Ανώτατο Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Πτυχιακή εργασία

«ΕΙΚΟΝΙΚΟΣ ΧΩΡΟΣ ΚΑΙ ΕΚΠΑΙΔΕΥΣΗ»

Δαμιανάκη Κωνσταντίνα

Επιβλέπων καθηγητής : Μαλάμος Αθανάσιος

Ηράκλειο - Νοέμβριος 2010

Ένα μεγάλο ευχαριστώ στην οικογένεια μου, για την υποστήριξη που μου προσέφερε απλόχερα, όλα αυτά τα χρόνια.

Synopsis

Over the past few years, new technologies, in the area of telecommunication systems, have achieved significant high data rates and managed to cover remote areas, as well. Today, each house has a direct and quick access to the internet.

The increased user demand, for fast and quick entertainment via the Internet, made many companies, to shift into the field of videogames production. Today, we can discover a huge range of all kinds of games, from the simplest one, for example puzzles, to more complex ones, such as massive 3d online multiplayer games. Furthermore, the industry of console and computer videogames, has experienced great prosperity, so great, that now is considered to be one of the most profitable. The present diploma paper focuses on the stages, of videogames development.

In the following chapters, are described thoroughly the whole process of designing and developing. The first section, covers the steps and choices that we have to make in order to accomplish a good start. In the second section, is given a tutorial on how to make a simple three-dimensional videogame, using two specialized applications. First we design 3d models and a 3d world that includes them. Then we add extra features, such as texture and lighting, so that models would reach a close approximation of real-life objects. There is a variety of modeling applications available, but the dominate one in this area, is considered to be *Autodesk,* with *3d studio max* and *Maya*. In the tutorial, the version that was used was *3d studio max 8*. Secondly, comes the complex phase of programming. *Adobe*, providing, *director*, simplified the whole procedure. The programming of the objects, is written in *lingo,* a scripting-style language embedded in *director*. The latest version of *director* (11.5), offers an even greater variety of features for authors, and the possibility for more realistic model movements. The simulation of the real world and the forces that act on bodies, is possible with the addition of a physics engine, such as *Ageia Physics*.

As authors, it is important to give equal weight to both programming and graphic design, to complete an elegant, yet entertaining video game.

Περίληψη

Με την πάροδο των τελευταίων ετών, νέες τεχνολογίες τηλεπικοινωνιακών συστημάτων έχουν επιτύχει σημαντικά υψηλούς ρυθμούς μετάδοσης δεδομένων, αλλά έχουν ακόμα καταφέρει, και να καλύψουν απομακρυσμένες περιοχές. Έτσι κάθε σπίτι, πλέον, έχει άμεση και γρήγορη πρόσβαση στο διαδίκτυο.

Η αυξημένη ζήτηση των χρηστών, για εύκολη και γρήγορη ψυχαγωγία μέσω του διαδικτύου, έκανε πολλές εταιρίες να κινηθούν στον τομέα παραγωγής βιντεοπαιχνιδιών. Σήμερα, μπορεί κανείς ν' ανακαλύψει μια τεράστια γκάμα όλων των ειδών, από τα πιο απλά π.χ. τύπου puzzles, μέχρι και πιο πολυσύνθετα, όπως massive 3d online multiplayer. Παράλληλα, η βιομηχανία παραγωγής βιντεοπαιχνιδιών για κονσόλες και υπολογιστές, γνώρισε ιδιαίτερη άνθιση, σε τέτοιο βαθμό ώστε ν' αποτελεί σήμερα, μια από τις πιο προσοδοφόρες. Η παρούσα εργασία εστιάζει στα στάδια ανάπτυξης ηλεκτρονικών παιχνιδιών.

Στα επόμενα κεφάλαια περιγράφεται αναλυτικά όλη η διαδικασία σχεδιασμού και ανάπτυξης βιντεοπαιχνιδιών. Στην πρώτη ενότητα, γίνεται λόγος για τα βήματα που πρέπει ν' ακολουθήσουμε καθώς και τις επιλογές που έχουμε, για να πραγματώσουμε ένα σωστό ξεκίνημα. Στη δεύτερη ενότητα παρατίθεται οδηγός ανάπτυξης ενός απλού τρισδιάστατου, βιντεοπαιχνιδιού με εξειδικευμένες εφαρμογές. Σχεδιάζεται ο 3d κόσμος και τα μοντέλα, προστίθενται επιπλέον χαρακτηριστικά, όπως υφή και φωτισμός ώστε τα μοντέλα να προσεγγίζουν όσο το δυνατόν περισσότερο, τα πραγματικά. Υπάρχει μια πληθώρα εφαρμογών μοντελοποίησης,, κυρίαρχος όμως του χώρου θεωρείται η *Autodesk* με τα προγράμματα *3d studio max* και *Maya*. Για τον οδηγό, χρησιμοποιήθηκε η έκδοση *3d studio max 8*. Ακολουθεί το πολύπλοκο στάδιο του προγραμματισμού. Η *Adobe* με το πρόγραμμα *director*, απλούστευσε αρκετά την όλη αυτή διαδικασία. Ο προγραμματισμός των αντικειμένων, γίνεται με χρήση της γλώσσας *lingo* τύπου scripting, η οποία βρίσκεται ενσωματωμένη στο *director*. Η νέα έκδοση της εφαρμογής (11.5), έδωσε ακόμα μεγαλύτερη ποικιλία λειτουργιών στους δημιουργούς και την δυνατότητα ρεαλιστικότερης κίνησης. Τέλος, η προσομοίωση του πραγματικού κόσμου φυσικής, οι δυνάμεις που ασκούνται στα σώματα, είναι εφικτή με χρήση μηχανών φυσικής, όπως η *Ageia Physics*.

Ως δημιουργοί, είναι σημαντικό, να δώσουμε βάρος εξίσου τόσο στον προγραμματισμό όσο και στο σχεδιασμό γραφικών, για την ολοκλήρωση ενός καλαίσθητου και συνάμα ψυχαγωγικού βιντεοπαιχνιδιού.

Πίνακας περιεχομένων

Πίνακας εικόνων

Λίστα πινάκων

1. Εισαγωγή

1.1. Κίνητρο για τη διεξαγωγή της εργασίας

Ο σχεδιασμός ψηφιακών μοντέλων, αποτελεί ισχυρό κίνητρο για κάθε σύγχρονο καλλιτέχνη. Η ευκολία και οι πολλαπλές λειτουργίες που προσφέρουν, τα σημερινά πακέτα εφαρμογών, δίνουν τα μέσα σε κάθε ενδιαφερόμενο, να δημιουργήσει ότι σκεφτεί με μοναδικό περιορισμό, τη φαντασία του. Το ενδιαφέρον αυξάνεται, όταν προστίθεται η ανάπτυξη κώδικα η οποία επιτρέπει, στο δημιουργό, να δώσει «πνοή» στα μοντέλα του, κάνοντάς τα να κινούνται και ν' αλληλεπιδρούν σ' ένα εικονικό περιβάλλον. Σήμερα είναι πιο εύκολο από ποτέ να αναπτύξει κάποιος ένα βιντεοπαιχνίδι, λόγω της πληθώρας εργαλείων που υπάρχουν διαθέσιμα, και μάλιστα δωρεάν. Τεχνολογία και τεχνογνωσία υπάρχει άφθονη, τα μόνα που χρειάζονται είναι ένας καλός οδηγός για το ξεκίνημα και αρκετή όρεξη.

1.2. Σκοπός και στόχοι εργασίας

Σκοπός της εργασίας, είναι η ανάλυση του τρόπο σχεδίασης και ανάπτυξης βιντεοπαιχνιδιών. Σημαντικός στόχος της πτυχιακής, είναι ν' αποτελέσει ένα εμπεριστατωμένο οδηγό για άπειρους αλλά και πιο προχωρημένους σχεδιαστές/ προγραμματιστές που θέλουν ν' ασχοληθούν με αυτόν τον τομέα. Επιδιώκοντας να εμπλουτίσουμε τις γνώσεις που απαιτούνται, όσον αφορά τη λειτουργία των εφαρμογών, πραγματοποιήθηκε εκτενής έρευνα και μελέτη πολλών και διαφόρων παραδειγμάτων. Έτσι, συντάχθηκε ειδικός οδηγός με όλα τα απαραίτητα βήματα που χρειάζονται για την κατασκευή ενός απλού τρισδιάστατου παιχνιδιού.

1.3. Ανάλυση κεφαλαίων

Το κεφάλαιο δύο ξεκινά με τον ορισμό και την ιστορία των βιντεοπαιχνιδιών, περιγράφεται εν συντομία η μακρόχρονη ιστορία του κλάδου, και συγκρίνεται το χθες με το σήμερα. Ακόμα αναλύονται ένα προς ένα, όλα τα στάδια ανάπτυξης βιντεοπαιχνιδιών.

Στο κεφάλαιο τρία βλέπουμε όλες τις τεχνικές ψηφιακής σχεδίασης τρισδιάστατων μοντέλων, και το πώς εξελίχτηκε η τρισδιάστατη αναπαράσταση. Αναλύονται προσεκτικά όλα τα βήματα σχεδίασης και ολοκλήρωσης τρισδιάστατων μοντέλων.

Η δεύτερη ενότητα της πτυχιακής εργασίας ξεκινά με το κεφάλαιο τέσσερα, όπου περιγράφεται ο τρόπος με τον οποίο μπορούμε να αναπτύξουμε ένα απλό βιντεοπαιχνίδι. Γίνεται μια μικρή εισαγωγή στο *adobe Photoshop* και στον τρόπο επεξεργασίας εικόνων. Επιδεικνύεται ο τρόπος με τον οποίο χρησιμοποιούμε αυτές τις επεξεργασμένες εικόνες ως οδηγό για να κατασκευάσουμε, στο πρόγραμμα *3d studio max 8*, ένα τρισδιάστατο όχημα. Έπειτα, «ντύνουμε» το όχημα με χρώμα και υφή και ρυθμίζουμε κατάλληλα το φωτισμό. Ολοκληρώνοντας την ανάπτυξη του παιχνιδιού, μαθαίνουμε πώς να προγραμματίζουμε με *lingo*. Για να υπάρχει φυσική στον εικονικό κόσμο, χρησιμοποιούμε τη μηχανή φυσικής *Ageia physics* που βρίσκεται ως extra στο *Adobe Director 11.5*.

Τέλος, στο κεφάλαιο πέντε, αξιολογούνται τ' αποτελέσματα της εργασίας και παρατίθενται προβληματισμοί και συμπεράσματα.

2. Βιντεοπαιχνίδια

Ένα βίντεο παιχνίδι είναι ένα ηλεκτρονικό παιχνίδι που περικλείει, την αλληλεπίδραση χρήστη και μια διεπαφή, από την οποία παράγεται μια οπτική ανάδραση σε μια συσκευή εξόδου. Η λέξη *βίντεο* παραδοσιακά, αναφέρεται σε βίντεο οθόνη. Εντούτοις, η δημοφιλή χρήση του όρου «βιντεοπαιχνίδι», τώρα πια υπονοεί οποιαδήποτε συσκευή προβολής. Τα ηλεκτρονικά συστήματα που χρησιμοποιούνται για να παίξουν τα βιντεοπαιχνίδια είναι γνωστά ως, *πλατφόρμες*. Παραδείγματα αυτών είναι οι προσωπικοί υπολογιστές και οι παιχνιδοκονσόλες. Αυτές οι πλατφόρμες διαφέρουν, από μεγάλους κεντρικούς υπολογιστές ως μικρές φορητές συσκευές (κινητά, pda κ.α.).

Τα βιντεοπαιχνίδια αποτελούν σήμερα μέσο ψυχαγωγίας για κάθε ηλικία και φύλο, με σχεδόν ολοκληρωτική διείσδυση στο νεανικό κοινό και τεράστια απήχηση στους ενήλικες. Είναι διαθέσιμα για κάθε ψηφιακή συσκευή – από κονσόλες μέχρι κινητά τηλέφωνα – και αποτελούν μια ισχυρή βιομηχανία που συναγωνίζεται σε κέρδη τη βιομηχανία του κινηματογράφου. Επιπλέον, αλλάζουν σταδιακά τον τρόπο με τον οποίο επικοινωνούμε, αλληλεπιδρούμε, μαθαίνουμε, και εργαζόμαστε και γι' αυτόν το λόγο, αποτελούν αντικείμενο ακαδημαϊκής έρευνας αλλά και κριτικής και κοινωνικής ανησυχίας. Αφομοιώνονται από και μετασχηματίζουν την κουλτούρα των νέων. Είναι ένα μέσο με σημαντική ιστορία, αλλά ακόμα σημαντικότερο μέλλον.

2.1.Σύντομη ιστορική αναδρομή και εξέλιξη

Τα πρώτα παιχνίδια που δημιουργήθηκαν δεν ήταν ιδιαίτερα ψυχαγωγικά, και ο τρόπος ανάπτυξής τους δεν εστίαζε στην εμπειρία χρηστών. Τα παιχνίδια απαιτούσαν ισχυρούς κεντρικούς υπολογιστές για να «τρέξουν». Το 1952, το «OXO» (τρίλιζα), ήταν από τα πρώτα ηλεκτρονικά παιχνίδια, όπου για την απεικόνιση του, έγινε χρήση μιας ψηφιακής συσκευή προβολής. Ένας φυσικός, το 1958, έφτιαξε ένα παιχνίδι με τίτλο «tennis for two», χρησιμοποιώντας ως συσκευή αναπαράστασης, ένα παλμογράφο. Το 1961, μια ομάδα σπουδαστών έφτιαξε το «Spacewar!». Η πλειοψηφία των πρώιμων ηλεκτρονικών παιχνιδιών, «έτρεχε» σε κεντρικούς υπολογιστές αμερικάνικων κολεγίων. Η ανάπτυξη παιχνιδιών γνώρισε άνθιση, τις δεκαετίες ΄60 και ΄70 από σπουδαστές που είχαν ως χόμπι τον προγραμματισμό και είχαν στη διάθεση τους ακριβό εξοπλισμό.

Εικόνα 1. Spacewar!: ένα από τα πρώτα βιντεοπαιχνίδια

Η χρυσή εποχή των arcades ξεκίνησε το 1978 με την έκδοση του παιχνιδιού «*Space Invader*» *της* εταιρίας *Taito*. Η τεράστια επιτυχία του, ενέπνευσε πολλούς κατασκευαστές να εισχωρήσουν στην αγορά. Το 1979, τα arcades έγιναν ακόμα πιο δημοφιλή με την άφιξη τίτλων, όπως το γνωστό «Pac-man» της *Namco*. Οι σχεδιαστές πολλών πρώιμων παιχνιδιών, αργότερα μεταβίβασαν την δουλειά τους στη βιομηχανία. Ογκώδης ηλεκτρονικές κονσόλες με αυτονομία, πρόσφεραν εύκολη και γρήγορη ψυχαγωγία, παρέχοντας ένα έγχρωμο παιχνίδι η κάθε μια. Το κόστος και το μέγεθός τους, τις καθιστούσε απαγορευτικές για οικιακή χρήση. Ήταν τοποθετημένες σε πολλά μέρη όπως μπαρ, εστιατόρια, ξενοδοχεία και χώρους ψυχαγωγίας και εμπορικά κέντρα, όπου οι χρήστες αντάλλαζαν ένα κέρμα ανά παιχνίδι. Τα παιχνίδια τύπου arcade ήταν αρκετά λαοφιλή τις δεκαετίες ’80 και ’90, βαθμιαία όμως, η δημοτικότητά τους μειώθηκε.

Εικόνα 2. Κονσόλα με το παιχνίδι Pac-man, που συναντούσαμε σε χώρους ψυχαγωγίας τις δεκαετίες ‘80 – ‘90

Ένα ουσιαστικό επιχειρηματικό πλάνο σχεδιάστηκε, όταν οι κονσόλες πρώτης γενιάς εισέβαλλαν στην αγορά προορισμένες για οικιακή χρήση. Στη βιομηχανία δεν υπήρχε ιδιαίτερη καινοτομία σε σχέδια παιχνιδιών, με αποτέλεσμα ένας μεγάλος αριθμός συσκευών, να έχει πολλά πανομοιότυπα παιχνίδια.

Συγχρόνως, οι προσωπικοί υπολογιστές έκαναν την εμφάνιση τους και πλέον μεμονωμένα άτομα μπορούσαν να προγραμματίσουν δικά τους παιχνίδια. Απλοϊκά παιχνίδια παράγονταν εύκολα και γρήγορα, δεδομένου ότι οι σχεδιαστές είχαν περιορισμούς μνήμης και γραφικών, οι οποίοι δεν επέτρεπαν πολυσύνθετη ανάπτυξη. Μεγαλύτερες επιχειρήσεις προσλάμβαναν μικρές ομάδες προγραμματιστών, που η κάθε μία αφιερωνόταν στην ανάπτυξη ενός τίτλου. Έτσι το 1984, οι κονσόλες παραμερίστηκαν και τα ηνία στις πωλήσεις πήραν, οι προσωπικοί υπολογιστές. Ένα χρόνο αργότερα, η ιαπωνική εταιρία *Nintendo* ζωντάνεψε ξανά την αγορά κονσόλας με μια νέα 8-bit συσκευή (NES) συνοδευμένη με τίτλους όπως, *Super Mario Bros.*

Η δεκαετία 1990 ξεχώρισε για την καινοτομία βιντεοπαιχνιδιών και τη μετάβαση γραφικών από 2d σε 3d. Νέα είδη, δημιουργήθηκαν όπως *first-person-shooter*, στρατηγικής και παιχνίδια διαδικτύου. Παράλληλα προστέθηκαν στην αγορά μικρές φορητές κονσόλες, όπως το *Gameboy*, με τεράστια απήχηση στο νεανικό κοινό. Ακολούθησαν οι κονσόλες 5ης γενιάς 32-64 bit και οι κασέτες αντικαταστάθηκαν με compact disks (cd).

Εικόνα 3. Gameboy: Η πρώτη φορητή κονσόλα με εκατοντάδες τίτλους παιχνιδιών

Τα κινητά τηλέφωνα μετατράπηκαν, το 1998, σε μίνι πλατφόρμες βιντεοπαιχνιδιών, όταν η φιλανδική *Nokia* εγκατέστησε στα μοντέλα της, το ασπρόμαυρο παιχνίδι *Snake*. Από τότε πολλές μεγάλες εταιρίες έκαναν το ίδιο προσθέτοντας, απλά παιχνίδια στα κινητά τους τηλέφωνα. Τέλη δεκαετίας του 1990, οι ογκώδης κονσόλες που λειτουργούσαν με κέρμα, αποσύρθηκαν, παραμένοντας στις αναμνήσεις μικρών και μεγάλων. Οι συνεχής αυξανόμενες επεξεργαστικές και γραφικές ικανότητες των προσωπικών υπολογιστών σε συνδυασμό με άλλες δυνατότητες που παρείχαν, όπως η σύνδεση στο διαδίκτυο, τους κατέστησε μια αρκετά πιο δημοφιλή επιλογή.

Η πρώτη δεκαετία του 2000 παρουσίασε καινοτομία στις κονσόλες, αλλά και στους προσωπικούς υπολογιστές. Συνάμα, η αγορά φορητών παιχνιδομηχανών έγινε περισσότερο ανταγωνιστική. Το 2001 πάνω από 3000 παιχνίδια εκδόθηκαν για PC, όμως μόνο 50 με 100 παιχνίδια απέφεραν κέρδη. Το 2005 το κόστος παραγωγής παιχνιδιών για κονσόλες έφτανε τα 3 με 6 εκατομμύρια δολάρια. Η βιομηχανία παιχνιδιών συνεχώς επεκτεινόταν. Τα εισοδήματα της βιομηχανίας πενταπλασιάστηκαν σε σύγκριση με τη δεκαετία του '90. Το 2007 η μερίδα του εισοδήματος ήταν $9,5 δισεκατομμύρια, υπερβαίνοντας αυτή, της βιομηχανίας κινηματογράφου.

Όσο οι ευρυζωνικές συνδέσεις διαδιδόταν, πολλοί εκδότες στράφηκαν στην ανάπτυξη βιντεοπαιχνιδιών διαδικτύου για έναν χρήστη και παιχνιδιών για πολλούς χρήστες όπου ταυτόχρονα συνδέονται και αλληλεπιδρούν μεταξύ τους (MMO). Το ενδιαφέρον των χρηστών επικεντρώθηκε και σε δωρεάν περιστασιακά παιχνίδια, που προσφέρονται σε διαδικτυακές κοινότητες.

Εν έτη 2010, την 7η γενιά χαρακτηρίζουν κονσόλες, που υποστηρίζουν πρόσβαση στο διαδίκτυο (playstation 3, Nintendo wii), αλλά και ισχυρές φορητές συσκευές (playstation psp, Nintendo ds), με καταπληκτικά γραφικά. Τέλος, τα κινητά τηλέφωνα 4ης γενιάς (iphone4) λαμβάνουν μεγάλο μερίδιο στην αγορά, όχι μόνο ως συσκευές τηλεπικοινωνίας αλλά και ως μίνι υπολογιστές με δυνατότητα ασύρματης πρόσβασης, και εγκατάστασης ψυχαγωγικών παιχνιδιών.

Τώρα, οι υπολογιστές και οι κονσόλες κατασκευάζονται για να υποδεχτούν βίντεο παιχνίδια φτιαγμένα με στερεοσκοπική τεχνική. Με αυτή την τεχνική, γίνεται χρήση διπλών εικόνων, που παράγονται από ελαφρώς διαφορετικές γωνίες προκειμένου να δημιουργηθεί μια τρισδιάστατη ψευδαίσθηση. Αυτό απαιτεί συνήθως χρήση ειδικών γυαλιών.

