαρτήτου πλατφορμας.

Οδηγοί συσκευών

Οι οδηγοί συσκευών (device drivers) αποτελούν χαμηλού επιπέδου (low-level) τμήματα λογισμικών που παρέχονται από των δημιουργό του λογισμικού ή του παραγωγού υλικού υπολογιστών. Οι οδηγοί αυτοί ελέγχουν τη κατανομή πόρων και θωρακίζουν το λειτουργικό σύστημα και τα ανώτερα στρώματα μίας μηχανής από τις λεπτομέρειες των διαφορετικών ειδών υλικού υπολογιστών που είναι διαθέσιμα. Στόχους τους είναι η αδιάκοπη και ομαλή εξέλιξη των λειτουργιών του συστήματος με κάθε συνδυασμό υλικού.

Λειτουργικό σύστημα

Σε έναν προσωπικό υπολογιστή το λειτουργικό σύστημα (operating system-OS) λειτουργεί διαρκώς. Ελέγχει και οργανώνει οποιαδήποτε εκτέλεση και λειτουργία κάποιου λογισμικού, με μία κατηγορία να αποτελείται από τα βιντεοπαιχνίδια.

Στις παλαιότερες κονσόλες βιντεοπαιχνιδιών το οποιαδήποτε λειτουργικό σύστημα, αν υπήρχε, αποτελούνταν από μια μικρή συλλογή βιβλιοθηκών που ήταν ενσωματωμένα στο ίδιο το παιχνίδι που εισήγαγε στη κονσόλα ο χρήστης. Σε αυτές τις κονσόλες το παιχνίδι λάμβανε τον πλήρη έλεγχο του συστήματος κατά τη εκτέλεση του. Πλέον αυτό δε συμβαίνει μιας και το λειτουργικό σύστημα

της κονσόλας μπορεί να διακόπτει τη ροή εκτέλεσης του παιχνιδιού και να δεσμεύσει συγκεκριμένους πόρους ώστε να προβάλει κάποιο μήνυμα ή την είσοδο κάποιου φίλου του χρήστη στο παιχνίδι. Επιπλέον στις κονσόλες PlayStation 4 και XBOX ONE εκτελούνται διαρκώς υπηρεσίες οι οποίες προσφέρουν τη δυνατότητα καταγραφής βίντεο από την οθόνη του χρήστη καθώς και τη δυνατότητα διαμοιρασμού του στο διαδίκτυο. Από τα παραπάνω καταλαβαίνει κανείς ότι το χάσμα μεταξύ κονσόλας και προσωπικού υπολογιστή σιγά σιγά ψαλιδίζεται.

Εργαλεία ανάπτυξης τρίτων

Οι περισσότερες μηχανές παραγωγής παιχνιδιών χρησιμοποιούν έναν αριθμό βοηθητικών εργαλείων οι οποίες έχουν αναπτυχθεί από τρίτους ώστε να διεκπεραιώσουν βασικές λειτουργίες.

Για παράδειγμα κατά την ανάπτυξη ενός βιντεοπαιχνιδιού δημιουργείται η ανάγκη για χρήση δομών δεδομένων που θα χρησιμοποιηθούν αποθήκες δεδομένων όπως και αλγορίθμων για την διεξαγωγή εργασιών πάνω σε αυτά τα δεδομένα. Παραδείγματα τέτοιων εργαλείοις είναι τα εξής :

- Η βιβλιοθήκη Boost, παρέχει δυνατότητες χρήσης δομών δεδομένων και αλγορίθμων μέσα από τη γλώσσα προγραμματισμού C++.
- Folly, η βιβλιοθήκη που έχει αναπτυχθεί μέσα στα πλαίσια του Facebook και σκοπό έχει την επέκταση της βασικής βιβλιοθήκης της C++ αλλά και της Boost.
- Loki, βιβλιοθήκη με λειτουργίες template meta programming και generic προγραμματισμού.

Η βασική βιβλιοθήκη της C++ παρέχει επίσης αρκετές από τις υπηρεσίες που προσφέρει η Boost. Το υποσύνολο της βασικής βιβλιοθήκης το οποίο υλοποιεί γενικού τύπου κλάσεις αποθήκευσης όπως το `std::vector` και `std::list` αναφέρεται συνήθως ως standard template library (STL).

Προσπάθειες για αυτές τις λειτουργίες είχαν γίνει καιρό πριν η C++ μπορούσε να παρέχει τέτοιες δυνατότητες. Με τη πάροδο των χρόνων αυτές απορροφήθηκαν στη γλώσσα ως επίσημα στοιχεία έπειτα από την έγκριση επίβλεψης της γλώσσας.

Εργαλεία απόδοσης γραφικών

Το τμήμα μιας μηχανής που ασχολείται με την βέλτιστη απόδοσης γραφικών στην οθόνη συχνά αποτελείται από βιβλιοθήκες όπως :

- Glide, η οποία παρείχε δυνατότητες τρισδιάστατων απεικονίσεων σε μία σειρά από τις πρώτες κάρτες γραφικών που είχαν κυκλοφορήσει, της σειράς Voodoo.
- OpenGL η οποία είναι ένας από τους δύο μεγάλους ανταγωνιστές του χώρου των γραφικών υπολογιστών πλέον. Είναι ανοιχτού κώδικα και υποστηρίζει πληθώρα πλατφορμών.
- DirectX είναι η εργαλειοθήκη για παραγωγή γραφικών και είναι σχεδιασμένη από τη Microsoft. Αποτελεί τον κύριο αντίπαλο της OpenGL.
- Libgcm είναι μια χαμηλού επιπέδου βιβλιοθήκη η οποία επιτρέπει τη χρήση του υλικού που βρίσκεται εντός των κονσολών της Sony.
- Vulkan, μία νέα βιβλιοθήκη από τους δημιουργούς και υποστηρικτές της OpenGL. Στόχο έχει την άμεση επικοινωνία με τις λειτουργίες μίας κάρτας γραφικών και λειτουργεί με την παροχή λιστών εντολών που στέλνονται προς εκτέλεση σε μία κάρτα γραφικών όπως επίσης και το πλήρη έλεγχο της μνήμης της κάρτας και των πόρων που μοιράζονται με τον επεξεργαστή.

Συστήματα προσομοίωσης συγκρούσεων

Η ανίχνευση συγκρούσεων και οι προσομοίωση φυσικών ιδιοτήτων μεταξύ οντοτήτων σε ένα εικονικό περιβάλλον όπως τα βιντεοπαιχνίδια είναι έργο απαιτητικό και γι αυτό έχουν δημιουργηθεί βιβλιοθήκες που καλύπτουν αυτές τις ανάγκες. Παρακάτω παρουσιάζονται ως εξής :

- Havok, είναι πιο γνωστή βιβλιοθήκη προσομοιώσεων που χρησιμοποιείται ακόμα και εκτός της βιομηχανίας βιντεοπαιχνιδιών.
- Η βιβλιοθήκη PhysX αποτελεί ακόμα μία πανίσχυρη μηχανή προσομοίωσης φυσικής και συγκρούσεων η οποία είναι αναπτυγμένη από την Nvidia.
- Open Dynamics Engine (ODE) είναι επίσης μία γνωστή ανοιχτού κώδικα εργαλειοθήκη ανίχνευσης συγκρούσεων μεταξύ οντοτήτων σε ένα εικονικό περιβάλλον.

Συστήματα δημιουργίας κινήσεων χαρακτήρων

Οι κύριες εργαλειοθήκες οι οποίες προσφέρουν βοηθητικές λειτουργίες για τη δημιουργία κινήσεων χαρακτήρων (animation) και χρησιμοποιούνται στη βιομηχανία είναι οι εξής :

- Το γνωστό σύστημα της Rad Game Tools' με την ονομασία Granny προσφέρει πληθώρα βοηθητικών εργαλείων για τη δημιουργία τρισδιάστατων κινούμενων μοντέλων τα οποία υποστηρίζονται από όλα τα σχεδιαστικά προγράμματα. Επίσης προσφέρει λειτουργίες ανάγνωσης λειτουργιών κίνησης που βρίσκονται ενσωματωμένα σε τρισδιάστατα μοντέλα καθώς και δυνατότητα επεξεργασίας αυτών.
- Havok Animation. Η διαφορά μεταξύ animation και δυνατοτήτων physics αρχίζει να μικραίνει μιας και ο βαθμός ρεαλισμού των τρισδιάστατων εικονικών περιβαλλόντων που σχεδιάζονται στις μέρες μας αυξάνεται ραγδαία. Η εταιρεία σχεδιασμού και ανάπτυξης του πακέτου Havok έχει δώσει ιδιαίτερη έμφαση στην γεφύρωση μεταξύ των τομέων του animation και physics.
- Η βιβλιοθήκη OrbisAnim η οποία αναπτύχθηκε για τη κονσόλα Playstation 4 από την εταιρεία SN σε συνεργασία με τις ομάδες σχεδιασμού Εργαλείων και Τεχνολογιών (Tools and Technology) της Sony Playstation και την ομάδα Προηγμένων Τεχνολογιών (Advanced Technology Group) της Sony Europe, περιλαμβάνει μία πανίσχυρη μηχανή παραγωγής animation και βέλτιστου υπολογισμού γεωμετρικών ιδιοτήτων για καλύτερη απόδοση γραφικών.

Συστήματα βιομηχανικών ιδιοτήτων τρισδιάστατων μοντέλων

Τα δύο πιο γνωστά πακέτα τα οποία περιλαμβάνουν τέτοιες λειτουργίες είναι τα Endorphin και Euphoria. Χαρακτηριστικό αυτών είναι η δυνατότητα δημιουργίας κινήσεων χαρακτήρων (character motion) με τη χρήση προηγμένων βιομηχανικών (biomechanical) μοντέλων που προέρχονται από τη ρεαλιστική κίνηση του ανθρώπινου σώματος.

Όπως αναφέρθηκε παραπάνω η διαχωριστική γραμμή μεταξύ των κινήσεων χαρακτήρων και των δυνατοτήτων physics έχει ξεκινήσει να θολώνει. Πακέτα όπως τα Havok Animation προσπαθούν να ενώσουν τις λειτουργίες σχετικά με τα physics και τις κινήσεις χαρακτήρων μέσα από τον σχεδιασμό των κινήσεων αυτών με ένα πρόγραμμα σχεδίασης τρισδιάστατων μοντέλων τα οποία χρησιμοποιούν physics ώστε να ενισχύσουν τη ρεαλιστικότητα διαφόρων κινήσεων.

Το Endorphin αποτελεί ένα πρόσθετο εργαλείο (plug in) που μπορεί να χρησιμοποιηθεί από το πρόγραμμα τρισδιάστατης σχεδίασης Maya το οποίο επιτρέπει στους σχεδιαστές να προσομοιάσουν ολόκληρες βιομηχανικές αναπαραστάσεις πάνω στα υπό σχεδίαση τρισδιάστατα μοντέλα και να εξάγουν τις κινήσεις που προήλθαν ως αποτέλεσμα της διαδικασίας χωρίς να τα σχεδιάσουν αυτή με το χέρι τους. Το βιομηχανικό μοντέλο που προκύπτει λαμβάνει υπόψη για κάθε

κίνηση τη βαρύτητα, τη κατανομή βάρους του μοντέλου και τον τρόπο με τον οποίο κινείται το ανθρώπινο σώμα όταν βρίσκεται υπό την επήρεια διάφορων δυνάμεων.

Το Euphoria από την άλλη αποτελεί τη μεταφορά του Endorphin για χρήση σε εφαρμογές πραγματικού χρόνου ώστε να παραχθούν ρεαλιστικές κινήσεις οντοτήτων βασισμένες στα βιομηχανικά μοντέλα του προγραμματιστικού κόσμου σε διαδραστικές εφαρμογές όπως τα βιντεοπαιχνίδια.

Χρήση του συγκεκριμένου πακέτου πραγματοποιήθηκε κατά την ανάπτυξη του βιντεοπαιχνιδιού GTAIV από την εταιρεία Rockstar, παιχνίδι το οποίο έγινε ιδιαίτερα δημοφιλές λόγω των ρεαλιστικών κινήσεων του κεντρικού χαρακτήρα σε περιπτώσεις σύγκρουσης και του τρόπου με τον οποίο ανταποκρινόταν το σώμα αυτού σε απότομες αλλαγές της ταχύτητας του.

Συστήματα ανεξαρτητοποίησης πλατφόρμας

Οι περισσότερες μηχανές παραγωγής βιντεοπαιχνιδιών πρέπει να λειτουργούν ομαλά σε παραπάνω από μία πλατφόρμες. Εταιρείες όπως η Electronic Arts και Activision Blizzard για παράδειγμα πάντα προορίζουν τα παιχνίδια τους για κυκλοφορίας σε πολλές διαφορετικές πλατφόρμες διότι με αυτόν τον τρόπο μπορούν να καλύψουν μεγαλύτερο μερίδιο αγοράς. Παραδοσιακά τα μόνα στούντιο βιντεοπαιχνιδιών τα οποία δεν στοχεύουν στην υποστήριξη πολλών διαφορετικών πλατφορμών είναι αυτά που αποτελούν τμήμα μιας εταιρείας παραγωγής και διάθεσης κονσολών. Αυτό γίνεται ώστε όποιος θέλει να παίξει κάποιο παιχνίδι που βρίσκεται αποκλειστικά σε μία τέτοια κονσόλα να είναι υποχρεωμένος να την αγοράσει, έτσι μειώνεται το ρίσκο απώλειας πωλήσεων. Για τους παραπάνω λόγους οι περισσότερες μηχανές υποστηρίζουν ένα στρώμα λειτουργιών το οποίο τους επιτρέπει να υποστηρίζουν πολλές διαφορετικές πλατφόρμες κερδίζοντας έτσι ανεξαρτησία. Το στρώμα αυτό βρίσκεται πάνω από τις λειτουργίες υλικού και τμημάτων λογισμικού και θωρακίζει τα κύρια τμήματα της μηχανής παραγωγής γραφικών από το να έχουν γνώσει του υλικού της πλατφόρμας προσφέροντας ενδιάμεσες λειτουργίες οι οποίες μετατρέπουν τη κάθε εντολή των προγραμματισμών στην αντίστοιχη της πλατφόρμας που θέτεται ως στόχος υποστήριξης.

Υπάρχουν δύο κύριοι λόγοι που προσφέρονται οι λειτουργίες που “τυλίζουν” τα κατώτερα στρώματα δυνατοτήτων σε αυτές τις περιπτώσεις. Πρώτον, κάποιες διεπαφές προγραμματισμού (Application Programming Interface – API), όπως αυτές που προσφέρονται από το λειτουργικό σύστημα ή όπως κάποιες συναρτήσεις (functions) σε γλώσσες προγραμματισμού όπως η C προσφέρει στη βασική βιβλιοθήκη της, διαφέρουν από πλατφόρμα σε πλατφόρμα. Με το να υπάρχει ένα στρώμα το οποίο βρίσκεται πάνω από τις συναρτήσεις που αναφέρθηκαν παραπάνω

τότε επιτυγχάνεται η ύπαρξη ενός σταθερού μέσου διεπαφής προγραμματισμού για όλες τις πλατφόρμες. Δεύτερον, ακόμα και σε περίπτωση που γίνεται χρήση μίας βιβλιοθήκης που ήδη έχει ενσωματωμένες λειτουργίες ανεξαρτητοποίησης πλατφόρμας υπάρχει η ανάγκη να προστατέψει κανείς τις λειτουργίες που ήδη έχει υλοποιήσει από το να τις μεταφέρει στο μέλλον σε ένα άλλο σύστημα. Με τους παραπάνω τρόπους του έργου των προγραμματιστών να υλοποιήσουν αλλά και να συντηρήσουν το λογισμικό τους σε ποικίλες πλατφόρμες, εξαιρετικά πιο απλό.

Βασικές βοηθητικές εργαλαιοθήκες

Κάθε μηχανή παιχνιδιών κάνει χρήση μίας ομάδας βασικών λειτουργιών οι οποίες είναι συχνά επαναλαμβανόμενες και συναντώνται αρκετές φορές κατά τα περισσότερα στάδια ανάπτυξης λογισμικού παραγωγής εικονικών περιβαλλόντων αλλά και γενικού σκοπού. Μερικές από αυτές είναι παρουσιάζονται ως εξής :

- Assertions, οι οποίες είναι λειτουργίες που λειτουργούν ως έλεγχοι λαθών και εισάγονται σε τμήματα κώδικα και σκοπό έχουν να επαληθεύουν τις αρχικές εκτιμήσεις που έκανε ο προγραμματιστής για το είδος διαφόρων τιμών που θα προκύψουν κατά την έναρξη του εκτελέσιμου αρχείου. Σε περίπτωση που δεν επαληθευθούν σωστά οι έλεγχοι αυτοί τότε εμφανίζονται απαραίτητα μηνύματα.
- Λειτουργίες ελέγχου μνήμης (memory management). Στη πραγματικότητα κάθε παιχνίδι υλοποιεί το δικό του ξεχωριστό τρόπο με τον οποίο δεσμεύει και αποδεσμεύει πόρους μνήμης από το υλικό ενός συστήματος-πλατφόρμας. Στόχος τέτοιων συστημάτων είναι η παραπάνω διεργασία να χαρακτηρίζονται από ταχύτητα, ακρίβεια και ασφάλεια ώστε να μη προξενθούν προβλήματα στην ομαλή λειτουργία του εκτελέσιμου.
- Βιβλιοθήκη μαθηματικών συναρτήσεων και υπολογισμών. Η ανάπτυξη λογισμικών βιντεοπαιχνιδιού είναι άμεσα συνδεδεμένη με τον τομέα των μαθηματικών. Γι αυτό τον λόγο όλες οι μηχανές βιντεοπαιχνιδιών χρησιμοποιούν έστω και μία αν όχι πολλαπλές βιβλιοθήκες οι οποίες διευκολύνουν υπολογισμούς μεταξύ διανυσμάτων (vectors), πινάκων (matrices), περιστροφών, τριγωνομετρίας, γεωμετρικών υπολογισμών με ευθείες, ακτίνες και σφαίρες μεταξύ άλλων.
- Ειδικά κατασκευασμένες δομές δεδομένων και αλγόριθμοι. Εκτός από τη περίπτωση οι σχεδιαστές μιας μηχανής να αποφάσισαν να βασιστούν εξολοκλήρου σε πακέτα υπηρεσιών σχεδιασμένα και διαθέσιμα από τρίτους όπως τις βιβλιοθήκες Boost και Folly που αναφέρθηκαν παραπάνω, οι προγραμματιστές δημιουργούν εξ αρχής τους απαραίτητους αλγορίθμους και δομές δεδομένων που θα χρησιμοποιήσουν.

Διαχειριστής πόρων

Ο διαχειριστής πόρων βρίσκεται σε κάθε παιχνίδι με κάποια μορφή και προσφέρει μία ολοκληρωμένη διεπαφή μέσω της οποίας μπορεί να πραγματοποιηθεί η βέλτιστη πρόσβαση των πόρων (assets) και άλλων ειδών δεδομένων εισόδου και εξόδου που χρησιμοποιεί το παιχνίδι. Αρκετές μηχανές υλοποιούν αυτή τη λειτουργία με έναν κεντρικό διαχειριστή όπως τα Unreal Packages ή κλάση Resource Manager στην μηχανή OGRE. Υπάρχουν βέβαια και μηχανές παραγωγής οι οποίες αφήνουν αυτό το έργο στον προγραμματιστή/χρήστη τους δίνοντας του την ευκαιρία να έχει άμεση πρόσβαση στα αρχεία στον δίσκο ενώ αυτά βρίσκονται στην απλή τους μορφή. Αυτή η περίπτωση συναντάται στο τρόπο που χειρίζεται τους πόρους ενός βιντεοπαιχνιδιού η μηχανή Quake.

Μηχανή απόδοσης γραφικών

Το στρώμα η αλλιώς μηχανή απόδοσης γραφικών είναι συχνά ένα από τα μεγαλύτερα και πιο σημαντικά τμήματα μίας μηχανής παιχνιδιών. Υπάρχουν διάφορες τεχνικές σχεδίασης και ανάπτυξης μιας μηχανής απόδοσης γραφικών. Παρόλο που υπάρχουν διαφορετικοί τρόποι προσέγγισης του προβλήματος της απεικόνισης ψηφιακών κόσμων σχεδόν όλες οι μηχανές απόδοσης μοιράζονται κοινά στοιχεία μιας και τα κύρια τμήματα τους βασίζονται στη μορφή και στο τρόπο κατασκευής του hardware που είναι διαθέσιμο εκείνη τη στιγμή στην αγορά. Ο πιο συνηθισμένος τρόπος ανάπτυξης της μηχανής αυτών είναι μέσω του σχεδιασμού στρωμάτων, των οποίων τις λειτουργίες θα εξετάσουμε παρακάτω.

Χαμηλού επιπέδου απόδοση γραφικών

Το στρώμα το οποίο είναι υπεύθυνο για τη χαμηλού επιπέδου απόδοση γραφικών (low level renderer) περιέχει όλες τις διαδικασίες που λαμβάνουν χώρα κατά τη παραγωγή μίας ψηφιακής εικόνας στην πιο απλή τους μορφή. Σε αυτό το στρώμα ο σχεδιασμός είναι βασισμένος στη σχεδίαση των γεωμετρικών σχημάτων όσο το δυνατόν πιο γρήγορα, χωρίς να λαμβάνονται περαιτέρω πολύπλοκες ανάγκες υπόψιν όπως το ποια τμήματα της παραγόμενης σκηνής είναι

ορατά ή όχι. Το στρώμα αυτό αποτελείται από περισσότερα υπό στρώματα όπως που έχουν σκοπό την απόδοση των εξής στοιχείων :

- Materials και Shaders, τα οποία περιγράφουν το τρόπο που το κάθε υλικό του χώρου ανταποκρίνεται στο φωτισμό μιας σκηνής-περιβάλλοντος
- Στατικός και δυναμικός φωτισμός, τον τρόπο με τον οποίο λειτουργεί ο φωτισμός στο χώρο αν αυτός υποστηρίζει μετατροπές κατά την εκτέλεση του λογισμικού
- Κάμερες, δηλαδή το τρόπο με τον οποίο κινούνται οι διάφορες κάμερες στη σκηνή και την οπτική που “βλέπουν” τον τρισδιάστατο περιβάλλον
- Κειμένων και γραμματοσειρών
- Βασικών σχημάτων (Primitives), τα βασικά στοιχεία που χρησιμοποιούνται για την απόδοση οποιουδήποτε πολύπλοκου μοντέλου. Περιλαμβάνει σφαίρες, τρίγωνα, κύβους.
- Ταπετσαρίες (textures), στοιχεία τα οποία κοσμούν τη κάθε οντότητα του παιχνιδιού
- Debug mode, το οποίο αποκρύβει τις περισσότερες λεπτομέρειες της τρισδιάστατης σκηνής με σκοπό την εύρεση λαθών

Διεπαφή συσκευής γραφικών

Οι διάφορες εργαλειοθήκες γραφικών όπως τα DirectX, OpenGL ή Vulkan απαιτούν τη δημιουργία μεγάλων τμημάτων κώδικα ώστε να αριθμήσουν τις διάφορες συσκευές υλικού γραφικών, να τις ενεργοποιήσουν και να πραγματοποιήσουν μια σειρά από άλλες ρυθμίσεις. Λόγω της δυσκολίας του έργου γραφής των παραπάνω έχει δημιουργηθεί ένα εργαλείο που ονομάζεται διεπαφή συσκευής γραφικών το οποίο αναλαμβάνει να διευκολύνει αυτές τις διεργασίες.

Λοιπές λειτουργίες της απόδοσης γραφικών

Τα υπόλοιπα τμήματα ενός renderer χαμηλού επιπέδου ασχολούνται με το έργο συλλογής δεδομένων βασικών γεωμετρικών αποδόσεων, τα οποία ονομάζονται πακέτα απόδοσης (render packets) και αποτελούνται από λίστες γραμμών, λίστες σημείων, τμήματα εδάφους και κομμάτια

κειμένου, με σκοπό να σχεδιαστούν και να αποδοθούν στην οθόνη με τη μεγαλύτερη ταχύτητα που είναι εφικτή.

Γράφος σκηνής / Βελτιστοποίηση οπτικοποίησης

Η μηχανή χαμηλού επιπέδου απόδοσης όπως είδαμε προηγουμένως σχεδιάζει οποιαδήποτε γεωμετρία της δοθεί ως είσοδος χωρίς να ρυθμίζει θέματα ορατότητας, εκτός από τις πίσω όψεις των γεωμετριών οι οποίες δε φαίνονται στην οπτική της κάμερας και πρέπει να μην είναι ορατές στο πλάνο. Γι' αυτό το λόγο μία λειτουργία υψηλότερου επιπέδου είναι απαραίτητη ώστε να καθοριστεί ο αριθμός των γεωμετριών που θα αποδοθούν στην οθόνη κάθε δεδομένη στιγμή με κριτήριο κάποιον βαθμό ορατότητας.

Για τρισδιάστατα περιβάλλοντα μικρής κλίμακας, ένα απλό frustrum cull με το οποίο αφαιρούνται από τη σκηνή πλήρως τα αντικείμενα τα οποία δεν είναι ορατά στο οπτικό πεδίο της κάμερας, αρκεί. Για μεγαλύτερης κλίμακας τρισδιάστατους κόσμους μία δομή δεδομένων που ονομάζεται διαστηματικός διαμοιρασμός (spatial subdivision) μπορεί να χρησιμοποιηθεί ώστε να καθορίσει με μεγάλη ταχύτητα τα ορατά και μη ορατά στη κάμερα τμήματα του χώρου. Συχνά αυτή η δομή δεδομένων ονομάζεται και γράφος σκηνής.

Λειτουργίες γραφικών εφέ

Οι σύγχρονες μηχανές γραφικών υποστηρίζουν ένα μεγάλο εύρος συστημάτων παραγωγής οπτικών εφέ όπως :

- Συστήματα σωματιδίων (particle systems) για απεικονίσεις φωτιάς, καπνού και βροχής
- Συστήματα στάμπας (decals) για απεικόνιση οπών οι οποίες προήρθαν από σφαίρες και πατημασιών.
- Χαρτογράφηση φωτισμού και περιβάλλοντος (light mapping και environment mapping). Ένας χάρτης φωτισμού είναι μια απλή ταπετσαρία που τοποθετείται στα τρισδιάστατα μοντέλα τα οποία αντί να περιέχουν τιμές χρωμάτων , όπως οι απλές ταπετσαρίες, περιέχουν τιμές των χρωμάτων που αντικατοπτρίζουν το τρόπο με τον οποίο θα ανταποκρινόταν αυτά στο φως.
- Δυναμική σκίαση (dynamic shadows), δηλαδή σκιές οι οποίες ανταποκρίνονται σε αλλαγές του φωτισμού σε πραγματικό χρόνο

- Εφέ πλήρους οθόνης, δηλαδή τεχνικές anti-aliasing για απόκρυψη των οδοντωμάτων κατά την απεικόνιση κυρτών γραμμών και διόρθωσης χρωμάτων

Αποτελεί κοινό φαινόμενο μία μηχανή παιχνιδιών να περιλαμβάνει ένα μεμονωμένο σύστημα οπτικών εφέ το οποίο εξειδικεύεται στη βέλτιστη απεικόνιση σωματιδίων (particles) και σταμπών (decals). Στη συγκεκριμένη περίπτωση οπτικών εφέ μάλιστα τα συστήματα υπεύθυνα για την απεικόνιση των παραπάνω είναι ξεχωρίστα και δίνονται ως είσοδο στον χαμηλού επιπέδου renderer. Από την άλλη τα εφέ της χαρτογράφησης φωτισμού και λειτουργίες δυναμικών σκιών διαχειρίζονται συνήθως μέσα στην ίδια τη μηχανή απόδοσης. Το ίδιο ισχύει και για τα οπτικά εφέ πλήρους οθόνης.

Λειτουργίες Front End

Τα περισσότερα παιχνίδια υλοποιούν κάποιου είδους δισδιάστατων γραφικών τα οποία τοποθετούνται πάνω στη τρισδιάστατη απεικόνιση ενός περιβάλλοντος διευκολύνοντας έτσι τις εξής ανάγκες που προκύπτουν κατά τη διαδικασία ανάπτυξης :

- Παροχή πληροφοριών σχετικά με τα ζωτικά σημεία του κεντρικού χαρακτήρα και άλλα χαρακτηριστικά χρήσιμα για την εμπειρία του παίκτη (heads-up display, HUD)
- Δημιουργία καταλόγων επιλογών, μενού ή ακόμα και κονσόλας πληροφοριών
- Σχεδιασμός του γενικότερου γραφικού περιβάλλοντος (graphical user interface, GUI) το οποίο μπορεί να παρέχει τη δυνατότητα στον παίκτη να αναδιοργανώνει τον εξοπλισμό και τους χώρους αποθήκευσης (inventory) του χαρακτήρα του όπως επίσης και να πραγματοποιήσει πληθώρα πολύπλοκων κινήσεων όπως τη τοποθέτηση των μονάδων του ή άλλων οντοτήτων στο τρισδιάστατο περιβάλλον του παιχνιδιού.

Οι δυνατότητες δισδιάστατων γραφικών αυτές στα πλαίσια ενός τρισδιάστατου παιχνιδιού αναπτύσσονται με τη χρήση δύο ζευγαριών ενωμένων τριγώνων τα οποία προβάλλονται στην οθόνη με τέτοιο τρόπο ώστε να δημιουργούν μία στρώση δύο διαστάσεων στο οπτικό πεδίο του παίκτη. Επιπρόσθετα υπάρχει και η δυνατότητα απόδοσης των τριγώνων αυτών σε πλήρη τρισδιάστατη μορφή αλλά με τη χρήση του billboard mode, σύμφωνα με το οποίο τα τρίγωνα αυτά θα είναι στραμμένα πάντα προς τη κατεύθυνση του οπτικού πεδίου του κεντρικού χαρακτήρα.

Ακόμη στις παραπάνω ανάγκες εντάσσεται συχνά και η παρουσίαση ολόκληρων κινούμενων στοιχείων (full motion video – FMV) τα οποία αποτελούν απεικονίσεις που πραγματοποιούνται σε

ολόκληρη την οθόνη και το οπτικό πεδίο του παιχνιδιού. Τα βίντεο αυτά έχουν δημιουργηθεί σε εξωτερικό περιβάλλον από αυτό του λογισμικού του παιχνιδιού και δεν αποτελούν απεικονίσεις πραγματικού χρόνου.

Σχετικό με τα προηγούμενα είναι στο σύστημα των in-game cinematics. Αυτό το στοιχείο επιτρέπει τη προβολή μικρών ταινιών η οποίες παράγονται μέσα από το ίδια τα μοντέλα του παιχνιδιού. Για παράδειγμα καθώς ο κεντρικός χαρακτήρας του παιχνιδιού περπατάει στο τρισδιάστατο περιβάλλον και περνάει δίπλα από διάφορους πράκτορες του κόσμου, ξεκινάει η αναπαραγωγή ενός cutscene που παρουσιάζει μία μικρή συζήτηση με έναν πράκτορα. Στις περισσότερες των περιπτώσεων ο παίκτης δεν έχει τον έλεγχο του χαρακτήρα του κατά την αναπαραγωγή τέτοιων σκηνών. Παιχνίδια όπως το Uncharted 4 έχει αντικαταστήσει πλήρως τα προκατασκευασμένα Full Motion Videos με In Game Cutscenes πραγματικού χρόνου, τα οποία πολλές φορές δίνουν τη δυνατότητα ελέγχου των κινήσεων του κεντρικού χαρακτήρα, διατηρώντας έτσι τον διαδραστικό χαρακτήρα της εμπειρίας ενός βιντεοπαιχνιδιού.

Εργαλεία debugging

Τα βιντεοπαιχνίδια αποτελούν συστήματα πραγματικού χρόνου κι έτσι οι μηχανικοί λογισμικού που σχεδιάζουν και αναπτύσσουν αυτά χρειάζονται να πραγματοποιήσουν ενέργειες profiling, που σκοπό έχουν την εύρεση τρόπων βελτιστοποίησης της απόδοσης. Επιπρόσθετα, οι πόροι μνήμης είναι συνήθως αρκετά περιορισμένοι σε ένα σύστημα και γι αυτό οι προγραμματιστές κάνουν χρήση εργαλείων ανάλυσης μνήμης (memory analysis tools). Όλα αυτά τα εργαλεία ανήκουν στην ομάδα των profiling και debugging tools. Συχνά σε αυτά τα συστήματα περιλαμβάνεται και η δυνατότητα καταγραφής και αναπαραγωγής προηγούμενων κινήσεων που έγιναν στο παιχνίδι και ειδικά μενού επιλογών προσβάσιμα μόνο από τους προγραμματιστές του παιχνιδιού για τη πιο εύκολη εύρεση λαθών.

Τα πιο γνωστά εμπορικά εργαλεία κατασκευασμένα γι αυτό τον σκοπό είναι τα εξής :

- Intel Vtune
- IBM'S Quantify και Purify
- Insure++
- Valgrind

Παρ' όλα αυτά οι περισσότερες μηχανές περιλαμβάνουν χειροποίητα εργαλεία profiling και debugging που μεταξύ άλλων περιλαμβάνουν τα εξής :

- Μηχανισμούς οι οποίοι με χειροκίνητο τρόπο επιτρέπουν την εκτέλεση τμημάτων του κώδικα σε συγκεκριμένους χρόνους
- Διεργασίες οι οποίες προβάλλουν τις πληροφορίες που εξάγουν τα εργαλεία profiling στην οθόνη, όπως θερμοκρασία επεξεργαστή και ποσοστό δέσμευσης μνήμης κάρτας γραφικών
- Λειτουργίες εξαγωγής δεδομένων profiling σε λογιστικά φύλλα (excel spreadsheets) ή απλά αρχεία κειμένου (txt)
- Μηχανισμοί παρουσίασης δέσμευσης μνήμης των διεργασιών του παιχνιδιού
- Διεργασίες που επιτρέπουν τη παρουσίαση πληροφοριών σε περίπτωση που το παιχνίδι τερματίσει την εκτέλεση του με απρόοπτο τρόπο
- Εργαλεία τα οποία εμφανίζουν διάφορα μηνύματα τύπου debug σε συγκεκριμένα τμήματα του κώδικα
- Μηχανισμοί καταγραφής των βημάτων εκτέλεσης μία δεδομένη στιγμή και τη δυνατότητα αναπαραγωγής τους

Ιδιαίτερη περίπτωση αποτελεί η μέθοδος με την οποία η κονσόλα Playstation 4 μπορεί να εξάγει πληροφορίες δεδομένων σε περίπτωση απρόοπτου τερματισμού του παιχνιδιού. Το Playstation 4 καταγράφει πάντοτε τα τελευταία 15 δευτερόλεπτα παιχνιδιού του παίκτη σε περίπτωση που αυτός θέλει να το διαμοιράσει στα κοινωνικά δίκτυα ή στους φίλους του. Έτσι οι προγραμματιστές του παιχνιδιού έχουν πρόσβαση όχι μόνο στη στοίβα εντολών που πραγματοποίησε το παιχνίδι πριν τερματίσει απρόοπτα αλλά και σε ένα βίντεο 15 δευτερολέπτων το οποίο τους δείχνει τι ακριβώς συνέβαινε με οπτικό τρόπο εκείνη τη στιγμή. Οι εξαγωγές αυτών των πληροφοριών μπορούν να ρυθμιστούν να στέλνονται αυτομάτως στους προγραμματιστές για την αντιμετώπιση των απρόοπτων τερματισμών.

Εργαλεία φυσικής και ανίχνευσης συγκρούσεων

Η ανίχνευση συγκρούσεων αποτελεί σημαντικό τμήμα στο περιβάλλον ενός παιχνιδιού. Χωρίς τη δυνατότητα εντοπισμών και κατάλληλου χειρισμού των διάφορων συγκρούσεων κατά την εκτέλεση ενός παιχνιδιού τότε το κάθε αντικείμενο θα διαπερνούσε το υπόλοιπα και δεν θα υπήρχε η δυνατότητα διάδρασης με το εικονικό περιβάλλον. Ανάλογα με το είδος του παιχνιδιού συναντάμε ρεαλιστικές ή ήμι ρεαλιστικές δυναμικές προσομοιώσεις τέτοιων συστημάτων. Το σύνολο αυτών των λειτουργιών ονομάζεται σύστημα φυσικής (physics system), αν και πολλές φορές χρησιμοποιείται ο όρος δυναμική άκαμπτων σωμάτων (rigid body dynamics) μίας και στα πλαίσια ενός παιχνιδιού μας απασχολεί κυρίως ο τρόπος κίνησης άκαμπτων σωμάτων καθώς και οι επίδραση που έχουν οι διάφορες δυνάμεις όπως η ορμή πάνω τους.

Ο έλεγχος συγκρούσεων και οι νόμοι της φυσικής ενός παιχνιδιού είναι συχνά το ένα και το αυτό. Αυτό συμβαίνει διότι στη περίπτωση την οποία ανιχνεύεται μία σύγκρουση μεταξύ δύο ή περισσότερων οντοτήτων χρησιμοποιούνται οι νόμοι της φυσικής για να επιλυθούν και να φανεί το αποτέλεσμα τους. Στις μέρες μας ελάχιστες εταιρίες επιλέγουν να αναπτύξουν τέτοια συστήματα και γι αυτό προβαίνουν στη χρήση των εργαλείων Havok ή PhysX που εξετάστηκαν σε προηγούμενες ενότητες.

Συστήματα παραγωγής κινήσεων οντοτήτων

Κάθε παιχνίδι το οποίο περιέχει οργανικές, ανθρωποειδή και ζώα, και ανόργανες οντότητες, βράχους και νερό, περιέχει και ένα σύστημα παραγωγής κινήσεων οντοτήτων (animation system). Ένα τέτοιο σύστημα περιλαμβάνει τα εξής τμήματα :

- Texture animation, τα οποία χειρίζονται τον τρόπο κινήσεων των διάφορων ταπετσαριών των μοντέλων του παιχνιδιού
- Animations άκαμπτων σωμάτων
- Skeletal animations, τα οποία χειρίζονται τον τρόπο κίνησης των άκρων των διάφορων οντοτήτων
- Vertex animations, τα οποία χειρίζονται τις κινήσεις μεμονωμένων σημείων των τρισδιάστατων μοντέλων

Τα τρισδιάστατα skeletal animations προσφέρουν τη δυνατότητα ρεαλιστικών κινήσεων των χαρακτήρων ενός παιχνιδιού με τη χρήση “αρθρώσεων” και “οστών” των οποίων οι κινήσεις που σχεδιάζονται μέσω ειδικών σχεδιαστικών προγραμμάτων μετατρέπονται σε κινήσεις των μοντέλων. Ουσιαστικά το σύστημα των skeletal animations γεφυρώνει το χάσμα μεταξύ του συστήματος απεικόνισης (renderer) και του γενικότερου συστήματος animations. Το animations system παράγει μία πόζα για κάθε άρθρωση του τρισδιάστατου σκελετού και έπειτα αυτές οι πόζες μεταφέρονται στο σύστημα απεικόνισης ως ομάδες πινάκων (matrices). Ο renderer με τη σειρά του μετατρέπει κάθε στοιχείο των πινάκων αυτών σε ολοκληρωμένες αλληλουχίες μεταβολών ενός τρισδιάστατου μοντέλου, πετυχαίνοντας έτσι την τελική απεικόνιση κινήσεων του.

Τέλος μία άλλη τεχνική απεικόνισης κινήσεων τρισδιάστατων μοντέλων είναι τα rag dolls, τα οποία αποτελούν το σημείο ένωσης των animation systems με τα physics systems. Ένα rag doll έχει συνήθως τη μορφή ενός πλαδαρού , και νεκρού, χαρακτήρα σε ένα εικονικό περιβάλλον του οποίου οι κινήσεις προσομοιώνονται εξολοκλήρου από το physics system. Το physics system καθορίζει τις

θέσεις και τις περιστροφές των διάφορων άκρων του τρισδιάστατου μοντέλου αντιμετωπίζοντας τα ως ένα περιορισμένο σύστημα συμπαγών σωμάτων.

Σύστημα διεπαφών ανθρώπου

Κάθε παιχνίδι χρειάζεται τις απαραίτητες λειτουργίες με τις οποίες θα επεξεργάζεται δεδομένα εισόδου τα οποία προέρχονται από το παίκτη, τα οποία ονομάζονται συστήματα διεπαφής ανθρώπων (human interface devices - HID) και περιλαμβάνουν τα εξής :

- Το πληκτρολόγιο και το ποντίκι
- Τα διάφορα χειριστήρια μοχλών (joypad)
- Άλλα χειριστήρια τα οποία έχουν τη μορφή τιμονιού ή πιστολιού με τη χρήση λέιζερ

Η μηχανή που χειρίζεται τα παραπάνω περιέχει δυνατότητες διαχωρισμού των πληροφοριών εισόδου υψηλού επιπέδου από αυτές του χαμηλού επιπέδου ανάλογα με το hardware που στοχεύουν οι σχεδιαστές του παιχνιδιού. Συχνά αυτά τα δεδομένα εισόδου διαχωρίζονται σε πληροφορίες σχετικά με τις νεκρές ζώνες γύρω από το κέντρο κάθε μοχλού σε ένα χειριστήριο, την ανίχνευση πατήματος ενός κουμπιού, την ανίχνευση αποδέσμευσης ενός κουμπιού και την ανίχνευση του ταυτόχρονου πατήματος μεγάλου αριθμού κουμπιών. Στα σύγχρονα χειριστήρια και συστήματα παρέχονται λειτουργίες ανίχνευσης δεδομένων αφής και αλλαγής ταχύτητας όπως επίσης και ήχου, εφόσον διαθέτουν μικρόφωνο και οθόνη αφής.

Σύστημα ήχου

Ο ήχος σε ιδανικές συνθήκες αποτελεί ίδιας σημασίας με αυτής των γραφικών κατά την ανάπτυξη μιας μηχανής παραγωγής βιντεοπαιχνιδιών. Δυστυχώς όμως ακόμα και σήμερα παραμένει ένα από τα πιο παραμελημένα τμήματα και συχνά λαμβάνει αρκετά λιγότερη σημασία από τα συστήματα απεικόνισης, φυσικής και κίνησης χαρακτήρων.

Τα συστήματα ήχου διαφέρουν αρκετά ως προς τη φιλοσοφία μέσω της οποίας διεκπεραιώνουν το έργο τους. Η μηχανή ήχου του παιχνιδιού Quake για παράδειγμα είναι αρκετά απλή και είναι στο χέρι των προγραμματιστών να προσθέσουν οποιαδήποτε επιπρόσθετη λειτουργία επιθυμούν. Από

την άλλη τα συστήματα ήχου που περιέχονται στη μηχανή Unreal Engine 4, παρέχουν δυνατότητες ήχου τριών διαστάσεων και άλλων λειτουργιών για ρεαλιστική παραγωγή ήχων σε κάθε εικονικό περιβάλλον. Παράδειγμα άλλων εταιριών αποτελούν τα εξής :

- Το σύστημα XAudio2 της Microsoft για χρήση στις κονσόλες της σειράς XBOX
- Το εξειδικευμένο σύστημα της εταιρείας Electronic Arts με όνομα SoundRiot το οποίο χρησιμοποιείται στα παιχνίδια της σειράς Battlefield
- Η μηχανή ήχου Scream η οποία έχει αναπτυχθεί από τη Sony και η χρήση της είναι εκτεταμένη σε τίτλους όπως τα Uncharted και The Last of Us.

Παρ' όλο το γεγονός ότι υπάρχουν διαθέσιμες αρκετές έτοιμες μηχανές ήχου προς χρήση για την ανάπτυξη μίας μηχανής παραγωγής παιχνιδιών, κάθε βιντεοπαιχνίδι απαιτεί ξεχωριστές ρυθμίσεις και αλλαγές.

Συστήματα online διασύνδεσης

Αρκετά παιχνίδια της αγοράς επιτρέπουν τη συμμετοχή πολλών παικτών σε τρόπους παιχνιδιού που ονομάζονται multiplayer. Οι κατηγορίες τέτοιων τρόπου παιχνιδιού που συναντάμε είναι οι εξής :

- Single – screen multiplayer: Δύο ή παραπάνω διεπαφές χρήστη (πληκτρολόγια, ποντίκια) συνδέονται σε ένα μοναδικό σύστημα το οποίο μπορεί να είναι ένας προσωπικός υπολογιστής ή μία κονσόλα. Πολλοί παίκτες συνδέονται την ίδια στιγμή σε έναν μοναδικό εικονικό σύμπαν και μία και μοναδική κάμερα παρακολουθεί τις κινήσεις τους και περιορίζει τη κίνηση τους εντός το προκαθορισμένου οπτικού πεδίου της. Ένα βιντεοπαιχνίδι το οποίο υποστηρίζει αυτόν τον τρόπο παιχνιδιού είναι το Gauntlet.
- Split – screen multiplayer: Πολλαπλοί παίκτες συνδέονται σε έναν μοναδικό ψηφιακό κόσμο κάνοντας χρήση πολλαπλών χειριστηρίων και συσκευών εισόδου. Ο κάθε παίκτης παρακολουθείται από τη δική του κάμερα με την οθόνη να είναι χωρισμένη σε τμήματα ανάλογα με τον αριθμό των παικτών τη δεδομένη στιγμή.
- Διαδικτυακό multiplayer: Πολλαπλοί υπολογιστές ή κονσόλες είναι συνδεδεμένοι μεταξύ τους με κάθε σύστημα να ανάλογη σε ένα παίκτη.
- Massively multiplayer: Υποστήριξη εκατοντάδων ή χιλιάδων χρηστών οι οποίοι μπορούν να παίζουν ταυτόχρονα την ίδια στιγμή σε ένα μόνιμο και μεγάλης κλίμακας εικονικό κόσμο.

Τα παιχνίδια τα οποία υποστηρίζουν τρόπους παιχνιδιού πολλών παικτών παρουσιάζουν αρκετά κοινά στοιχεία με αυτά που υποστηρίζουν μόνο έναν παίκτη. Παρ' όλα αυτά η απόφαση παροχής τέτοιων δυνατοτήτων παιχνιδιού κατά τα στάδια ανάπτυξης ενός βιντεοπαιχνιδιού έχει σημαντικές επιπτώσεις στον τρόπο σχεδίασης των επιμέρους τμημάτων της μηχανής. Το σύστημα παραγωγής του εικονικού περιβάλλοντος, το σύστημα απεικόνισης μοντέλων καθώς και η διαχείριση των δεδομένων εισόδου επηρεάζονται άμεσα. Συνήθως η προσθήκη στοιχείων multiplayer σε ένα παιχνίδι το οποίο δεν έχει σχεδιαστεί εξ αρχής να υποστηρίζει αυτές τις λειτουργίες είναι έργο δύσκολο και γι αυτό αποφεύγεται.

Από την άλλη η μετατροπή ενός multiplayer παιχνιδιού σε single player είναι έργο απλό και πολλά παιχνίδια αντιμετωπίζουν το single player τμήμα ενός βιντεοπαιχνιδιού και του τρισδιάστατου περιβάλλοντος του ως ένα multiplayer τρόπο παιχνιδιού αλλά που χρησιμοποιείται από έναν μόνο παίκτη.

Συστήματα κόσμων και οντοτήτων

Η εισαγωγή στρωμάτων σχετικά με τον τρόπο παιχνιδιού (gameplay) σε ένα βιντεοπαιχνίδι ορίζει έννοιες όπως κόσμος παιχνιδιού (game world) ο οποίος περιέχει στατικές ή δυναμικές οντότητες και στοιχεία. Τα περιεχόμενα ενός εικονικού τρισδιάστατου περιβάλλοντος σχεδιάζονται συχνά κάνοντας χρήση της λογικής των αντικειμένων του πραγματικού κόσμου γεγονός που κάνει εύκολη την επιλογή γλωσσών αντικειμενοστραφούς προγραμματισμού για την υλοποίηση αυτών. Το μοντέλο αντικείμενων (game object model) αποτελεί το κύριο μοντέλο το οποίο χρησιμοποιείται στη βιομηχανία ανάπτυξης βιντεοπαιχνιδιών ακόμα και σήμερα. Παραδείγματα game objects αποτελούν τα παρακάτω :

- Στατικές γεωμετρίες όπως κτίρια, δρόμοι και εδάφη
- Δυναμικά άκαμπτα σώματα όπως πέτρες και έπιπλα
- Χαρακτήρες οι οποίοι βρίσκονται υπό τον έλεγχο του χρήστη (Playable characters - PCs)
- Χαρακτήρες – πράκτορες οι οποίοι δεν ελέγχονται από τη χρήστη και έχουν υποστηρικτικό ρόλο (Non playable characters – NPCs)
- Αντικείμενα προς χρήση (Usable items) όπως όπλα
- Οχήματα
- Δυναμικά ή στατικά αντικείμενα τα οποία εκπέμπουν φως για τη παρουσίαση της σκηνής του κόσμου

- Κάμερες

Σύστημα συμβάντων

Οι διάφορες οντότητες οι οποίες βρίσκονται στο κόσμο του παιχνιδιού απαιτούν δυνατότητες επικοινωνίας με τα υπόλοιπα στοιχεία του χώρου. Το έργο αυτό μπορεί να επιτευχθεί με πολλούς τρόπους. Για παράδειγμα κάθε φορά που ένα αντικείμενο θέλει να στείλει μία πληροφορία – μήνυμα σε ένα άλλο μπορεί να καλεί μία συνάρτηση του αντικειμένου-παραλήπτη. Η αρχιτεκτονική σχεδίασης η οποία βασίζεται σε συμβάντα (event driven architecture) συναντάται κυρίως σε εφαρμογές γραφικού περιβάλλοντος. Σε ένα τέτοιο σύστημα ένας πομπός αποστέλλει μία μικρή δομή δεδομένων η οποία ονομάζεται συμβάν (event) ή μήνυμα (message) η οποία περιέχει πληροφορίες σχετικά με το τύπο του μηνύματος και κάθε άλλο δεδομένο που θέλει να λάβει ο παραλήπτης. Σε αυτό το σημείο μία διεργασία που ονομάζεται χειριστής συμβάντων (event handler) λαμβάνει το έργο ο σωστός παραλήπτης να λάβει το μήνυμα με ασφάλεια και χωρίς να αλλοιωθεί το περιεχόμενο του. Τα διάφορα συμβάντα κατά την εκτέλεση του παιχνιδιού μπορούν να αποθηκευθούν προσωρινά και σε μία δομή δεδομένων ουράς για χρήση σε μετέπειτα χρόνο.