2.2. Ανάπτυξη βιντεοπαιχνιδιών

Η ανάπτυξη βιντεοπαιχνιδιών όπως και άλλες μορφές ψυχαγωγίας, έχει μετατραπεί σε διεπιστημονικό τομέα.

Για τη δημιουργία ενός παιχνιδιού τα συμβαλλόμενα μέλη χωρίζονται σε δύο κύριες κατηγορίες, σε προγραμματιστές και σχεδιαστές γραφικών. Με τα πέρασμα των χρόνων ο τομέας έχει εξελιχθεί τόσο, ώστε για μια ολοκληρωμένη δουλειά είναι απαραίτητοι, ειδικοί και από άλλους τομείς, όπως της μουσικής, του κινηματογράφου, της τηλεόρασης, μουσικοί συνθέτες, τεχνικοί ήχου κ.α. Αύτη η ομάδα ανθρώπων διευθύνεται από παραγωγούς όπου ρόλος τους είναι να παρέχουν τελευταίας τεχνολογίας μηχανήματα και να προωθούν το τελικό προϊόν.

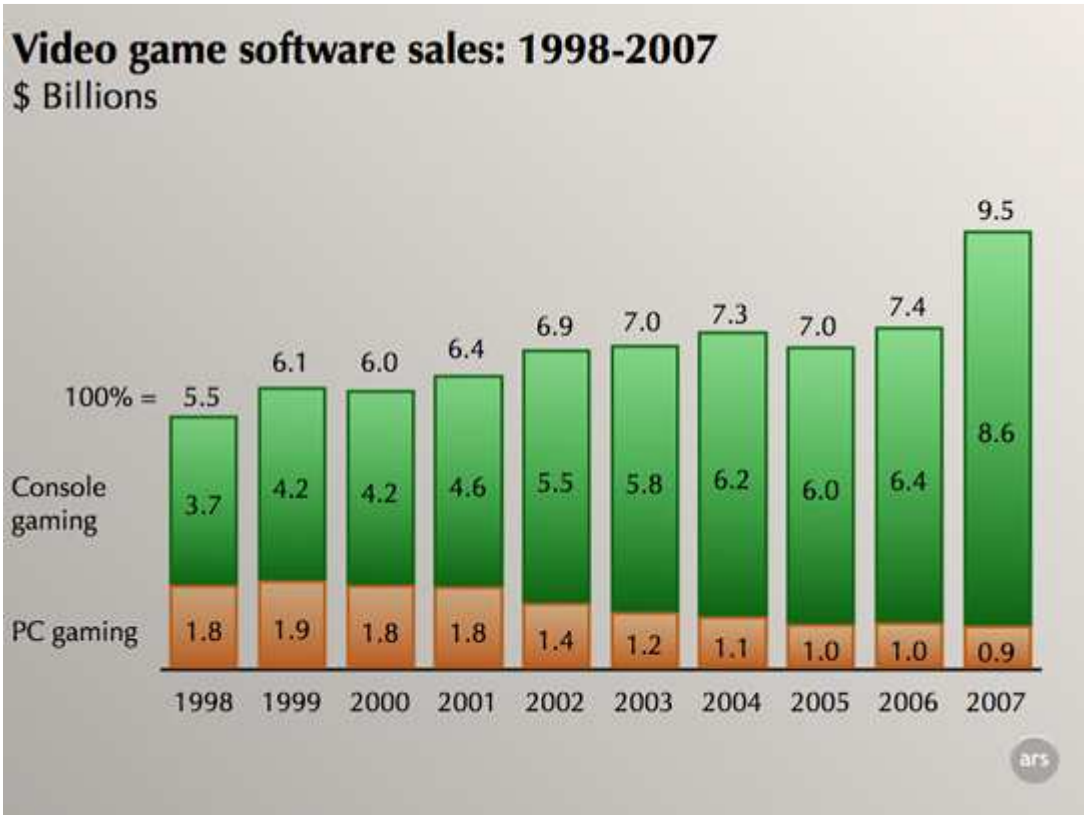
2.2.1. Βιομηχανία παραγωγής βιντεοπαιχνιδιών

Developer είναι ένα άτομο ή μία ομάδα ατόμων ή μία εταιρία, που έχει σκοπό τη σχεδίαση/ανάπτυξη ενός βιντεοπαιχνιδιού. Οι developer εταιρίες χωρίζονται σε τέσσερις βασικές κατηγορίες: 1) τη first-party developer εταιρία, η οποία είναι η ίδια εταιρία που κατασκευάζει παιχνιδομηχανές, όπως είναι η Sony, η Nintendo και η Microsoft, 2) τη second-party developer εταιρία, η οποία, αν και διαφορετική "εταιρική οντότητα", επιλέγει (κυρίως λόγω συμβολαίων) να αναπτύσσει αποκλειστικά βιντεοπαιχνίδια, μόνο για μία εταιρία κατασκευής παιχνιδρομηχανών (όπως είναι η Rare για την Microsoft ή η Insomniac για τη Sony), 3) τη third-party developer εταιρία, η οποία αναπτύσσει βιντεοπαιχνίδια για λογαριασμό άλλων εταιριών, κυρίως publisher, για μία ή περισσότερες πλατφόρμες και 4) την independent developer εταιρία, η οποία είναι συνήθως μικρού μεγέθους όπου αναπτύσσει και δημοσιεύει (publishing) ένα βιντεοπαιχνίδι από μόνη της. Οι independent εταιρίες προωθούν κυρίως το προϊόν τους μέσω του διαδικτύου αλλά και μέσο του "Word of Mouth". Σαφέστατα, βιντεοπαιχνίδια που προέρχονται από ανεξάρτητες εταιρίες, δεν έχουν την ίδια αναγνωρισιμότητα (άρα και επιτυχία) από το gaming κοινό. Δημοφιλείς developer εταιρίες είναι η BioWare, η Bungie, η Infinity Ward, η Insomniac, η Naughty Dog κ.α.

Στα πρώιμα στάδια της βιομηχανίας, ήταν πιο διαδεδομένο ν' αναλαμβάνει και να διαχειρίζεται όλες αυτές τις αρμοδιότητες, ένα μεμονωμένο πρόσωπο. Δεδομένου ότι οι υπολογιστές και οι βίντεο κονσόλες, έχουν γίνει πιο σύνθετες και πιο ισχυρές, όσον αφορά τον τύπο του υλικού που μπορούν να παρουσιάσουν, απαιτούνται μεγαλύτερες ομάδες για να την σύνταξη προγραμματισμού, για την κινηματογραφία, και τα γραφικά. Αυτό δεν σημαίνει, ότι ένας και μόνο άνθρωπος δε μπορεί να δημιουργήσει το δικό του παιχνίδι, η δουλειά του οποίου μπορεί να βρεθεί σε μια αγορά, όπου τα λιγότερο απαιτητικά παιχνίδια επικρατούν, όταν προορίζονται για μηχανήματα με τεχνικούς περιορισμούς, όπως περιορισμένη μνήμη RAM ή έλλειψη εξαρτημάτων αναπαράστασης 3d γραφικών (π.χ. PDAs).

Με την αύξηση του μεγέθους των ομάδων ανάπτυξης στη βιομηχανία, το πρόβλημα του κόστους αυξήθηκε. Τα στούντιο ανάπτυξης πρέπει να είναι σε θέση να πληρώσουν το

προσωπικό τους μια ανταγωνιστική αμοιβή προκειμένου να προσελκύσουν και να συγκρατήσουν τα καλύτερα ταλέντα. Ενώ οι εκδότες κοιτάζουν συνεχώς πώς να μειώσουν τις δαπάνες προκειμένου να διατηρήσουν την κερδοφορία της επένδυσή τους. Χαρακτηριστικά, μια ομάδα ανάπτυξης παιχνιδιών μπορεί να κυμανθεί σε μεγέθη τάξεως 5 - 50 ατόμων, ενώ κάποιες άλλες υπερβαίνουν τα 100. Η αύξηση του μεγέθους των ομάδων σε συνδυασμό με τη πίεση, να βγουν ολοκληρωμένα παιχνίδια στην αγορά σε γρήγορο χρονικό διάστημα ώστε να γίνει απόσβεση του κόστους, οδήγησε πολλές εταιρίες στη χρεοκοπία, στο πρόσφατο παρελθόν.



Πίνακας 1. Πωλήσεις βιντεοπαιχνιδιών κατά την περίοδο 1998 – 2007. Σύγκριση πωλήσεων παιχνιδιών για υπολογιστές με παιχνίδια για κονσόλες

2.3. Στάδια ανάπτυξης βιντεοπαιχνιδιών

Τα τρία βασικά στάδια ανάπτυξης βιντεοπαιχνιδιών είναι ο σχεδιασμός, η υλοποίηση και ο έλεγχος λειτουργίας του.

2.3.1. Σχεδιασμός

Η ανάπτυξη ενός παιχνιδιού ξεκινά με μια ιδέα. Τα παιχνίδια, όπως όλα τα είδη μυθοπλασίας, απαιτούν μια έμπνευση για να είναι επιτυχή. Υπάρχουν δύο μέθοδοι για την ανάπτυξη αυτής της ιδέας. Είτε επιλέγοντας να προχωρήσουμε με βάση ένα γνωστό τεχνολογικό πρότυπο είτε δημιουργώντας ένα νέο. Αυτό σημαίνει ότι, είτε διαλέγουμε ως βάση, ένα είδος παιχνιδιού από τα υπάρχοντα, είτε βαδίζουμε ακολουθώντας ένα πρωτότυπο ιδέα. Σε αυτό το στάδιο, αναλύουμε με λεπτομέρεια όλη την πλοκή του παιχνιδιού.

Σχεδιάζουμε τους χαρακτήρες και το περιβάλλον, γράφουμε το σενάριο, περιγράφουμε τον τρόπο, με τον οποίο θα εξελίσσεται το παιχνίδι. Απαιτείται, η χρήση πίνακα με διάταξη σκηνών (storyboard) ώστε να έχουμε ένα ολοκληρωμένο υπόδειγμα, που θα μας βοηθήσει στην συνέχεια. Συμπερασματικά, ένα παιχνίδι για να είναι παιχνίδι πρέπει να περιέχει τα παρακάτω στοιχεία:

- **Όρια**: κάθε παιχνίδι λαμβάνει χώρα μέσα σε ένα φανταστικό κόσμο με καθορισμένα όρια
- **Στόχος**: ο παίκτης που συμμετέχει στο παιχνίδι έχει κάποια επιδίωξη, κάποιον απώτερο στόχο
- **Διαδικασίες**: οι διαδικασίες ενός παιχνιδιού είναι όλες οι κινήσεις που μπορεί να πραγματοποιήσει ένας παίκτης
- **Κανόνες**: οι κανόνες σε ένα παιχνίδι καθορίζουν ποιες κινήσεις και διαδικασίες επιτρέπεται να πραγματοποιηθούν και ποιες όχι
- **Πόροι**: πόρος σε ένα παιχνίδι είναι οποιοδήποτε αντικείμενο αξίας μπορεί να βοηθήσει τον παίκτη να πετύχει το σκοπό του
- **Σύγκρουση**: τα εμπόδια που ο παίκτης συναντά στη διαδρομή για την επίτευξη ενός στόχου
- **Αποτέλεσμα**: ένα παιχνίδι καταλήγει σε μια τελική κατάσταση με κάποιο αποτέλεσμα (νίκη ή μη) για τον παίκτη

Τα συστατικά αυτά υπάρχουν σε όλα τα παιχνίδια, από τα πιο απλά μέχρι τα πιο πολύπλοκα. Χωρίς αυτά δεν μπορούν να δημιουργηθούν οι κατάλληλες συνθήκες για «παιχνίδι». Όταν λοιπόν ξεκινάμε την δημιουργία ενός παιχνιδιού πρέπει, πριν αρχίσουμε τον προγραμματισμό ή τη δημιουργία του περιεχομένου, να σχεδιάσουμε το παιχνίδι, να καθορίσουμε, δηλαδή τα στοιχεία της παραπάνω λίστας. Αυτή είναι η δυσκολότερη διαδικασία που συναντάμε, κατά την ανάπτυξη ενός παιχνιδιού. Σε περίπτωση που δεν καθοριστεί σωστά ο σκελετός, τότε ελλοχεύει κίνδυνος, το παιχνίδι να είναι ακατανόητο και μη ψυχαγωγικό με αποτέλεσμα, να μην κεντρίσει το ενδιαφέρον του παίκτη, όσο καλά κι αν είναι τα γραφικά του. Ο σχεδιασμός ενός παιχνιδιού είναι τέχνη και τεχνική μαζί.

Παράδειγμα: Ο παίκτης είναι στρατιώτης και πρέπει να καταλάβει τον πύργο επικοινωνίας του εχθρού στην κορυφή του λόφου. Γιατί να το κάνει αυτό; Διότι έτσι θα διακόψει την επικοινωνία ενός μεγάλου τμήματος του αντίπαλου στρατού με τη κεντρική διοίκηση. Πως θα φτάσει στον πύργο; Είτε με τα πόδια, είτε με κάποιο άρμα μάχης. Τι εμπόδια θα συναντήσει στην πορεία; Αντίπαλους στρατιώτες, κακή ορατότητα λόγω καιρικών συνθηκών, εχθρικά βομβαρδιστικά. Τι κατάληξη θα έχει το παιχνίδι; Είτε θα χτυπηθεί από τον εχθρό, είτε θα χαθεί μέσα στην ομίχλη είτε θα καταφέρει να καταλάβει τον πύργο.

Ο σχεδιασμός ενός παιχνιδιού είναι η καρδιά της δημιουργίας του. Δεν είναι ιδιαίτερα δύσκολο να σχεδιάσουμε παιχνίδια αν λάβουμε υπόψη, τις παραπάνω απαιτήσεις.

2.3.2. Υλοποίηση

Πριν προχωρήσουμε στο στάδιο της υλοποίησης, είναι απαραίτητο να επιλέξουμε:

- την πλατφόρμα (υπολογιστής, κονσόλα, κινητό τηλέφωνο, ιστοσελίδες), για την οποία θα προορίζεται το παιχνίδι
- το λειτουργικό σύστημα πάνω στο οποίο θα δουλέψουμε
- αν το παιχνίδι θα είναι 3d ή 2d
- αν θα υποστηρίζει πολλούς παίκτες ταυτόχρονα

- και τέλος, πρέπει ν' αποφασίσουμε αν θέλουμε να δουλέψουμε πάνω σε μια μηχανή παιχνιδιών (game engine) ή να φτιάξουμε μια νέα από την αρχή και να συνθέσουμε το παιχνίδι, πάνω σε αυτή.

2.3.2.1. Game engines

Πολλά δημοφιλή παιχνίδια είναι χτισμένα πάνω σε επαγγελματικά πακέτα λογισμικού γνωστά και ως, μηχανές παιχνιδιών (game engines). Είναι ειδικά εργαλεία με τα οποία οργανώνονται και αναπτύσσονται τα βιντεοπαιχνίδια. Οι ισχυρές αυτές μηχανές διαθέτουν λειτουργίες, όπως μηχανή απόδοσης ("renderer") για 2D ή 3D γραφικά, μηχανή φυσικής, ανίχνευση σύγκρουσης (και απόκριση σύγκρουσης), ήχο, γλώσσα προγραμματισμού, βίντεο, τεχνητή νοημοσύνη, δικτύωση, διαχείριση μνήμης κ.α. Τα είδη διαφέρουν αναλόγως, το λειτουργικό σύστημα πάνω στο οποίο θα «τρέξουν» (Mac OS x, windows, Linux), την πλατφόρμα για την οποία προορίζονται τα παιχνίδια, το είδος παιχνιδιού (first person shooter, strategy, adventure), τη γλώσσα που διαθέτουν, όπως και αν θέλουμε τα παιχνίδια να έχουν 3d ή 2d γραφικά. Ακόμα, μερικές από τις μηχανές, επιτρέπουν όχι μόνο την εισαγωγή και τη διαχείριση τρισδιάστατων αντικειμένων άλλα και την μοντελοποίηση τους. Συνήθως, οι μηχανές είναι συνδεδεμένες με κάποια scripting γλώσσα (python, lua, ruby) και έτσι μπορούμε να κωδικοποιήσουμε το πραγματικό παιχνίδι σε μια τέτοια γλώσσα, εστιάζοντας στον σχεδιασμό, παρά στις λεπτομέρειες χαμηλότερων επιπέδων. Γενικά, πρέπει να είμαστε απόλυτα σαφής σχετικά με το τι θέλουμε να κάνουμε, ώστε να επιλέξουμε την κατάλληλη εφαρμογή. Στο διαδίκτυο προσφέρονται δωρεάν εφαρμογές με συλλογές έτοιμων αντικειμένων, κόσμων παραδειγμάτων και οδηγών που βοηθούν στην οργάνωση/σχεδίαση και ανάπτυξη βιντεοπαιχνιδιών.

Δύο όροι που ακούγονται συχνά στον τομέα ανάπτυξης, οι οποίοι έχουν στενή σχέση με τις μηχανές, είναι «API» (application programming interface) και «SDK» (software development kit). Μια διασύνδεση προγραμματισμού εφαρμογών είναι μια διεπαφή που εκτελείται, κυρίως, από λειτουργικά συστήματα. Είναι το μέσο επικοινωνίας και αλληλεπίδρασης μεταξύ διαφορετικών συστημάτων. Τα λειτουργικά συστήματα, οι βιβλιοθήκες και οι υπηρεσίες είτε παρέχονται, είτε τις εγκαθιστά ο ίδιος ο χρήστης, έτσι ώστε να μπορεί να επωφεληθεί των ιδιαίτερων χαρακτηριστικών τους.

SDK είναι μια συλλογή από βιβλιοθήκες, APIs, και εργαλείων, διαθέσιμα για τον προγραμματισμό των λειτουργικών συστημάτων και των υπηρεσιών. Οι περισσότερες μηχανές, είναι εφοδιασμένες με APIs, ενσωματωμένες στα πακέτα SDK τους. Για παράδειγμα, η *Unreal Engine*, παρέχει μια διεπαφή για τους προγραμματιστές ώστε να μπορούν να δημιουργούν παιχνίδια, τόσο μέσω scripting γλώσσας με ονομασία *UnrealScript*, όσο και μέσω βιβλιοθηκών, που παρέχονται κατόπιν άδειας και έρχονται στο ίδιο πακέτο μαζί με άλλα εργαλεία, όπως ο συντάκτης *UnrealEd*.

Τα τελευταία χρόνια, πολλές επιχειρήσεις δημιουργούσαν μηχανές παιχνιδιών και κρατούσαν την τεχνολογία αυτή μυστική. Καθώς οι υπολογιστές βελτιώνονταν, πιο προηγμένες εκδόσεις απαιτούνταν, έτσι ανάπτυσσαν τις μηχανές. Μηχανές όπως, η *SCUMM* από την *LucasArts* και *SCI* από την *Sierra*, παραδείγματος χάριν, τροφοδοτούσαν τα περισσότερα από τα παιχνίδια περιπέτειας, που οι επιχειρήσεις απελευθέρωσαν προς το τέλος της δεκαετίας του '80 και στη μέση της δεκαετίας του '90. Πιο πρόσφατα, μηχανές όπως η *Id tech* (απ' την οποία δημιουργείται η σειρά παιχνιδιών *Quake*) και η *Unreal*, ξεκίνησαν ως εσωτερικές τεχνολογίες, όμως εξελίχθηκαν και βρίσκονται πλέον στην αγορά. Το κόστος μιας μηχανής έχει αυξηθεί σημαντικά, και όλο και περισσότερες επιχειρήσεις έχουν αρχίσει να ειδικεύονται στην κατασκευή είτε πλήρων μηχανών, είτε τμημάτων μηχανών. Πολλά studios, προτιμούν να αγοράζουν μηχανές παρά να πληρώνουν προγραμματιστές για να κατασκευάζουν νέες μηχανές. Ως αποτέλεσμα, σχεδόν όλα τα υλικά, βρίσκονται στην αγορά σε διάφορες τιμές. Η ποικιλία εμπορικών και δωρεάν μηχανών είναι μεγάλη και σχεδόν όλες, μπορούν να ταξινομηθούν στις παρακάτω κατηγορίες:

- ***Έτοιμες μηχανές***

Αυτές οι μηχανές περιέχουν αρκετά εργαλεία, τα οποία τις καθιστούν έτοιμες για την ανάπτυξη βιντεοπαιχνιδιών. Περιλαμβάνουν δυνατότητες απόδοσης (rendering), GUI, φυσικής κ.α. Πολλές έχουν ακόμη, χρήσιμες αλυσίδες εργαλείων έτσι δεν είναι απαραίτητο να φτιάξουμε τις δικές μας. Σε αυτή την κατηγορία ανήκουν μηχανές όπως, η *OGRE* και η *Genesis3D*, οι οποίες είναι ανοικτού πηγαίου κώδικα, την *Torque* τη βρίσκουμε στο εμπόριο σε χαμηλή τιμή. Πιο γνωστές, με υψηλότερο κόστος είναι η *Gamebryo*, η *Unreal* και η *id Tech*. Όλες αυτές οι μηχανές δέχονται περεταίρω οργάνωση, όπως πρόσθεση υλικών, για να υπηρετήσουν καλύτερα τις ανάγκες του χρήστη. Υπάρχουν, περιορισμοί στις αρχικές δυνατότητες αλλά δέχονται βελτιστοποιήσεις.

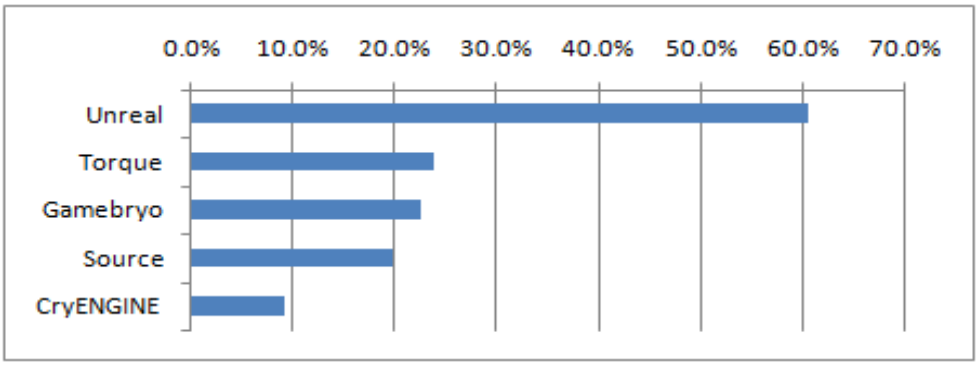
- ***Μηχανές point-and-click***

Οι μηχανές αυτές, κερδίζουν όλο και περισσότερο έδαφος, καθώς περιλαμβάνουν μια πλήρη αλυσίδα εργαλείων που επιτρέπει στους χρήστες να διαμορφώνουν τα παιχνίδια τους, με το απλό σύστημα point-and-click. Στην κατηγορία ανήκουν, οι *GameMaker*, *Game Builder* και *Unity3D*. Είναι κατασκευασμένες έτσι ώστε να προσφέρουν απλότητα, ευχρηστία και απαιτούν ελάχιστο προγραμματισμό. Φυσικά, η προσθήκη κώδικα, είναι πάντα μεγάλη βοήθεια, αλλά δεν είναι τόσο απαραίτητη όσο είναι στις άλλες κατηγορίες. Το πρόβλημα με πολλές από αυτές, είναι ότι μπορούν να είναι εξαιρετικά περιοριστικές. Κάποιες είναι άριστες για την δημιουργία, έναν ή δύο τύπων παιχνιδιού, άλλες διαθέτουν έναν ή δύο μεθόδους γραφικής αναπαράστασης. Αυτό δεν σημαίνει ότι είναι άχρηστες. Ακόμη και αντιμέτωποι με αυτούς τους περιορισμούς, είναι δυνατόν να κατασκευαστούν ιδιαίτερα δημιουργικά παιχνίδια ή ακόμα και να βρεθούν δημιουργικοί τρόποι, ώστε να ξεπεράσουμε τους περιορισμούς. Το θετικότερο με αυτές τις μηχανές, είναι ότι μας επιτρέπουν να εργαστούμε γρηγορότερα και χωρίς πολλή δουλειά.

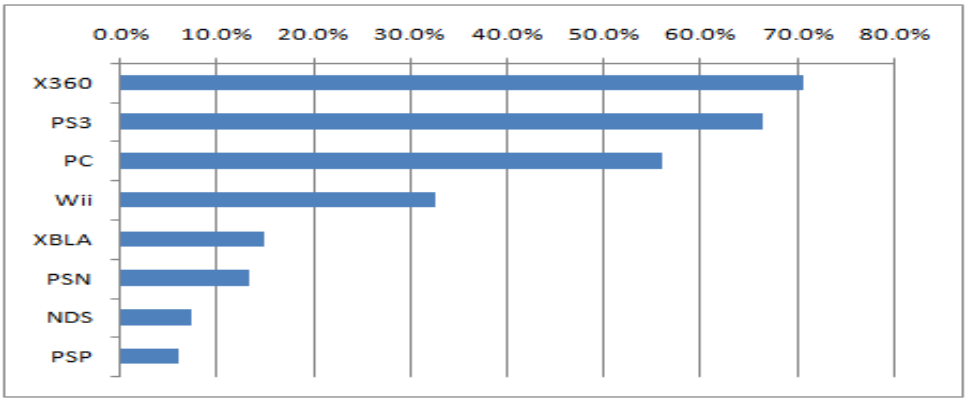
- ***Μηχανή συναρμολογημένη, από υλικά της επιλογής μας***

Παρά το κόστος, πολλές επικρατούσες στο χώρο επιχειρήσεις, προσπαθούν να αναπτύσσουν τις δικές τους μηχανές. Χρησιμοποιούν δημόσια διαθέσιμες APIs όπως XNA, DirectX, OpenGL, τα APIs των windows και των Linux και SDL, για τη δημιουργία δικών τους μηχανών. Επιπλέον, μπορούν να κάνουν χρήση κι άλλων βοηθητικών εργαλείων, όπως βιβλιοθήκες, εμπορικές ή μη και με ανοικτό πηγαίο κώδικα. Αυτές οι βιβλιοθήκες περιλαμβάνουν βιβλιοθήκες φυσικής όπως *Havok* και *ODE*, βιβλιοθήκες γραφικών παραστάσεων σκηνής, όπως *OpenSceneGraph*, και GUI όπως *AntTweakBar*. Γενικά, αυτά τα συστήματα δίνουν στους προγραμματιστές μεγάλη ευελιξία, αφήνοντας τους να επιλέξουν τα συστατικά που θέλουν, ενσωματώνοντάς τα ακριβώς όπως θέλουν. Όμως η κατασκευή τους είναι χρονοβόρα. Επιπλέον, οι προγραμματιστές συχνά, θα πρέπει να «χτίσουν» την αλυσίδα εργαλείων από την αρχή, δεδομένου ότι σπάνια μπορούν να στηριχθούν στην συνεργασία όλων αυτών των βιβλιοθηκών.

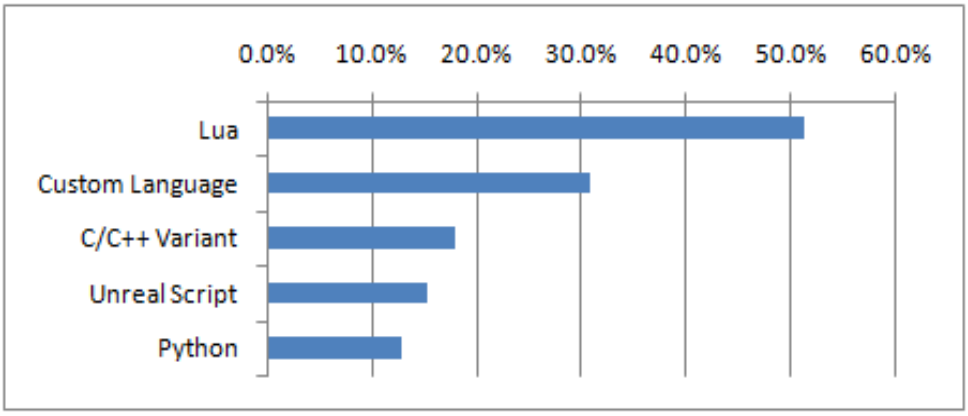
Στον πίνακα 2, διακρίνονται οι 5 πιο γνωστές μηχανές που χρησιμοποιούν οι επαγγελματίες. Στον πίνακα 3, παρατηρούμε ότι το 70% των κατασκευαστών, αυτή τη στιγμή, απασχολούνται με την δημιουργία παιχνιδιών, που προορίζονται για την παιχνιδομηχανή *Microsoft Xbox360*. Και στον πίνακα 4, διακρίνονται οι δημοφιλέστερες γλώσσες, που χρησιμοποιούνται για την ανάπτυξη βιντεοπαιχνιδιών.



Πίνακας 2. Τοπ - 5 δημοφιλών μηχανών



Πίνακας 3. Οι Τοπ - 8 πιο δημοφιλείς πλατφόρμες, για τις οποίες κατασκευάζονται παιχνίδια αυτή τη στιγμή



Πίνακας 4. Τοπ - 5 πιο δημοφιλής γλώσσες προγραμματισμού για παιχνίδια

2.3.2.2. Προγραμματισμός

Υπάρχει ακόμα και η επιλογή να κατασκευάσουμε εξολοκλήρου από την αρχή τα υλικά που θα συνθέσουν μια μηχανή ή το ίδιο το παιχνίδι. Για να γίνει αυτό πρέπει να

συντάξουμε κώδικα, συνήθως σε μια παραδοσιακή low – level γλώσσα, και να χρησιμοποιήσουμε ένα compiler.

Είναι πολύ σημαντικό να κατανοήσουμε όλες τις βασικές έννοιες, όσον αφορά τον προγραμματισμό. Αν δεν ξέρουμε πώς να προγραμματίζουμε, μπορούμε να ξεκινήσουμε μαθαίνοντας μια σχετικά εύκολη γλώσσα όπως την Python. Αυτή η γλώσσα αφαιρεί το χαμηλό επίπεδο λεπτομερειών και επιτρέπει στον προγραμματιστή να επικεντρωθεί στις έννοιες.

Αφού μπούμε στο πνεύμα του προγραμματισμού, θα πρέπει ν' αποφασίσουμε τη γλώσσα με την οποία θα φτιάξουμε τη μηχανή παιχνιδιών μας. Οι προγραμματιστές ξεκινούν δημιουργώντας τον πυρήνα της μηχανής, όλα τ' αντικείμενα, τις αρμοδιότητές και τα εργαλεία που θα βοηθήσουν στην σύνθεση ενός παιχνιδιού. Σε αυτή τη διαδρομή, θα πρέπει να ασχοληθούμε με όλες τις λεπτομέρειες χαμηλού επιπέδου προγραμματισμού. Για αυτό, προτείνεται η C ή C ++, καθώς οι περισσότερες μηχανές είναι κωδικοποιημένες σε αυτές τις γλώσσες. Είναι βασικό, να είμαστε πολύ πειθαρχημένοι σχετικά με την κατανομή μνήμης και εξοικειωμένοι με τεχνικές βελτιστοποίησης.

Όταν μάθουμε πώς να προγραμματίζουμε, πρέπει να γνωρίσουμε και τη σημασία της βιβλιοθήκης. Αν δούμε το θέμα από την άποψη επιπέδων, στο χαμηλότερο επίπεδο βρίσκονται οι βιβλιοθήκες, όπως OpenGL, Pygame κ.α. Ναι μεν, παρέχουν πολλές λειτουργίες, εντούτοις δεν πρέπει να βασιζόμαστε μόνο σε αυτό. Πρέπει να κωδικοποιήσουμε εξολοκλήρου τη ροή του παιχνιδιού. Οι βιβλιοθήκες είναι, επιπλέον «μπαλώματα» κώδικα, τα οποία συνδέονται στο δικό μας κώδικά. Για να κάνουμε παιχνίδια, χρειαζόμαστε βιβλιοθήκες με γραφικά, με χειρισμούς γεγονότων (event handler) κ.λπ. Αν ξεκινήσουμε με Python, η βιβλιοθήκη Pygame είναι άριστη για αρχαρίους, και παρέχει πολλά από αυτά τα υλικά. Για την C/C++, έχουμε *ALLEGRO* και *SDL*. Μια απλή αναζήτηση στο *Google*, μας παραθέτει σε πολλούς κατάλογους βιβλιοθηκών προγραμματισμού, για την γλώσσα της επιλογής μας.

Οι μηχανές περιλαμβάνουν κάποιο είδος διερμηνέα ή μεταγλωττιστή (compiler) που μεταφράζει τις εντολές του χρήστη, σε γλώσσα την οποία καταλαβαίνει ο υπολογιστής. Ο πυρήνας μιας μηχανής παιχνιδιών είναι φτιαγμένος από χαμηλού επιπέδου γλώσσες όπως η C++, ενώ για τη διαχείριση των υλικών στηριζόμαστε σε πιο απλές γλώσσες τύπου script (Lua, UnrealScript, Javascript κ.α.).

Ο υπολογιστής καταλαβαίνει δυαδικό κώδικα, που ονομάζεται, γλώσσα μηχανής (Machine Language). Είναι μια συλλογή από δυαδικά δεδομένα που λέει στον υπολογιστή (που είναι ουσιαστικά μια αριθμομηχανή), ποιές αριθμητικές πράξεις να κάνει και που ν' αποθηκεύσει τ' αποτελέσματα. Αυτή η γλώσσα είναι το χαμηλότερο επίπεδο. Δυστυχώς, η γλώσσα μηχανής είναι δύσκολη στην κατανόηση. Ένας compiler, όπως ο *Microsoft Visual Studio*, μεταφράζει μια τεχνητή γλώσσα (όπως τη C++) σε Machine language. Όταν γράφουμε σε C++, ο compiler θα μεταγλωττίσει τον κώδικα και θα δημιουργήσει ένα αρχείο exe, στη συνέχεια η διεπαφή-χρήστη θα εκτελέσει το αρχείο αυτό και θα εμφανίσει το αποτέλεσμα. Αυτό είναι το μόνο που χρειάζεται να κάνουμε για να φτιάξουμε ένα απλό παιχνίδι, με χρήση C++.

Μια μηχανή παιχνιδιών αποτελείται από μια συλλογή κώδικα (συνήθως C/C++), η οποία μπορεί να ενσωματωθεί στα δικά μας C/C++ προγράμματα ώστε ν' αποφύγουμε τις επαναλήψεις. Ακόμα και τα πιο απλά προγράμματα απαιτούν πολύ κώδικα για να λειτουργήσουν. Αυτό, κάνει την σύνταξη κώδικα πιο εύκολη, αλλά δεν είναι υποχρεωτικό για να ολοκληρώσουμε ένα παιχνίδι. Πολλοί προγραμματιστές γράφουν ποικίλες ποσότητες κώδικα ξεκινώντας από το μηδέν. Άλλοι προγραμματιστές χρησιμοποιούν έτοιμες μηχανές, ή αγοράζουν συλλογές και παίρνουν τμήματα κώδικα. Ορισμένες μηχανές είναι τόσο προχωρημένες, που έχουν ενσωματωμένους μεταγλωττιστές, για άλλες γλώσσες. Με τον τρόπο αυτό, έχουμε πρόσβαση στον εσωτερικό κώδικα της μηχανής (π.χ. Unity3D).

Τα scripts συντάσουν έναν κατάλογο εσωτερικών λειτουργιών που έχουν μεταγλωττιστεί και είναι έτοιμα να κληθούν. Η μηχανή διαβάζει το αρχείο, ταιριάζει τις λειτουργίες, και τις εκτελεί. Οι μεταγλωττιστές κάνουν τα πράγματα διαφορετικά. Δέχονται απλά τον κώδικα και τον εκτελούν. Δεν υφίσταται καμία ανάγκη για τον προσδιορισμό των

λειτουργιών. Ομοίως, όταν μαθαίνουμε για πρώτη φορά πως λειτουργεί μια συσκευή, το πρώτο βήμα είναι να διαβάσουμε τον οδηγό λειτουργίας. Αφού σκεφτούμε τι πρέπει να κάνουμε, εκτελούμε τις κατάλληλες ενέργειες. Όμως, για κάτι το οποίο είναι ήδη εγγενώς ενσωματωμένο στον εγκέφαλό μας, όπως η αναπνοή ή η κίνηση, δεν είναι απαραίτητη καμία μορφή επεξεργασίας, συμβαίνει υποσυνείδητα και χωρίς σκέψη. Οι γλώσσες προγραμματισμού, είναι ο τρόπος επικοινωνίας μεταξύ προγραμματιστών και υπολογιστών. Παραδοσιακές τεχνητές γλώσσες είναι οι : C#, Delphi, Java, Objective-C, Qbasic και Visual Basic. Ενώ, γνωστές script-style γλώσσες, είναι οι : Lua, ruby, ActionScript (Flash), JavaScript, KonsolScript, και Squirrel.

Γλώσσες προγραμματισμού	**Πλεονεκτήματα**	**Μειονεκτήματα**
Assembly	*Χαμηλό overhead*	*Error-prone, αργή ανάπτυξη, δύσκολη για αρχάριους, δεν είναι φορητή*
C	*Υπάρχουν ευρέως γνωστά και ποικίλα εργαλεία*	*Καμία ενσωματωμένη υποστήριξη αντικειμενοστραφή προγραμματισμού, δύσκολη για μεγάλα project και πολλαπλές πλατφόρμες*
C++	*Ενσωματωμένη υποστήριξη αντικειμενοστραφή προγραμματισμού, γνωστά και ποικίλα εργαλεία*	*Δεν υπάρχει ενσωματωμένος GC (garbage collector), καμιά προστασία κατά της διαρροής μνήμης*
C#	*Αρκετά αντικειμενοστραφής, RAD (rapid application development) γλώσσα, εύκολη στη χρήση*	*Χρησιμοποιεί υψηλό ποσοστό μνήμης*
Java	*Αντικειμενοστραφής, εύκολη στη χρήση, φορητή*	*Δεν ενδείκνυται για τον προγραμματισμό παιχνιδιών κονσόλας, υψηλή χρήση μνήμης*
Eiffel, Smalltalk, Ada	*Αντικειμενοστραφής*	*Ελάχιστα εργαλεία ανάπτυξης*
Scripting γλώσσες όπως **Lua** και **Python**	*Συχνά χρησιμοποιούνται για τη σεναριογραφία του gameplay*	

Πίνακας 5. Γνωστές γλώσσες - μειονεκτήματα και πλεονεκτήματα

2.3.2.3. Επιμέρους στοιχεία

Για τη δημιουργία ενός καλοστημένου παιχνιδιού, πρέπει να ενσωματώσουμε και άλλα συστατικά στο πεδίο δράσης του εικονικού κόσμου, όπως φυσική. Είναι πολύ σημαντικό, να γνωρίζουμε τις έννοιες πίσω από τις αλληλεπιδράσεις του πραγματικού κόσμου και όχι τόσο, το πώς ακριβώς εφαρμόζονται στα παιχνίδια. Εάν διαθέτουμε μια ισχυρή γνώση των θεμελιωδών νόμων της φυσικής, τότε είμαστε σε θέση να αναπτύξουμε τη δική μας μηχανή φυσικής. Η φυσική στα παιχνίδια, σκοπό έχει να παρουσιάζει τις ενέργειες που συμβαίνουν στο παιχνίδι, όσο το δυνατό πιο ρεαλιστικές, στον παίκτη. Ουσιαστικά, η προσομοίωση φυσικής είναι μια κοντινή προσέγγιση της πραγματικής φυσικής, όπου ο υπολογισμός γίνεται με διακριτές τιμές. Τα στοιχεία που διαμορφώνουν τα συστατικά της προσομοίωσης είναι:

- η μηχανή φυσικής, που είναι έτοιμος κώδικας και χρησιμοποιείται για να μιμηθεί τους νόμους του Νεύτων μέσα στο εικονικό περιβάλλον.
- το σύστημα ανίχνευσης σύγκρουσης, που χρησιμοποιείται για ν' αντιμετωπίσει το πρόβλημα πρόσκρουσης, μεταξύ δύο ή περισσότερων αντικειμένων.

Ένας γνωστός τύπος σύγκρουσης, είναι η έκρηξη. Στα πρώιμα παιχνίδια υπολογιστών, απλά χρησιμοποιούσαν το ίδιο εφέ έκρηξης σε κάθε περίσταση. Εντούτοις, στον πραγματικό κόσμο μια έκρηξη μπορεί να ποικίλει ανάλογα με το περιβάλλον, την ένταση πρόσκρουσης και των όγκο των στερεών οργανισμών. Ανάλογα την ισχύ της επεξεργαστικής ικανότητας, τα αποτελέσματα μιας έκρηξης διαφέρουν. Για παράδειγμα, μ' ένα ισχυρό επεξεργαστή μπορούμε ν' απεικονίσουμε μια έκρηξη, μοντελοποιώντας πολλά αποσπώμενα κομμάτια των αντικειμένων που ήρθαν σε σύγκρουση, τη στιγμή της έκρηξης, σε συνδυασμό με εφέ καπνού. Αυτό είναι εφικτό με τη βοήθεια ενός συστήματος προσομοίωσης μορίων (particle system). Ένα τέτοιο σύστημα, επιτρέπει σε ποικίλα άλλα φυσικά φαινόμενα να προσομοιωθούν, όπως ο καπνός, η κίνηση, το νερό και άλλα. Τα μεμονωμένα μόρια μέσα στο σύστημα διαμορφώνονται χρησιμοποιώντας άλλα στοιχεία των κανόνων προσομοίωσης φυσικής, πάντα με περιορισμό την υπολογιστική ισχύ.

Πέραν το σημαντικό κομμάτι του προγραμματισμού και της φυσικής για την κατασκευή ρεαλιστικότερων παιχνιδιών, είναι καλό να προσθέσουμε και άλλα συστατικά. Μια από τις μέγιστες προκλήσεις για τους προγραμματιστές, είναι η δημιουργία δυνατής τεχνητής νοημοσύνης (ΤΝ). Η τεχνητή νοημοσύνη στα παιχνίδια, κατέχει σημαντική θέση πλέον. Πολλές φορές, η εμπορική επιτυχία ενός παιχνιδιού εξαρτάται συχνά από την ποιότητα αυτής. Η ΤΝ στα παιχνίδια, είναι κυρίως νευρωνικά συστήματα δικτύων και σύνθετες μαθηματικές δομές, είναι συμπεριφοριστική λογική, και όχι επιστημονική. Ακόμα δε θα έπρεπε να παραλείψουμε στοιχεία όπως, μουσική υπόκρουση, ηχητικά και οπτικά εφέ.

Αφού διαμορφώσουμε και προγραμματίσουμε ένα ικανοποιητικό gameplay, το οποίο θα πρέπει να είναι ξεκάθαρο και κατανοητό, το επόμενο βήμα μας παραθέτει στη δημιουργία γραφικών. Μπορούμε είτε να σχεδιάσουμε δικά μας μοντέλα και κόσμους, χρησιμοποιώντας εξειδικευμένα προγράμματα (3d studio max, maya, blender) είτε να χρησιμοποιήσουμε έτοιμα μοντέλα. Οι γραφικές τέχνες στα παιχνίδια, έχουν έναν από τους σημαντικότερους ρόλους. Η υπερβολική προσοχή στις λεπτομέρειας του τρόπου παιξίματος, μπορεί να αποβεί μοιραίο λάθος, χωρίς ποιοτικά γραφικά, αλλά και αντίστροφα. Η προσοχή μας λοιπόν θα πρέπει να είναι στραμμένη, εξίσου και στα δύο μέρη.