Συστήματα scripting

Πολλές μηχανές παιχνιδιών χρησιμοποιούν μία γλώσσα scripting ώστε να κάνουν τη διαδικασία ανάπτυξης συγκεκριμένων κυρίως game play στοιχείων πιο εύκολη και γρήγορη. Χωρίς τη χρήση βοηθητικών λειτουργιών όπως η υποστήριξη μίας γλώσσας scripting θα πρέπει το εκτελέσιμο να περνάει από τα στάδια compiling και linking κάθε φορά που πραγματοποιείται μία αλλαγή. Με την χρήση μίας γλώσσας scripting αλλαγές στα συστήματα λογικής (logic) και δεδομένων (data) μπορούν να πραγματοποιηθούν με την επαναφόρτωση (reloading) του κώδικα του script. Κάποιες μηχανές παραγωγής παιχνιδιών επιτρέπουν τη πραγματοποίηση αλλαγών σε πραγματικό χρόνο καθώς το παιχνίδι βρίσκεται στο στάδιο της εκτέλεσης ενώ άλλες απαιτούν το τερματισμό τους και την επανεκκίνηση τους.

Συστήματα τεχνητής νοημοσύνης

Ο τομέας της τεχνητής νοημοσύνης συνδέεται αρκετά στενά με τον τομέα των βιντεοπαιχνιδιών. Παρ' όλα αυτά μόνο τα τελευταία χρόνια έχει ξεκινήσει να αποτελεί ενσωματωμένο τμήμα των μηχανών παραγωγής βιντεοπαιχνιδιών. Συνήθως αυτά τα τμήματα περιέχουν λειτουργίες χαμηλού επιπέδου διεργασιών τεχνητής νοημοσύνης που προσφέρουν δυνατότητες εύρεσης μονοπατιού (path finding), παραγωγή μοντέλων προσανατολισμού (navigation mesh generation), δυναμική αποφυγή αντικειμένων (dynamic object avoidance) και εντοπισμό ευπαθειών (vulnerability identification).

Υποσυστήματα παιχνιδιών κατά περίπτωση

Πάνω από το στρώμα των βασικών λειτουργιών gameplay μιας μηχανής και των υπόλοιπων χαμηλού επιπέδου συστημάτων οι προγραμματιστές και οι σχεδιαστές έχουν συνεργαστεί ώστε να δημιουργήσουν στρώματα που αφορούν ξεχωριστές περιπτώσεις βιντεοπαιχνιδιών. Συνήθως αυτά τα ανώτερα στρώματα περιλαμβάνουν δυνατότητες σχετικά με τους κανόνες κίνησης του κεντρικού χαρακτήρα, αριθμό ρυθμίσεων για το τρόπο συμπεριφοράς των διάφορων καμερών του εικονικού χώρου, συστήματα όπλων και εξοπλισμού. Αυτά τα στρώματα αποτελούν τη νοητή διαχωριστική γραμμή μεταξύ μιας μηχανής που κρύβεται πίσω από ένα παιχνίδι και του ίδιου το παιχνιδιού. Από τα παραπάνω καταλαβαίνουμε πως ενώ τα περισσότερα συστήματα είναι κοινά σε μεγάλο αριθμό βιντεοπαιχνιδιών, αυτό που τα κάνει να ξεχωρίζουν σε τελικό στάδιο είναι τα ανώτερα στρώματα με τα οποία προγραμματιστές και σχεδιαστές μπορούν να δημιουργήσουν την αρχική τους ιδέα και να της δώσουν την αίσθηση που είχαν στο μυαλό τους, παράγοντας έτσι καθηλωτικές διαδραστικές εμπειρίες.

ΚΕΦΑΛΑΙΟ Γ'

ΔΙΑΔΙΚΑΣΙΑ ΑΝΑΠΤΥΞΗΣ ΕΚΤΕΛΕΣΙΜΟΥ ΕΡΓΑΣΙΑΣ

Στα προηγούμενα κεφάλαια πραγματοποιήσαμε μία ιστορική αναδρομή στο χώρο των γραφικών υπολογιστών όπως επίσης και στη πορεία εξέλιξης των βιντεοπαιχνιδιών ανά τα χρόνια. Έπειτα εξετάσαμε τα κύρια τμήματα μίας μηχανής παραγωγής βιντεοπαιχνιδιών και τρισδιάστατων

ψηφιακών περιβαλλόντων και παρακολουθήσαμε τις κύριες σχεδιαστικές τεχνικές που χρησιμοποιούνται κατά την ανάπτυξη ενός τέτοιου έργου.

Στο κεφάλαιο αυτό θα εξετάσουμε τα βασικά εργαλεία που χρησιμοποιήθηκαν για το σχεδιασμό και την ανάπτυξη του εκτελέσιμου της εργασίας. Θα καλυφθούν τα γενικότερα κριτήρια επιλογής των τεχνολογιών που ακολουθούν καθώς και τα διάφορα στοιχεία των μέσων υλοποίησης αυτών ξεχωριστά, τα οποία συνέβαλαν στη δημιουργία ενός τρισδιάστατου παιχνιδιού πρώτου προσώπου το οποίο εκτελείται σε περιβάλλον browser κάνοντας χρήση της γλώσσας προγραμματισμού Javascript, των μέσων διεπαφής WebGL και BabylonJS και του προγράμματος τρισδιάστατης σχεδίασης Blender.

Γ.1 Η γλώσσα προγραμματισμού Javascript

Η Javascript δημιουργήθηκε το 1995 ως ένας τρόπος σχεδιασμού και ανάπτυξης διαδικτυακών σελίδων για τον φυλλομετρητή Netscape Navigator. Από τότε η γλώσσα υιοθετήθηκε από όλους τους διαθέσιμους φυλλομετρητές της αγοράς.

Γ1.1 Γενικές πληροφορίες

Με τη χρήση της επιτεύχθηκε η δημιουργία του σύγχρονου ιστού όπως τον γνωρίζουμε σήμερα, με τη δυνατότητα διαδικτυακών διαδραστικών εφαρμογών οι οποίες μπορούν να χρησιμοποιηθούν χωρίς να απαιτείται ανανέωση της σελίδας για κάθε πάτημα ενός κουμπιού. Από τότε η Javascript έχει εξελιχθεί αρκετά με αποτέλεσμα να τροφοδοτεί τη διαδικασία ανάπτυξης μεγάλης πληθώρας ιστοσελίδων, εφαρμογών διαδικτύου αλλά και εφαρμογών για περιβάλλοντα έξυπνων κινητών.

Σε αυτό το σημείο είναι σημαντικό να αναφερθεί το γεγονός πως η γλώσσα προγραμματισμού Javascript δεν έχει καμία σχέση με τη γλώσσα προγραμματισμού Java. Το παρόμοιο όνομα προέκυψε για λόγους marketing μιας και οι αρχικοί σχεδιαστές αναλογίστηκαν το γεγονός ότι λόγω της κυριαρχίας της γλώσσας προγραμματισμού Java στην αγορά θα ήταν καλή ιδέα να χρησιμοποιήσουν κάποια πλησιέστερη ονομασία με σκοπό την κέντρωση του ενδιαφέροντος του προγραμματιστικού κόσμου που θα οδηγήσει σε απόσπαση ενός μεριδίου της αγοράς χρηστών. Τελικά το όνομα καθιερώθηκε και επίσημα.

Έπειτα από την υιοθεσία της γλώσσας προγραμματισμού αυτής εκτός του φυλλομετρητή Netscape, ένα έγγραφο προδιαγραφών (standard document) συντάχθηκε για τη Javascript το οποίο περιέγραφε το τρόπο λειτουργίας των τμημάτων και των δυνατοτήτων της ώστε να είναι σίγουρο πως τα διάφορα κομμάτια λογισμικού που ανέφεραν πως κάνουν χρήση της νέας γλώσσας αυτής, αναφέρονταν στην ίδια γλώσσα. Το standard document αυτό ονομάστηκε ECMAScript και πήρε την ονομασία του από το Ecma International Organization το οποίο επιμελήθηκε της σύνθεσης του. Πρακτικά οι όροι ECMAScript και Javascript μπορούν να αντικαταστήσουν ο ένας τον άλλον μιας και απευθύνονται στο ίδιο πράγμα, με τον πρώτο να αποτελεί σε κάθε περίπτωση τον πιο επίσημο από τους δύο.

Συχνά η Javascript αποτελεί αντικείμενο φαντάζει ελλείψεις και ανεκτική σε περιπτώσεις λαθών τα οποία είναι δυνατό να οδηγήσουν σε προβλήματα λειτουργίας μίας εφαρμογής ή ιστοσελίδας. Παρ' όλα αυτά, το χαρακτηριστικό αυτό είναι που τη κάνει να αποτελεί μία από τις πιο προσιτές επιλογές γλώσσας για νέους προγραμματιστές ακόμα και αν δυσκολεύει το έργο της εύρεσης λαθών σε τμήματα κώδικα από προχωρημένους χρήστες.

Η ευελιξία που παρέχει η Javascript έχει και τα πλεονεκτήματα της όπως αναφέραμε παραπάνω. Ένα από αυτά είναι ότι αφήνει περισσότερο χώρο για την υλοποίηση τεχνικών που δε θα μπορούσαν να πραγματοποιηθούν υπό τους περιορισμούς-κανονισμούς άλλων γλωσσών προγραμματισμού.

Μέχρι τώρα έχουν υπάρξει αρκετές εκδόσεις της Javascript. Η τρίτη έκδοση του ECMAScript ήταν αυτή με τη μεγαλύτερη υποστήριξη από το κοινό των προγραμματιστών τη περίοδο 2000-2010. Από αυτό το σημείο και μετά, κατά τη διαδικασία ανάπτυξης της έκδοσης 4 πάρθηκαν πρωτοβουλίες για την προσθήκη αρκετών αλλαγών που μεταμόρφωναν ριζικά τη γλώσσα. Δυστυχώς αυτές οι αλλαγές δεν είχαν απήχηση διότι το κοινό που είχε υιοθετήσει τη Javascript ως κύρια γλώσσα είχε αυξηθεί ραγδαία κι πλέον ήταν λιγότερο ανεκτικό σε ριζικές αλλαγές. Για τους λόγους αυτούς η έκδοση 4, έτος 2008, ακυρώθηκε με τις λιγότερο ριζικές αλλαγές να εισχωρούν στην έκδοση 5 του 2009. Έπειτα από ένα μεγάλο διάστημα η έκδοση 6, έτος 2015, αναπτύχθηκε και περιείχε αρκετά στοιχεία που δεν υλοποιήθηκαν από την έκδοση 4. Από τότε υιοθετήθηκε ένας πιο σταθερός κύκλος αλλαγών ο οποίος λαμβάνει χώρα κάθε χρόνο. Με αυτόν τρόπο αποφεύγονται προβλήματα όπως αυτά που συναντήθηκαν κατά τη περίοδο σχεδιασμού της έκδοσης 4.

Το γεγονός ότι η Javascript με αναπτύσσεται με ραγδαίους ρυθμούς σημαίνει πως το ίδιο είναι αναγκασμένοι να κάνουν και οι φυλλομετρητές που είναι διαθέσιμη στη διαφορά ακολουθώντας γρήγορους κύκλους ανάπτυξης ώστε να είναι σε θέση να χρησιμοποιήσουν τις νέες λειτουργίες που παρέχονται στο νέο έγγραφο προδιαγραφών αλλά και να διορθώσουν τυχόν προηγούμενα bugs.

Τέλος οι φυλλομετρητές δεν αποτελούν τη μόνη πλατφόρμα που πραγματοποιείται χρήση της γλώσσας Javascript. Οι βάσεις δεδομένων MongoDB και CouchDB χρησιμοποιούν τη γλώσσα Javascript ως τη κύρια γλώσσα επερωτήσεων (query) και scripting. Πλέον στις μέρες έχουν αναπτυχθεί συστήματα όπως το NodeJS τα οποία παρέχουν τη δυνατότητα εκτέλεσης της γλώσσας Javascript εκτός του περιβάλλοντος ενός φυλλομετρητή.

Γ1.2 Γιατί επιλέχθηκε η γλώσσα Javascript

Η παραδοσιακή επιλογή ως προς τις γλώσσες προγραμματισμού για την ανάπτυξη ενός βιντεοπαιχνιδιού είναι η C++. Αυτό συμβαίνει διότι οι συγκεκριμένες δύο περιπτώσεις αποτελούν compiled και non garbage collected γλώσσες ενώ προσφέρουν δυνατότητες αντικειμενοστραφούς προγραμματισμού ο οποίος όπως είδαμε αποτελεί ένα από τα πιο εύχρηστα μοντέλα αναπαράστασης και διαχείρισης των οντοτήτων σε ένα τρισδιάστατο εικονικό περιβάλλον. Compiled διότι το κάθε τμήμα κώδικα πρέπει να μεταφραστεί σε γλώσσας μηχανής πριν να εκτελεσθεί σε ένα σύστημα και non garbage collected επειδή δεν διαθέτει αυτοματοποιημένα συστήματα διαχείρισης δεσμεύσεων πόρων μνήμης. Με αυτόν τον τρόπο οδηγούμαστε εύκολα στο συμπέρασμα ότι τα δύο αυτά χαρακτηριστικά προσθέτουν δυνατότητες γρήγορων επιδόσεων αλλά και δίνουν τον πλήρη έλεγχο ενός συστήματος στα χέρια ενός προγραμματιστή ώστε με χειροκίνητες μεθόδους να δεσμεύσει και να αποδεσμεύσει κάθε δεδομένη στιγμή τους πόρους που απαιτεί. Χαρακτηριστικά τα οποία αποτελούν σημαντικά κατά την ανάπτυξη εφαρμογών πραγματικού χρόνου, όπως είναι τα παιχνίδια και οι εφαρμογές εικονικών κόσμων, που απαιτούν σταθερές και ταχύτατες επιδόσεις οι οποίες δεν κάνουν κακή χρήση του περιορισμένου αριθμού πόρων ενός συστήματος προσωπικού υπολογιστή, μίας κονσόλας ή ενός έξυπνου κινητού.

Από την άλλη η Javascript αποτελεί μία interpreted και managed (garbage collected) γλώσσα, πράγμα που αντιτίθεται στα χαρακτηριστικά που αναφέρθηκαν παραπάνω. Interpreted σημαίνει πως η Javascript δεν απαιτεί το τμήμα της μετάφρασης(compilation) σε γλώσσας μηχανής ώστε να εκτελεστεί και σε αντίθεση ένας διερμηνευτής (interpreter) ο οποίος περιέχετε ενσωματωμένος στο περιβάλλον ενός φυλλομετρητή αναλαμβάνει να μετατρέψει κάθε γραμμή κώδικα σε γλώσσας μηχανής. Αυτό στους μοντέρνους φυλλομετρητές πραγματοποιείται με τη χρήση Just In Time (JIT) compilers ο οποίος μεταφράζει τμήματα της γλώσσας Javascript σε πραγματικό χρόνο. Επίσης από τη στιγμή που η γλώσσα διαθέτει garbage collector τότε η δέσμευση και η αποδέσμευση μνήμης αποτελεί αυτοματοποιημένο έργο.

Τα δύο χαρακτηριστικά αυτά σίγουρα δεν την κάνουν την πλέον ελκυστική γλώσσα ώστε να αναπτύξει κανείς μία απαιτητική εφαρμογή όπως ένα βιντεοπαιχνίδι. Παρ' όλα αυτά λόγω της ραγδαίας αύξησης των χρηστών της γλώσσας και της υιοθέτησης της σε περιβάλλοντα εκτός browser όπως επίσης και νέες τεχνολογίες όπως η WebGL που επιτρέπει τη χρήση των πόρων της κάρτας γραφικών ενός συστήματος από έναν φυλλομετρητή μείωσαν το γιγαντιαίο χάσμα απόδοσης ανάμεσα στη Javascript και τη C++ στον τομέα των τρισδιάστατων εφαρμογών πραγματικού χρόνου. Επιπρόσθετα λόγω του μεγάλου αριθμού χρηστών ολοένα και περισσότερες βοηθητικές βιβλιοθήκες και εργαλεία αναπτύσσονται κάθε μέρα τα οποία γλυτώνουν τον κάθε προγραμματιστή από το να “ανακαλύψει ξανά τον τροχό” με το να ξεκινήσει από το μηδέν υλοποιώντας τμήματα κώδικα τα οποία έχουν υλοποιηθεί πολλάκις στο παρελθόν από τρίτους.

Ακόμη η διαδικασία ανάπτυξης ενός βιντεοπαιχνιδιού με τις συμβατικές μεθόδους απαιτεί την ανάπτυξη ενός μεγάλου τμήματος προπαρασκευαστικού κώδικα ο οποίος έχει στόχο τη δημιουργία ενός γραφικού περιβάλλοντος. Σε αυτό το σημείο η Javascript επιτρέπει σε κάποιον να ξεκινήσει κατευθείαν να υλοποιεί τις ιδέες του μιας και όχι μόνο βλέπει τις αλλαγές του σε πραγματικό χρόνο, μιας και δεν υπάρχει το στάδιο μετάφρασης (compilation), αλλά έχει ταχύτατα οπτικά αποτελέσματα επειδή το γραφικό περιβάλλον/παράθυρο της εκάστοτε εφαρμογής αποτελεί ο ίδιος ο canvas σχεδίασης που έχει ο κάθε φυλλομετρητής ενσωματωμένο.

Τέλος το ίδιο το περιβάλλον του φυλλομετρητή, Google Chrome στη παρούσα εργασία, παρέχει out of the box πληθώρα εργαλείων debugging και profiling τα οποία όπως είδαμε σε προηγούμενες ενότητες αποτελούν τεράστιας σημασίας εργαλεία κατά την ανάπτυξη διαδραστικών εφαρμογών εικονικών κόσμων. Επίσης σήμερα οι φυλλομετρητές καλύπτουν όλο το φάσμα ηλεκτρικών συσκευών ψυχαγωγίας οπότε καταφέρνουμε με αυτόν τον τρόπο να καλύπτουμε μεγαλύτερο μερίδιο της αγοράς μιας και έχουμε πλέον τη δυνατότητα να υλοποιήσουμε μία εφαρμογή μία φορά και να έχουμε τη δυνατότητα να την εκτελέσουμε σε πληθώρα πλατφορμών χωρίς να χρειάζεται να κάνουμε αλλαγές.

Όλα τα παραπάνω κάνουν την Javascript να αποτελεί ένα εύχρηστο και ευπρόσιτο περιβάλλον για έναν αρχάριο προγραμματιστή που θέλει να ασχοληθεί με τη διαδικασία σχεδιασμού και ανάπτυξης τρισδιάστατων περιβαλλόντων ακόμα και αν θυσιάζονται, ελάχιστα στις μέρες μας πλέον, μέτρα απόδοσης κατά την διαδικασία εκτέλεσης του τελικού προϊόντος.

Γ.2 Διεπαφή προγραμματισμού εφαρμογών WebGL

Η WebGL αποτελεί μία διεπαφή προγραμματισμού εφαρμογών (API) το οποίο κάνει χρήση της γλώσσας Javascript για τη παραγωγή δισδιάστατων και τρισδιάστατων γραφικών σε περιβάλλοντα φυλλομετρητών. Το μέσο διεπαφής αυτό είναι πλήρως ενσωματωμένο με τις υπόλοιπες προδιαγραφές διαδικτύου και επιτρέπει τη χρήση των πόρων της κάρτας γραφικών ενός συστήματος για την προσομοίωση των physics και την αναπαράσταση οπτικών εφέ στον καμβά (canvas) μίας ιστοσελίδας. Με αυτό το τρόπο πετυχαίνουμε τη δημιουργία GPU accelerated διαδικτυακών εφαρμογών τα οποία μπορούν να αναμιχθούν με στοιχεία HTML σε έναν ιστότοπο. Επιπρόσθετα οι κύριες λειτουργίες που προσφέρει η WebGL καλούνται μέσω κώδικα γραμμένο στη γλώσσα Javascript και τα προγράμματα σκίασης (shaders) χρησιμοποιούν τη διάλεκτο ES Shading Language της OpenGL που ο λόγος δημιουργίας ήταν η υποστήριξη GPU accelerated λογισμικού σε συσκευές όπως έξυπνα κινητά και τάμπλετ. Η WebGL σχεδιάστηκε και υποστηρίζεται από την μην κερδοσκοπικού χαρακτήρα ομάδα, Khronos Group.

Πιο συγκεκριμένα η διεπαφή προγραμματισμού εφαρμογών αυτή αποτελεί μία χαμηλού επιπέδου λύση για τη παραγωγή γραφικών υπολογιστών και εικονικών περιβαλλόντων ενσωματωμένα σε browsers. Αυτό σημαίνει πως οι εντολές που παρέχει αντιστοιχούν με ακρίβεια στο τρόπο λειτουργίας μίας κάρτας γραφικών. Από τα παραπάνω καταλαβαίνει κανείς ότι μπορεί κανείς να πετύχει εγχειρήματα που παλαιότερα ήταν μόνο εφικτά με μέσα διεπαφής όπως το OpenGL και Direct3D που αναφέρθηκαν σε προηγούμενα κεφάλαια.

Η WebGL αποτελεί τόσο χαμηλού επιπέδου διεπαφή που ο χαρακτηριστικός “διεπαφή τρισδιάστατων γραφικών” δεν θα ήταν πλήρως σωστός μίας και το υλικό (hardware) παραγωγής γραφικών δεν αναγνωρίζει τη διαφορά μεταξύ των δύο και των τριών διαστάσεων. Πιο συγκεκριμένα μετά την έκδοση της πρώτης έκδοσης της OpenGL η οποία είχε ξεκάθαρα τον σκοπό να αναδείξει τις δυνατότητες παραγωγής τρισδιάστατων γραφικών υπολογιστών του νέου υλικού που ήταν διαθέσιμο στην αγορά, οι επόμενες εκδόσεις σχεδιάστηκαν ώστε να είναι πιο γενικού σκοπού. Αυτό είχε ως αποτέλεσμα η απεικόνιση δισδιάστατων και τρισδιάστατων γραφικών να αποτελούν απλά δύο πιθανές περιπτώσεις λύσης του ίδιου προβλήματος, της απεικόνισης στοιχείων με τη χρήση μίας κάρτας γραφικών και όχι του επεξεργαστή. Η WebGL ουσιαστικά αποτελεί τη μεταφορά των επόμενων εκδόσεων της OpenGL, OpenGL 2.0, στο χώρο των τεχνολογιών διαδικτύου.

Γ2.1 Γενικά χαρακτηριστικά

Κάθε ιστότοπος διαθέτει ένα στοιχείο καμβά (canvas element) το οποίο χρησιμοποιείται για το σχεδιασμό δισδιάστατων, αρχικά, γραφικών με τη γλώσσα Javascript και η χρήση του έγινε

διαθέσιμη με την έκδοση της HTML 5. Η WebGL πρόσθεσε τη δυνατότητα σχεδίασης και τρισδιάστατων στο canvas μιας ιστοσελίδας γεγονός που για να γίνει πράξη απαιτεί τη δέσμευση του από τις λειτουργίες της. Αυτό πραγματοποιείται με το ακόλουθο τμήμα κώδικα :

Από αυτό το σημείο και μετά το έργο της απόδοσης γραφικών (rendering) πραγματοποιείται καλώντας τις διαθέσιμες συναρτήσεις (functions) του μέσου διεπαφής WebGL στο στοιχείο gl το οποίο δεσμεύσαμε και περιέχει τον καμβά μίας ιστοσελίδας. Η WebGL δεν παρέχει ποτέ single-buffered τρόπους απεικόνισης γραφικών, γεγονός που δείχνει ότι κάθε εικόνα που δημιουργείται για να αποδοθεί στην οθόνη, δεν είναι ποτέ η ίδια που απεικονίζεται στο στοιχείο καμβά. Η λειτουργία αυτή έχει σκοπό την αποφυγή απόδοσης στον καμβά των καρτέ που δεν έχουν περάσει πλήρως από τη διαδικασία rendering. Η εικόνα η οποία γίνεται rendering κάθε στιγμή αποτελεί το WebGL framebuffer ή backbuffer. Από την άλλη η εικόνα η οποία τελικά απεικονίζεται στον καμβά ονομάζεται front buffer. Αυτή η διαδικασία γίνεται αυτοματοποιημένα από τον εκάστοτε web browser μιας και ο προγραμματιστής δεν έχει πρόσβαση σε αυτά τα τμήματα λειτουργιών. Αυτό σημαίνει πως ο browser έχει τη δυνατότητα να αντιγράψει τον backbuffer στον frontbuffer οποιαδήποτε στιγμή εφόσον δεν εκτελείται κάποιο τμήμα κώδικα Javascript ταυτόχρονα. Με αυτόν τον τρόπο καταλαβαίνουμε πως ένα ολόκληρο καρέ (frame) θα πρέπει να ολοκληρωθεί με το κάλεσμα μίας συνάρτησης callback μέσω Javascript.

Η WebGL επιτρέπει τη σχεδίαση σημείων, γραμμών ή τριγώνων. Τα τρίγωνα αποτελούν τη συνηθέστερη επιλογή των οποίων η σκίαση και η μορφή πραγματοποιείται με τη χρήση δύο μικρών τμημάτων κώδικα, ή υποπρογραμμάτων. Αυτά είναι ο pixel shader και ο fragment shader τα οποία θα χρησιμοποιηθούν για κάθε ένα pixel του τριγώνου και θα προκύψει έτσι ο χρωματισμός αυτών. Για τη παραγωγή άλλων σχημάτων όπως τετράγωνα, απλά χρησιμοποιούμε δύο ενωμένα τρίγωνα. Σε ένα τρισδιάστατο περιβάλλον πραγματοποιείται η ίδια εργασία με τη τμηματοποίηση (tessalation) κάθε πολύπλοκου σχήματος τριών διαστάσεων σε έναν μεγάλο αριθμό τριγώνων. Αυτό συμβαίνει διότι κάθε ομάδα τριών σημείων σε ένα τρισδιάστατο χώρο ορίζει τις κορυφές ενός τριγώνου. Δε θα μπορούσαν να χρησιμοποιηθούν παραπάνω σημεία, για παράδειγμα τέσσερα για τον ορισμό ενός παραλληλογράμμου, διότι θα ήταν πολύ δύσκολο αυτά να τηρούν τη προϋπόθεση πως βρίσκονται στο ίδιο επίπεδο.

```
var gl;
try {
  gl = canvas.getContext("experimental-webgl");
} catch(e) {}
```

Κώδικας 1: Δέσμευση στοιχείου καμβά ιστοσελίδας

Οι συνταγμένες που διαθέτει ο φαινομενικά δισδιάστατος καμβάς για τη σχεδίαση τριγώνων συνεχίζουν να βασίζονται σε ένα σύστημα συντεταγμένων τριών διαστάσεων αλλά με τη διαφορά πως οι τιμές του άξονα z, ο οποίος υποδηλώνει το βάθος του χώρου και κυμαίνεται από το -1.0 έως το 1.0 όπως και οι υπόλοιποι άξονες, έχει μηδενικές τιμές. Με αυτόν τρόπο αποδίδουμε στην οθόνη γραφικά δύο διαστάσεων ενώ αντιμετωπίζουμε ένα χώρο τριών.

Γ2.2 Διασωλήνωση γραφικών στη WebGL

Η διαδικασία διασωλήνωσης παραγωγής γραφικών είναι ένα στοιχείο που όπως είδαμε αποτελεί το πυρήνα των τεχνικών απόδοσης γραφικών σε μία μηχανή γραφικών για την απεικόνιση ενός ολοκληρωμένου περιβάλλοντος.

Οι κάρτες γραφικών ως παράλληλοι επεξεργαστές δεδομένων αποτελούνται από ένα σύνολο υπολογιστικών μονάδων οι οποίες είναι σχεδιασμένες και κατασκευασμένες έτσι ώστε να λειτουργούν παράλληλα η μία με την άλλη αλλά και με τον επεξεργαστή. Αυτό συμβαίνει ακόμα και στη περίπτωση του hardware των κινητών συσκευών. Οι διεπαφές προγραμματισμού εφαρμογών όπως η WebGL έχουν σχεδιαστή έχοντας λάβει υπόψη αυτές τις ιδιαιτερότητες στην αρχιτεκτονική υλικού. Η WebGL επιτρέπει τη πλήρη δέσμευση των πόρων μίας κάρτας γραφικών και την αδιάκοπη λειτουργία της την οποία διατηρεί ξεχωριστεί από αυτή του επεξεργαστή. Έτσι το ένα τμήμα δε χρειάζεται να είναι σε αναμονή λόγω της λειτουργίας του άλλου και γι αυτό τα προγράμματα σκίασης, vertex και fragment shader, είναι γραμμένα σε μία πιο φιλική προς το υλικό γλώσσα η οποία επιτρέπει την εκτέλεση αυτών στη κάρτα γραφικών.

Όπως φαίνεται στο παραπάνω διάγραμμα η διαδικασία διασωλήνωσης ξεκινάει με τη δημιουργία των vertex και fragment shader. Έπειτα η WebGL αποκτάει τους απαραίτητους πόρους – δεδομένα τα οποία και θα τροφοδοτήσει στα προγράμματα σκίασης. Τα δεδομένα αυτά είναι της μορφής bitmap data (ή αλλιώς textures) τα οποία θα χρησιμοποιηθούν στον fragment shader και vertex data τα οποία θα χρησιμοποιηθούν στον vertex shader και περιγράφουν τα τρίγωνα που πρέπει να σχεδιαστούν. Μόλις αυτή η διαδικασία ολοκληρωθεί, ακολουθεί η εκτέλεση του κώδικα που περιέχει ο vertex shader. Έτσι καθορίζονται οι συντεταγμένες των τριγώνων πάνω στον καμβά σχεδίασης και έπειτα αυτά περνάνε από τη διαδικασία rasterization. Αυτό σημαίνει πως καθορίζεται η λίστα των εικονοστοιχείων (pixels) που θα πρέπει να χρωματιστούν. Έπειτα ξεκινάει η εκτέλεση του κώδικα του fragment shader για κάθε pixel ξεχωριστά, καθορίζοντας έτσι το κατάλληλο χρώμα που θα χρησιμοποιηθεί. Τέλος κάποιος framebuffer καθορίζει πως το

υπολογισμένο χρώμα του προηγούμενου σταδίου θα επηρεάσει τον τελικό framebuffer και το χρώμα του pixel στο ίδιο σημείο. Σε αυτό το τελικό στάδιο θα υλοποιηθούν τα υπόλοιπα οπτικά εφέ όπως ορατότητα και έλεγχος βάθους.

Γ2.3 Μνήμη και WebGL

Κάποιες κάρτες γραφικών ιδιαίτερα σε σταθερούς υπολογιστές διαθέτουν την δικιά τους μνήμη ενώ άλλες μοιράζονται τη μνήμη ολόκληρου του συστήματος. Ως προγραμματιστές WebGL δεν έχουμε τη δυνατότητα να γνωρίζουμε πάντα σε τι σύστημα θα εκτελεστεί το λογισμικό μας έτσι ως μέτρο ασφαλείας η WebGL και οι λειτουργίες τις περιορίζονται στο τρόπο σχεδιασμού κατάλληλο για περιπτώσεις μεμονωμένης μνήμης GPU. Πρακτικά ο τρόπος που αντιμετωπίζει τη μνήμη η WebGL είναι ο εξής :

- Τα δεδομένα απόδοσης (rendering data) φορτώνονται σε μία ειδική δομή δεδομένων που παρέχει η WebGL. Πιο συγκεκριμένα πραγματοποιείται μία αντιγραφή από μία μνήμη γενικού τύπου στη μνήμη ειδικού WebGL τύπου. Αυτές οι ειδικές δομές δεδομένων ονομάζονται WebGL Textures και WebGL Buffers.
- Η διαδικασία αυτή απαιτεί αρκετό χρόνο και έτσι χαρακτηρίζεται ως αργή.
- Μόλις όμως τελειώσει το προηγούμενο αυτό στάδιο η απεικόνιση των δεδομένων με τη μορφή γραφικών είναι ταχύτατη διαδικασία.

Συνοπτικά η WebGL αντιμετωπίζει τη κάρτα γραφικών ενός συστήματος ως ένα τμήμα αργό προς την διαδικασία επικοινωνίας αλλά ταχύτατο κατά τη διαδικασία επεξεργασίας δεδομένων. Ο περιορισμός αυτός ως προς τον τρόπο που αντιμετωπίζει το hardware η WebGL εξασφαλίζει την ομαλή εκτέλεση της διαδικασίας απόδοσης γραφικών σε κάθε περίπτωση.

Γ3 Το πλαίσιο BabylonJS

Η διαδικασία ανάπτυξης ενός ολοκληρωμένου τρισδιάστατου περιβάλλοντος όπως είναι η δημιουργία ενός παιχνιδιού πρώτου προσώπου αποτελεί εγχείρημα ιδιαίτερα απαιτητικό. Η WebGL παρέχει μόνο βοηθητικές μεθόδους για την απλή σχεδίαση και απεικόνιση σχημάτων και οπτικών εφέ γεγονός που κάνει το παραπάνω έργο ακόμα πιο δύσκολο. Για το λόγο αυτό στα πλαίσια της εργασίας χρησιμοποιήθηκε το πλαίσιο (framework) BabylonJS το οποίο σχεδιάστηκε, αναπτύχθηκε και υποστηρίζεται από την εταιρεία Microsoft. Το συγκεκριμένο framework παρέχει βοηθητικές

μεθόδους (functions) οι οποίες αποτελούν μία υψηλότερου επιπέδου μορφή των βασικών μεθόδων που παρέχει η WebGL, με έμφαση και τη παροχή διευκολύνσεων για τη δημιουργία διαδραστικών εφαρμογών διαδικτύου. Μέθοδοι δημιουργίας κάμερας, κίνησης στο χώρο, βιβλιοθήκες μαθηματικών καθώς και υποστηρικτικές λειτουργίες σχετικά με τη δυνατότητα ανίχνευσης συγκρούσεων και φόρτωσής τρισδιάστατων μοντέλων για χρήση σε ένα εικονικό περιβάλλον είναι μονάχα μερικές από τις υπηρεσίες που παρέχει.

Γ3.1 Δυνατότητες

Οι σχεδιαστές του framework έχουν προνοήσει και έχουν υλοποιήσει αρκετές βοηθητικές μεθόδους και λειτουργίες που κάνουν το BabylonJS να αποτελεί ιδανική επιλογή για τον προγραμματισμό διαδραστικών τρισδιάστατων εφαρμογών. Μεταξύ άλλων τα κύρια χαρακτηριστικά είναι τα εξής :

- Ολοκληρωμένος γράφος τρισδιάστατης σκηνής
- Ανίχνευση συγκρούσεων
- Διασύνδεση με συστήματα physics
- Σύστημα παραγωγής σωματιδίων
- Animation System

Ο ολοκληρωμένος τρισδιάστατος γράφος σκηνής αποτελεί μία δομή δεδομένων η οποία έχει τη δυνατότητα να αποθηκεύει πληροφορίες σχετικά με τις οντότητες ενός τρισδιάστατου περιβάλλοντος. Επιπρόσθετα αποθηκεύει και της σχέσης που μπορεί να έχουν τα αντικείμενα μεταξύ τους κι έτσι μπορεί κανείς να ορίσει σχέσης πατέρα – τέκνου με αποτέλεσμα να δημιουργείται ένας μεγάλος γράφος συσχετίσεων μεταξύ των διαφόρων οντοτήτων. Πιο συγκεκριμένα κάθε σύστημα παραγωγής διαδραστικού περιεχομένου τρισδιάστατων περιβαλλόντων περιέχει λειτουργίες γράφου σκηνής μιας και σε αυτό αποθηκεύονται οι πληροφορίες των οντοτήτων οι οποίες θα απεικονιστούν στην οθόνη από τη μηχανή απόδοσης. Αυτό επιτυγχάνεται με τη διάσχιση των κόμβων και των παιδιών τους με τους περισσότερους γράφους σκηνής να έχουν ιεραρχική δομή όπως αυτό το οποίο θα μπορούσε να χρησιμοποιηθεί για τη προσομοίωση ενός ηλιακού συστήματος. Στο χώρο δημιουργείται μία κεντρική οντότητα με τις υπόλοιπες οντότητες να αποτελούν παιδιά αυτής και άλλες οντότητες να αποτελούν τους δορυφόρους τους. Χαρακτηριστικό παράδειγμα αποτελεί η σχέση ήλιου (parent), γης (child ήλιου) και φεγγαριού (child γης). Με αυτόν τρόπο δημιουργίας σχέσεων μεταξύ των οντοτήτων ενός τρισδιάστατου περιβάλλοντος μέσω του γράφου σκηνής μπορούμε να υλοποιήσουμε κινήσεις οι οποίες πραγματοποιούνται σε σχέση ενός άλλο αντικείμενο. Παράδειγμα στη συγκεκριμένη περίπτωση αποτελεί η κίνησης της γης γύρω από τον ήλιο.

Το ενσωματωμένο σύστημα ανίχνευσης συγκρούσεων που παρέχει βοηθάει στην δημιουργία διαδραστικών συμβάντων μεταξύ τρισδιάστατων οντοτήτων. Επιπλέον η εύκολη και γρήγορη διασύνδεση με τις βιβλιοθήκες `cannon.js` και `ammo.js` δίνει τη δυνατότητα χρήσης ενός ολοκληρωμένου συστήματος `physics` για ρεαλιστικές αναπαραστάσεις ιδιοτήτων σωμάτων όπως εκτόξευση πυρών από ένα όπλο.

Οι ενσωματωμένες λειτουργίες παραγωγής σωματιδίων που παρέχει το `BabylonJS` προσφέρουν πληθώρα δυνατοτήτων. Με τη χρήση μίας μη ορατής στη κάμερα του τρισδιάστατου περιβάλλοντος οντότητας, η οποία λειτουργεί ως πομπός (`emitter`) και εκπέμπει μικρά σωματίδια μπορούμε να δημιουργία οπτικών εφέ όπως αυτό της φωτιάς κατά το στάδιο απόδοσης των γραφικών στην οθόνη. Ο πομπός αυτός παρέχει δυνατότητες ρυθμίσεων σχετικά με το μέγεθος των σωματιδίων και το ρυθμό παραγωγής τους, δίνοντας έτσι τον πλήρη έλεγχο στο προγραμματιστή ώστε να υλοποιήσω το καλύτερο δυνατό οπτικό αποτέλεσμα.

Ένας τρόπος υλοποίησης `animation` σε ένα τρισδιάστατο περιβάλλον είναι η πραγματοποίηση αλλαγών στις ιδιότητες μίας οντότητας μεταξύ κάθε προπαρασκευαστικού σταδίου απόδοσης γραφικών. Σταδιακές προσθέσεις ή μειώσεις μονάδων στις συντεταγμένες, του μεγέθους και της περιστροφής εντός αντικειμένου θα έχει ως αποτέλεσμα τη σύνθεση κινήσεων. Παρ' όλα αυτά το `BabylonJS` παρέχει τη δυνατότητα χρήσης χρονομετρημένων καρέ δεδομένων που με την είσοδο δεδομένων υπολογίζεται ο σχεδιασμός του σωστού καρέ ανεξαρτήτως του ρυθμού ανανέωσης της οθόνης. Πιο συγκεκριμένα ένα `animation` στο `BabylonJS` αναπαρίστανται με τη χρήση ενός αντικειμένου στη γλώσσα `JavaScript` και όπως ένα αντικείμενο ταπετσαρίας μπορεί να προστεθεί σε ένα τρισδιάστατο μοντέλο έτσι μπορεί να πραγματοποιηθεί και με ένα `animation object`. Στη πραγματικότητα το αντικείμενο αυτό περιγράφει μία αλλαγή στις ιδιότητες ενός τρισδιάστατου μοντέλου και αποτελείται από τα εξής στοιχεία :

- **Performer**: το οποίο δέχεται ως είσοδο ένα αντικείμενο το οποίο μπορεί να γίνει `animated` όπως ένα τρισδιάστατο μοντέλο, ένα φως ή μία κάμερα
- **Frame**: ένα καρέ της διαδικασίας `animation`
- **Animation**, το οποίο αποτελείται από :
 - Η ιδιότητα που θα μεταβληθεί
 - Ο ρυθμός αλλαγής της ιδιότητας
 - Το είδος της ιδιότητας που θα μεταβληθεί και παίρνει τιμές όπως `float` και `vec3`
 - Δείκτη επανάληψης στο τέλος του
 - Οι τιμές της ιδιότητας που μεταβάλλεται σε συγκεκριμένες χρονικές στιγμές

- Scripted performer: Το μοντέλο και το σύνολο των προσκολλημένων animation του
- Performance: Η επόμενη πράξη μετά το τέλος αναπαραγωγής του animation
- Clip, το είδος της εκτέλεσης:
 - Movie Clip: Ο παίκτης χάνει τον έλεγχο της κάμερας κατά την αναπαραγωγή και λειτουργεί ως απλός θεατής των αποτελεσμάτων
 - Game Clip: Ο παίκτης έχει τον έλεγχο της κάμερας κατά την αναπαραγωγή επηρεάζοντας την εκτέλεση του animation
- Cartoon: Μία σειρά από clips που αναπαράγονται σε συγκεκριμένα χρονικά τμήματα

Γ3.2 Παραδείγματα τμημάτων κώδικα

Σε αυτή την ενότητα θα εξετάσουμε κάποια παραδείγματα κώδικα που παρουσιάζουν το τρόπο χρήσης του BabylonJS μέσα από τη γλώσσα Javascript.

Αρχικά παρατηρούμε το παράδειγμα της δημιουργίας μιας απλής σκηνής με φωτισμό και ένα αντικείμενο παραλληλογράμμου. Σημαντική αποτελεί η διασύνδεση του καμβά του κώδικα html με την κάμερας της σκηνής μας. Έπειτα η δημιουργία στοιχείων φωτισμού και η τοποθέτηση αυτών στο χώρο μας επιτρέπει να έχουμε πλήρη ορατότητα αυτού. Τέλος η δημιουργία του σχήματος του παραλληλογράμμου γίνεται με τη συνάρτηση CreateBox η οποία περιλαμβάνεται στη κλάση MeshBuilder η οποία προσφέρει γενικές μεθόδους κατασκευής γεωμετρικών σχημάτων. Ο καθορισμός των συντεταγμένων τοποθέτησης γίνεται με τη χρήση τριών διαστάσεων διανυσμάτων για κάθε στοιχείο που εισέρχεται στο χώρο.

```
var createScene = function () {
    // Δημιουργία της σκηνής
    var scene = new BABYLON.Scene(engine);
    // Προσθήκη κάμερας στη σκηνή και σύνδεση αυτής με το αντικείμενο καμβά
    var camera = new BABYLON.ArcRotateCamera("Camera", 3 * Math.PI / 4, Math.PI
        / 4, 4, BABYLON.Vector3.Zero(), scene);
    camera.attachControl(canvas, true);
    // Προσθήκη αντικειμένων φωτισμού
    var light1 = new BABYLON.HemisphericLight("light1", new BABYLON.Vector3
        (1, 1, 0), scene);
    var light2 = new BABYLON.PointLight("light2", new BABYLON.Vector3(0, 1, -
        1), scene);
    // Προσθήκη παραλληλογράμμου στο περιβάλλον
    var box = BABYLON.MeshBuilder.CreateBox("box", {height: 1, width: 0.75, dept
    h: 0.25}, scene);
    return scene;
};
```

Κώδικας 2: Παρουσίαση δημιουργίας σκηνής με ένα παραλληλόγραμμο

Το παράδειγμα που ακολουθεί περιέχει τα στάδια του προηγούμενου, Κώδικας 2, για τη δημιουργία κόσμου με τη διαφορά όμως πως γίνεται χρήση της διαδικασίας Import για τη φόρτωση ενός μοντέλου που έχει δημιουργηθεί με ειδικό εξωτερικό πρόγραμμα σχεδίασης τρισδιάστατων μοντέλων. Η συνάρτηση αυτή φορτώνει απευθείας το αρχείο του μοντέλου από το σκληρό μας δίσκο και δημιουργεί ένα αντικείμενο Javascript που αντιστοιχεί σε αυτό. Επιπρόσθετα στη διαδικασία registerBeforeRender μεταβάλλουμε τη θέση του φωτισμού ώστε να ακολουθεί αυτή της κάμερας. Οτιδήποτε βρίσκεται στη διαδικασία αυτή πραγματοποιείται μεταξύ κάθε καρέ ανανέωσης και γι αυτό χρησιμοποιείται για την υλοποίηση αλλαγών όπως είναι η κίνηση οντοτήτων. Σημειώνεται πως η χρήση του γράφου σκηνής και η αξιοποίηση των δυνατοτήτων που προσφέρει σχετικά με την ανάπτυξη σχέσεων μεταξύ του κόμβου φαίνεται από το γεγονός ότι έχουμε ένα αντικείμενο σκηνής στο οποίο προσθέτουμε οντότητες κι έτσι μπορούμε να έχουμε λειτουργίες όπως, light.position = camera.position για την εύκολη πραγματοποίηση αλλαγών σχετικών με άλλες οντότητες του περιβάλλοντος μας.

```
var createScene = function () {  
  
    var scene = new BABYLON.Scene(engine);  
  
    //Προσθήκη φωτισμού  
    var light = new BABYLON.PointLight("Omni", new BABYLON.Vector3(20, 20, 100), scene);  
  
    //Δημιουργία περιστροφικής κάμερας  
    var camera = new BABYLON.ArcRotateCamera("Camera", 0, 0.8, 100, BABYLON.Vector3.Zero  
(), scene);  
    camera.attachControl(canvas, false);  
  
    // Φόρτωση μοντέλου  
    BABYLON.SceneLoader.ImportMesh("", "scenes/", "skull.babylon", scene, function (newM  
eshes) {  
        // Η κάμερα βλέπει στη τοποθεσία του μοντέλου  
        camera.target = newMeshes[0];  
    });  
  
    // Η θέση του φωτός είναι αυτή της κάμερας  
    scene.registerBeforeRender(function () {  
        light.position = camera.position;  
    });  
  
    return scene;  
}
```

Κώδικας 3: Φόρτωση τρισδιάστατου μοντέλου που σχεδιάστηκε με εξωτερικό πρόγραμμα

Στο τμήμα κώδικα που ακολουθεί παρατηρείται μία από τις πιο σημαντικές δυνατότητες στην ανάπτυξη διαδραστικών εφαρμογών και ιδιαίτερα στον σχεδιασμό βιντεοπαιχνιδιών πρώτου προσώπου όπως θα παρουσιαστεί στα επόμενα κεφάλαια. Η δυνατότητα της επιλογής με τη χρήση του κλικ του ποντικιού ενός σημείου στο τρισδιάστατο χώρο. Το παράδειγμα κώδικα αυτό παράγει ένα τοίχο στον οποίο έχουμε τη δυνατότητα να πραγματοποιήσουμε βολές οι οποίες αφήνουν στάμπα, επιλέγοντας το σημείο ρίψης με το ποντίκι. Η δημιουργία της στάμπας, αποτέλεσμα βολής, γίνεται με τον συνδυασμό εικόνων από τον σκληρό δίσκο και η ανίχνευση του κλικ ποντικιού γίνεται με την συνάρτηση ανίχνευσης συμβάντων που διαθέτει το BabylonJS.

```
var createScene = function () {  
    var scene = new BABYLON.Scene(engine);  
  
    // Ρύθμιση περιβάλλοντος  
    var light0 = new BABYLON.PointLight("Omni", new BABYLON.Vector3(0, 10, 20), scene);  
    var freeCamera = new BABYLON.FreeCamera("FreeCamera", new BABYLON.Vector3(0, 0, -  
30), scene);  
  
    // Δημιουργία αποτελέσματος βολής - στάμπας  
    var impact = BABYLON.Mesh.CreatePlane("impact", 1, scene);  
    impact.material = new BABYLON.StandardMaterial("impactMat", scene);  
    impact.material.diffuseTexture = new BABYLON.Texture("textures/impact.png", scene);  
    impact.material.diffuseTexture.hasAlpha = true;  
    impact.position = new BABYLON.Vector3(0, 0, -0.1);  
  
    // Δημιουργία επιπέδου - στόχου  
    var wall = BABYLON.Mesh.CreatePlane("wall", 20.0, scene);  
    wall.material = new BABYLON.StandardMaterial("wallMat", scene);  
    wall.material.emissiveColor = new BABYLON.Color3(0.5, 1, 0.5);  
  
    //Καθώς ενεργοποιείται ένα κλικ πράξε τα ακόλουθα  
    scene.onPointerDown = function (evt, pickResult) {  
        // Αν η βολή πετύχει το επίπεδο  
        // δημιουργήσε στάμπα με κατάλληλες συντεταγμένες  
        if (pickResult.hit) {  
            impact.position.x = pickResult.pickedPoint.x;  
            impact.position.y = pickResult.pickedPoint.y;  
        }  
    };  
  
    return scene;  
}
```

Κώδικας 4: Scene Picking, επιλογή στοιχείων στον τρισδιάστατο χώρο μέσω του κλικ ποντικιού

Γ3.3 Αιτιολόγηση χρήσης

Ο κύριος λόγος που χρησιμοποιήθηκε το πλαίσιο ανάπτυξης εφαρμογών BabylonJS ως το μέσω σχεδιασμού και προγραμματισμού του εκτελέσιμου παιχνιδιού πρώτου προσώπου που θα αναλυθεί στα πλαίσια της εργασίας έχει να κάνει κυρίως με την απλότητα και τη πληθώρα συναρτήσεων υψηλού επιπέδου. Όπως είδαμε σε προηγούμενη ενότητα η ανάπτυξη ενός βιντεοπαιχνιδιού από ένα άτομο το οποίο επιλέγει την εκκίνηση της διαδικασίας με ένα χαμηλού επιπέδου μέσου διεπαφής όπως η WebGL ή OpenGL αποτελεί έργο δύσκολο ως ακατόρθωτο. Η ταυτόχρονη γνώση στοιχείων μαθηματικών, γραφικών υπολογιστών, γενικότερων αρχών προγραμματισμού καθώς και εξοικείωσης με το ίδιο το μέσω διεπαφής είναι μερικά από τα στοιχεία που θα μετατρέψουν τη διαδικασία ανάπτυξης ως μη κατάλληλη για διδακτικό σκοπό και θα μπουν εμπόδιο στο στάδιο της δημιουργίας.