Τέλος, αν θέλουμε να κατασκευάσουμε διαδικτυακά παιχνίδια, μπορούμε να δουλέψουμε με διαφορετικά προγράμματα. Σε αυτή την κατηγορία ανήκουν δύο αρκετά γνωστές πολυμεσικές εφαρμογές. Με το *adobe Flash*, οι σχεδιαστές δημιουργούν παιχνίδια όχι ιδιαίτερα πολύπλοκα και συνήθως δεν έχουν τρισδιάστατα γραφικά. Η εφαρμογή που δίνει τη δυνατότητα χειρισμού 3d γραφικών είναι η *adobe director*. Παιχνίδια και εφαρμογές διοχετεύονται στις ιστοσελίδες με το *shockwave* player. Στην αγορά υπάρχουν ειδικά

προγράμματα και εργαλεία ακόμα και για την δημιουργία παιχνιδιών κι εφαρμογών, με προορισμό τα κινητά τηλέφωνα.

2.3.3. Δοκιμή και επιδιορθώσεις

Δεν υπάρχει καμία συγκεκριμένη ακολουθία για τη δοκιμή παιχνιδιών, και οι περισσότερες μεθοδολογίες αναπτύσσονται διαφορετικά από τις εταιρίες παραγωγής. Επανακαθορίζονται συνεχώς και μπορούν να διαφέρουν ανάλογα με τον τύπο παιχνιδιών. Δανειζόμαστε όμως μεθόδους, από τις τεχνικές εξέτασης λογισμικού. Ακολουθούν μερικές από τις σημαντικότερες:

- **Η δοκιμή λειτουργίας (Functionality testing)** δεν απαιτεί εκτενείς τεχνικές γνώσεις. Οι ελεγκτές παίζοντας το παιχνίδι, ερευνούν τα γενικά προβλήματα μέσα στο ίδιο ή στη διεπαφή-χρήστη.
- **Ο έλεγχος συμμόρφωσης (Compliance testing)** είναι ο λόγος ύπαρξης εξειδικευμένων εργαστηρίων δοκιμής παιχνιδιών. Για να κυκλοφορήσει ένα παιχνίδι στην αγορά πρέπει να τηρεί συγκεκριμένες προδιαγραφές. Είναι, ένας τεχνικός έλεγχος στοιχείων, όπως διαχείριση μνήμης, προστασία πνευματικών δικαιωμάτων κ.α.
- **Ο έλεγχος συμβατότητας** συνήθως απαιτείται για τίτλους PC, που πλησιάζουν στο τέλος της ανάπτυξής τους. Οι ελεγκτές βλέπουν κατά πόσο είναι συμβατό το παιχνίδι, σε διαφορετικά είδη λειτουργικού συστήματος (software) και υλικού (hardware).
- **Με τον έλεγχο multiplayer,** οι ελεγκτές εξασφαλίζουν ότι όλοι οι τρόποι σύνδεσης (lan, διαδίκτυο) λειτουργούν και ότι πολλοί παίκτες μπορούν να παίζουν ταυτόχρονα.
- **Με τη δοκιμή φόρτωσης** εξετάζονται τα όρια ενός συστήματος, όπως ο μέγιστος αριθμός χρηστών που μπορούν να παίζουν παράλληλα.

Η πρωταρχική λειτουργία δοκιμής των παιχνιδιών, είναι η ανακάλυψη και η επίλυση ελαττωμάτων του λογισμικού (bugs). Ο έλεγχος ψυχαγωγικού λογισμικού είναι ένας εξαιρετικά τεχνικός τομέας, που απαιτεί γνώσεις πληροφορικής, αναλυτική ικανότητα, και δεξιότητες αξιολόγησης.

Συμπερασματικά, οι επιλογές για την ανάπτυξη παιχνιδιών είναι πολλές, οπότε πρέπει να είμαστε απόλυτα σαφής, όσον αφορά το τι ακριβώς θέλουμε να φτιάξουμε και για ποιο μηχάνημα, θα προορίζεται το βιντεοπαιχνίδι. Αν έχουμε κλήση στον προγραμματισμό και θέλουμε να πειραματιστούμε, μπορούμε να επιλέξουμε μια τεχνητή γλώσσα και να ξεκινήσουμε από εκεί. Εάν, δεν θέλουμε να εμπλακούμε πολύ με προγραμματισμό, αλλά μας ενδιαφέρει περισσότερο ο σχεδιασμός του παιχνιδιού, τότε θα ήταν πιο συνετό να χρησιμοποιήσουμε μια έτοιμη μηχανή, σε συνδυασμό με άλλες βιβλιοθήκες.

Εικόνα 4. Κονσόλες τελευταίας τεχνολογίας

Στην επόμενη ενότητα, ακολουθεί, η ανάπτυξη ενός απλού τρισδιάστατου βιντεοπαιχνιδιού. Παραθέτεται η εισαγωγή στον τρόπο δημιουργίας 3d μοντέλων με χρήση του προγράμματος *3d studio max 8* και στη συνέχεια επισημαίνεται ο τρόπος με τον οποίο ελέγχουμε όλα αυτά τα στοιχεία, και το πώς αντιδρούν με βάση τις εντολές του χρήστη. Αυτό επιτυγχάνεται χρησιμοποιώντας το *adobe director* και τη γλώσσα *lingo* η οποία είναι τύπου scripting.

3. Σχεδίαση τρισδιάστατων μοντέλων (3d modeling)

3.1. Εισαγωγή

Στην ηλεκτρονική γραφιστική, σχεδίαση τρισδιάστατων μοντέλων είναι η διαδικασία μαθηματικής αναπαράστασης μιας οποιασδήποτε τρισδιάστατης επιφάνειας αντικειμένου (άψυχου ή έμψυχου) μέσω, εξειδικευμένου λογισμικού. Το προϊόν της αναπαράστασης, ονομάζεται τρισδιάστατο μοντέλο. Τα μοντέλα μπορούν να δημιουργηθούν είτε αυτόματα (με αλγόριθμους), είτε με το χέρι.

Η σχεδίαση τρισδιάστατων μοντέλων δεν αφορούσε πάντα λογισμικό υψηλής τεχνολογίας, και πολύπλοκες μαθηματικές εξισώσεις. Σήμερα, όταν ακούμε τη φράση «3d modeling» η σκέψη μας πηγαίνει σε ομάδες ατόμων με ιδιαίτερη εξειδίκευση, που κάθονται πίσω από γιγαντιαίες οθόνες υπολογιστών, υπολογίζοντας τις περίπλοκες μορφές και τα σχέδια. Αν και αυτό είναι μια αρκετά ακριβής περιγραφή των ανθρώπων που κρύβονται πίσω από επιτυχημένες εταιρίες του χώρου, η εκκίνηση του κλάδου ήταν διαφορετική.

3.2. Ιστορική αναδρομή

Αυτή η χρήσιμη μορφή αντιπροσώπευσης αντικειμένων μεγαλύτερης κλίμακας σε μια πολύ μικρότερη κλίμακα, ξεκίνησε ως χόμπι και ως τρόπος πρόκλησης του μυαλού. Οι πρώτοι σχεδιαστές δεν είχαν στη διάθεση τους εξειδικευμένα εργαλεία. Για να υπολογίσουν σωστά, το επιθυμητό μέγεθος του αντικειμένου και να το σχεδιάσουν βάση μιας μικρότερης κλίμακας, έπρεπε να επιλύσουν δύσκολες και πολύπλοκες μαθηματικές εξισώσεις.

Το 1975 ο βρετανός ερευνητής Martin Newell, σχεδίασε στο χαρτί μια τσαγέρα γνωστή ως «Utah teapot» ή «Newell teapot» (βλ. Εικόνα 5). Θεώρησε πως το σχήμα της περιείχε ένα σύνολο στοιχείων, τα οποία την έκαναν ιδανική για πειραματισμούς. Ιδιότητες όπως, ότι είναι στερεά, κυλινδρική και μερικώς κυρτή. Αφού ανέλυσε και επεξεργάστηκε τα δεδομένα, κατάφερε ν' αναπαραστήσει επιτυχώς, ένα τρισδιάστατο μοντέλο της τσαγέρας, στον υπολογιστή. Δημοσίευσε τ' αποτελέσματα και αρκετοί ερευνητές μετέπειτα, χρησιμοποίησαν το μοντέλο αυτό, ως αναφορά για δικές τους μελέτες. Προς τιμήν του ερευνητή, η τσαγέρα υπάρχει, ακόμη και σήμερα, σε πολλές εφαρμογές ως έτοιμο μοντέλο. Ακόμη, στα κλασσικά μοντέλα συγκαταλέγονται, αεροπλάνα, πλοία, κτήρια, διάσημα αγάλματα και τοπία.

Εικόνα 5. Αρχικό 3d μοντέλο του M.Newell

Εικόνα 6. Σύγχρονο 3d μοντέλο τσαγέρας

Τα οφέλη της μοντελοποίησης σύντομα ανακαλύφτηκαν, από τον επιχειρησιακό κόσμο μετά από μια βιομηχανική επανάσταση. Συγκεκριμένα, ο πρώτος κλάδος που ευνοήθηκε αρκετά, ήταν αυτός της αρχιτεκτονικής. Έδωσε τα μέσα, για μια φυσική υπόσταση των αρχιτεκτονικών σχεδίων (Architectural visualization). Σήμερα, η τρισδιάστατη μοντελοποίηση έχει προσαρμοστεί για να ικανοποιεί τις ανάγκες των σχεδιαστών σε πολλούς τομείς. Στην ιατρική βιομηχανία, δημιουργούνται λεπτομερή μοντέλα των ανθρώπινων οργάνων. Στη βιομηχανία του κινηματογράφου αξιοποιούνται, για τη δημιουργία εξολοκλήρου τρισδιάστατων ταινιών (3d animation movies). Απαραίτητα και στην βιομηχανία βιντεοπαιχνιδιών. Στον τομέα της επιστήμης, χρησιμοποιούνται ιδιαίτερα, για την αναπαράσταση χημικών ενώσεων. Στην τηλεοπτική βιομηχανία για τη δημιουργία διαφημιστικών σποτ. Στην εφαρμοσμένη μηχανική σχεδιάζονται νέες συσκευές, οχήματα και άλλες κατασκευές. Πιο πρόσφατα, η γεωεπιστήμη άρχισε με τη σειρά της να εκμεταλλεύεται τα πλεονεκτήματα που προσφέρουν τα τρισδιάστατα γεωλογικά μοντέλα. Όλες αυτές οι βιομηχανίες στηρίζονται σημαντικά, στις ομάδες σχεδιαστών τους, στα εξειδικευμένα προγράμματα και σε μηχανήματα υψηλής τεχνολογίας, για την υλοποίηση και εφαρμογή των ιδεών τους. Με το πέρασμα του χρόνου ο τομέας έχει εξελιχθεί τόσο, ώστε πλέον δε θεωρείται μια απλή ενασχόληση, αλλά ένα δημιουργικό επάγγελμα με υψηλές αποδοχές.

Εικόνα 7. Architectural visualization

3.3. Μοντέλα

Τα τρισδιάστατα μοντέλα αντιπροσωπεύουν ένα τρισδιάστατο αντικείμενο χρησιμοποιώντας μια συλλογή σημείων στο τρισδιάστατο διάστημα, που συνδέεται με διάφορες γεωμετρικές οντότητες όπως τα τρίγωνα, οι γραμμές, οι κυρτές επιφάνειες, κ.τ.λ. Χρησιμοποιούνται ευρέως, οπουδήποτε στο τομέα των τρισδιάστατων γραφικών. Σχεδόν όλα τα τρισδιάστατα μοντέλα μπορούν να διαιρεθούν σε δύο κατηγορίες:

- Συμπαγή (solid)

  Αυτά καθορίζουν τον όγκο του αντικειμένου που αντιπροσωπεύουν (όπως έναν βράχο). Σαφώς είναι ρεαλιστικότερα, αλλά η σχεδίασή τους είναι δυσκολότερη. Τα συμπαγή

αντικείμενα χρησιμοποιούνται συνήθως σε μη οπτικές προσομοιώσεις όπως, οι ιατρικές και οι προσομοιώσεις εφαρμοσμένης μηχανικής.

- Μη συμπαγή (shell/boundary)
  Αντιπροσωπεύουν την επιφάνεια, π.χ. τα όρια του αντικειμένου, και όχι τον όγκο του (όπως το λεπτό κέλυφος του αυγού). Η σχεδίασή τους είναι ευκολότερη. Σχεδόν όλα τα μοντέλα που χρησιμοποιούνται στα παιχνίδια και τις ταινίες, είναι τέτοιου είδους.

3.4. Μέθοδοι μοντελοποίησης

Για το σχεδιασμό τρισδιάστατων μοντέλων χρησιμοποιούνται οι εξής μέθοδοι:

- Polygonal modeling
  Σημεία στο 3D χώρο, που ονομάζονται κορυφές, συνδέονται με ευθύγραμμα τμήματα για να σχηματίσουν ένα πολυγωνικό πλέγμα. Σε μεγάλη πλειοψηφία τα 3D μοντέλα, φτιάχνονται και με υφή, καθώς η σημερινή τεχνολογία υπολογιστών επιτρέπει γρήγορο rendering. Ωστόσο, το μειονέκτημα της πολυγωνικής σχεδίασης είναι ότι όλα τ' αντικείμενα αποτελούνται από μικροσκοπικές επίπεδες επιφάνειες και για να προσεγγίσει τις καμπυλωτές επιφάνειες χρησιμοποιούνται πολλά πολύγωνα.

- Non-uniform rational basis spline (N.U.R.B.S.) modeling
  Οι NURBS είναι επιφάνειες, οι οποίες, καθορίζονται από καμπύλες splines και επηρεάζονται από σημεία ελέγχου με «βάρος». Η καμπύλη ακολουθεί (αλλά δεν παρεμβάλλει απαραιτήτως), τα σημεία. Αυξάνοντας το βάρος ενός σημείου, έχει ως αποτέλεσμα η καμπύλη να πλησιάσει το σημείο αυτό. Οι NURBS είναι βασικά, λείες επιφάνειες και όχι προσεγγίσεις που χρησιμοποιούν μικρές επίπεδες επιφάνειες. Αυτό τις καθιστά ιδιαίτερα κατάλληλες για βιολογική μοντελοποίηση, όπως ζώα και ανθρώπους. Τα πιο γνωστά εμπορικά λογισμικά που χρησιμοποιούν NURBS εγγενώς, είναι το *Maya*, το *3d Rhino* και το *solidThinking*.

- Splines και Patches modeling
  Ο όρος «spline» συχνά αποδίδεται ως, παραμετρική καμπύλη. Η απλότητα της κατασκευής τους, τα καθιστά μια δημοφιλή επιλογή κατά το σχεδιασμό ψηφιακών μοντέλων. Ο αλληλεπιδραστικός σχεδιασμός καμπύλων, επιτρέπει την εύκολη χειραγώγησή τους. Τα Splines μπορούν να χρησιμοποιηθούν είτε για μονοδιάστατες, είτε για πολυδιάστατες οντότητες. Splines και Patches εξαρτώνται από κυρτωμένες γραμμές για τον καθορισμό της ορατής επιφάνειας. Η χρήση Patches είναι ένας συνδυασμός πολυγωνικής σχεδίασης και NURBS.

- Primitive modeling
  Αυτή η διαδικασία χρησιμοποιεί έτοιμα στοιχειώδη γεωμετρικά σχήματα, όπως κυλίνδρους, σφαίρες, κώνους, κύβους και άλλα δομικά στοιχεία για πιο πολύπλοκα μοντέλα. Τα οφέλη είναι, η γρήγορη και εύκολη κατασκευή σωμάτων. Τα σχήματα είναι μαθηματικά ορισμένα, και ως εκ τούτου είναι απολύτως ακριβή. Αυτή η τεχνική μοντελοποίησης είναι κατάλληλη για τεχνικές εφαρμογές και όχι για οργανικά σχήματα.

- Sculpt modeling
  Αν και αρκετά νέα μέθοδος, δικαίως έχει γίνει πολύ δημοφιλής. Επί του παρόντος, υπάρχουν δύο τύποι sculpt modeling. Η τεχνική μετατόπισης και η τεχνική ογκομετρικής.

Όλες αυτές οι μέθοδοι έχουν πλεονεκτήματα και μειονεκτήματα. Στο ερώτημα ποια είναι η καλύτερη, η απάντηση εξαρτάται από το τι θέλουμε να σχεδιάσουμε καθώς και από το πόσο λεπτομερές θέλουμε να είναι.

3.5. Φωτορεαλισμός

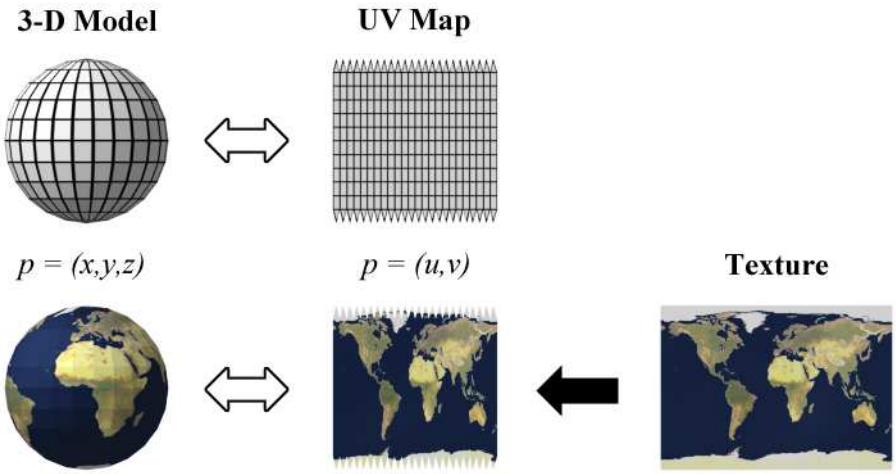
Εικόνα 8. Φωτορεαλιστική αναπαράσταση αντικειμένων

3.5.1. Οργάνωση σκηνής

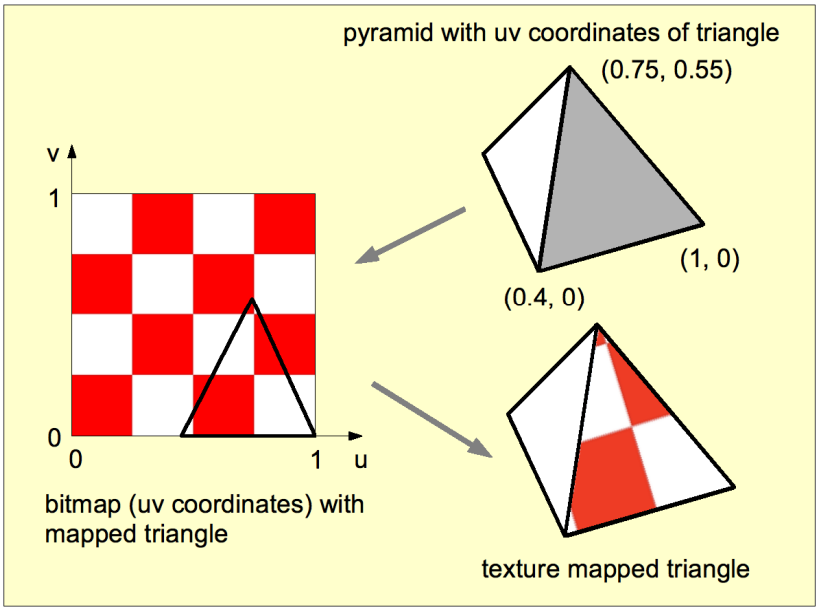
Το στάδιο μοντελοποίησης αποτελείται από τη διαμόρφωση επιμέρους εικονικών αντικειμένων, τα οποία στη συνέχεια χρησιμοποιούνται στη σκηνή. Η οργάνωση της σκηνής, περιλαμβάνει την τακτοποίηση των αντικειμένων και την τοποθέτηση άλλων οντοτήτων, όπως κάμερες και φώτα. Ο φωτισμός είναι μια σημαντική πτυχή, που συντελεί στη σωστή απεικόνιση της σκηνής. Όπως συμβαίνει στην πραγματικότητα, ο φωτισμός είναι ένας σημαντικός συμβάλλοντας παράγοντας στην προκύπτουσα αισθητική και οπτική ποιότητα. Υπό τον σωστό φωτισμό, τα μοντέλα μπορούν να προσεγγίσουν ρεαλιστικότερα αποτελέσματα.

3.5.2. Χρώμα και υφή

Τέλος, είναι συνήθως επιθυμητό η προσθήκη υφής και χρώματος στην επιφάνεια των αντικείμενων. Κάποιες εφαρμογές επιτρέπουν την επιλογή χρώματος καθώς τ' αντικείμενα σχεδιάζονται. Η πιο κοινή μέθοδος προσθήκης υφής σ' ένα τρισδιάστατο μοντέλο είναι η εφαρμογή μιας δισδιάστατης εικόνας στην επιφάνεια του μοντέλου, μέσω μιας διαδικασίας αποκαλούμενης «χαρτογράφηση υφής» (texture mapping). Οι εικόνες αυτές, δε διαφέρουν σε τίποτα, από άλλες ψηφιακές εικόνες. Κατά τη διάρκεια της διαδικασίας χαρτογράφησης, ειδικά κομμάτια πληροφορίας (texture coordinates ή UV coordinates) αποθηκεύονται, προσδιορίζοντας ποια σημεία της επιφάνειας του μοντέλου θα καλυφτούν, από ποια μέρη της δισδιάστατης εικόνας. Η χρήση δισδιάστατων εικόνων, καταφέρνει να δώσει την αίσθηση πιστής αναπαράστασης υφής.



Εικόνα 9. Texture mapping



Εικόνα 10. Συντεταγμένες UV

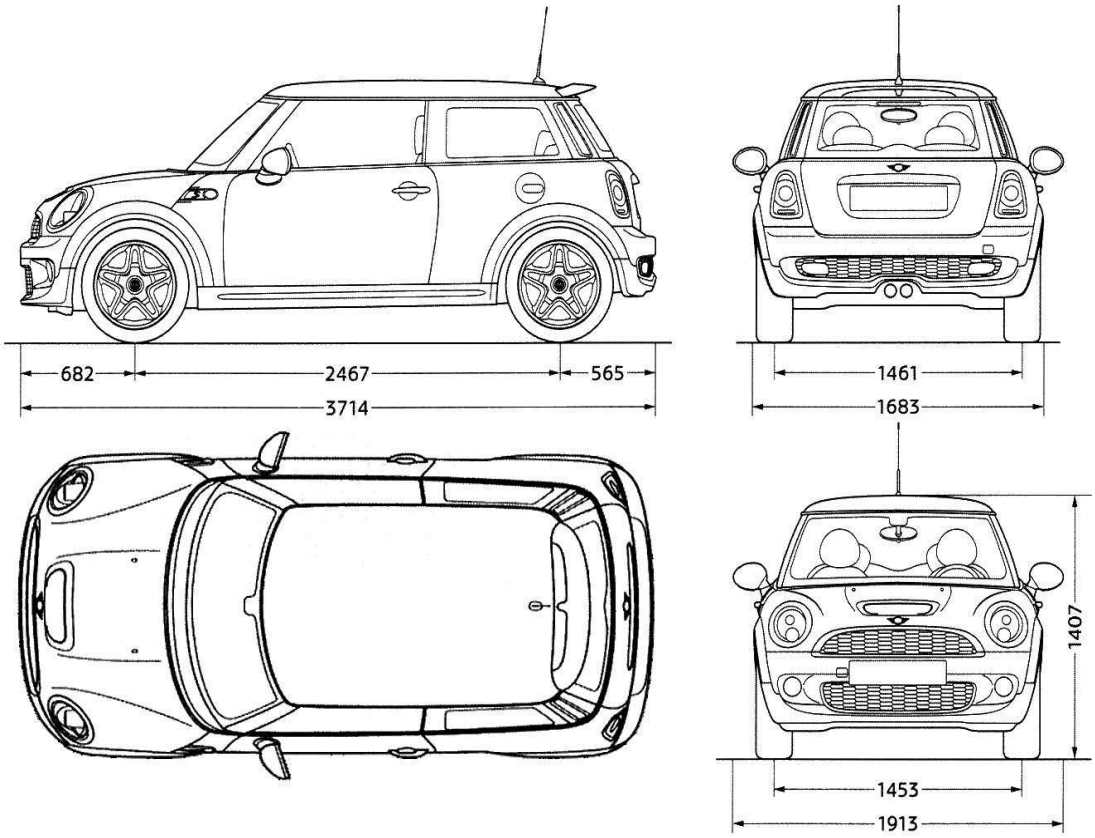
4. Οδηγός κατασκευής ηλεκτρονικού 3d παιχνιδιού

Παρακάτω βλέπουμε πως μπορούμε να κατασκευάσουμε ένα απλό τρισδιάστατο βιντεοπαιχνίδι με τη δυνατότητα να «τρέξει» σε πρόγραμμα περιήγησης (web browser). Αρχικά γίνεται χρήση του Photoshop, μετέπειτα του *3ds max 8* και τέλος του *director*, το οποίο είναι ειδικό για διαδικτυακά παιχνίδια.

4.1. Επεξεργασία εικόνας με Photoshop

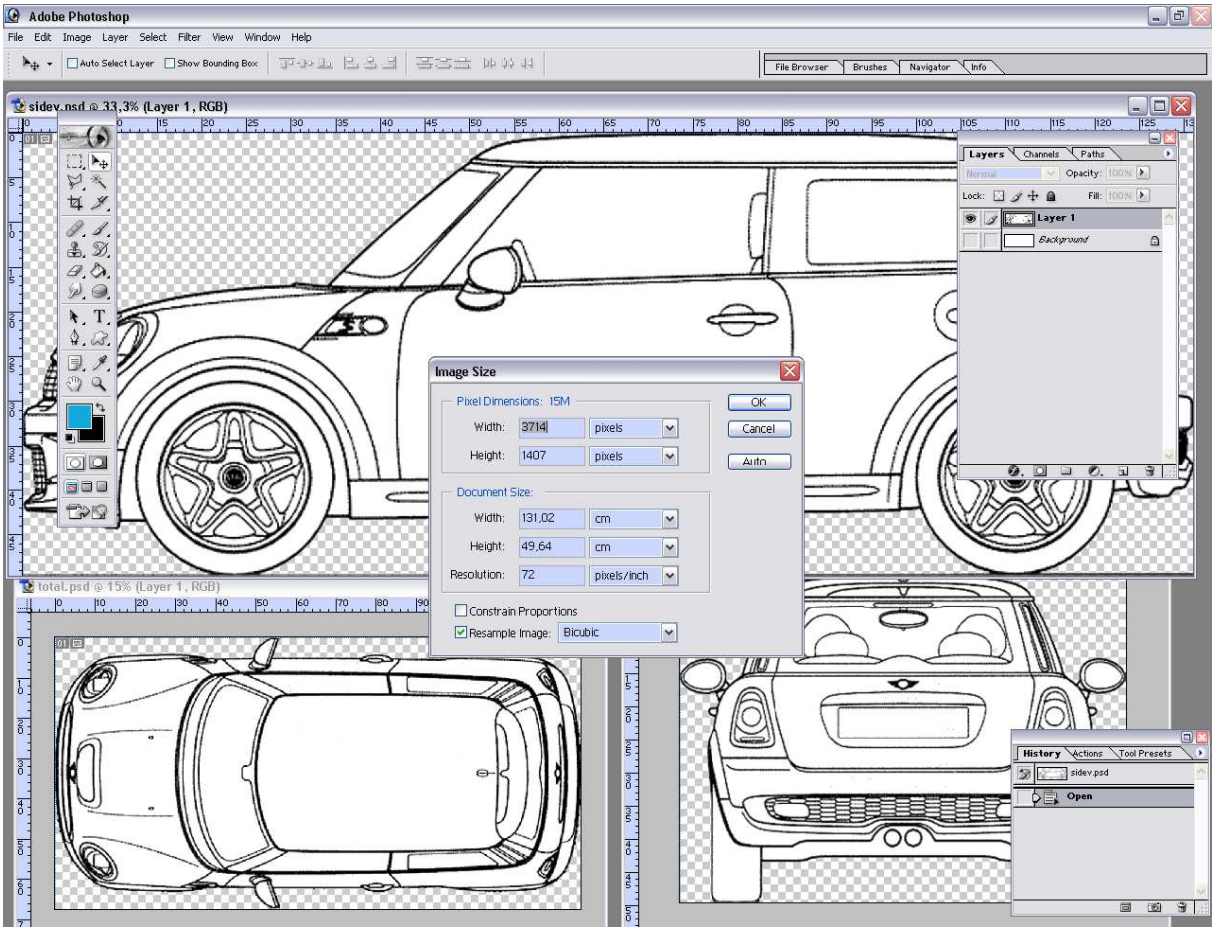
Το *adobe Photoshop* χρησιμοποιείται κατά κόρον από γραφίστες και φωτογράφους. Είναι από τα καλύτερα στην κατηγορία του. Ένα πρόγραμμα πολύ χρηστικό, με πολλές εργαλειοθήκες και χρωματικές παλέτες, οι οποίες βοηθούν στη δημιουργία λογότυπων, διαφημιστικών εξώφυλλων, αφισών, μουσικών booklets και άλλων. Επίσης είναι χρήσιμο στην επεξεργασία φωτογραφίας. Μπορεί το Photoshop είχε σχετικά σύντομο ρόλο στην όλη διαδικασία, αυτό όμως, δε το καθιστά λιγότερο σημαντικό. Αν και η επεξεργασία εικόνας προσφέρεται και από άλλες εφαρμογές, η ευχρηστία και οι ποικιλία λειτουργιών του λογισμικού, ήταν οι αιτίες επιλογής αυτού του προγράμματος.

Στην παρακάτω εικόνα (βλ. Εικόνα 11) βλέπουμε τέσσερις διαφορετικές όψεις ενός αυτοκινήτου, κάτοψη, όψη, αριστερή και πίσω πλευρά. Δηλαδή τις πραγματικές διαστάσεις του, αλλά σε μικρότερη κλίμακα. Στο διαδίκτυο υπάρχουν πολλές βιβλιοθήκες με σχέδια αντικειμένων σε διαφορετικές κλίμακες ονομαζόμενα και ως «blueprints». Αφού «κατεβάσουμε» το επιθυμητό blueprint, το ανοίγουμε με το πρόγραμμα. Έπειτα ξεχωρίζουμε την κάθε όψη για να δημιουργήσουμε τέσσερα διαφορετικά jpeg αρχεία.



Εικόνα 11. Blueprint

Παρατηρούμε τις διαστάσεις, έτσι ώστε κάθε αρχείο να είναι ακριβές και να συνάπτει με τις διαστάσεις των άλλων όψεων. Στο συγκριμένο αρχείο η μονάδα μέτρησης είναι τα pixel. Στην εικόνα 12, διακρίνονται οι διαστάσεις της αριστερής όψης και η αναλογία pixel/εκατοστών. Δηλαδή 3714pixel = 131,02cm. Είναι απαραίτητο να σημειώσουμε τις διαστάσεις κάθε αρχείου, ώστε να προχωρήσουμε στο επόμενο βήμα. Ολοκληρώνοντας αυτό το στάδιο, αποθηκεύουμε τις εικόνες με προτιμώμενο φορμάτ το jpeg, διότι έχει την καλύτερη αναλογία συμπίεσης/ποιότητας. Στο επόμενο κεφάλαιο, θα δούμε πως γίνεται εισαγωγή των εικόνων στο 3ds max, το οποίο υποστηρίζει jpeg καθώς και άλλα φορμάτ εικόνας.



Εικόνα 12. Blue print στο Photoshop

4.2. Autodesk 3d studio max 8

4.2.1. Περιγραφή

Το 3ds Max είναι ένα ευρέως γνωστό πρόγραμμα που χρησιμοποιείται στη μοντελοποίηση τρισδιάστατων αντικειμένων και κόσμων με φωτορεαλιστική απόδοση. Οι δυνατότητές του είναι απεριόριστες, καθώς διαθέτει πολλές εργαλειοθήκες, αρκετά plug-ins

και extras. Αποτελεί ένα σημαντικό εργαλείο στην αρχιτεκτονική, στην παραγωγή βιντεοπαιχνιδιών, στη δημιουργία διαφημιστικών σποτ, ταινιών και ειδικών εφέ. Η νέα έκδοση παρέχει ακόμα:

- **Εκτενής εργαλειοθήκη μοντελοποίησης 3D**
  Περισσότερα από 100 προηγμένα πολυγωνικά εργαλεία μοντελοποίησης και ελεύθερης σχεδίασης 3D.

- **Σκίαση και υφή**
  Μεγάλη ποικιλία επιλογών για υφή, χρωματισμό, χαρτογράφηση και στρώσεις.

- **Κινούμενα σχέδια**
  Εξελιγμένη εργαλειοθήκη για τη δημιουργία χαρακτήρων και κινουμένων σχεδίων 3D υψηλής ποιότητας.

- **Δυναμική, εφέ και προσομοίωση**
  Εργαλειοθήκες υψηλής απόδοσης, καταξιωμένες κατά την παραγωγική διαδικασία για τη δημιουργία δυναμικών καταστάσεων και εφέ.

- **Πανίσχυρες δυνατότητες απόδοσης (rendering)**

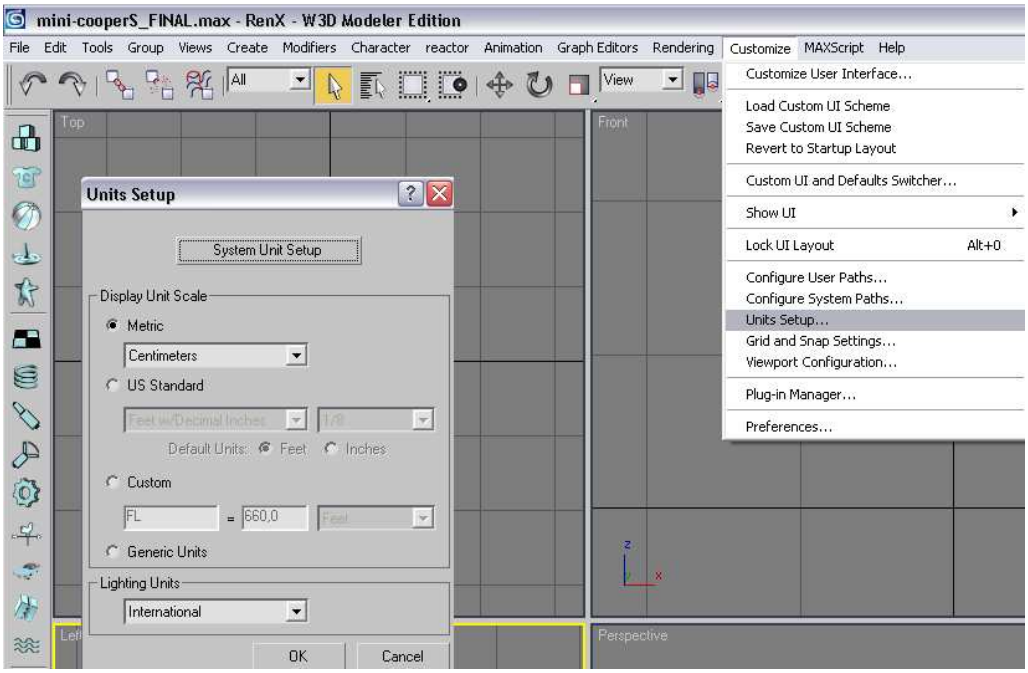
- **Ολοκλήρωση στη γραμμή παραγωγής**
  Εισαγωγή δεδομένων από πολλές και διαφορετικές πηγές και μεταφορά δεδομένων 3ds Max και 3ds Max Design μέσα από επαναληπτικές διαδικασίες αρχείων, εφαρμογές λογισμικού, χρήστες και στάδια εργασιών.

- **Συνεργατική ροή εργασιών**
  Συλλογή και διαμοιρασμός δεδομένων σε πολύπλοκες σκηνές, δίνοντας τη δυνατότητα σε πολλαπλούς χρήστες να συμβάλλουν αποτελεσματικά στη ροή των εργασιών.

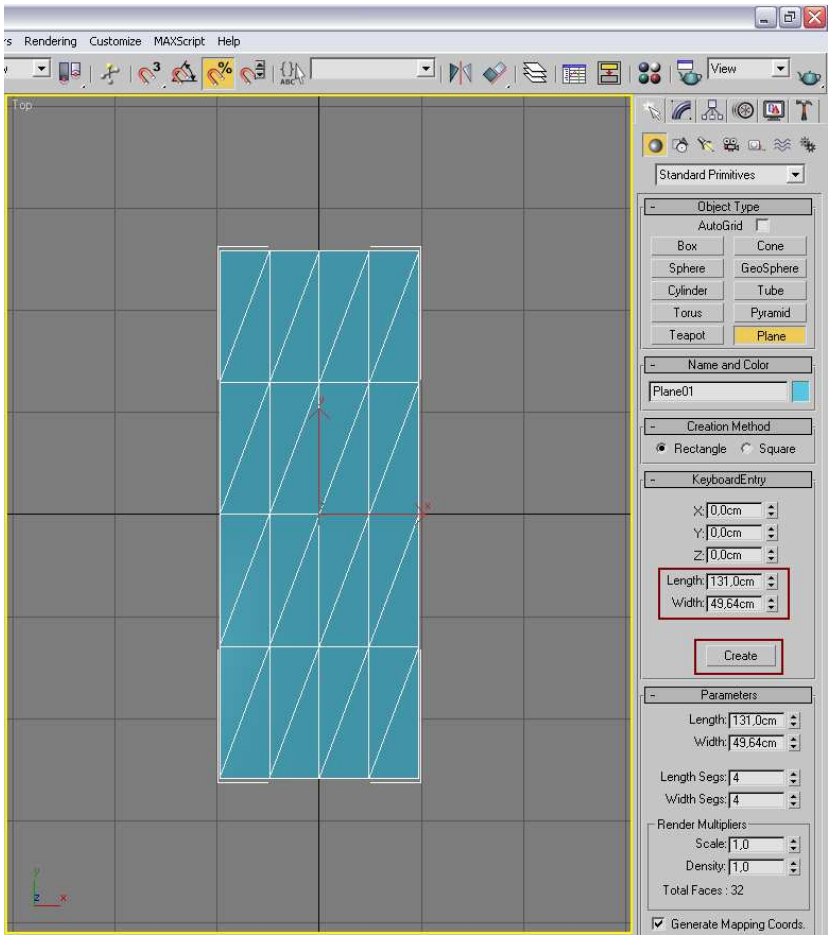
4.2.2. Μοντελοποίηση με 3ds max

Το 3ds max αν και δεν είναι ιδιαίτερα εύχρηστο, ανταμείβει με τα εκπληκτικά αποτελέσματα. Κάθε επαγγελματίας, οφείλει να διαθέτει τις γνώσεις ώστε να εκμεταλλεύεται εις στο έπακρο τις απεριόριστες δυνατότητες που έχει.

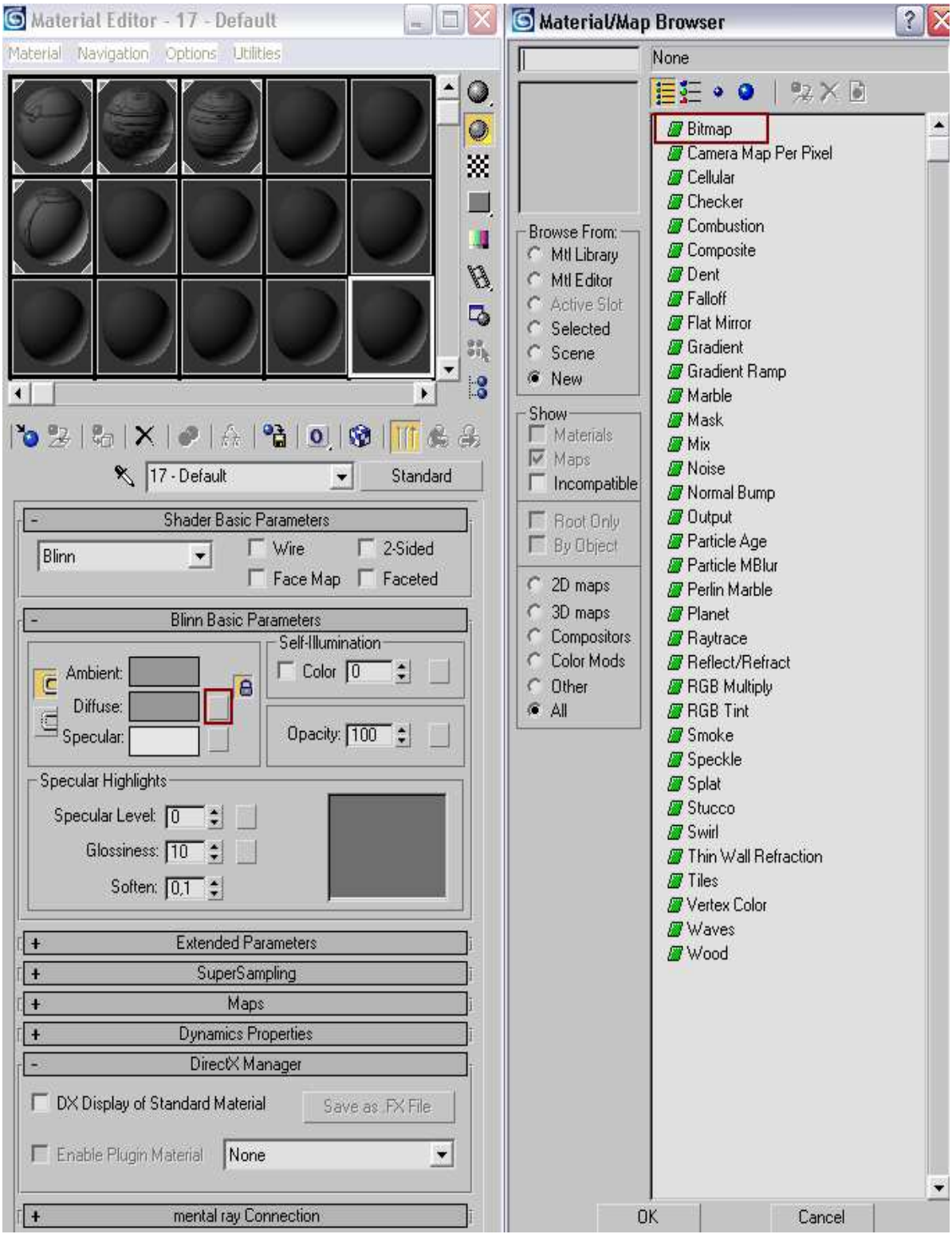
Ξεκινώντας, μετατρέπουμε τις μονάδες μέτρησης του συστήματος, στην προκειμένη περίπτωση, σε cm (βλ. Εικόνα 13). Έπειτα δημιουργούμε τέσσερις επίπεδες επιφάνειες. Από την εργαλειοθήκη «create» πατάμε το κουμπί «plane». Προσέχουμε ώστε η διάσταση του κάθε plane να είναι ανάλογη της κάθε εικόνας (βλ. Εικόνα 14). Σε κάθε ένα από τα planes, προσθέτουμε, ως texture, τις εικόνες που επεξεργαστήκαμε στο Photoshop. Για συντομία πατάμε απ' το πληκτρολόγιο, «m». Το νέο παράθυρο που ανοίγει έχει τίτλο «material editor». Επιλέγουμε μια άδεια θέση (slot) και στη συνέχεια κάνουμε κλικ στο κουμπί που βρίσκεται δίπλα στη λέξη «diffuse». Από το νέο παράθυρο, επιλέγουμε «bitmap» και ψάχνουμε τις αποθηκευμένες jpeg εικόνες (βλ. Εικόνα 15).