Από την άλλη η υλοποίηση ενός τρισδιάστατου εικονικού περιβάλλοντος με τη χρήση μίας ολοκληρωμένης μηχανής παραγωγής παιχνιδιών τριών διαστάσεων όπως η Unity και Unreal θα δημιουργούσε επίσης σύγχυση σε έναν προγραμματιστή μη εξοικειωμένο με παρόμοια εγχειρήματα αλλά για λόγους διαφορετικούς από αυτούς που αναφέρθηκαν παραπάνω. Μία μηχανή μεγάλου βεληνεκούς αγγίζει υψηλά επίπεδα abstraction που σημαίνει πως η υπερβολική απλοποίηση των συστημάτων που έχουν χτιστεί πάνω στα χαμηλότερου επιπέδου στρώματα των διαδικασιών παραγωγής γραφικών υπολογιστών, προσθέτει ακόμα περισσότερες γνώσεις που πρέπει να ληφθούν υπόψη από έναν προγραμματιστή ενός συμπεριλαμβάνουν και αυτές που αναφέρθηκαν παραπάνω. Πιο συγκεκριμένα οι γνώσεις των εσωτερικών μηχανισμών μιας μηχανής γραφικών καθώς και η ικανότητα σχεδιασμού επαρκούς αρχιτεκτονικής για ένα μεγάλο σύστημα βιντεοπαιχνιδιού είναι μερικές από τις δεξιότητες που απαιτεί η χρήση τέτοιων μηχανών. Φυσικά σε αυτά προστίθενται και οι εκμάθηση των πιο ισχυρών αλλά ταυτόχρονα και πιο απαιτητικών γλωσσών προγραμματισμού που υποστηρίζουν (στα συγκεκριμένα παραδείγματα που δόθηκαν C++ και C#) όπως επίσης και η εξερεύνηση του πολύπλοκου γραφικού περιβάλλοντος και του τεραστίου αριθμού λειτουργιών που παρέχουν. Τα χαρακτηριστικά αυτά καθιστούν σε κάθε περίπτωση τα εργαλεία αυτά ως πανίσχυρα αλλά ταυτόχρονα αποτελούν πρόσθετο θόρυβο στη σκέψη και στα πλάνα ενός μη μεγάλης εμπειρίας προγραμματιστή.

Φυσικά τα παραπάνω δε σημαίνουν πως το BabylonJS δεν επωφελείται από βαθύτερες γνώσεις των χαμηλού επιπέδου λειτουργιών ενός συστήματος, μίας μηχανής απόδοσης γραφικών υπολογιστών ή των μαθηματικών για την απεικόνιση τρισδιάστατων κόσμων. Αντιθέτως, προσφέρει μεθόδους των

οποίων το επίπεδο βρίσκεται στη χρυσή τομή μεταξύ ενός μέσου διεπαφής και μίας ολοκληρωμένης μηχανής παραγωγής βιντεοπαιχνιδιών. Συγκεκριμένα, είναι προφανές πως δεν θα μπορούσε να γίνει ομαλή χρήση του BabylonJS αν ο χρήστης δε γνωρίζει τη σημασία των διανυσμάτων στη γεωμετρία ή τις διάφορες έννοιες που κυριαρχούν σε έναν τρισδιάστατο χώρο όπως. Ουσιαστικά το παρών πλαίσιο προγραμματισμού απελευθερώνει τον προγραμματιστή από την διαδικασία εκμάθησης πολλών διαφορετικών τμημάτων γνώσεις και του επιτρέπουν να ξεκινήσει να ασχολείται με παραγωγή τρισδιάστατων κόσμων ταχύτατα. Έτσι αποκτάει γρήγορα γνώση του πως λειτουργεί ένας τρισδιάστατος χώρος γεγονός που συχνά αποτελεί πεδίο σύγχυσης πολλών επίδοξων προγραμματιστών διαδραστικών εφαρμογών και του αφήνει το περιθώριο να εξερευνήσει σιγά σιγά τις υπόλοιπες δυνατότητες μέσα από πολύ καλά δομημένες κλάσεις και συναρτήσεις.

Επιπρόσθετα οι σχεδιαστές του BabylonJS παρέχουν ένα απίστευτα δομημένο εγχειρίδιο, documentation, το οποίο περιγράφει λεπτομερώς τη λειτουργία κάθε κλάσεις και μεθόδου όπως επίσης και παρέχει διαδραστικά παραδείγματα τα οποία εκτελούνται σε πραγματικό χρόνο. Το γεγονός αυτό επιτυγχάνεται με τη χρήση του Babylon Playground που αποτελεί ένα διαδικτυακό εργαλείο δοκιμών, sandbox, για την εύκολη δημιουργία αυτόνομων παραδειγμάτων κώδικα δίνοντας τη δυνατότητα σε έναν προγραμματιστή να δοκιμάσει διάφορες θεωρίες και ιδέες που έχει χωρίς να περάσει από τη μεγάλη διαδικασία αρχικών ρυθμίσεων, setup, που απαιτούν άλλα εργαλεία.

Τη στιγμή που γράφεται η παρούσα εργασία σχεδιάζονται και εργαλεία γραφικού περιβάλλοντος όπως editor για τη εύκολη πραγματοποίηση αλλαγών σε έναν τρισδιάστατο κόσμο και node material editor ο οποίος επιτρέπει τη δημιουργία προγραμμάτων σκίασης, shader programs, χωρίς να απαιτείται η χρήση προγραμματισμού. Οπότε σιγά σιγά αρχίζει να παρέχει και δυνατότητες ολοκληρωμένης μηχανής γραφικών που είναι προαιρετικές προς τη χρήση.

Τέλος το BabylonJS αποτελεί δωρεάν και ανοιχτού κώδικα εργαλείο γεγονός που δίνει τη δυνατότητα σε έμπειρους χρήστες να συνεισφέρουν ενεργά στην ανάπτυξη της προτείνοντας και υλοποιώντας αλλαγές. Φυσικά ο πηγαίος κώδικας του πλαισίου αυτού μπορεί να χρησιμοποιηθεί και για διδακτικούς σκοπούς πάνω στους τομείς της αρχιτεκτονικής εργαλείων παραγωγής διαδραστικού λογισμικού.

Γ4 Το εργαλείο σχεδιασμού Blender

Κατά τον σχεδιασμό και την ανάπτυξη του εκτελέσιμου της παρούσας εργασίας χρησιμοποιήθηκαν προκατασκευασμένα και στη πλειοψηφία τους animated τρισδιάστατα μοντέλα. Παρ' όλα αυτά η

μετατροπή τους σε τέτοια μορφή ώστε να χρησιμοποιηθούν με ευκολία στα πλαίσια του προγραμματισμού με τη χρήση του BabylonJS ήταν έργο το οποίο απαιτούσε τη χρήση ενός εξειδικευμένου προγράμματος τρισδιάστατης σχεδίασης.

Γ4.1 Γενικές πληροφορίες

Το Blender αποτελεί μία δωρεάν και ανοιχτού πηγαίου κώδικα σουίτα τρισδιάστατου σχεδιασμού. Υποστηρίζει όλες τις απαιτήσεις τις διασωλήνωσης τρισδιάστατων γραφικών όπως modeling, rigging, animation, simulation, rendering, motion tracking και ακόμα επεξεργασία βίντεο και δημιουργία βιντεοπαιχνιδιών. Προχωρημένοι χρήστες μπορούν να χρησιμοποιήσουν το μέσω διεπαφής προγραμματισμού εφαρμογών που προσφέρει το Blender με τη γλώσσα προγραμματισμού Python, με σκοπό να δημιουργήσουν νέα εργαλεία τα οποία εξυπηρετούν τις ατομικές ανάγκες τα οποία έχουν πιθανότητες να συμπεριληφθούν σε επόμενες επίσημες εκδόσεις. Τα παραπάνω το κάνουν να αποτελεί ιδανικό λογισμικό για άτομα αλλά και μικρές ομάδες που δε θα μπορούσαν να καλύψουν τα έξοδα συνδρομής για τη χρήση άλλων προγραμμάτων τρισδιάστατης σχεδίασης που βρίσκονται διαθέσιμα στην αγορά.

Επιπρόσθετα υποστηρίζει όλες τις πλατφόρμες Windows, Linux και Macintosh και το ότι βρίσκεται υπό τη προστασία της άδειας GNU General Public License (GPL), δίνει τη δυνατότητα στο κοινό να προτείνει μικρές ή μεγάλες αλλαγές ή ακόμα και να διορθώσει λάθη στο πηγαίο κώδικα του λογισμικού με σκοπό τη προσθήκη νέων λειτουργιών και την εγγύηση της σταθερότητας χρήσης του.

Τέλος τα τελευταία χρόνια έχει δημιουργηθεί το Blender Fund στο οποίο μεγάλες εταιρείες έχουν επενδύσει μεγάλα ποσά για τη διευκόλυνση της ανάπτυξης του Blender που μεταξύ άλλων περιλαμβάνουν της εταιρείες Epic, Microsoft και Google. Το γεγονός αυτό έχει ανεβάσει το συγκεκριμένο λογισμικό αρκετά ψηλά στις προτιμήσεις των σχεδιαστών με αποτέλεσμα μεγάλα στούντιο ταινιών να το χρησιμοποιούν.

Γ4.2 Αιτιολόγηση χρήσης του Blender

Τα προσχεδιασμένα από τρίτους μοντέλα τα οποία συμπεριλήφθηκαν στην ανάπτυξη των λειτουργιών των συστημάτων του βιντεοπαιχνιδιού πρώτου προσώπου που αναπτύχθηκε στα πλαίσια της εργασίας απαιτούσαν αρκετές μετατροπές στο σύνολο τους ώστε να χρησιμοποιηθούν ομαλά από τις μεθόδους φόρτωσης του BabylonJS.

Πιο συγκεκριμένα πραγματοποιήθηκε η μετονομασία πολλών στοιχείων άκρων (bones) για χρήση στα animations και η μετατροπή του τρόπου σκίασης των ταπετσαριών στον κατάλληλο που υποστηρίζει το BabylonJS. Επιπρόσθετα σε αρκετά τρισδιάστατα μοντέλα προστέθηκαν γεωμετρικά σχήματα κύβων τα οποία λειτουργούν ως Bounding αλλά και Hitboxes για τη διευκόλυνση της ανίχνευσης συγκρούσεων με το κεντρικό χαρακτήρα αλλά και την εύκολη ρίψη βολών σε αυτά.

Ακόμα με τη χρήση του λογισμικού αυτού πραγματοποιήθηκε η γενικότερη διαδικασία level designing με το σχεδιασμό των επιπέδων στα οποία κινείται ο παίκτης καθώς και της τοποθέτησης των διάφορων στοιχείων του κόσμου όπως κτίρια, λίμνης, φυτών καθώς και τον ίδιων των εχθρών.

Ο συνδυασμός του μηδενικού κόστους καθώς και η πληθώρα πηγών εκμάθησης χρήσης ενός τέτοιου πολύπλοκου λογισμικού μετέτρεψαν την όλη εμπειρία σχεδιασμού, εμπειρία αρκετά δύσκολη για τους περισσότερους προγραμματιστές, σε κατανοητή, ομαλή και εύκολη.

ΚΕΦΑΛΑΙΟ Δ΄

ΠΑΡΟΥΣΙΑΣΗ ΕΚΤΕΛΕΣΙΜΟΥ ΕΡΓΑΣΙΑΣ

Όπως αναφέρθηκε στα προηγούμενα κεφάλαια, στα πλαίσια της παρούσας εργασίας πραγματοποιήθηκε η διαδικασία ανάπτυξης ενός βιντεοπαιχνιδιού πρώτο προσώπου χρησιμοποιώντας τη γλώσσα προγραμματισμού Javascript, το πλαίσιο προγραμματισμού BabylonJS το οποίο περιέχει βοηθητικές λειτουργίες κατασκευασμένες πάνω στο μέσω διεπαφής προγραμματισμού εφαρμογών WebGL, το λογισμικό σχεδιασμού τρισδιάστατων μοντέλων Blender και το σύστημα εκδόσεων GIT. Το εγχείρημα αυτό είχε σκοπό την ανάδειξη των όσων συζητήθηκαν στα προηγούμενα κεφάλαια δείχνοντας έτσι τις κύριες λειτουργίες που απαιτούνται για την ανάπτυξη ενός ολοκληρωμένου τρισδιάστατου περιβάλλοντος. Λαμβάνοντας υπόψιν τα στοιχεία που θα παρουσιαστούν σε αυτήν ενότητα θα γνωρίσει κανείς από πλευράς προγραμματισμού τα τμήματα από τα οποία αποτελείται ένα βιντεοπαιχνίδι πρώτου προσώπου παρουσιάζοντας έτσι μία πιο χαμηλού επιπέδου άποψη στα αφαιρετικά μοντέλα τα οποία παρουσιάστηκαν στα κεφάλαια περιγραφής των συστημάτων απόδοσης γραφικών υπολογιστών και περιγραφής βασικών στοιχείων μηχανών βιντεοπαιχνιδιών.

Δ1 Γενικές πληροφορίες και Σκοπός Παιχνιδιού

Σε αυτήν την υπο-ενότητα θα παρουσιαστούν οι γενικότερες πληροφορίες του παιχνιδιού σχετικά με το γραφικό περιβάλλον του, τον διαθέσιμο εξοπλισμό, το περιβάλλοντα χώρο, τις δυνατότητες του παίκτη καθώς και το βασικό σκοπό που καλείται να εκπληρώσει ο παίκτης.

Τα κύρια τεχνικά χαρακτηριστικά του παιχνιδιού ως το γραφικό περιβάλλον είναι τα εξής :

- Μπάρα απεικόνισης πόντων ζωής με κόκκινο χρώμα
 - Μέγιστος αριθμός 20
- Μπάρα απεικόνισης πόντων ενέργεια με μπλε
 - Μέγιστος αριθμός 20
- Δείκτης ποσότητας διαθέσιμων πυρομαχικών
 - Αριθμός ανάλογα με το επιλεγμένο όπλο

Στο τμήμα του εξοπλισμού, ο παίκτης έχει στη διάθεση του εξαρχής τις ακόλουθες επιλογές οπλισμού :

1. Πιστόλι (pistol)
 - Πυρομαχικά (Ammo) : 50
 - Ζημιά (Damage) : 10
 - Βεληνεκές (Range) : 15
2. Καραμπίνα με δύο κάννες (double barrel shotgun)
 - Πυρομαχικά (Ammo) : 25
 - Ζημιά (Damage) : 30
 - Βεληνεκές (Range): 20
3. Πολυβόλο Καλάσνικοφ (AK47)
 - Πυρομαχικά (Ammo) : 25
 - Ζημιά (Damage) : 15
 - Βεληνεκές (Range) : 25
4. Όπλο λείζερ (ray gun),
 - Πυρομαχικά (Ammo) : 30
 - Ζημιά (Damage) : 20
 - Βεληνεκές (Range): 30
5. Όπλο κεραυνού (lightning gun)
 - Πυρομαχικά (Ammo) : 10
 - Ζημιά (Damage) : 40
 - Βεληνεκές (Range): 25

Από τα παραπάνω φαίνεται πως τα όπλα είναι σχεδιασμένα με το είδος της ζημιάς τους να είναι προοδευτικό και αυτά να αποτελούν κατάλληλα για μικρή-μέση-μεγάλη απόσταση. Χαρακτηριστικό ιδιαίτερο αποτελεί πως τα πιο “ίσχυρά” όπλα έχουν λιγότερα διαθέσιμα πυρομαχικά.

Επιπρόσθετα υπάρχουν διάσπαρτα στη πίστα πακέτα πυρομαχικών, τα οποία χρησιμοποιούν το μοντέλου του όπλου για να απεικονισθούν και επαναφέρουν συγκεκριμένο αριθμό πυρομαχικών το καθένα ως εξής :

- Πιστόλι : +20
- Καραμπίνα : +10
- Καλάσνικοφ : + 10
- Όπλο λείζερ : + 15
- Όπλο κεραυνού : + 5

Ως προς τον σχεδιασμό του τρισδιάστατου περιβάλλοντος, ο κόσμος του παιχνιδιού αποτελείται από ένα επίπεδο σε σχήμα σταυρού το οποίο αποτελείται από μικρά υπό επίπεδα το καθένα με το δικό του χαρακτήρα και στοιχεία φύσης, τα οποία παρουσιάζονται ως εξής:

- Χωριό προστατευμένο πίσω από τοίχος και κλειστή πύλη
- Έρημος με κάκτους και λόφους
- Περιφραγμένη λίμνη με νούφαρα και φυτά
- Δάσος με φράχτη και βλάστηση
- Κεντρικός χώρος (main hub(που ξεκινάει ο παίκτης στο κέντρο του κόσμου

Στο χωριό συγκεκριμένα περιέχεται εφέ σωματιδίων φωτιάς (fire particle effect) καθώς και πληθώρα κτιρίων.

Στα επίπεδα αυτά ο παίκτης έχει τη δυνατότητα να αναμετρηθεί με τα ακόλουθα είδη αντιπάλων που βρίσκονται στο κάθε ένα :

1. Ιπτάμενα κρανία (enemy) στο χώρο της λίμνης :
 - Πόντοι ζωής : 50
 - Ζημιά που προκαλούν στον παίκτη με κάθε εκτόξευση βλήματος : 0.15
2. Ιπτάμενος δράκος (dragon) στο χώρο της ερήμου:
 1. Πόντοι ζωής : 50
 2. Ζημιά : 0.09
3. Απέθαντος (zombie) :
 1. Πόντοι ζωής : 150

2. Ζημιά : 0.6
4. Σκελετός (skeleton) εντός των πυλών του χωριού :
 1. Πόντοι ζωής : 200
 2. Ζημιάς : 0.06

Οι εχθροί δημιουργήθηκαν ώστε να αναπαριστούν διαφορετικά επίπεδα δυσκολίας αντιμετώπισης. Επιπρόσθετα ο κάθε εχθρός έχει το δικό του ξεχωριστό animation κίνησης και επίθεσης στον παίκτη.

Ο παίκτης έχει τη δυνατότητα να αυξήσει τη ταχύτητα κίνησης του στο χώρο καταναλώνοντας πόντους ενεργείας οι οποίοι μπορούν να αναπληρωθούν με την απόκτηση ενός από τον διάσπαρτων πακέτων ενέργειας που υπάρχουν στη περιβάλλον. Κάθε τέτοιο πακέτο ενέργειας που υπάρχει διαθέσιμο προσδίδει στο παίκτη 5 μονάδες ενέργειας.

Επιπρόσθετα πακέτα πόντων ζωής τα οποία προσδίδουν 5 πόντους με την απόκτηση τους υπάρχουν επίσης διάσπαρτα στο περιβάλλοντα χώρο και σκοπό έχουν την αναπλήρωση ποσοτήτων ζωής που χάθηκαν έπειτα από τα χτυπήματα που δέχθηκε ο παίκτης κατά την επαφή του με τους εχθρούς που αναφέρθηκαν πιο πάνω.

Τέλος όπως κάθε παιχνίδι έτσι και αυτό περιέχει έναν συγκεκριμένο σκοπό τον οποίο πρέπει να εκπληρώσει ο παίκτης μέσω του χειρισμού του κεντρικού χαρακτήρα. Ο σκοπός αυτός καθορίζει το βασικό κύκλο επανάληψης του παιχνιδιού (gameplay loop) καθώς και τις συνθήκες τερματισμού. Πιο συγκεκριμένα ο παίκτης καλείται να σώσει το χωριό εξοντώνοντας όλους τους εχθρούς που το έχουν καταλάβει. Αρχικά ο παίκτης με το που εμφανίζεται στο χώρο του παιχνιδιού και το main hub δεν έχει πρόσβαση στο χωριό λόγω της κλειστής πύλης και των τειχών που το περιβάλλουν. Για να ξεκλειδώσει τη πύλη, καλείται να εξουδετερώσει όλους τους εχθρούς που βρίσκονται γύρω του. Με το που εκπληρώσει αυτή τη συνθήκη, ο φυλλομετρητής παγώνει και εμφανίζει το μήνυμα πως η πύλη του χωριού έχει ανοίξει. Έτσι με την είσοδο του παίκτη στο χωριό και την τελική εξουδετέρωση των εχθρών, σκελετών, που απομένουν ο φυλλομετρητής πάλι παγώνει και εμφανίζει κατάλληλο μήνυμα νίκης. Σε κάθε άλλη περίπτωση την οποία η πόντοι ζωής του παίκτη τελειώσουν ο φυλλομετρητής ανανεώνεται και ο παίκτης ξεκινάει από την αρχή.

Δ2 Αναλυτική παρουσίαση κώδικα

Στην ενότητα αυτή θα παρουσιάσουμε αναλυτικά τα επιμέρους τμήματα κώδικα τα οποία συνθέτουν το σύνολο των λειτουργιών ενός βιντεοπαιχνιδιού πρώτου προσώπου. Θα ακολουθήσει η περιγραφή των κλάσεων και των χαρακτηριστικών κάθε εχθρού, αντικειμένου εξοπλισμού, πακέτου ενέργειας, του κεντρικού χαρακτήρα και τέλος του πιο σημαντικού της κύριας κλάσης που συνθέτη τη μηχανή των περισσότερων λειτουργιών και συνδέει τα διάφορα αντικείμενα του τρισδιάστατου περιβάλλοντος μεταξύ τους.

Πιο συγκεκριμένα τα αρχεία-κλάσεις που συνθέτουν τη πλήρες εκτελέσιμο της εργασίας είναι τα εξής :

- `index`: το οποίο αποτελεί το κύριο αρχείο και ενεργοποιεί τις βασικές λειτουργίες του εκτελέσιμου καθώς και το `render loop`
- `engine`: το οποίο αρχικοποιεί τις ρυθμίσεις του κόσμου και περιέχει χρήσιμες λειτουργίες όπως τη φόρτωση των τρισδιάστατων μοντέλων
- `character`: το οποίο περιέχει τα χαρακτηριστικά του παίκτη και τις συναρτήσεις που υλοποιούν τις δυνατότητες του
- `enemy`, `skeleton`, `zombie`, `dragon`: περιέχει χαρακτηριστικά εχθρών και συναρτήσεις κίνησης και πραγματοποίησης ζημιάς στον κεντρικό χαρακτήρα
- `weapon`: χρησιμοποιείται για τη δημιουργία όπλου και τη ρύθμιση των `animation` του
- `collectible`: περιέχει τον ορισμό των πακέτων ζωής και ενέργειας

Δ2.1 Αρχείο index.js

Το αρχείο αυτό αποτελεί το σημείο κατά το οποίο δημιουργείται το αντικείμενο `game` μέσω της κλάσης `engine()`, των βασικών λειτουργιών του τρισδιάστατου περιβάλλοντος που δημιουργήσαμε, και το αντικείμενο του κεντρικού χαρακτήρα `player` μέσω της κλάσης `character()`.

```
var game, player;
function main(){
  game = new Engine();
  player = new Character(game);
  game.player = player;
  game.assetManager();
  game.pointerLock();
  player.characterController(game.hud);
  game.render();
}
main();
```

Κώδικας 5: Αρχείο index.js : εκκίνηση βασικών λειτουργιών και εκτελέσιμου

Επιπρόσθετα ενεργοποιείται η λειτουργία δέσμευσης του κέρσορα του ποντικιού από το παράθυρο του εκτελέσιμου μέσω της συνάρτησης `game.pointerLock()` και έπειτα ακολουθεί η ενεργοποίηση του κύριου rendering loop από τη συνάρτηση `render()` του αντικειμένου `game`.

Δ2.2 Αρχείο engine.js

Όπως είδαμε στο αρχείο `index` δημιουργείται το αντικείμενο `game` το οποίο αποτελεί instance της κλάσης `engine`. Ο constructor αυτής, ρυθμίζει βασικά στοιχεία όπως τη δέσμευση του καμβά, τη δημιουργία της σκηνής, τη δημιουργία της κάμερας και των ιδιοτήτων της τα οποία βοηθούν στη διαδικασία ανίχνευσης συγκρούσεων. Επιπρόσθετα λαμβάνει χώρα η δημιουργία των λιστών αποθήκευσης των οντοτήτων εχθρών(`enemyList[]`) αλλά και των πακέτων ενέργειας (`collectibleList[]`) στις γραμμές (lines) 27 και 28 του πηγαίου κώδικα, όπως και η ενεργοποίηση του

συστήματος παραγωγής σωματιδίων (particle system) στις γραμμές 67. Αποσπάσματα του πηγαίου κώδικα παρουσιάζονται παρακάτω :

```
constructor(){
    // Scene setup
    this.canvas = document.getElementById("renderCanvas");
    this.engine = new BABYLON.Engine(this.canvas);
    this.scene = new BABYLON.Scene(this.engine);
    // Entities
    this.player;
    this.enemyList = [];
    this.collectibleList = [];
}
```

Κώδικας 6: Βασικά στοιχεία σκηνής και λίστες εχθρών και πακέτων ενέργειας και ζωής

```
// Particle system setup
this.particleSystem = new BABYLON.ParticleSystem("particles", 2000, this.scene);
this.particleSystem.particleTexture = new
BABYLON.Texture("../assets/textures/flare.png", this.scene);
this.particleSystem.emitRate = 0;
this.particleSystem.start();
```

Κώδικας 7: Ενεργοποίηση συστήματος παραγωγής σωματιδίων

Όπως παρατηρείται στο απόσπασμα κώδικα 7 , το BabylonJS παρέχει τη δυνατότητα ρύθμισης του ρυθμού παραγωγής πολλαπλών σωματιδίων. Στη συγκεκριμένη περίπτωση ο αριθμός παραγωγής σωματιδίων έχει οριστεί 0 διότι θα θέλαμε το στοιχείο να αναπαραχθεί μία μόνο φορά.

Επιπρόσθετα στο ίδιο αρχείο συναντάμε την μέθοδο `assetManager()` της οποίας οι λειτουργίες περιστρέφονται γύρω από το θέμα της φόρτωσης τρισδιάστατων μοντέλων, που έχουν δημιουργηθεί μέσω του προγράμματος σχεδίασης Blender, καθώς και της δημιουργίας μερικών στοιχείων του τρισδιάστατου περιβάλλοντος όπως του ουρανού (skybox) και του υλικού λίμνης (lake material). Παρακάτω ακολουθούν κάποια τμήματα κώδικα της μεθόδους αυτής που παρουσιάζουν ιδιαίτερη σημασία.

```
// Setting up the collision detection for various world props

let buildings = scene.getTransformNodeByName("villageBuildings").getChildMeshes();

    buildings.forEach(function(entry){

        entry.checkCollisions = true;

    });

let villageWalls = scene.getTransformNodeByName("villageWalls").getChildMeshes();

    villageWalls.forEach(function(entry){

        entry.checkCollisions = true;

    });

let trees = scene.getTransformNodeByName("trees").getChildMeshes();

    trees.forEach(function(entry){

        entry.checkCollisions = true;

    });
```

Κώδικας 8: Γραμμές 149-163 : Ενεργοποίηση ανίχνευσης συγκρούσεων σε στοιχεία του χώρου

Μέσω του σχεδιαστικού προγράμματος Blender διάφορα στοιχεία του περιβάλλοντα χώρου ομαδοποιήθηκαν σε κατηγορίες για πιο εύκολη επεξεργασία. Στο τμήμα Κώδικας 8, παρουσιάζεται η διάσχιση των παιδιών του κάθε κόμβου και η ενεργοποίηση των συστημάτων ανίχνευσης συγκρούσεων ώστε να αναγνωρίζονται κατάλληλα με αποτέλεσμα η κάμερα – παίκτης να μη μπορεί να τα διαπεράσει.

Ακόμη η μέθοδος pointerLock() εξυπηρετεί στη δέσμευση του ποντικιού εντός του καμβά που χρησιμοποιείται από το BabylonJS ώστε να αποκτήσει τον έλεγχο της κάμερας ο παίκτης. Μέσω του έλεγχου συμβάντων με την ανίχνευση του κλικ ποντικιού ελέγχεται μέσω της τεχνικής ray picking το αν χτυπήθηκε κάποιος από τους εχθρούς του χώρου, τμήμα της ακόλουθης διαδικασίας παρουσιάζεται παρακάτω στο τμήμα Κώδικα 9. Χαρακτηριστικά μέσα από τη λίστα εχθρών

εντοπίζεται αυτός που επαληθεύθηκε από τον έλεγχο συμβάντος του ποντικιού μέσω του ονόματος του. Έπειτα αφαιρείται από τη ζωή του ποσό ίσο με αυτό της ζημιάς που προκαλεί το όπλο που χειρίζεται τη δεδομένη στιγμή της βολής ο παίχτης. Σε περίπτωση που ο εχθρός έχει μηδενικούς πόντους ζωής, αφαιρείται από το κόσμο του παιχνιδιού.

```
if(enemyList[i].name == model.parent.name || (model.parent.parent &&
enemyList[i].name == model.parent.parent.name)){

// Damage enemy
if(enemyList[i].health > 0){

    enemyList[i].health -= currentWeapon.damage;
    console.log("Target Hit :" + model.parent.name + " Health :" +
    enemyList[i].health );
}

// Destroy enemy

if(enemyList[i].health <= 0){

    enemyList[i].destroy();
    break;
}
```

Κώδικας 9: Γραμμές 336 – 346 : Αναζήτηση εντοπισμένου εχθρού μέσα από τη λίστα εχθρών

Στη συνέχεια τη δημιουργία του γραφικού περιβάλλοντος, heads up display, λαμβάνει η μέθοδος hudManager(). Στοιχεία GUI.Rectangle, για τις πληροφορίες πόντων ζωής και ενέργειας και GUI.TextBlock, για τις πληροφορίες πυρομαχικών, δημιουργούνται με διάφορα μεγέθη και χρώματα και εισάγονται σε μία ταπετσαρία πλήρους οθόνης με τη χρήση της μεθόδου του υποσυστήματος GUI του BabylonJS, που ονομάζεται CreateFullScreenUI(). Τμήμα των ενεργειών αυτών ακολουθεί.

```
// GUI setup
var advancedTexture = new GUI.AdvancedDynamicTexture.CreateFullscreenUI("UI", true,
this.scene);
// Array filled with our three info bars
var hudComponents = [];
// Health bar
var healthBar = new GUI.Rectangle("healthBar");
healthBar.left = -500;
healthBar.top = 400;
healthBar.width = 0.2;
healthBar.height = "20px";
healthBar.cornerRadius = 20;
healthBar.color = "white";
healthBar.thickness = 4;
healthBar.background = "red";
advancedTexture.addControl(healthBar);
```

Κώδικας 10: Γραμμές 378 – 393 : Δημιουργία μπάρας πληροφοριών πόντων ζωής

Ακόμη η κλάση engine παρέχει και τη μέθοδο render() η οποία όπως αναφέραμε στην ανάλυση του αρχείου index.js αποτελεί το βασικό κύκλο απόδοσης γραφικών (rendering loop). Κατά τη διαδικασία αυτή οποιοσδήποτε κινήσεις, υπολογισμοί ή γενικότερα μεταβολές στο χώρο οριστικοποιούνται και απεικονίζονται στην οθόνη.

```
render(){

    // Render every frame

    this.engine.runRenderLoop(() => {

        this.scene.render();

    });

}
```

Κώδικας 11: Γραμμές 420 – 425 : Κύκλος απόδοσης γραφικών

Πρόσθετες βοηθητικές συναρτήσεις περιέχονται στο αρχείο engine. Κύριος σκοπός τους είναι η χρήση τους εντός της μεθόδου assetManager() για την πιο εύκολη και γρήγορη επεξεργασία, φόρτωση και ρύθμιση των τρισδιάστατων μοντέλων των εχθρών των όπλων αλλά και των πακέτων

ενέργειας που βρίσκονται στο αρχείο που δημιουργήθηκε μέσω του Blender. Πιο συγκεκριμένα η συνάρτηση `addPistol` αναλαμβάνει το έργο της ρύθμισης των διαστάσεων των διάφορων τρισδιάστατων μοντέλων των όπλων. Η συνάρτηση `addEnemy` δημιουργεί τα αντικείμενα των εχθρών και παρέχει στους κατάλληλους constructors τις επιθυμητές τιμές αρχικοποίησης των ιδιοτήτων τους. Η συνάρτηση `addCollectible` δημιουργεί και προσθέτει στη λίστα των πακέτων ενέργειας τα αντικείμενα αυτών προορίζοντας τα με αυτόν τον τρόπο για χρήση στα υπόλοιπα συστήματα του παιχνιδιού. Οι παραπάνω συναρτήσεις προσθέτουν στην ουρά απόδοσης γραφικών τις ενεργές λειτουργίες όπως κίνηση εχθρών και περιστροφή πακέτων ενέργειας.

Δ2.3 Αρχείο `character.js`

Το συγκεκριμένο αρχείο υλοποιεί τη κλάση του κεντρικού χαρακτήρα και αρχικοποιεί τη λίστα εξοπλισμού καθώς και τους πόντους ζωής και ενέργειας του παίκτη. Επιπρόσθετα συνδέει τα στοιχεία γραφικού περιβάλλοντος τα οποία δημιουργήθηκαν στη κλάση `engine`.

```
// World info
this.game = game;
this.scene = game.scene;
this.camera = game.camera;
this.camera.position.z = 5;
// Player info
this.gunLoadout = [];
this.health = 20;
this.energy = 20;
this.running = false;
this.walking = false;
this.standing = true;
// Index marking weapon in use
this.currentWeapon = 0;
// Getting the camera's physics impostor
this.cameraImpostor = this.camera.getChildren();
// Hud
this.hud = game.hud;
this.debug = false;
```

Κώδικας 12: Γραμμές 7 - 25 : Αρχικοποίηση στοιχείων κεντρικού χαρακτήρα - παίκτη

Η ίδια κλάση ωστόσο παρέχει μεθόδους οι οποίες χειρίζονται τις αυξομειώσεις στους πόντους ζωής, πόντους ενέργειας αλλά και του αριθμού πυρομαχικών για κάθε όπλο.

```
energyUp(){  
  
    console.log("Energy Up");  
  
    if(this.energy < 20){  
  
        this.energy += 5;  
  
        this.hud[1].width = this.energy/100;  
  
    }  
  
}
```

```
energyDown(){  
  
    //console.log("Energy Down");  
  
    if(this.energy >= 0){  
  
        this.energy -= 0.05;  
  
        this.hud[1].width = this.energy/100;  
  
        return true;  
  
    }  
  
    else{  
  
        return false;  
  
    }  
  
}
```

Κώδικας 13: Γραμμές 52-70 : Μέθοδοι αύξησης και μείωσης πόντων ενέργειας

Έπειτα ακολουθεί η μέθοδος character controller η οποία αποτελεί το κέντρο ελέγχου συμβάντων σχετικών με τη χρήση των πλήκτρων W A S D και Shift του πληκτρολογίου τα οποία ως αποτέλεσμα έχουν τη κίνηση του παίκτη στο περιβάλλοντα χώρο και την αύξηση της ταχύτητας του.

```
// Create our own manager:
var FreeCameraKeyboardRotateInput = function () {
this._keys = [];
this.keysLeft = [65]; // A
this.keysRight = [68]; // D
this.keysForward = [87]; // W
this.keysBackward = [83]; // S
};
```

Κώδικας 14: Γραμμές 134 – 141 : Καθορισμός πλήκτρων περιήγησης στο χώρο

Επίσης παρακάτω πραγματοποιείται έλεγχος συμβάντων μέσω της συνάρτησης onKeyboardObservable.add() η οποία έχει τη δυνατότητα προσθήκης επιπλέον χαρακτηριστικών τα οποία ελέγχονται σε κάθε κύκλο απόδοσης γραφικών μέσω του συστήματος ελέγχου συμβάντων που προσφέρει το BabylonJS.

```
case "1":
console.log("Pistol selected");
this.currentWeapon = 0;
ammoHud.text = String(gunLoadout[this.currentWeapon].ammo);
weaponSwitch(gunLoadout, this.currentWeapon);
Break;

case "2":
console.log("Shotgun selected");
this.currentWeapon = 1;
ammoHud.text = String(gunLoadout[this.currentWeapon].ammo);
weaponSwitch(gunLoadout, this.currentWeapon);
break;
```

Κώδικας 15: Γραμμές 265 - 277 : Επιλογή κατάλληλου όπλου με βάση τα πλήκτρα

Ακόμη στο τμήμα κώδικα παρακάτω παρουσιάζεται ο τρόπος ελέγχου των καταστάσεων του παίκτη. Με τις τρεις περιπτώσεις να κυμαίνονται ως ακίνητος (standing), κινείται (walking) και κινούμενος ταχύτατα (running) μπορούμε να εξακριβώσουμε το αν του επιτρέπεται να τρέξει.

```
scene.onBeforeRenderObservable.add(function(){

//console.log("Standing : " + player.standing + " Walking : " + player.walking + "
Running : " + player.running)

if(player.running && !player.standing && player.walking && player.energyDown()) {

//console.log("Running");

player.camera.speed = 0.8;

}

else{

//console.log("Not Running");

player.camera.speed = 0.4;

}
```

Κώδικας 16: Γραμμές 334 – 343 : Έλεγχος κατάστασης χαρακτήρα

Η κλάση που αναλύθηκε στη συγκεκριμένη ενότητα αποτελεί μεγάλης σημασίας διότι χωρίς αυτήν δεν θα μπορούσαμε εύκολα και γρήγορα να μεταβάλουμε τις ιδιότητες και τα χαρακτηριστικά του παίκτη καθώς το παιχνίδι εξελίσσεται. Οι μέθοδοι που παρέχει το αντικείμενο αυτό μπορούν να κληθούν από τα υπόλοιπα συστήματα του βιντεοπαιχνιδιού.

Δ2.4 Αρχείο weapon.js

Στο αρχείο weapon.js υλοποιείται η κλάση weapon η οποία περιέχει τα κύρια στοιχεία που συνθέτουν τον τρόπο λειτουργίας και αναπαράστασης των όπλων που βρίσκονται διαθέσιμα στο παιχνίδι. Ο constructor της κλάσης αυτής, αρχικοποιεί της τιμές ονόματος, το μοντέλο, τον αριθμό διαθέσιμων πυρομαχικών και ζημιάς μεταξύ άλλων.

```
// Weapon properties
this.name = name;
this.mesh = mesh;
this.ammo = ammo;
this.damage = damage;
this.range = range;
this.scene = scene;
```

Κώδικας 17: Γραμμές 5 - 11 : Αρχικοποίηση ιδιοτήτων όπλου

Επιπρόσθετα η μέθοδος της κλάσης setAnimations() καλείται κατά τη δημιουργία του αντικειμένου όπλου μέσα από τον constructor. Η μέθοδος αυτή ρυθμίζει τα animations του κάθε όπλου ανάλογα με το όνομα του μιας και αυτά ενώ βρέθηκαν προ κατασκευασμένα, δεν συμπεριλαμβάνονταν λειτουργίες animation. Αυτό είχε ως αποτέλεσμα να πρέπει να δημιουργηθούν χειροκίνητα απευθείας μέσω της γλώσσας προγραμματισμού Javascript και τις μεθόδους που παρέχει το BabylonJS. Κατάλληλο απόσπασμα από τον πηγαίο κώδικα ακολουθεί.

```

if(this.name == "ak47"){

    //this.damage = 5;

    //this.range = 50;

    end.x -= Math.PI/100;

    this.animationSpeed = 5;


    this.bar.setPositionWithLocalVector(new BABYLON.Vector3(0.70, -0.74, 3.30));

}

```

Κώδικας 18: Γραμμές 67 - 74 : Ρύθμιση στοιχείων animation όπλου AK47

Μέσω της εντολής end.x μπορούμε να καθορίσουμε την τελική θέση που επιθυμούμε να έχει το αντικείμενο όπλου στο χώρο. Στη συγκεκριμένη περίπτωση ορίζουμε τη τελική θέση μίας περιστροφικής κίνησης, με τις κινήσεις αυτές.

Επιπρόσθετα οι βοηθητικές συναρτήσεις που είναι παρούσες στο ίδιο αρχείο συντελούν στη δημιουργία των οπτικών εφέ βολής για κάθε όπλο. Χαρακτηριστικό παράδειγμα αποτελεί η συνάρτηση που παρουσιάζεται παρακάτω και δημιουργεί το εφέ λείζερ του όπλου ray gun.

```

if(dest == null){
    dest = org;
}

var dist = BABYLON.Vector3.Distance(org, dest);
var orb1 = orbMesh.clone("orb1");
orb1.isVisible = true;
orb1.isPickable = false;
var spark1 = sparkMesh.clone("spark");
spark1.material.emissiveTexture = getLazerTexture(256, 128, scene);
spark1.material.opacityTexture = spark1.material.emissiveTexture;
spark1.isVisible = true;
spark1.scaling.z = dist;
spark1.position = org.clone();
spark1.lookAt(dest);

```

Κώδικας 19: Γραμμές 241 – 261 : Δημιουργία οπτικού εφέ λείζερ

Δ2.5 Αρχείο collectible.js

Το αρχείο collectible.js περιέχει την υλοποίηση της κλάσης η οποία συντελεί στη δημιουργία των αντικειμένων πακέτων πόντων ζωής αλλά και ενέργειας. Στον constructor της κλάσης ορίζονται οι βασικές ιδιότητες του αντικειμένου όπως το όνομα και το τρισδιάστατο μοντέλο που θα χρησιμοποιηθεί. Η μέθοδος destroy() που ακολουθεί, αποτελεί κύριο συστατικό στοιχείο του αντικείμενου μιας και χειρίζεται τη συνθήκη καταστροφής και αφαίρεσης από το κόσμο του παιχνιδιού, του πακέτου ζωής ή ενέργειας. Αυτό πραγματοποιείται με τη μέτρηση της απόστασης που έχει το αντικείμενο από τον παίκτη. Τη στιγμή που η απόσταση αυτή distVec αποκτάει τιμή μικρότερη του 4, το αντικείμενο καταστρέφεται. Η συγκεκριμένη τιμή ελέγχου επιλέχθηκε διότι προσομοιώνει με καλύτερο τρόπο την αίσθηση ότι το εξωτερικό τμήμα του παίκτη έχει επαφή με το περιστρεφόμενο αντικείμενο και έτσι το ακουμπάει. Σε αντίθετη περίπτωση που είχε επιλεχθεί η τιμή 0 για τον έλεγχο αυτόν, ο παίκτης θα έπρεπε να διαπεράσει εξολοκλήρου το περιστρεφόμενο αντικείμενο για να το αποκτήσει, γεγονός που δεν θα ταίριαζε με τις καθιερωμένες πρακτικές που χρησιμοποιούνται στην ανάπτυξη διαδραστικών εικονικών περιβαλλόντων.

```
destroy(){
let mesh = this.mesh;
let scene = this.scene;
let camera = scene.activeCamera;
let distVec = BABYLON.Vector3.Distance( camera.position, mesh.absolutePosition);

    if(mesh != null){
        // Remove collectible from the scene
        if(distVec <= 4){
            mesh.dispose();
        }
    }
}
```

Κώδικας 20: Γραμμές 63 - 78 : Μέθοδος καταστροφής περιστρεφόμενου συλλεκτικού πακέτου

Επιπρόσθετα κατά την δημιουργία του αντικειμένου λόγω του ότι αυτόματα καλείται και ο constructor της κλάσης, μπορούμε να καλέσουμε την μέθοδο setup() ώστε να ρυθμίσουμε κάποιες χρήσιμες λειτουργίες που επιβάλλεται να έχουν τα συλλεκτικά πακέτα που βρίσκονται στο χώρο. Τμήμα της μεθόδου αυτής ακολουθεί.

```

setup(){
    let player = this.player;
    if(this.name == "healthPack"){
        this.mesh.onDisposeObservable.add(function(){
            player.healthUp();
        });
    }

    if(this.name == "energyPack"){
        this.mesh.onDisposeObservable.add(function(){
            player.energyUp();
        });
    }

    if(this.name.search("Ammo") ){
        let name = this.name
        this.mesh.onDisposeObservable.add(function(){
            player.ammoUp(name);
        });
    }
}

```

Κώδικας 21: Γραμμές 22 - 45 : Αρχική ρύθμιση βασικών λειτουργιών συλλεκτικών πακέτων

Όπως φαίνεται παραπάνω, με τη συνάρτηση `onDisposeObservable.add(...)`, μπορούμε να προσθέσουμε διάφορες ενέργειες στη ουρά των καθηκόντων που θα ενεργοποιηθούν μόλις ανιχνευθεί πως το συγκεκριμένο περιστρεφόμενο έχει αφαιρεθεί από τη πίστα. Στη συγκεκριμένη περίπτωση με τη καταστροφή του αντικειμένου καλούνται οι κατάλληλες μέθοδοι για αύξηση των πόντων ζωής (`healthUp`), πόντων ενέργειας(`energyUp`) και ποσότητας πυρομαχικών(`ammoUp`) οι οποίες είναι ενσωματωμένες στη κλάση του παίκτη (`player`). Τέλος η μέθοδος που χειρίζεται την αδιάκοπη περιστροφή των αντικειμένων συναντάται στο ίδιο αρχείο.

Δ2.6 Αρχεία κλάσεων εχθρών

Τα αρχεία `enemy`, `zombie`, `dragon` και `skeleton.js` αποτελούν της κλάσης που συντελούν στη δημιουργία των αντικειμένων των 4 διαφορετικών εχθρών που συναντώνται στο ολοκληρωμένο τρισδιάστατο περιβάλλον του παιχνιδιού. Οι λειτουργίες των κλάσεων αυτών ακολουθούν παρόμοια δομή και διαφέρουν μόνο στα στοιχεία πόντων ζημιάς, ταχύτητας κάλυψης απόστασης καθώς κινούνται και στους πόντους ζωής.

Αρχικά σε όλα τα αρχεία και για όλους τους τύπους εχθρών ο `constructor` της κλάσης ορίζει το όνομα, το τρισδιάστατο μοντέλο που θα χρησιμοποιηθεί και τους πόντους ζωής του εχθρού. Έπειτα καλείται κατά τη δημιουργία του αντικειμένου η μέθοδος `setup` με την οποία καθορίζονται τα `animations` αδράνειας, κίνησης και επίθεσης στον παίκτη όπου αυτά υπάρχουν. Τμήμα της μεθόδου αυτής ακολουθεί.

```
for(let i = 0; i < scene.animationGroups.length; i++){
  var targetMesh =
  scene.animationGroups[i]._targetedAnimations[0].target.parent.parent.name;
  if(targetMesh == name){
    animations.push(scene.animationGroups[i]);
  }
}

for(let i = 0; i < animations.length ; i++){
  if(animations[i].name == "ZombieIdle_Zombie.001"){
    this.animationIdle = animations[i];
  }
  else if(animations[i].name == "ZombieWalk_Zombie.001"){
    this.animationAttack = animations[i];
  }
}
```

Κώδικας 22: Γραμμές 50-67 : Ανίχνευση και καθορισμός animation εχθρού

Με τη χρήση της μεταβλητής `targetMesh` ορίζεται το όνομα του μοντέλου του οποίου τα `animation` προσπαθούμε να εντοπίσουμε μέσα στη γενικότερη λίστα που αποθηκεύει τα `animations` της

σκηνής του κόσμου. Από τη στιγμή που τα ανιχνεύουμε, ανάλογα με το όνομα τους εισάγονται στις κατάλληλες μεταβλητές ιδιοτήτων οι οποίες με τη σειρά τους θα χρησιμοποιηθούν στην μέθοδο `move()`, τμήμα της οποίας ακολουθεί.

```
let initVec = mesh.position.clone();
let distVec = BABYLON.Vector3.Distance(camera.position, mesh.position);

let targetVec = camera.position.subtract(initVec);
let targetVecNorm = BABYLON.Vector3.Normalize(targetVec);

// Enemy always faces the player
mesh.lookAt(camera.position, Math.PI);
// Move enemy towards the player and stops slightly ahead
if(distVec < 40 && distVec >=5){
    distVec -= 0.03;
    mesh.translate(targetVecNorm, 0.03, BABYLON.Space.WORLD);
    mesh.position.y = 0;
    animationAttack.start();
}

// Idle animation play distance
if(distVec >= 40){
    animationIdle.start();
}

// Attack animation play distance
if(distVec < 5){
    animationAttack.start();
    player.healthDown(0.6);
}
```

Κώδικας 23: Γραμμές 85 - 110 : Υλοποίηση κίνησης εχθρού

Στο τμήμα αυτό φαίνεται η χρήση της συνάρτησης `lookAt` στην οποία τροφοδοτούμε τη τοποθεσία της κάμερας, που ουσιαστικά είναι η ίδια με αυτής του παίκτη, καθώς και τη γωνία θέασης. Με αυτόν τον τρόπο καταφέρνουμε να κάνουμε πάντα το μοντέλο του εχθρού να είναι στραμμένο προς τον παίκτη.

Επίσης κάνοντας χρήση της συνάρτησης `distance`, που παρέχει η κλάση `Vector3` του `BabylonJS` και διαχειρίζεται ζητήματα που αφορούν πράξεις με διανύσματα, μπορούμε να υπολογίσουμε την

απόσταση μεταξύ του παίκτη και του εχθρού. Η τιμή αυτή αποθηκεύεται στη μεταβλητή `distVec`. Έπειτα με μία συνθήκη ελέγχου, προσδιορίζουμε το αν η απόσταση αυτή είναι μικρότερη από τον αριθμό 40 γεγονός που ενεργοποιεί το animation κίνησης του εχθρού και τη μετατόπιση αυτού μέσω της συνάρτησης `translate` προς το μέρος του παίκτη. Στη περίπτωση που η απόσταση μεταξύ τους είναι μεγαλύτερη από 40 τότε ενεργοποιείται σε κατάσταση επανάληψης το animation αδράνειας, `idle` του εχθρού. Σε αυτό το σημείο αξίζει να τονιστεί πως ο εχθρός `enemy-skull` έχει μία επιπλέον συνθήκη επίθεσης η οποία είναι μοναδική σε αυτόν και διαπραγματεύεται τη διαδικασία εκτόξευσης βλημάτων προς τον παίκτη ανά τακτά χρονικά διαστήματα.

```
let scene = this.scene;
let name = this.name;
let mesh = this.mesh;
let timeNow = Date.now();

if(mesh.isEnabled()){
    // Checking if enemy's bullet is still in the scene
    if(scene.getMeshByName("Bullet"+name) == null)
    {
        // Enemy shoots every nth second
        let timePassed = (timeNow - this.timeThen) / 1000;
        let firingRate = 1.5;
        if( timePassed >= firingRate){
            fireBullet(scene, mesh, name );
            this.timeThen = timeNow;
        }
    }
}
```

Κώδικας 24: Γραμμές 74 - 90 αρχείο `enemy` : Εκτόξευση βλημάτων

Όπως φαίνεται στο παραπάνω τμήμα κώδικα τη διαδικασία εκτόξευσης βλημάτων αυτή καθεαυτή διαχειρίζεται η βοηθητική συνάρτηση που υπάρχει στο αρχείο enemy. Η μορφή της παρουσιάζεται στο τμήμα κώδικα που ακολουθεί.

```
var bulletName = "Bullet" + name;
var bullet = BABYLON.MeshBuilder.CreateSphere(bulletName, { segments: 3, diameter: 0.3 },
scene);
bullet.isPickable = false;
// Name bullet , Bullet + rand number then get name via bullet.name;
bullet.position = mesh.getAbsolutePosition();
bullet.physicsImpostor = new BABYLON.PhysicsImpostor(bullet,
    BABYLON.PhysicsImpostor.SphereImpostor,
    { mass: 0.1, friction: 0.5, restitution: 0.3 },
    scene);

var dir = scene.activeCamera.position.subtract(mesh.getAbsolutePosition());
bullet.physicsImpostor.applyImpulse(dir.scale(0.5), mesh.getAbsolutePosition());
bullet.life = 0;

bullet.step = ()=>{
    bullet.life++;
    if(bullet.life> 20 && bullet.physicsImpostor){
        bullet.physicsImpostor.dispose();
        bullet.dispose();
    }
};
scene.onBeforeRenderObservable.add(bullet.step);
```

Κώδικας 25: Γραμμές 107 -132 : Υλοποίηση εκτόξευσης βλημάτων

Αρχικά δημιουργείται ένα σφαιρικό αντικείμενο με τη συνάρτηση MeshBuilder στο οποίο προστίθενται λειτουργίες physics μέσω της βιβλιοθήκης cannon.js. Έπειτα εκτοξεύουμε το κάθε βλήμα με συγκεκριμένη δύναμη και μέσω της συνάρτησης step το καταστρέφουμε μετά από ορισμένο, ή αν χτυπήσει τον εχθρό.