Εικόνα 13. Αλλαγή μονάδας μέτρησης



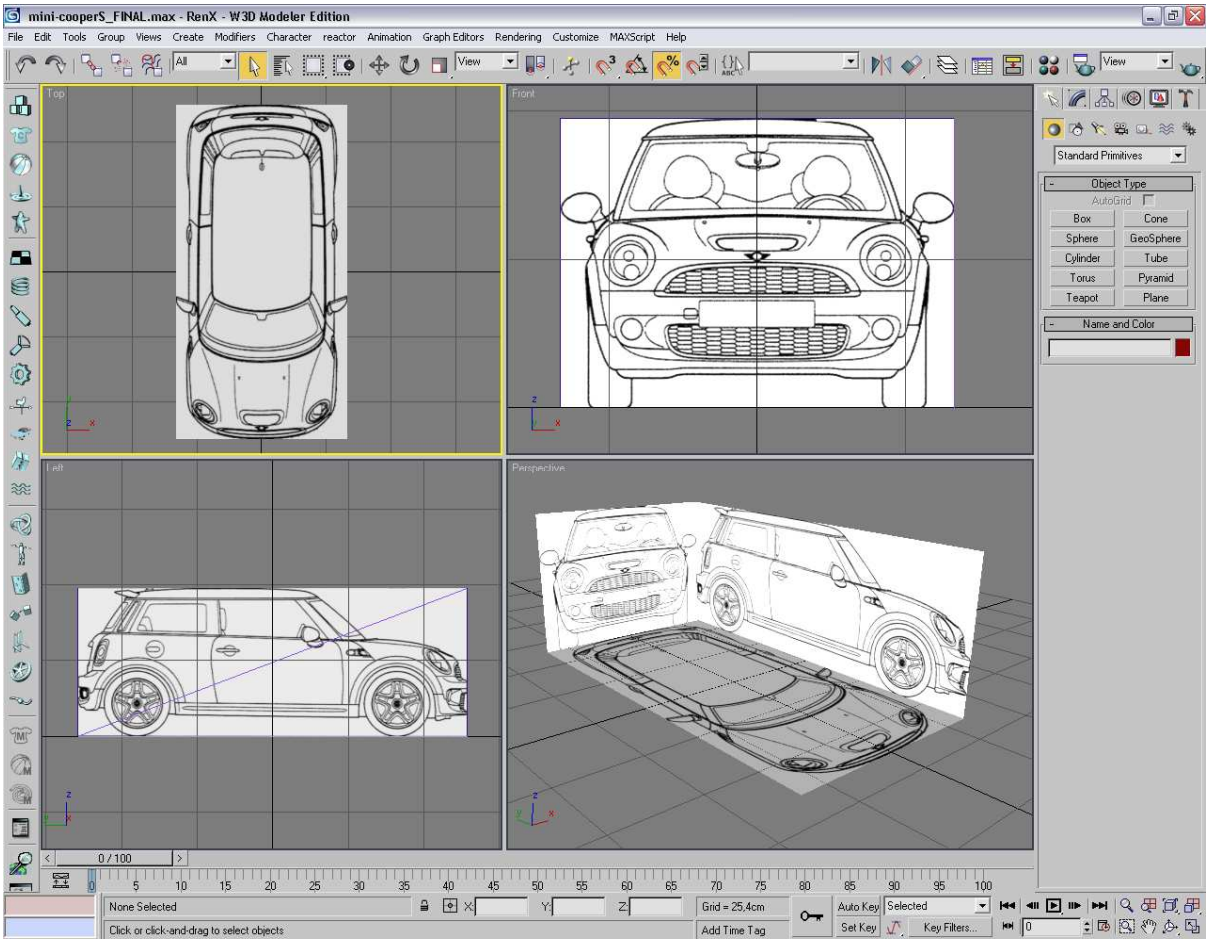
Εικόνα 14. Δημιουργία plane



Εικόνα 15. Material Editor

Περιστρέφοντας τις επίπεδες επιφάνειες με το κουμπί rotate  από το κυρίως μενού και τραβώντας τις έτσι ώστε να είναι κοντά η μια στην άλλη, σχηματίζεται ένα ορθογώνιο παραλληλεπίπεδο με τέσσερις ορατές έδρες, όπως φαίνεται στην εικόνα 16. Αυτό το σχήμα θ' αποτελέσει στη συνέχεια τον οδηγό, τον οποίο ακολουθώντας τον, θα έχουμε μια πιστή εικονική αναπαράσταση του οχήματος. Έπεται η επιλογή της μεθόδου σχεδίασης,

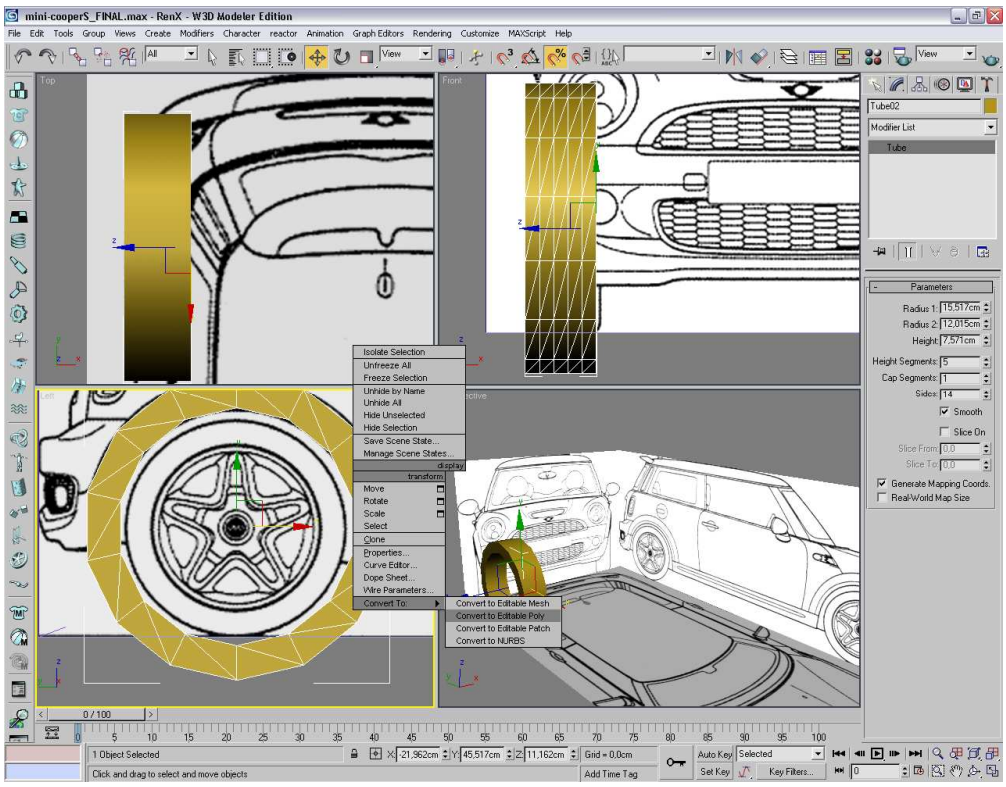
στην προκειμένη περίπτωση είναι η μέθοδος της πολυγωνικής σχεδίασης, καθώς είναι από τις πιο δημοφιλής και επιτρέπει ακρίβεια και άμεση διόρθωση.



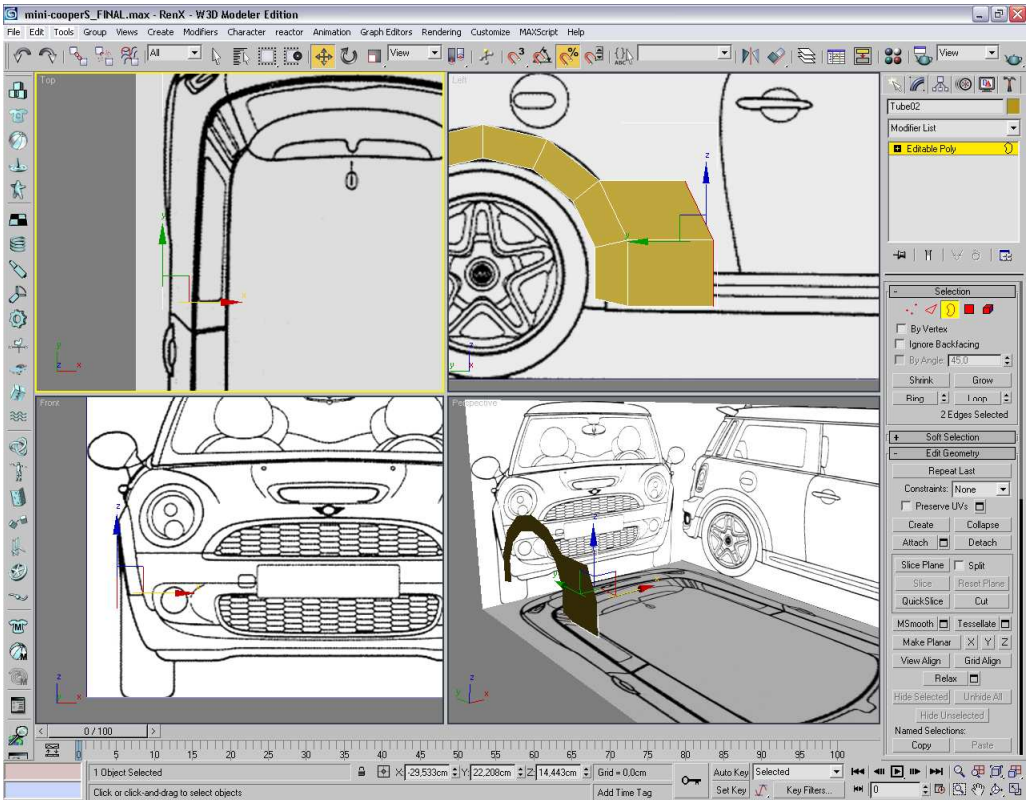
Εικόνα 16. Οδηγός σχεδίασης

Ξεκινάμε κατά προτίμηση, από τη δεξιά οπτική γωνία, δηλαδή κοιτάμε τη δεξιά όψη του αυτοκινήτου. Από το «create panel», επιλέγουμε το κουμπί «tube» έτσι δημιουργούμε ένα στοιχειώδη σωλήνα. Προσέχουμε ώστε τα όρια να είναι σύμφωνα με την εικόνα. Δηλαδή να μη ξεπερνούν, τα όρια του πλαϊνού προφυλακτήρα που βρίσκεται πάνω από τη δεξιά πίσω ρόδα. Δημιουργούμε τον σωλήνα με κέντρο, το κέντρο της ρόδας. Κοιτάζοντας και τις άλλες οπτικές γωνίες (πάνω και μπροστά) ως οδηγό, μετακινούμε τον σωλήνα στο σωστό σημείο, όπως φαίνεται στην εικόνα 17, με το πλήκτρο μετακίνησης . Με δεξί κλικ πάνω στο αντικείμενο επιλέγουμε «editable poly». Έτσι ο σωλήνας μετατρέπεται σ' ένα έτοιμο προς επεξεργασία αντικείμενο, αποτελούμενο από πολλά πολύγωνα. Στόχος είναι να φτιάξουμε πρώτα το σασί του αυτοκίνητου και μετά τις ρόδες. Στη συνέχεια πατάμε το κουμπί «polygon» από το «selection» πάνελ, κι επιλέγουμε όλα τ' άχρηστα πολύγωνα. Αφού τα μαρκάρουμε, πατάμε το πλήκτρο «delete», έτσι ώστε, να μείνει ένα επίπεδο καμπυλωτό σχήμα (βλ. Εικόνα 18). Ξανά από το «selection» πάνελ, βρίσκουμε το κουμπί «edge» και μαρκάρουμε τις δύο τελευταίες δεξιές ακμές. Αφού επιλέξουμε το αντικείμενο και πατήσουμε το κουμπί μετακίνησης «select and move» , κρατώντας πατημένο το πλήκτρο shift μετακινούμε ελαφρώς τον κέρσορα προς τα δεξιά. Παρατηρούμε ότι με αυτό το τρόπο δημιουργούνται νέα πολύγωνα. Το 3ds max, μας επιτρέπει να επεξεργαστούμε κάθε πολύγωνο, κορυφή και ακμή δίνοντας στον χρήστη μια πληθώρα εργαλείων π.χ. μπορούμε

να προσθέσουμε, ν' αφαιρέσουμε παραπάνω στοιχεία, να τα μετακινήσουμε ή και να τα ενώσουμε μεταξύ τους.

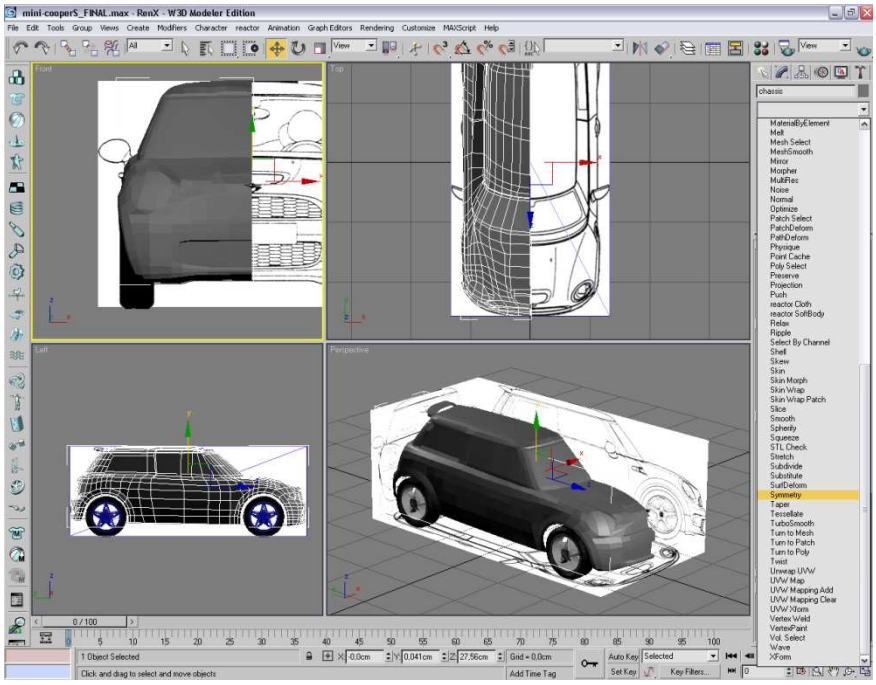


Εικόνα 17. Μετατροπή tube σε editable poly

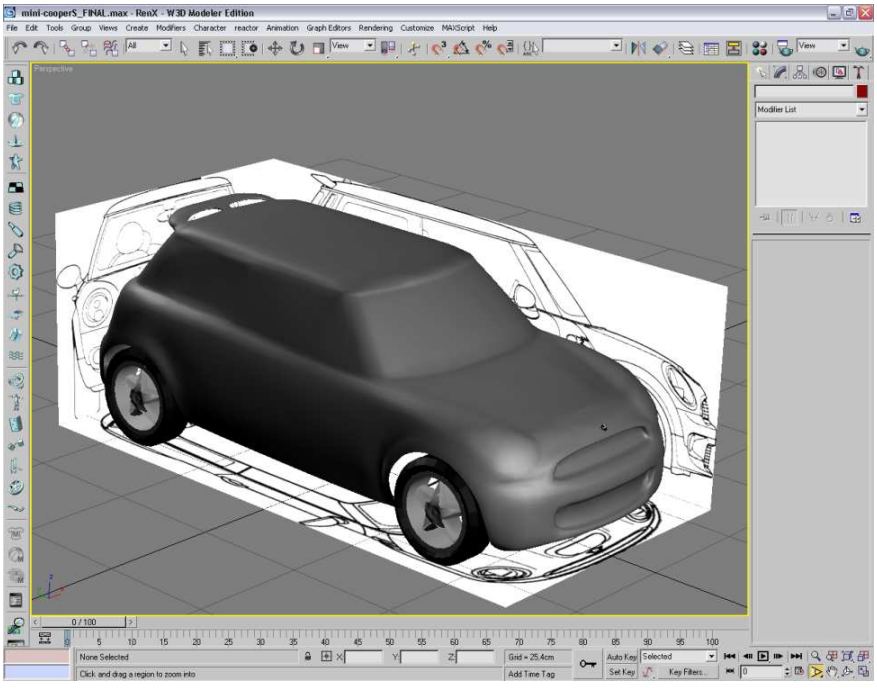


Εικόνα 18. Επεξεργασία ακμών

Αξίζει να σημειωθεί, ότι στη μοντελοποίηση συνηθίζεται, τ' αντικείμενα να σχεδιάζονται μόνο κατά το ήμισυ. Ο λόγος είναι ότι, σχεδόν όλα τα πακέτα εφαρμογών επιτρέπουν στο χρήστη, να συμπληρώσει αυτόματα το άλλο μισό, χρησιμοποιώντας διάφορα εργαλεία. Ανοίγοντας το «modifier list» επιλέγουμε «Symmetry» και στις παραμέτρους επιλέγουμε τον άξονα Z (βλ. Εικόνα 19). Αφού το εφαρμόσουμε το μοντέλο γίνεται πλήρης. Με τον ίδιο τρόπο δημιουργούμε τις ρόδες. Αφού ολοκληρωθεί η διαδικασία σχεδίασης, έχουμε ένα αρκετά πιστό αντίγραφο του οχήματος (βλ. Εικόνα 20).



Εικόνα 19. Συμμετρία



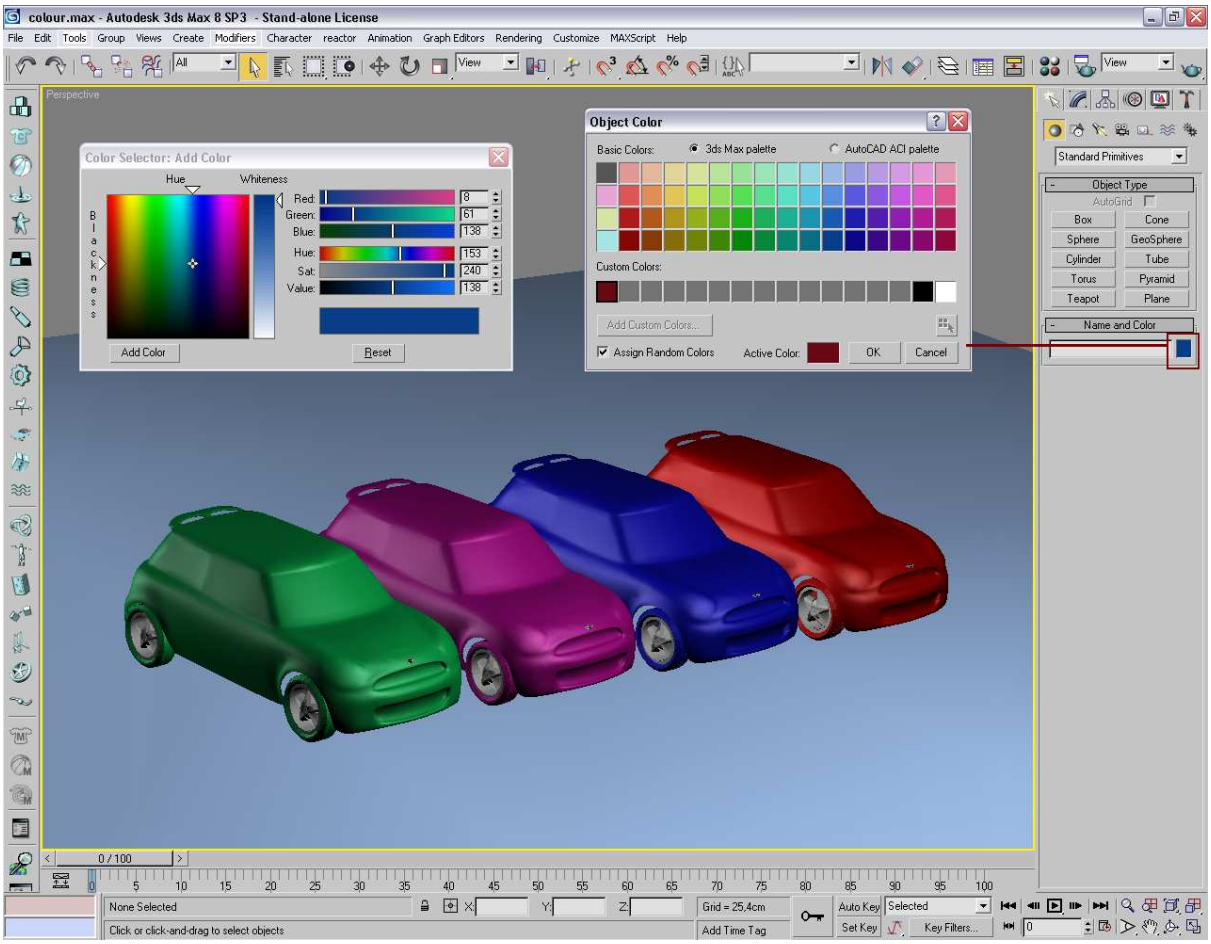
Εικόνα 20. Ολοκληρωμένο 3d μοντέλο αυτοκινήτου

4.2.3. Φωτορεαλιστική αναπαράσταση

Το χρώμα και η υφή είναι ουσιαστικά, αυτά τα οποία δίνουν ζωή και ταυτότητα στο εικονικό μοντέλο.

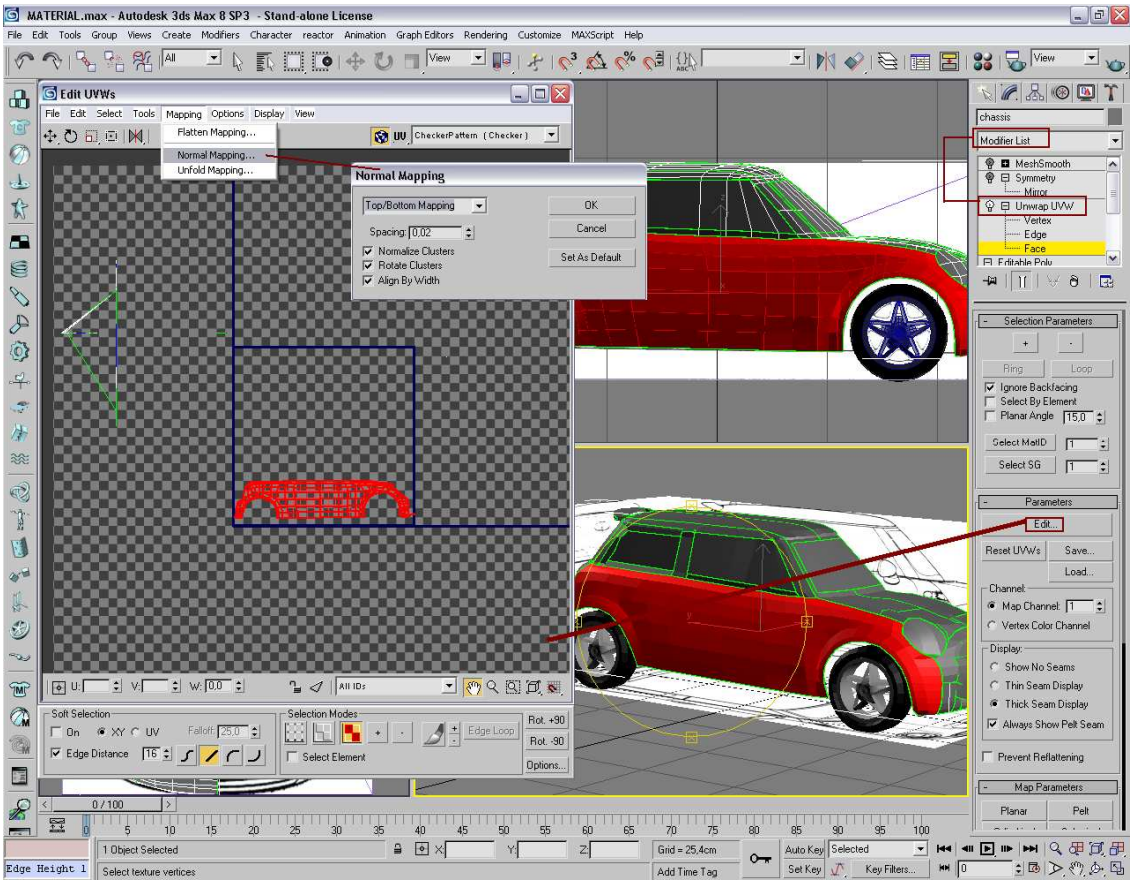
4.2.3.1. Χρώμα και υφή

Παρατηρούμε ότι, κατά τη δημιουργία έτοιμων μοντέλων (κύβοι, πυραμίδες κ.ο.κ.), το 3ds max, επιλέγει αυτόματα ένα τυχαίο χρώμα, για κάθε μοντέλο. Φυσικά, ο χρήστης μπορεί να επιλέξει κάποιο άλλο, από τη χρωματική παλέτα που παρέχεται. Αυτός είναι και ο πιο απλός τρόπος ανάθεσης και επεξεργασίας χρώματος αντικειμένου (βλ. Εικόνα 21). Συνηθίζεται όμως, τα πιο πολύπλοκα μοντέλα ν' απαρτίζονται από πολλά κομμάτια, διαφορετικού χρώματος και υφής. Έτσι το πρόβλημα που υφίσταται είναι, το πώς θα χρωματίσουμε τα τμήματα ξεχωριστά. Υπάρχουν πολλές τεχνικές για τον καθορισμό χρώματος. Μια απλή λύση είναι, η ομαδοποίηση των πολυγώνων και η ανάθεση διαφορετικού χρώματος ανά ομάδα. Πατώντας ***m*** ή  ανοίγουμε το πάνελ «material editor» (βλ. Εικόνα 15). Από εκεί πατάμε στο γκρίζο πλαίσιο που βρίσκεται δίπλα στη λέξη «diffuse» και επιλέγουμε άλλο χρώμα από την αναδυόμενη παλέτα.



Εικόνα 21. Παλέτες χρώματος

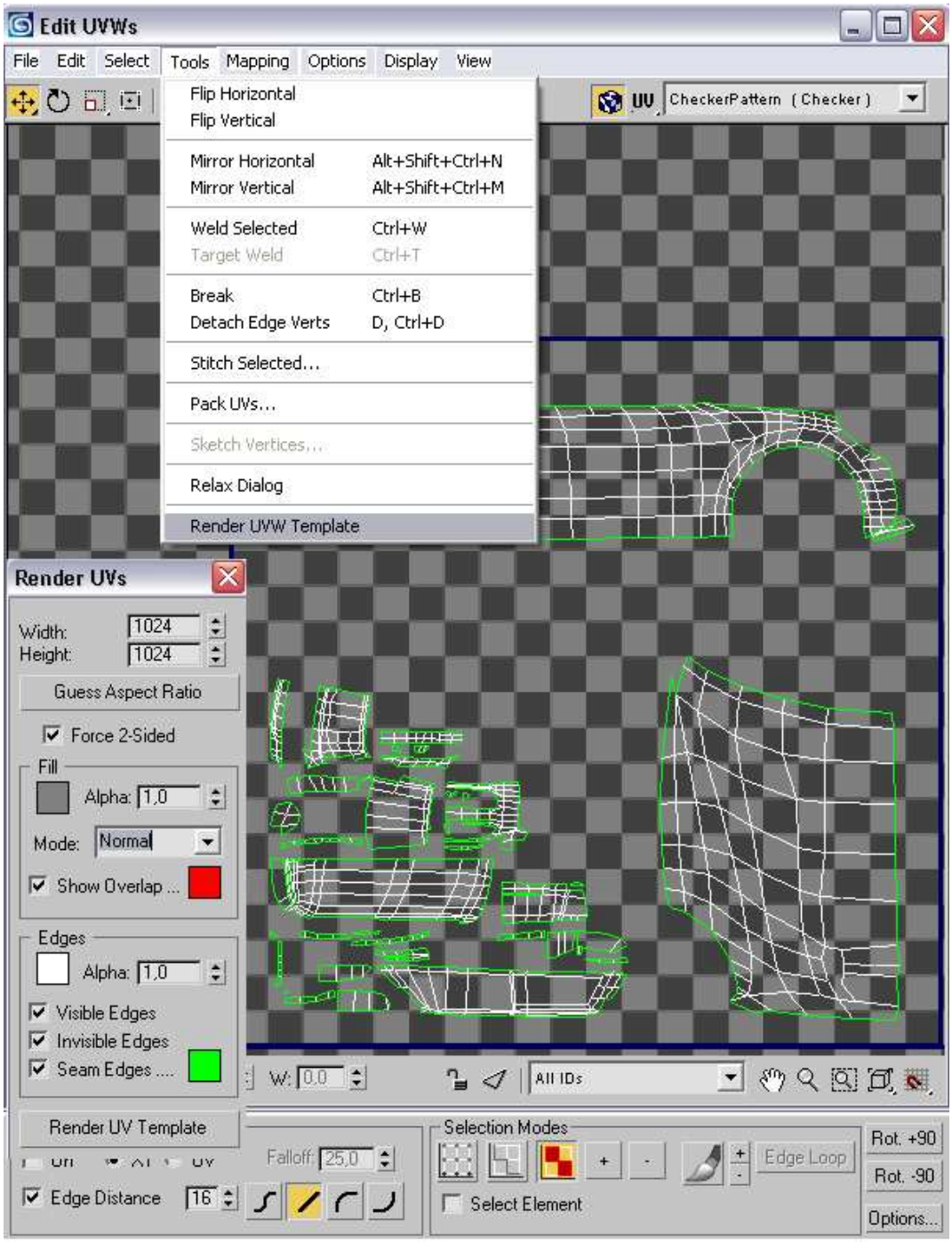
Μια δημοφιλής αλλά πολύπλοκη τεχνική (Unwrap UVW), είναι να «ξετυλίξουμε» το περίβλημα του μοντέλου και να επεξεργαστούμε την εικόνα που θα προκύψει στο Photoshop. Αφού επιλέξουμε το σασί, πατάμε «modifier list», και από το roll-on menu βρίσκουμε το «unwrap UVW». Θέλοντας να ξεχωρίσουμε την πλαϊνή μεριά του αυτοκινήτου, πατάμε «face» και μαρκάρουμε όλα τα πολύγωνα (βλ. Εικόνα 22). Στο μενού των παραμέτρων πατάμε «edit». Στο νέο παράθυρο διακρίνεται με κόκκινο, το επιλεγμένο τμήμα, αλλά όχι ευκρινώς. Από το μενού «Mapping», πατάμε «Normal mapping» κι επιλέγουμε «Top/Bottom mapping». Στην εικόνα που προκύπτει, μπορούμε να διακρίνουμε πλέον ξεκάθαρα το πλαϊνό τμήμα του αυτοκινήτου μαζί με ένα σύνολο ακαθόριστων σχημάτων που αποτελούν τα υπόλοιπα μέρη του οχήματος. Με την ίδια λογική μπορούμε να ξεχωρίσουμε όλα τα κομμάτια του αυτοκίνητου. Στην εικόνα 23 βρίσκονται ξεχωριστά το καπό και το πλαϊνό τμήμα. Η εργαλειοθήκη που παρέχεται από κάτω είναι σημαντική, αφού επιτρέπει την άμεση επεξεργασία θέσης κορυφών και ακμών, όπως επίσης και άλλες λειτουργίες. Από το μενού του ίδιου παραθύρου στο «tools» πατάμε «UVW render». Αλλάζουμε το mode σε normal και πατάμε «Render UV template» (βλ. Εικόνα 23). Αυτό που βλέπουμε είναι ουσιαστικά το «ξεδιπλωμένο» περίβλημα του οχήματος.



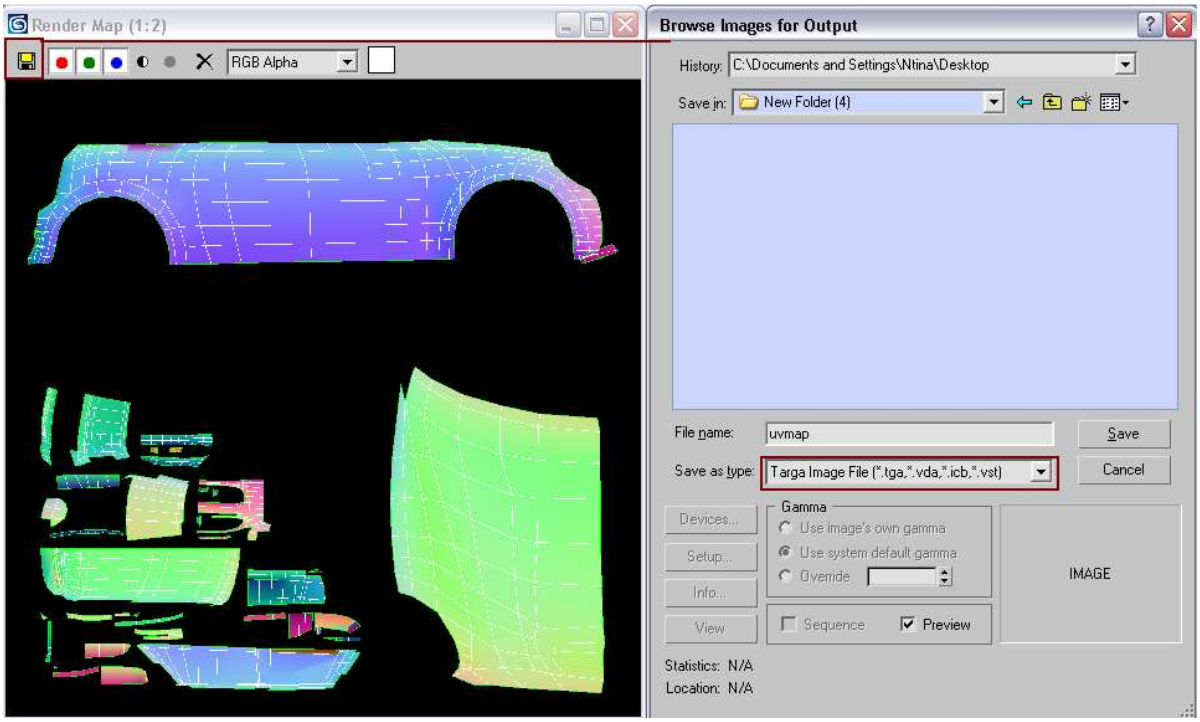
Εικόνα 22. Unwrap UVW

Τα ανοιχτόχρωμα μέρη είναι τα σημεία, όπου πέφτει περισσότερο φώς. Αποθηκεύουμε την νέα εικόνα με φορματ Targa Image (βλ. Εικόνα 24). Στη συνέχεια θα δούμε πόσο καλά συνεργάζονται Photoshop και 3ds max. Εισάγουμε τις εικόνες στο Photoshop και με την προϋπόθεση ότι θα επεξεργαστούμε την εικόνα αρκετές φορές, είναι προτιμώμενο να αποθηκεύσουμε το αρχείο ως Photoshop αρχείο (φορματ .psd). Επεξεργαζόμαστε την εικόνα, για παράδειγμα, σχεδιάζοντας διάφορα σχήματα στο καπό. Αποθηκεύουμε τις αλλαγές κι επιστρέφουμε στο 3ds max. Μαρκάρουμε το σασί και ανοίγουμε το material editor. Πατάμε

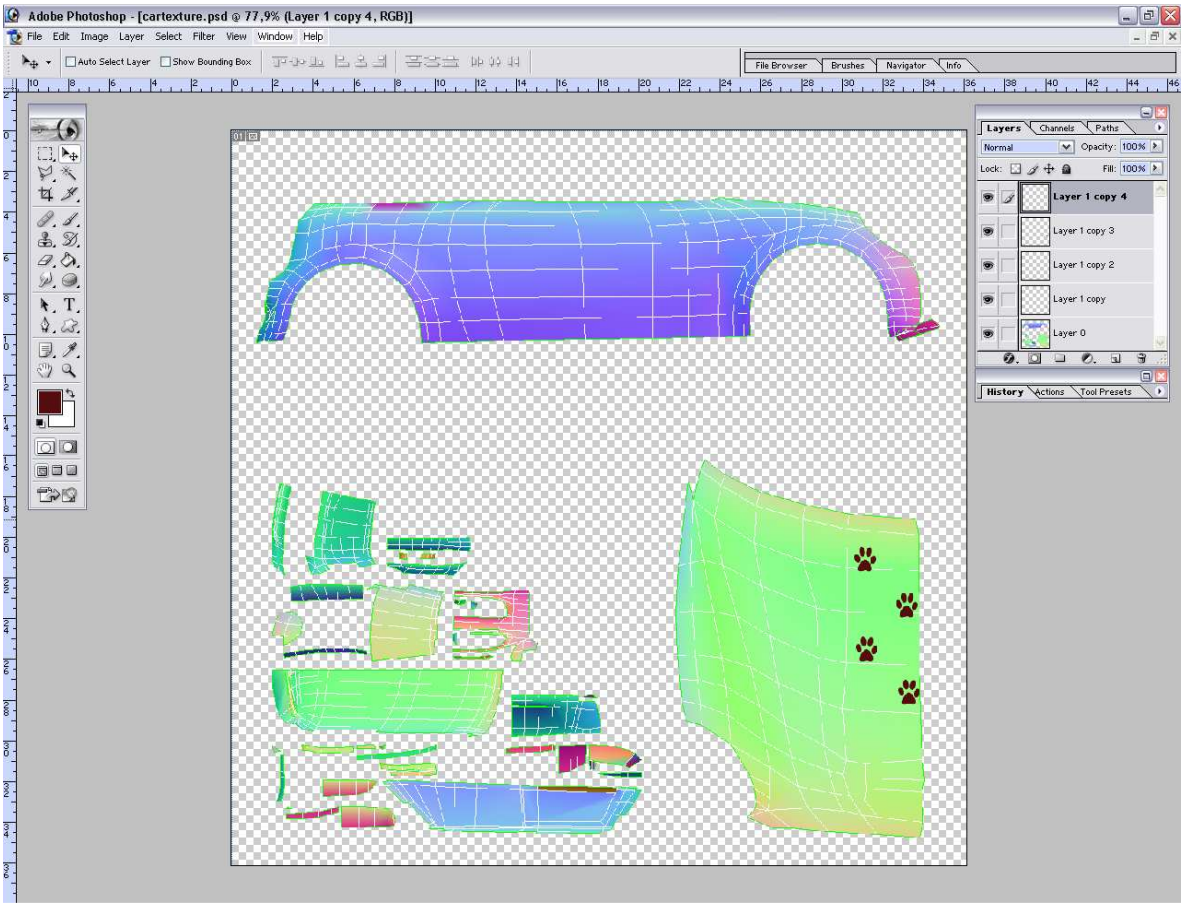
το γκρι κουμπί δίπλα από το diffuse, επιλέγουμε bitmap και αναθέτουμε στο μοντέλο το αρχείο με φορματ Photoshop. Πατώντας το κουμπί «show map in viewport» από το material editor, βλέπουμε το διαφοροποιημένο περίβλημα του αμαξιού. Κάθε επεξεργασία που υφίσταται η εικόνα στο Photoshop, είναι ορατή στο 3ds max, αμέσως μετά την αποθήκευση των αλλαγών. Επαναλαμβάνοντας τη διαδικασία, βλέπουμε στην εικόνα 26 το αποτέλεσμα. Όταν εξοικειωθούμε με όλη αυτή τη τεχνική, θα διαπιστώσουμε πόσο σημαντική είναι η συνεργασία Photoshop και 3ds max. Με αυτό τον τρόπο μπορούμε όχι μόνο ν' αλλάξουμε χρώμα στ' αντικείμενα αλλά να δημιουργήσουμε και τα δικά μας υλικά, όπως μέταλλο, ξύλο ύφασμα κ.α. με τα οποία θα «ντύσουμε» τα μοντέλα μας.