Ακόμη σε κάθε αρχείο εχθρού εκτός από τη μέθοδο που αναλαμβάνει το έργο της καταστροφής του εχθρού, παρέχεται και μία βοηθητική συνάρτηση η οποία χειρίζεται τη κατασκευή του στοιχείου γραφικού περιβάλλοντος το οποίο είναι υπεύθυνο για την απεικόνιση των πόντων ζωής του κάθε εχθρού. Το τμήμα κώδικα της συνάρτησης enemyHUD παρουσιάζεται παρακάτω.

```
// GUI
var plane = BABYLON.Mesh.CreatePlane("plane2", 2);
plane.parent = mesh;
plane.position.y = 1.3;
plane.position.z = -1.65;
var advancedTexture = new GUI.AdvancedDynamicTexture.CreateForMesh(plane, 600, 600);
var healthBar = new GUI.Rectangle("enemyHealthBar2");
advancedTexture.addControl(healthBar);
healthBar.width = 10;
healthBar.height = "20px";
healthBar.cornerRadius = 20;
healthBar.color = "white";
healthBar.thickness = 2;
healthBar.background = "red";

return healthBar;
```

Κώδικας 26: Γραμμές 130-150 : Heads Up Display πόντων ζωής εχθρού

Δ3.1 Σχεδιασμός επιπέδου με το Blender

Η χρήση του Blender ήταν περιορισμένη κατά τα στάδια της ανάπτυξης του βιντεοπαιχνιδιού μιας και τα μοντέλα ήταν προκατασκευασμένα και απαίτησαν μικρές αλλαγές όπως αυτές των χρωμάτων και η αλλαγή της ονομασίας τους για την ομαλή επεξεργασία και φόρτωση τους στον τρισδιάστατο περιβάλλοντα κόσμου του παιχνιδιού.

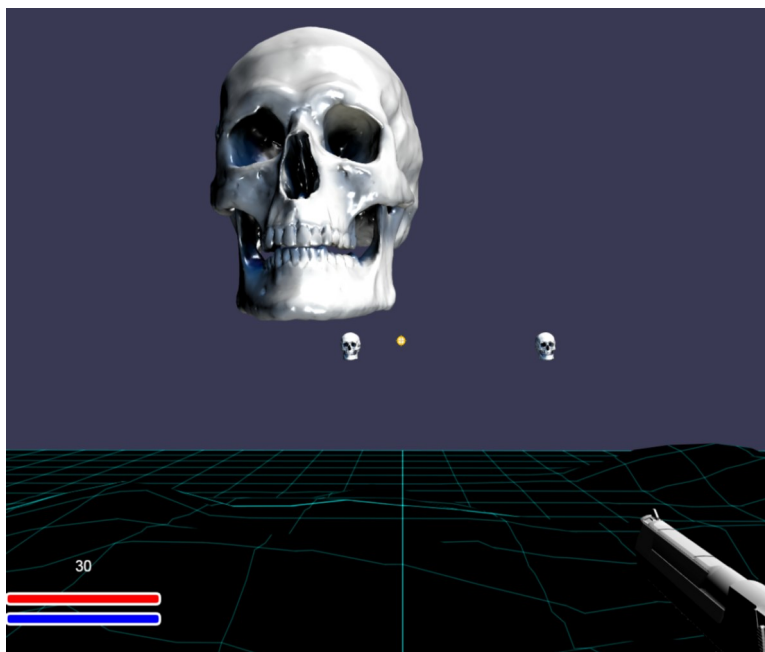
Παρ' όλα αυτά το Blender πήρε τη θέση του Level Editor για τη σχεδίαση του χώρου καθώς χρησιμοποιήθηκαν τα απλά εργαλεία που προσφέρει για μετατόπιση, περιστροφή αλλά και αλλαγή του μεγέθους ενός μοντέλου. Επιπρόσθετα η λειτουργία ομαδοποίησης μοντέλων υπό έναν κόμβο / αόρατο μοντέλο, αποτέλεσαν ιδιαίτερα χρήσιμες μίας και λόγω του μεγάλου αριθμού μοντέλων ήταν δύσκολη η φόρτωση τους ένα προς ένα από τις μεθόδους που παρέχει το BabylonJS.

Κεφάλαιο Ε΄

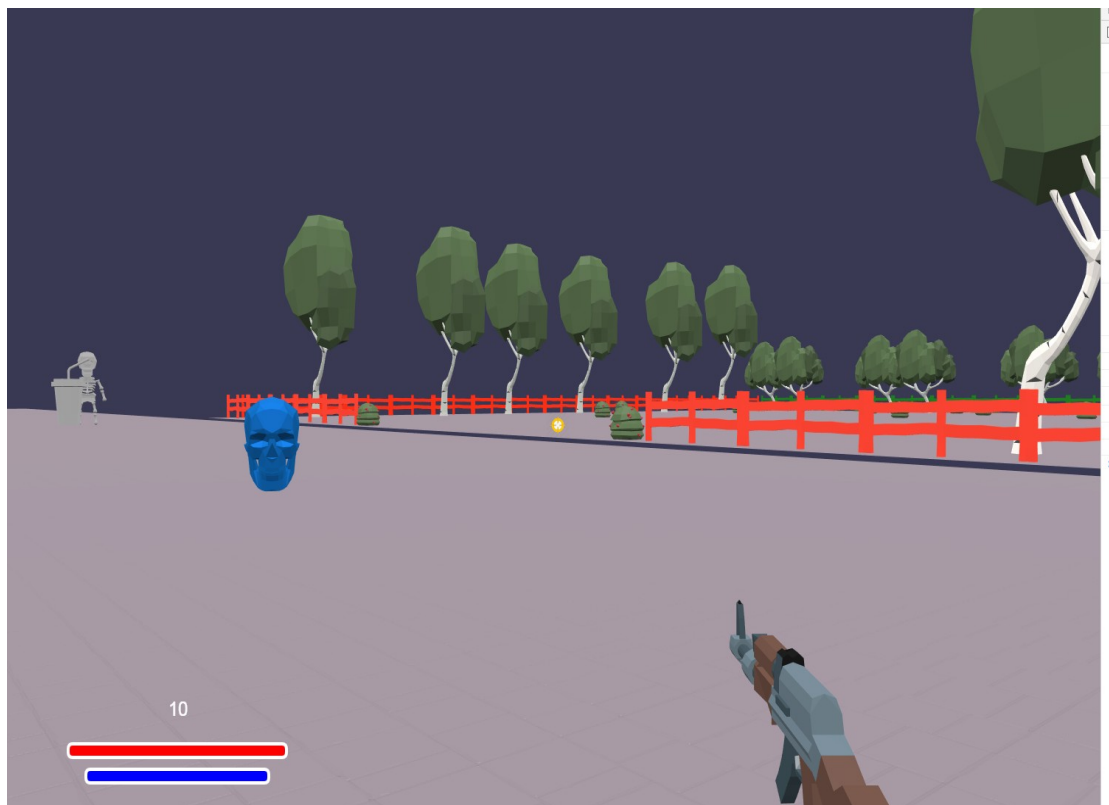
Στο κεφάλαιο αυτό θα παρουσιαστούν εικόνες από διάφορα στάδια της ανάπτυξης του βιντεοπαιχνιδιού που δημιουργήθηκε για τις απαιτήσεις της παρούσας πτυχιακής εργασίας. Με αυτό το τρόπο ο αναγνώστης θα έχει μία καλή ιδέα για τις τεχνικές white boxing, κατά τις οποίες στα αρχικά στάδια σχεδιασμού ενός βιντεοπαιχνιδιού χρησιμοποιούνται τρισδιάστατα μοντέλα απλής μορφής που σκοπό έχουν απλά να επιβεβαιώσουν πως οι λειτουργίες που προγραμματίστηκαν λειτουργούν ομαλά. Επιπρόσθετα θα αναφερθούν και μερικές από τις δυσκολίες, οι οποίες συναντήθηκαν κατά την δημιουργία του τελικού εκτελέσιμου καθώς και ιδέες για μελλοντικές βελτιώσεις και συμβουλές για επίδοξους προγραμματιστές που θα ήθελαν να ακολουθήσουν ένα παρόμοιο εγχείρημα.

Ε.1 Στιγμιότυπα ανάπτυξης

Σε αυτή την ενότητα θα παρουσιαστούν κάποιες εικόνες-στιγμιότυπα (screenshots) από τα αρχικά στάδια του τρισδιάστατου εικονικού περιβάλλοντος που δημιουργήθηκε καθώς και εικόνες από την τελική μορφή του και τη διαδικασία σχεδιασμού επιπέδου μέσα από το περιβάλλον του Blender.



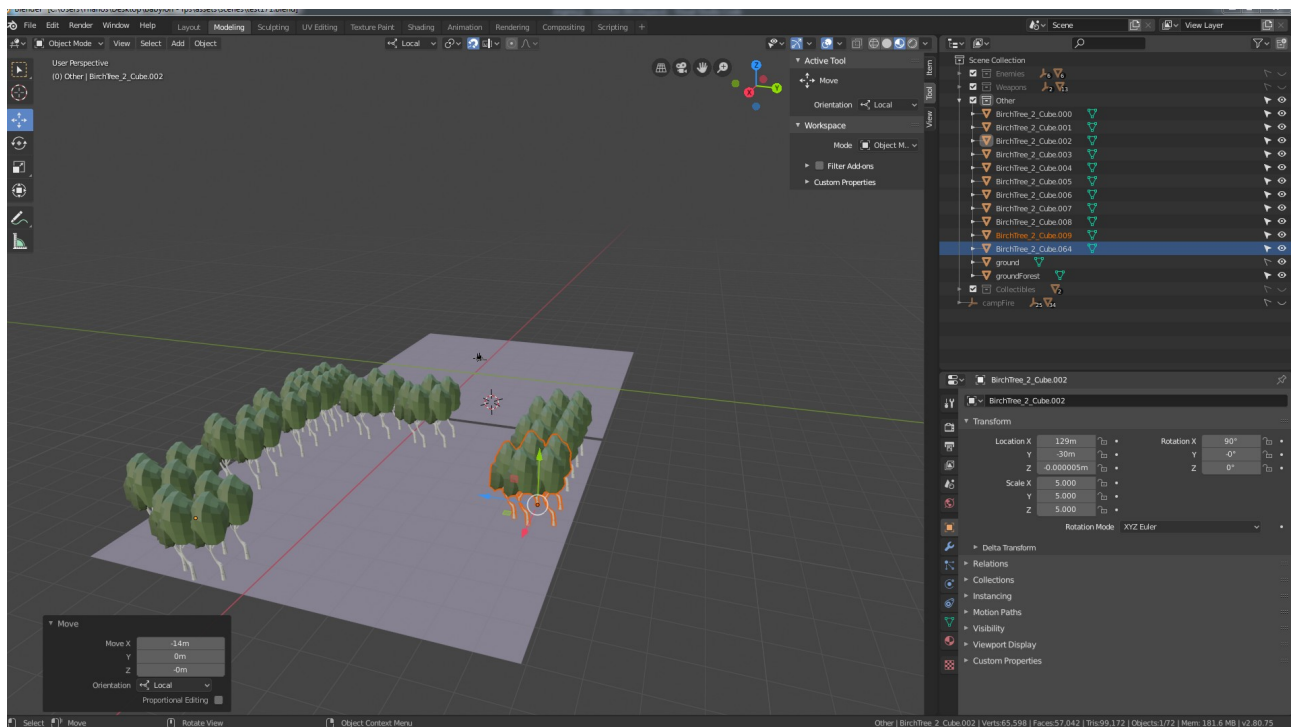
Screenshot 1 Πρώτη επαφή με το πλαίσιο BabylonJS



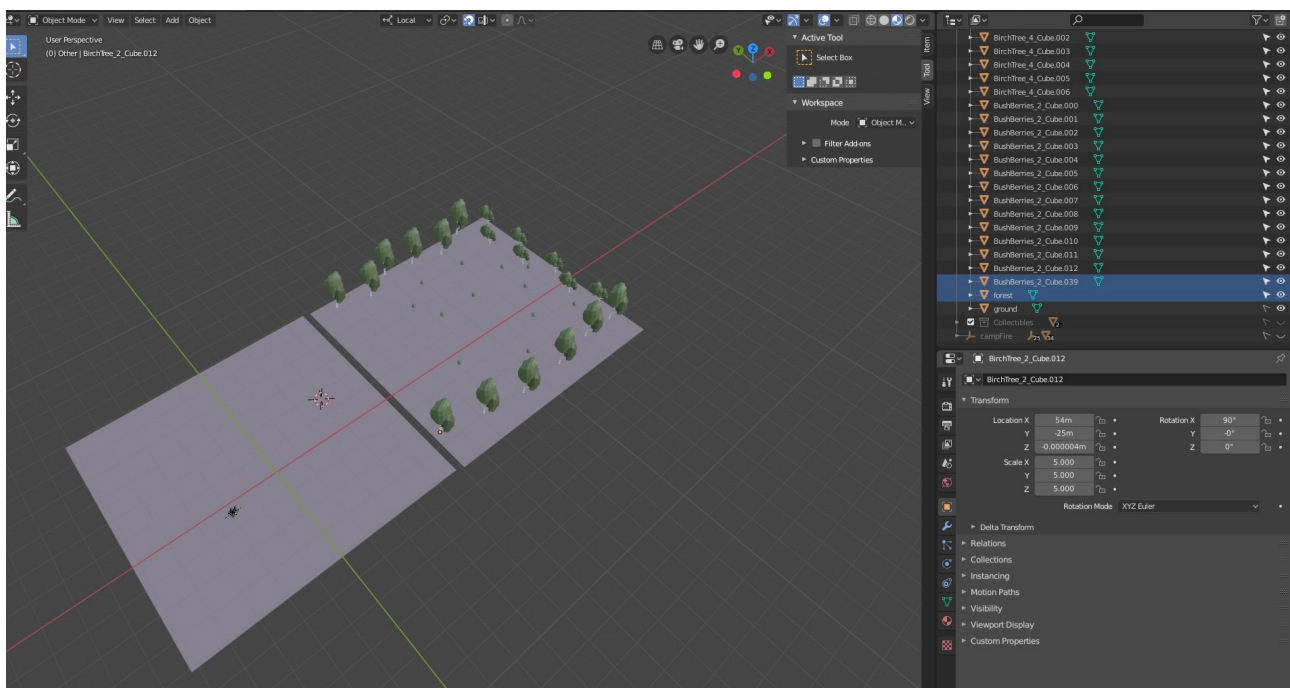
Screenshot 2 Τοποθέτηση αρχικών οντοτήτων περιβάλλοντος



Screenshot 3 Προσθήκη λίμνης και ουρανού



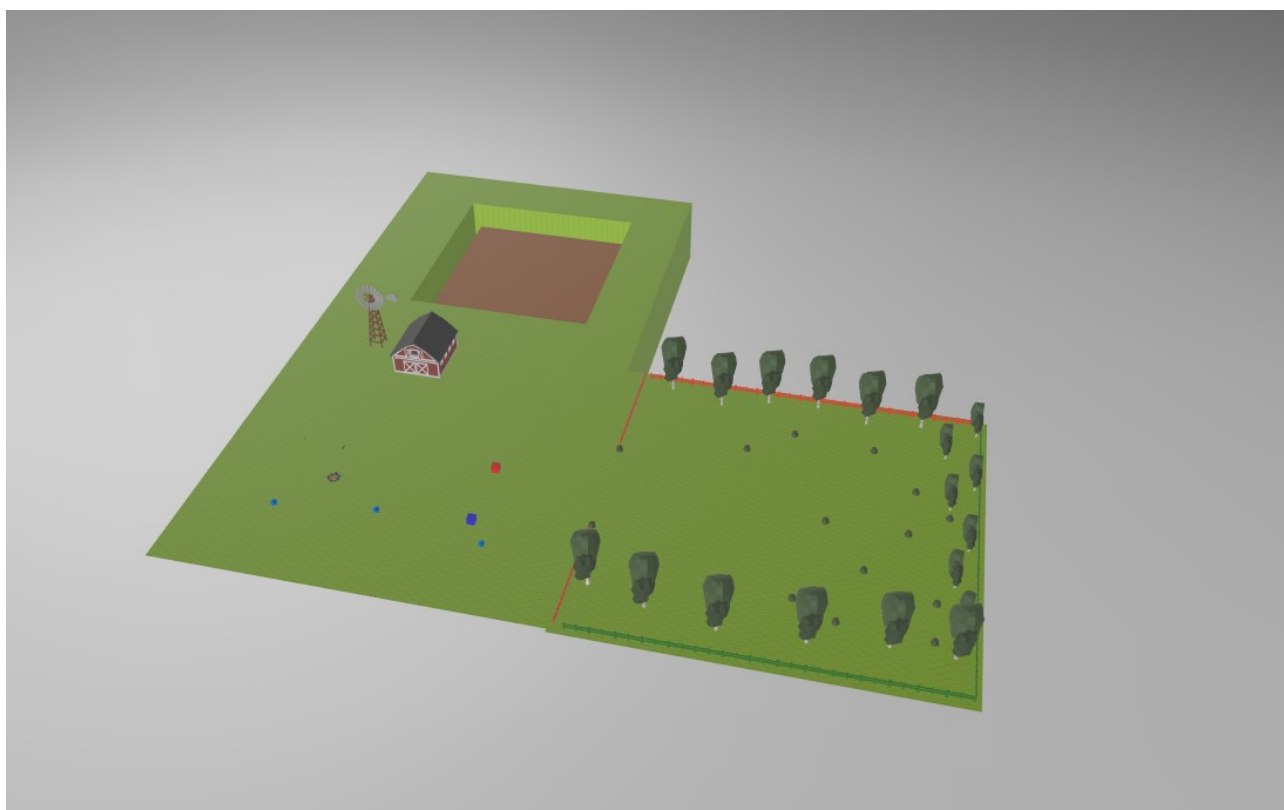
Screenshot 4: Δημιουργία δάσους



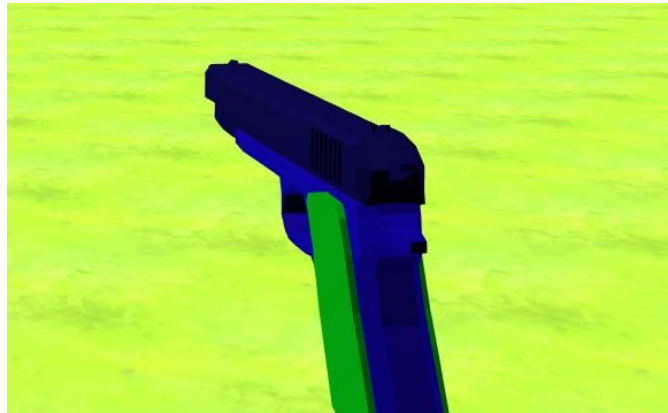
Screenshot 5: Δημιουργία δάσους 2



Screenshot 6: Στοιχείο φωτιάς



Screenshot 7: Σχεδιασμός κόσμου



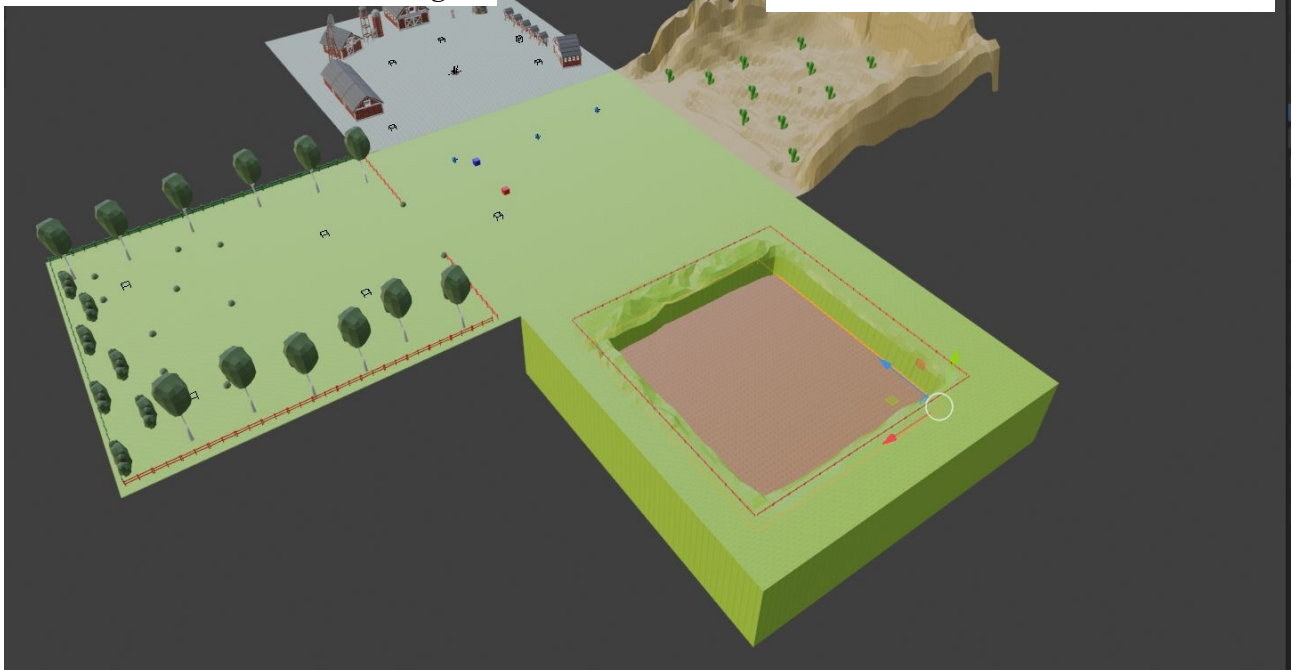
Screenshot 9: Μοντέλο όπλου pistol



Screenshot 11 Μοντέλο όπλου Shotgun



Screenshot 10 Μοντέλο όπλου AK



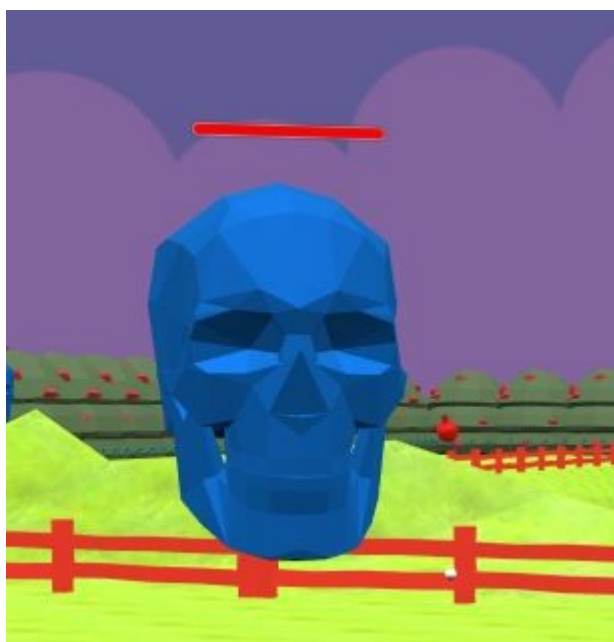
Screenshot 8: Οριστικοποίηση μορφής κόσμου



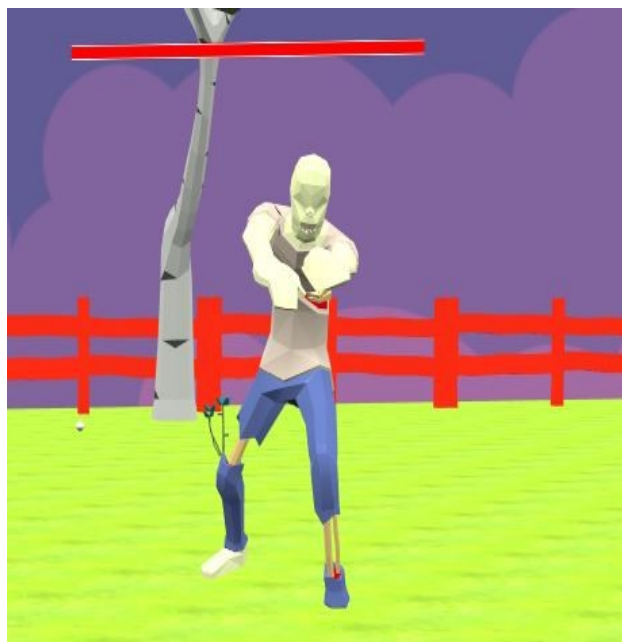
Screenshot 13 Μοντέλο όπλου Raygun



Screenshot 12: Μοντέλο όπλου lightning gun



Screenshot 14: Εχθρός Skull



Screenshot 15: Εχθρός Zombie



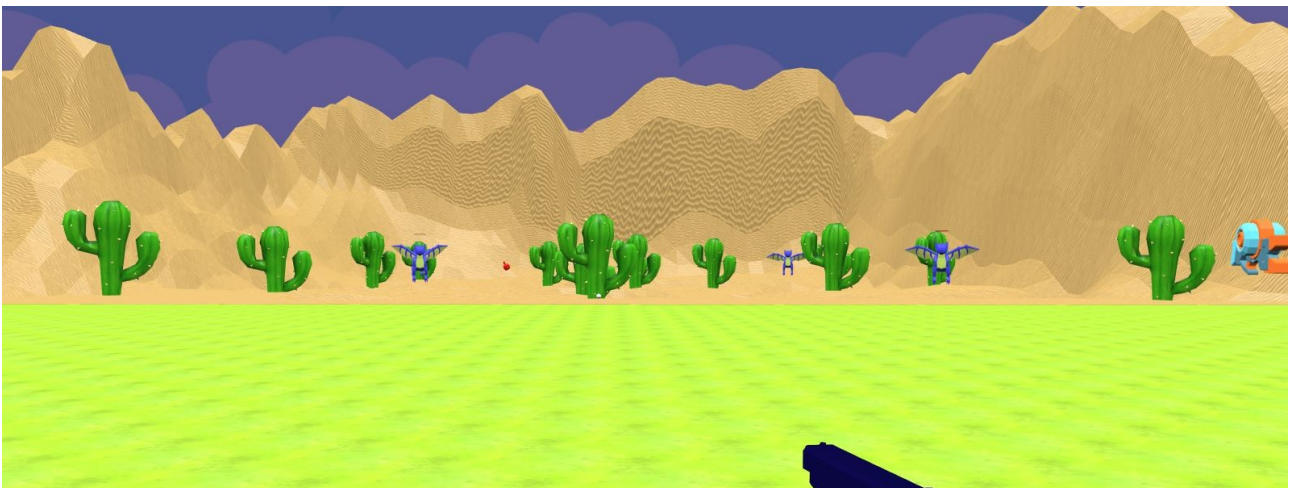
Screenshot 16: Εχθρός Dragon



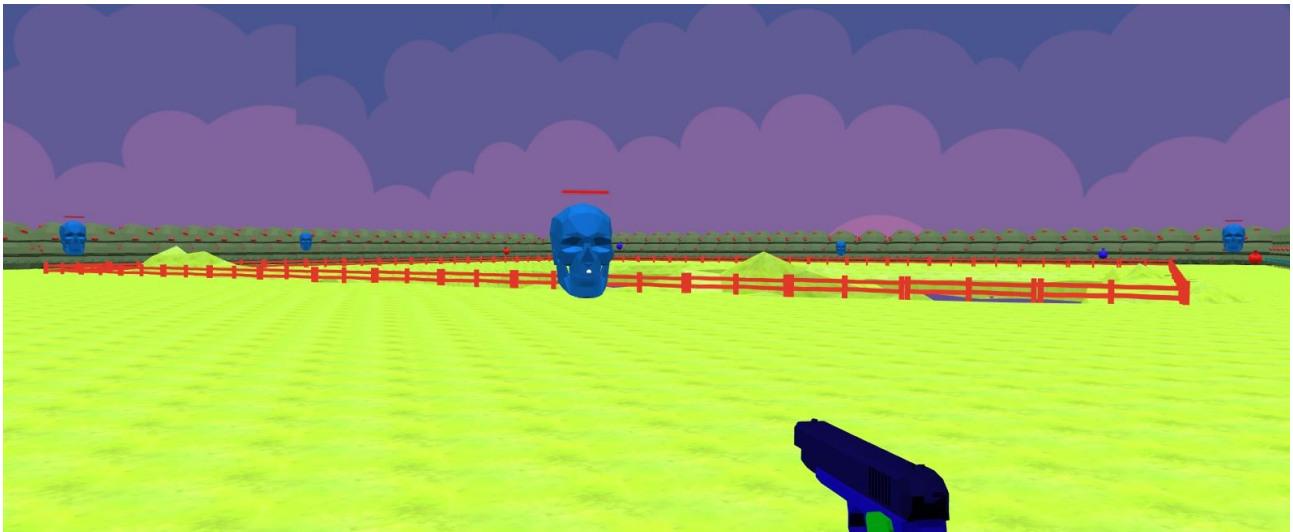
Screenshot 17: Εχθρός Skeleton



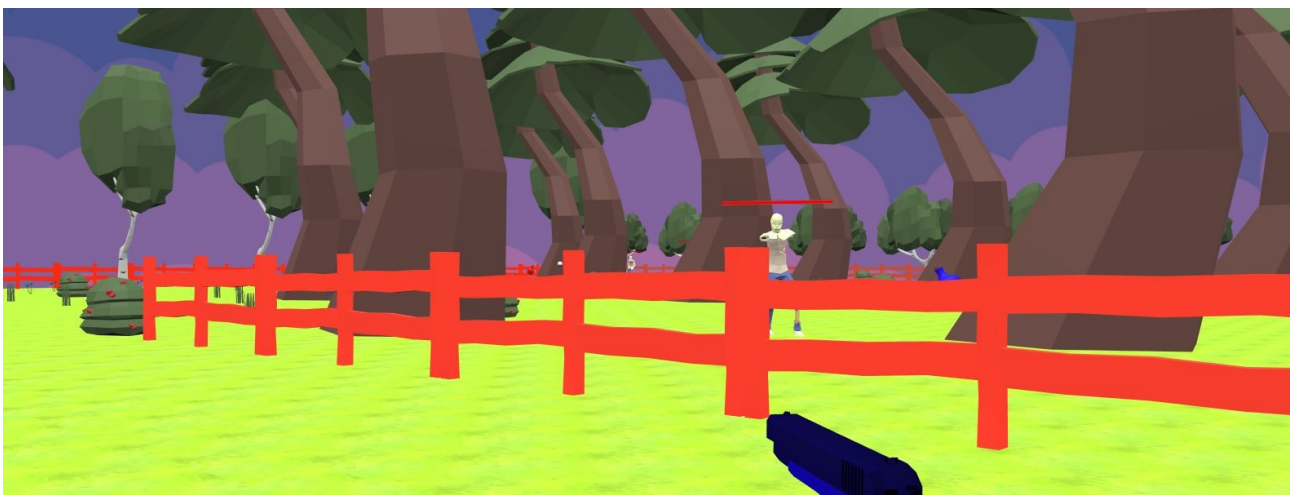
Screenshot 18: Κλειστή πύλη χωριού



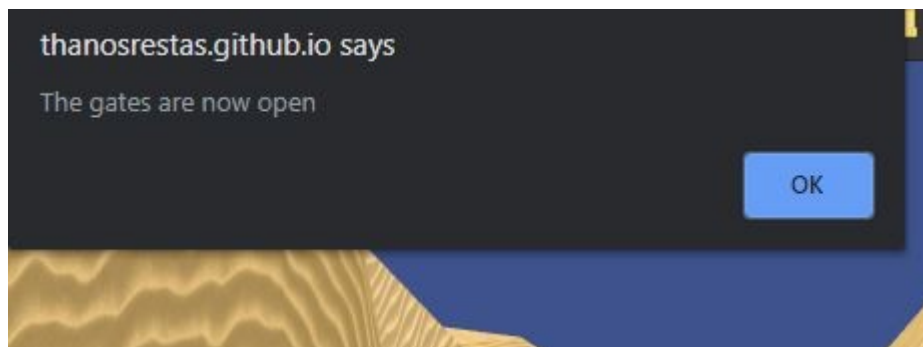
Screenshot 19: Έρημος



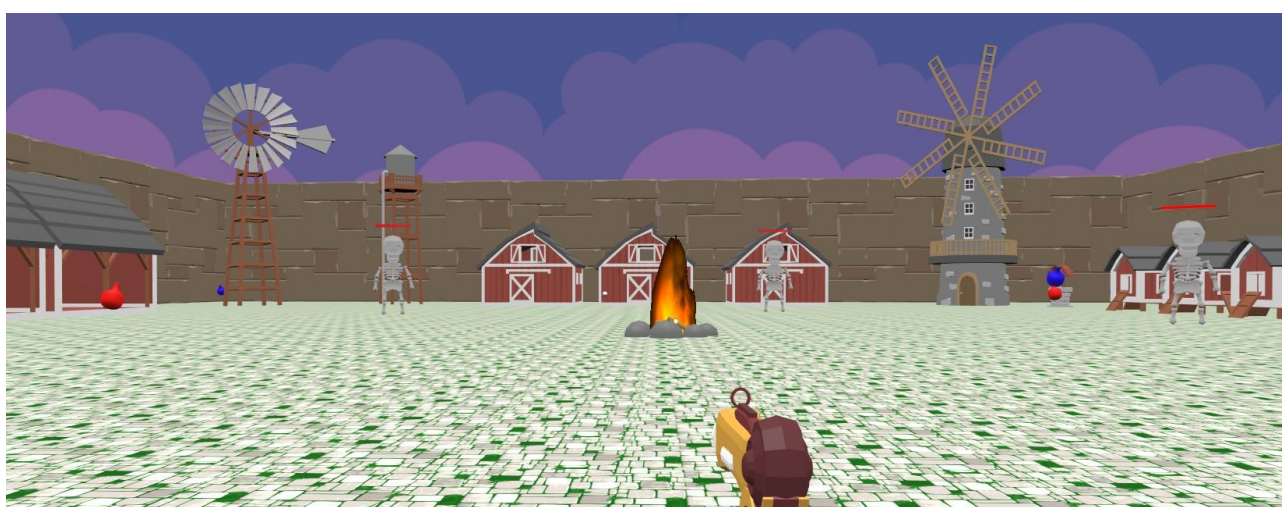
Screenshot 20: Λίμνη



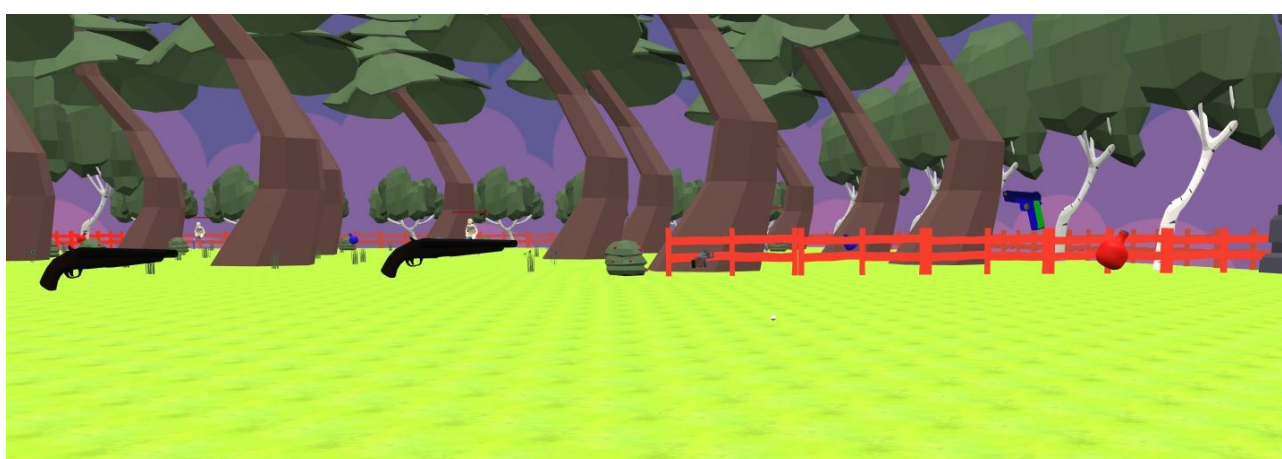
Screenshot 21: Δάσος



Screenshot 22: Μήνυμα ανοίγματος πόλης



Screenshot 23: Εσωτερικό χωριού



Screenshot 24: Περιστρεφόμενα πακέτα πυρομαχικών και ζωής

E.2 Μελλοντικές βελτιώσεις

Το εκτελέσιμο που αναπτύχθηκε για τις απαιτήσεις της παρούσας εργασίας αν και ολοκληρωμένο σε μεγάλο βαθμό ώστε να προσφέρει διαφορετικά ήδη εχθρών και όπλων αλλά και ένα τυπικό κύκλο παιχνιδιού με μία σαφώς καθορισμένη αποστολή για τον παίκτη, σίγουρα αφήνει περιθώρια για μελλοντικές βελτιώσεις.

Οι περισσότερες αλλαγές που θα μπορούσαν να πραγματοποιηθούν έχουν να κάνουν με τη δομή του κώδικα διότι στα τελευταία στάδια σχεδιασμού επιτεύχθηκε μεγαλύτερη εξοικείωση με τις δυνατότητες που παρέχει τόσο η Javascript όσο και το πλαίσιο προγραμματισμού εφαρμογών BabylonJS. Πιο συγκεκριμένα ο σαφέστερος καθορισμός των λειτουργιών που υλοποιεί το αρχείο πυρήνας engine.js όπως επίσης και η δημιουργία μίας γενικής κλάσης εχθρού από την οποία θα κληρονομούν ιδιότητες οι ξεχωριστές υποκατηγορίες εχθρών είναι μερικά από τα στοιχεία τα οποία θα μπορούσαν να βελτιώσουν την αναγνωσιμότητα και λειτουργικότητα του πηγαίου κώδικα.

Ακόμη η τεχνική φόρτωσης των διάφορων τρισδιάστατων μοντέλων στο κόσμο του παιχνιδιού θα μπορούσε να αυτοματοποιηθεί δημιουργώντας μία ολόκληρη κλάση γύρω από τις λειτουργίες της μεθόδου assetManager του BabylonJS, η οποία θα αντικαθιστούσε το υπάρχον σύστημα με τις επαναλαμβανόμενες κλήσεις συναρτήσεων.

Τέλος μία ακόμα μελλοντική βελτίωση είναι η μεταφορά του εκτελέσιμου βιντεοπαιχνιδιού σε άλλες πλατφόρμες όπως έξυπνα κινητά γεγονός που θα αξιοποιούσε τις δυνατότητες του οικοσυστήματος λειτουργιών, modules και βιβλιοθηκών που προσφέρει η γλώσσα Javascript και οι διάφοροι σύγχρονοι φυλλομετρητές. Το εγχείρημα αυτό θα μπορούσε να πραγματοποιηθεί με τη χρήση του πλαισίου προγραμματισμού εφαρμογών Electron, το οποίο συντηρείται από την εταιρεία GitHub. Με αυτό το τρόπο επιτυγχάνεται η προσομοίωση ενός μικρού σε μέγεθος φυλλομετρητή όπως ο Chrome για τη δημιουργία ενός περιβάλλοντος με το οποίο μπορεί κανείς να φτιάξει εφαρμογές πολλών πλατφορμών με τη γλώσσα Javascript. Έτσι μπορεί κανείς να πετύχει την εύκολη και γρήγορη υποστήριξη πλατφορμών web, desktop και smartphone χωρίς να χρειαστεί να ξανασχεδιάσει από το μηδέν τον πηγαίο κώδικα της εφαρμογής.

E.3 Συμπεράσματα

Στα πλαίσια της εργασίας αυτής παρουσιάσαμε τον σχεδιασμό και την ανάπτυξη ενός ολοκληρωμένου τρισδιάστατου περιβάλλοντος μέσα από τη δημιουργία ενός βιντεοπαιχνιδιού πρώτου προσώπου το οποίο αξιοποιεί τις δυνατότητες των WebGL και BabylonJS μέσα από τη γλώσσα προγραμματισμού Javascript.

Η προσπάθεια αυτή είχε ως αποτέλεσμα να αποσαφηνιστούν τα στάδια σχεδιασμού ενός πολύπλοκου λογισμικού όπως αυτό που ανήκει στον τομέα των γραφικών υπολογιστών και των ψηφιακών κόσμων. Πιο συγκεκριμένα τονίστηκε πως ο προσεκτικός σχεδιασμός των βασικών συστημάτων κίνησης, φόρτωσης αρχείων και διασύνδεσης των διάφορων οντοτήτων ενός εικονικού περιβάλλοντος, είναι αντικείμενα υψίστης σημασίας και αποτελούν έργο δύσκολο για έναν μόνο προγραμματιστή ακόμα και στη περίπτωση της μικρής κλίμακας του εκτελέσιμου που παρουσιάστηκε. Το γεγονός αυτό συμβαίνει διότι τα συστήματα αυτά είναι που διευκολύνουν τη μετέπειτα προσθήκη λειτουργιών και δυνατοτήτων τα οποία με τη σειρά τους θα ολοκληρώσουν την απολαυστική διαδραστική εικονική εμπειρία που καλείται να βιώσει ο παίκτης.

Τέλος παρ όλες τις δυσκολίες που συναντήθηκαν σε όλα τα στάδια ανάπτυξης, αποδείχθηκε πως με τη σωστή επιλογή σύγχρονων εργαλείων και τη μελέτη των βασικών τεχνικών ανάπτυξης ενός βιντεοπαιχνιδιού που χρησιμοποιούνται από τις αρχές του τομέα ως και σήμερα, μπορεί κάποιος ξεκινώντας από το μηδέν να δημιουργήσει ένα ικανοποιητικής κλίμακας ολοκληρωμένο τρισδιάστατο περιβάλλον.

Βιβλιογραφία

Γραφικά και Εικονική Πραγματικότητα, Κ.Μουστάκας, Ι.Παλιόκας, Α.Τσακίρης, Δ. Τζοβάρας, 2015

Game Engine Architecture, Jason Gregory 2018

Real Time Rendering, Tomas Akenine -Moller, Eric Haines, Naty Hoffman, Angelo Pesce, Michael Iwanicki, Sebastien Hillaire, 2018

WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL, Kouichi Matsuda, 2013

Learning BabylonJS, Julian Chenard, 2017

Professional Javascript for Web Developers, Nickolas C. Zakas, 2012

ΠΗΓΕΣ

BabylonJS: <https://www.babylonjs.com/>

Blender: <https://www.blender.org/>

Τρισδιάστατα μοντέλα : <https://quaternius.itch.io/>

Αποθετήριο ολοκληρωμένου πηγαίου κώδικα εργασίας : <https://github.com/ThanosRestas/Theasis>

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

SpaceWar, το πρώτο βιντεοπαιχνίδι.....	3
Sketchpad, το πρώτο πρόγραμμα σχεδίασης.....	3
Το ηλεκτρονικό ποντίκι.....	4
Ο αλγόριθμος σχεδίασης ευθείας γραμμής.....	4
Παρουσίαση σχεδίασης γραμμής.....	5
Pong, το πρώτο βιντεοπαιχνίδι σε Raster Displays.....	5
Apple 2 : Ο πρώτος προσωπικός υπολογιστής με γραφικό περιβάλλον.....	5
Luxor Jr. Pixar 1986, διάρκεια 2 λεπτών.....	6
Χρήση ψηφιακών γραφικών στη ταινία Tron.....	6
Jurassic Park 1993, απόσπασμα που παρουσιάζει τη χρήση Computer Generated Imagery.....	6
AutoCAD 3D, το περιβάλλον της πρώτης τρισδιάστατης έκδοσης.....	7
Toy Story, Pixar 1995, ταινία εξ ολοκλήρου φτιαγμένη με Computer Generated Imagery.....	8
Crash Bandicoot, Playstation 1 1996.....	8
Star Fox, Super Nintendo Entertainment System 1993.....	8
Final Fantasy, Square Enix.....	9
Doom 3, νέες ικανότητες φωτισμού με τη δύναμη της ID Tech 4.....	10
Το σύμβολο της εταιρείας Nvidia.....	11
Το σύμβολο της ATI.....	11
Μοντέλο της GeForce 3.....	11
Παράδειγμα εφαρμογής προγράμματος σκίασης.....	12
Ρεαλιστικό κινούμενο εφέ φωτιάς.....	12
Τεχνικές αλλοίωσης μορφής μοντέλου.....	12
Παρακολούθηση μέλους σώματος και αναπαράσταση αυτού σε ψηφιακό περιβάλλον σε πραγματικό χρόνο.....	13
Κίνηση νερού με βάση τη τοπολογία του χώρου.....	13
Ρεαλιστική απεικόνιση τριχών.....	13
Ανακατασκευή οικιακών συγκροτημάτων μέσω data mining σε πραγματικό χρόνο.....	14
Ρεαλιστική κίνηση ζώων σε εικονικό χώρο.....	14
Προσομοίωση ρεαλιστικής μορφολογίας και κίνησης βλεφάρων.....	14
Μιμητικό μοντέλο αναρριχητικών ικανοτήτων ανθρώπινου σώματος.....	15
Προσομοίωση ελαστικών ιδιοτήτων διαφόρων υλικών.....	15
Δυναμική διάσπαση μοντέλων διάφορων υλικών.....	16
Παραγωγή κόσμων με έμφαση στη ρεαλιστική και προοδευτική απεικόνιση και εξάπλωση ενός οικοσυστήματος.....	16
Περιήγηση σε αρχαιολογικό χώρο με τη χρήση εξοπλισμού εικονικής πραγματικότητας.....	17
Βιντεοπαιχνίδι πρώτο προσώπου με τρόπο παιχνιδιού εικονικής πραγματικότητας.....	18
Το συνεχές της μεικτής πραγματικότητας.....	18
Χειριστήριο για χρήση σε περιβάλλοντα εικονικής πραγματικότητας.....	20
Συσκευή κεφαλής Headset.....	20
Προβολείς μικτής πραγματικότητας.....	20
Συσκευή κεφαλής μικτής πραγματικότητας.....	20

ΕΥΡΕΤΗΡΙΟ ΤΜΗΜΑΤΩΝ ΚΩΔΙΚΑ

Δέσμευση στοιχείου καμβά ιστοσελίδας.....	67
Παρουσίαση δημιουργίας σκηνής με ένα παραλληλόγραμμο.....	72
Φόρτωση τρισδιάστατου μοντέλου που σχεδιάστηκε με εξωτερικό πρόγραμμα.....	73
Scene Picking, επιλογή στοιχείων στον τρισδιάστατο χώρο μέσω του κλικ ποντικιού.....	74
Αρχείο index.js : εκκίνηση βασικών λειτουργιών και εκτελέσιμου.....	84
Βασικά στοιχεία σκηνής και λίστες εχθρών και πακέτων ενέργειας και ζωής.....	85
Ενεργοποίηση συστήματος παραγωγής σωματιδίων.....	85
Γραμμές 149-163 : Ενεργοποίηση ανίχνευσης συγκρούσεων σε στοιχεία του χώρου.....	86
Γραμμές 336 – 346 : Αναζήτηση εντοπισμένου εχθρού μέσα από τη λίστα εχθρών.....	87
Γραμμές 378 – 393 : Δημιουργία μπάρας πληροφοριών πόντων ζωής.....	88
Γραμμές 420 – 425 : Κύκλος απόδοσης γραφικών.....	88
Γραμμές 7 - 25 : Αρχικοποίηση στοιχείων κεντρικού χαρακτήρα - παίκτη.....	89
Γραμμές 52-70 : Μέθοδοι αύξησης και μείωσης πόντων ενέργειας.....	90
Γραμμές 134 – 141 : Καθορισμός πλήκτρων περιήγησης στο χώρο.....	91
Γραμμές 265 - 277 : Επιλογή κατάλληλου όπλου με βάση τα πλήκτρα.....	91
Γραμμές 334 – 343 : Έλεγχος κατάστασης χαρακτήρα.....	92
Γραμμές 5 - 11 : Αρχικοποίηση ιδιοτήτων όπλου.....	93
Γραμμές 67 - 74 : Ρύθμιση στοιχείων animation όπλου AK47.....	94
Γραμμές 241 – 261 : Δημιουργία οπτικού εφέ λείζερ.....	94
Γραμμές 63 - 78 : Μέθοδος καταστροφής περιστρεφόμενου συλλεκτικού πακέτου.....	95
Γραμμές 22 - 45 : Αρχική ρύθμιση βασικών λειτουργιών συλλεκτικών πακέτων.....	96
Γραμμές 50-67 : Ανίχνευση και καθορισμός animation εχθρού.....	97
Γραμμές 85 - 110 : Υλοποίηση κίνησης εχθρού.....	98
Γραμμές 74 - 90 αρχείο enemy : Εκτόξευση βλημάτων.....	99
Γραμμές 107 -132 : Υλοποίηση εκτόξευσης βλημάτων.....	100
Γραμμές 130-150 : Heads Up Display πόντων ζωής εχθρού.....	101

ΕΥΡΕΤΗΡΙΟ ΣΤΙΓΜΙΟΤΥΠΩΝ ΕΚΤΕΛΕΣΙΜΟΥ

Πρώτη επαφή με το πλαίσιο BabylonJS.....	102
Τοποθέτηση αρχικών οντοτήτων περιβάλλοντος.....	103
Προσθήκη λίμνης και ουρανού.....	103
Δημιουργία δάσους.....	104
Δημιουργία δάσους 2.....	104
Στοιχείο φωτιάς.....	105
Σχεδιασμός κόσμου.....	106
Οριστικοποίηση μορφής κόσμου.....	106
Μοντέλο όπλου pistol.....	107
Μοντέλο όπλου AK.....	107
Μοντέλο όπλου Shotgun.....	107
Μοντέλο όπλου lightning gun.....	107
Μοντέλο όπλου Raygun.....	107
Εχθρός Skull.....	108
Εχθρός Zombie.....	108
Εχθρός Dragon.....	108
Εχθρός Skeleton.....	108
Κλειστή πύλη χωριού.....	109
Έρημος.....	109
Λίμνη.....	110
Δάσος.....	110
Μήνυμα ανοίγματος πύλης.....	111
Εσωτερικό χωριού.....	111
Περιστρεφόμενα πακέτα πυρομαχικών και ζωής.....	111