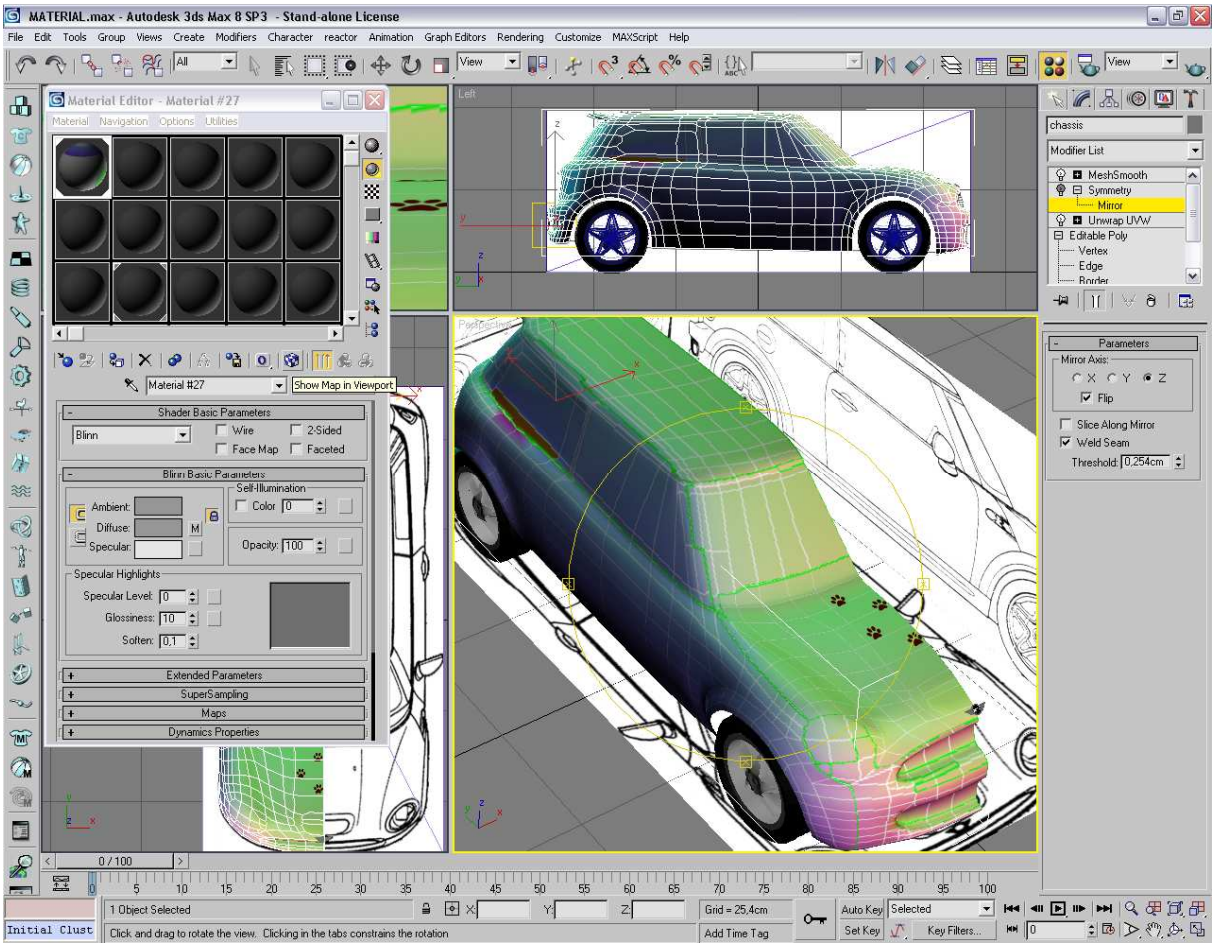
Εικόνα 23. Επεξεργασία UVW



Εικόνα 24. Αποθήκευση ως Targa file



Εικόνα 25. Επεξεργασία επιφάνειας αυτοκινήτου, στο Photoshop



Εικόνα 26. Άμεσα αποτελέσματα στο 3ds max

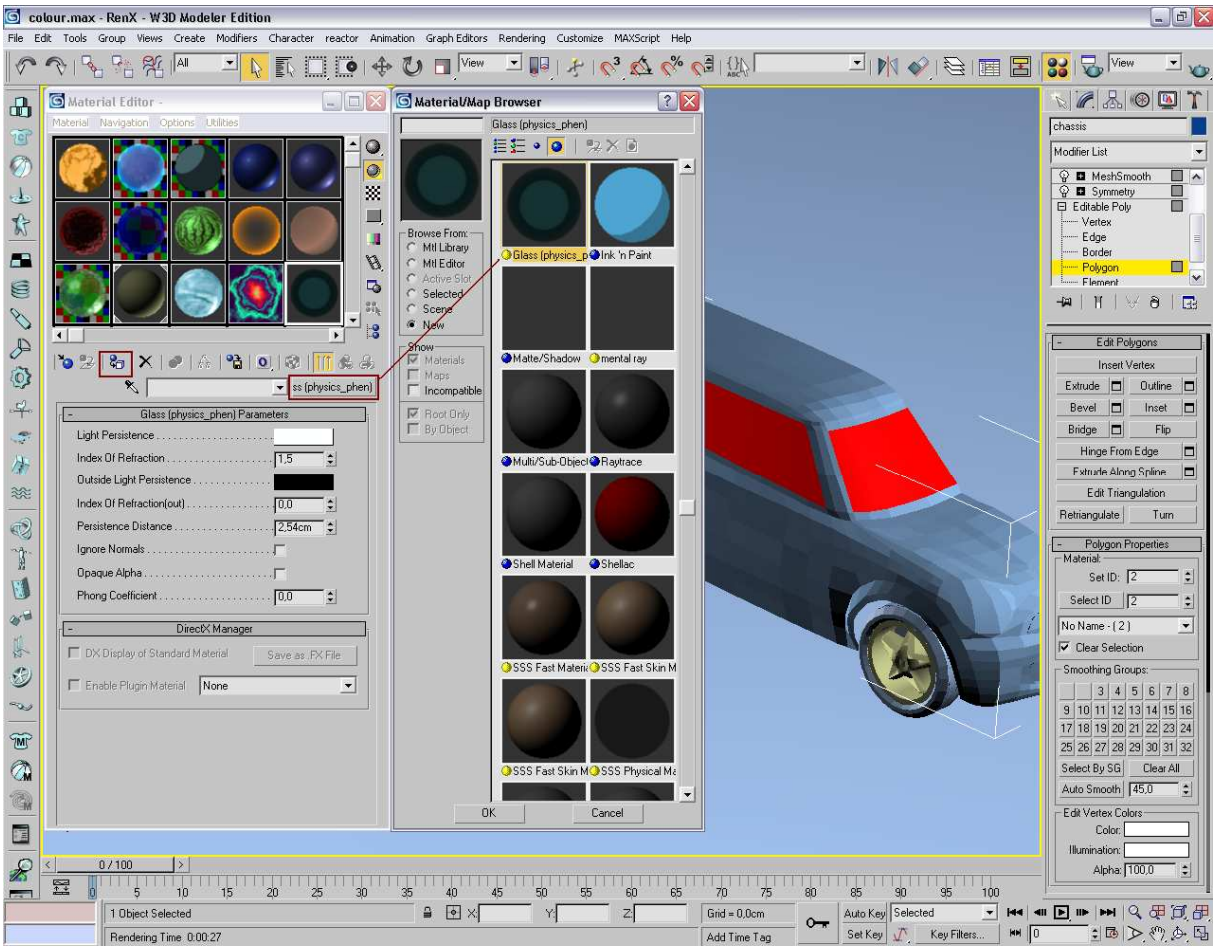
Ένα αυτοκίνητο είναι κατασκευασμένο κυρίως από μέταλλο. Η εικονική απόδοση του μετάλλου και κάθε υλικού, γίνεται με την σωστή επεξεργασία εικόνων και τον κατάλληλο φωτισμό.

Παραπάνω είδαμε πως επεξεργαζόμαστε εικόνες για τη δημιουργία υλικών, χρησιμοποιώντας παράλληλα και μια άλλη εφαρμογή. Το 3ds max παρέχει μια συλλογή από έτοιμα υλικά, που διακρίνονται απ' το πάνελ «material editor» (βλ. Εικόνα 15). Μπορούμε είτε να τα επεξεργαστούμε περεταίρω μέσω ρυθμίσεων όπως specular και diffuse ή να προσθέσουμε χάρτες (maps), όπως bump για αίσθηση βαθουλωμάτων, reflection για λείες επιφάνειες κ.α.

Η εργαλειοθήκη υλικών του 3ds max είναι πανίσχυρη, αφού παρέχει απεριόριστες λειτουργίες,. Η μεταλλική μπλε απόδοση του οχήματος, έγινε μετά από προσθήκη νέων εικόνων (maps) και την επεξεργασία όλων αυτών. Πρώτα το μοντέλο χωρίζεται σε τμήματα διαφορετικού υλικού. Αυτό γίνεται, ομαδοποιώντας τα πολύγωνα του κάθε τμήματος και αναθέτοντας σε κάθε ομάδα πολυγώνων ένα υλικό. Πιο συγκεκριμένα το παρμπρίζ είναι από γυαλί, ο προφυλακτήρας από πλαστικό η μάσκα από γυαλιστερό μέταλλο και τα φώτα από σκληρό πλαστικό. Προσπαθούμε να προσομοιώσουμε όλα αυτά τα υλικά.

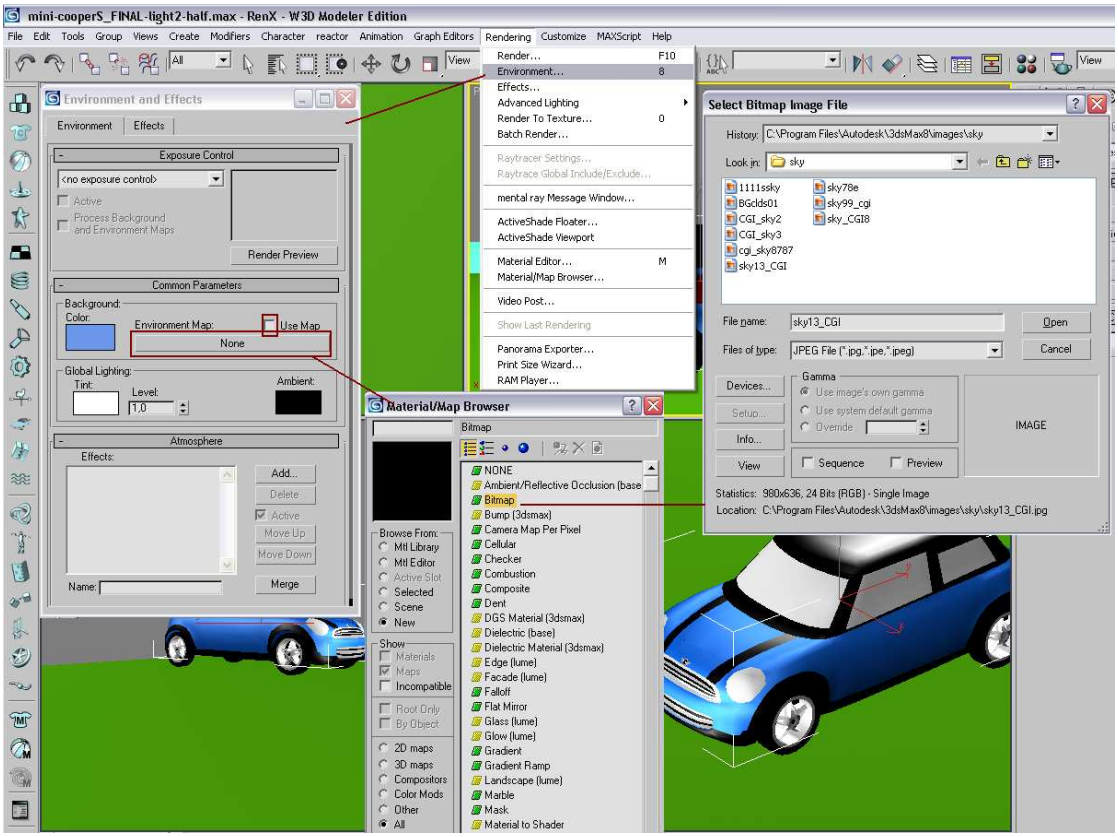
Επιλέγουμε το μοντέλο πατάμε από το selection πάνελ, polygon  και μαρκάρουμε τα πολύγωνα ενός τμήματος. Στην εικόνα 27, είναι μαρκαρισμένα τα πολύγωνα των παραθύρων, του μπροστινού και του πίσω παρμπρίζ. Με αυτό τον τρόπο ξεχωρίσαμε τα τμήματα τα οποία θα «ντυθούν» στη συνέχεια με γυαλί. Στο διαδίκτυο μπορούμε να βρούμε πολλές δωρεάν βιβλιοθήκες με έτοιμα υλικά. Μια από αυτές φαίνεται στην παρακάτω εικόνα. Ανοίγουμε ξανά το «material editor» πατάμε «standard», επιλέγουμε «glass» (αν

υπάρχει στην έκδοση που δουλεύουμε) και πατάμε το κουμπί «assign material to Selection» .

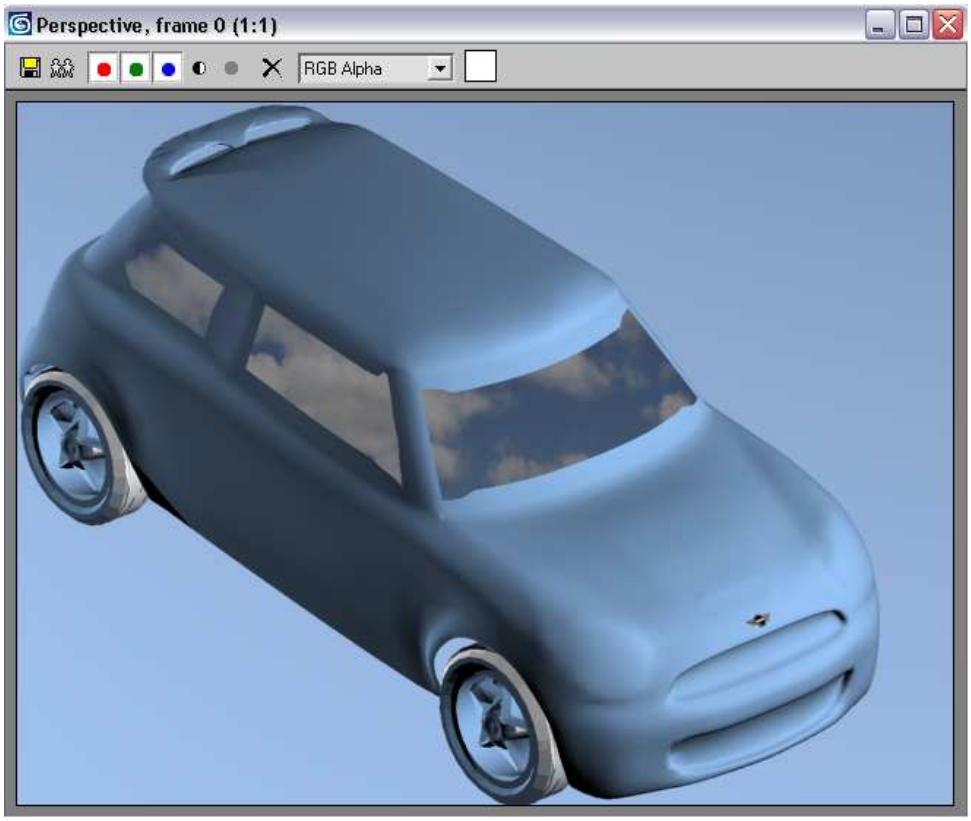


Εικόνα 27. Ανάθεση υλικού σε ομάδα πολυγώνων

Μια από τις ιδιότητες του γυαλιού είναι η αντανάκλαση αντικειμένων πάνω σε αυτό. Από το μενού «Rendering» επιλέγουμε environment. Από το νέο παράθυρο, στην κατηγορία «common parameters» πατάμε το κουμπί «none» κι επιλέγουμε μια εικόνα. Αυτή η εικόνα θα καθορίζει το περιβάλλον του εικονικού κόσμου. Εδώ, επιλέχτηκε μια φωτογραφία με θέμα σύννεφα (βλ. Εικόνα 28). Τέλος, από το πληκτρολόγιο πατάμε F9 για να ελέγξουμε το αποτέλεσμα. Στην εικόνα 29 παρατηρούμε ότι τώρα τα σύννεφα, αντανακλούνται σε όλα τα παράθυρα. Δηλαδή σε όλα τα τμήματα, όπου είχε διοριστεί το υλικό του γυαλιού. Με τον ίδιο τρόπο διαμορφώνουμε όλα τα υπόλοιπα υλικά. Η απεικόνιση όλων αυτών των στοιχείων γίνεται πάντα μέσω μιας διαδικασίας ονόματι *rendering*. Rendering είναι η διαδικασία απεικόνισης της τελικής μορφής του βιντεοπαιχνιδιού στην οθόνη. Το rendering χωρίζεται σε δύο βασικές κατηγορίες: 1) το Real-time rendering, κατά το οποίο, τα μοντέλα εμφανίζονται στην οθόνη και για το αποτέλεσμα που δημιουργείται εκείνη τη δεδομένη χρονική στιγμή, γίνεται χρήση του hardware και 2) το Pre-rendering, κατά το οποίο, το τελικό αποτέλεσμα είναι προκατασκευασμένο από κάποιο άλλο σύστημα, δε γίνεται δηλαδή χρήση του hardware, παίζει απλά το ρόλο του video player. Ο υπολογιστής, αναλύει τους μαθηματικούς αλγόριθμους και αναπαράγει μια εικόνα, της σκηνής, από την οπτική της κάμερας.



Εικόνα 28. Επιλογή περιβάλλοντος εμφανές κατά το Rendering



Εικόνα 29. Τα σύννεφα αντανακλούν στα τζάμια

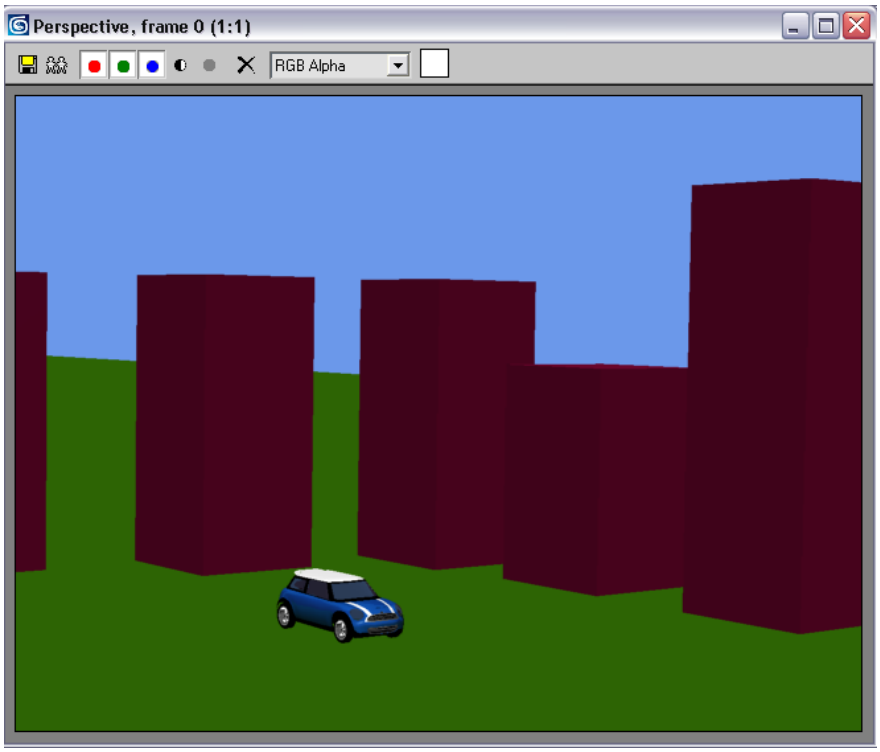
4.2.3.2. Φωτισμός

Οι στόχοι του φωτισμού, στα 3D γραφικά υπολογιστών, είναι λιγότερο ή περισσότερο οι ίδιοι με τους στόχους του πραγματικού φωτισμού. Ο φωτισμός λειτουργεί ως μέσο ανάδειξης του σχήματος των αντικειμένων που απεικονίζονται, από την οπτική της κάμερας. Δίνει μια δισδιάστατη εικόνα στην οθόνη, με την ψευδαίσθηση της τρίτης διάστασης σε βάθος. Παρέχει στην εικόνα, προσωπικότητα και χαρακτήρα. Μία επιδέξια παραλλαγή, του φωτισμού της σκηνής, μπορεί να αναδείξει το συναίσθημα της ευτυχίας, της θλίψης, του φόβου κλπ. Μαζί με προσωπικότητα και χαρακτήρα, ο φωτισμός γεμίζει μια σκηνή με συναισθήματα τα οποία διαβιβάζονται άμεσα στον θεατή.

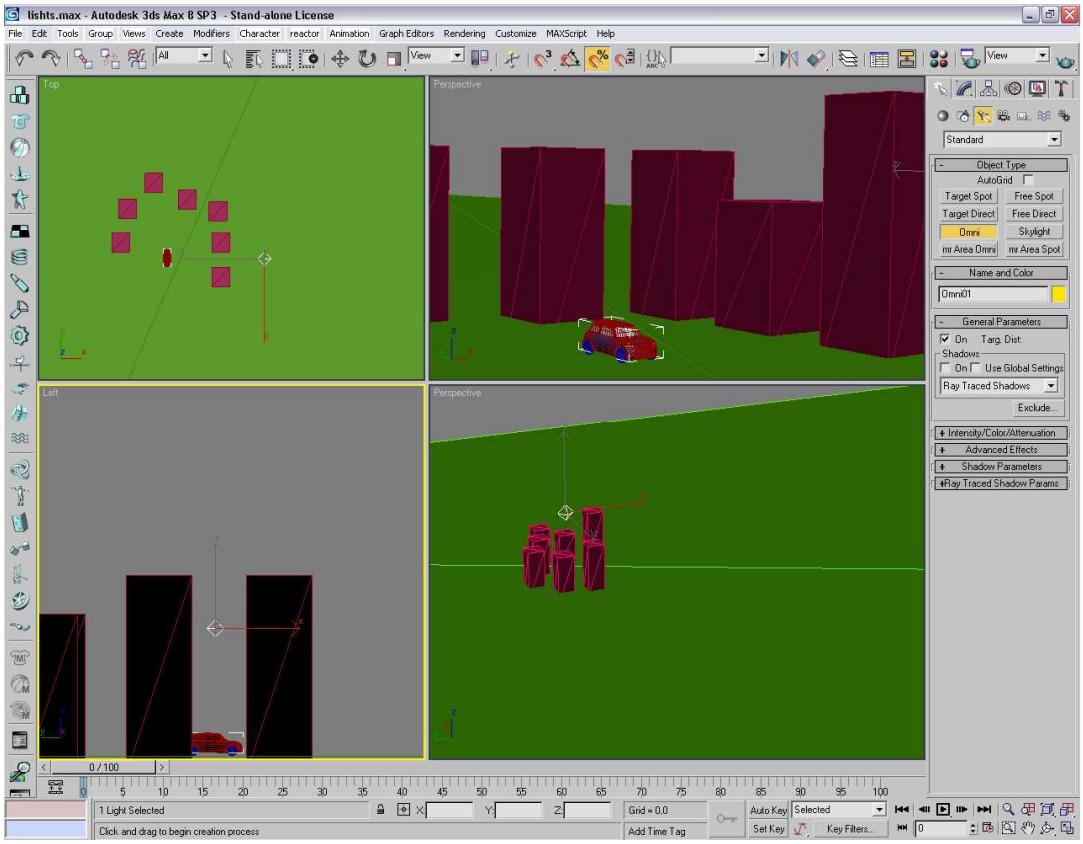
Η προσπάθεια που χρειάζεται για την προσομοίωση πραγματικού περιβάλλοντος με τεχνητά μέσα, μπορεί να είναι αποθαρρυντική. Κάνοντας τις 3D απεικονίσεις να φαίνονται φωτορεαλιστικές είναι ένα δύσκολο έργο. Γι' αυτό είναι απαραίτητο να επεξεργαστούμε σωστά κάθε φωτεινή πηγή, λαμβάνοντας υπ' όψιν τέσσερις βασικές συνιστώσες:

- Ένταση
- Κατεύθυνση
- Χρώμα
- Μέγεθος

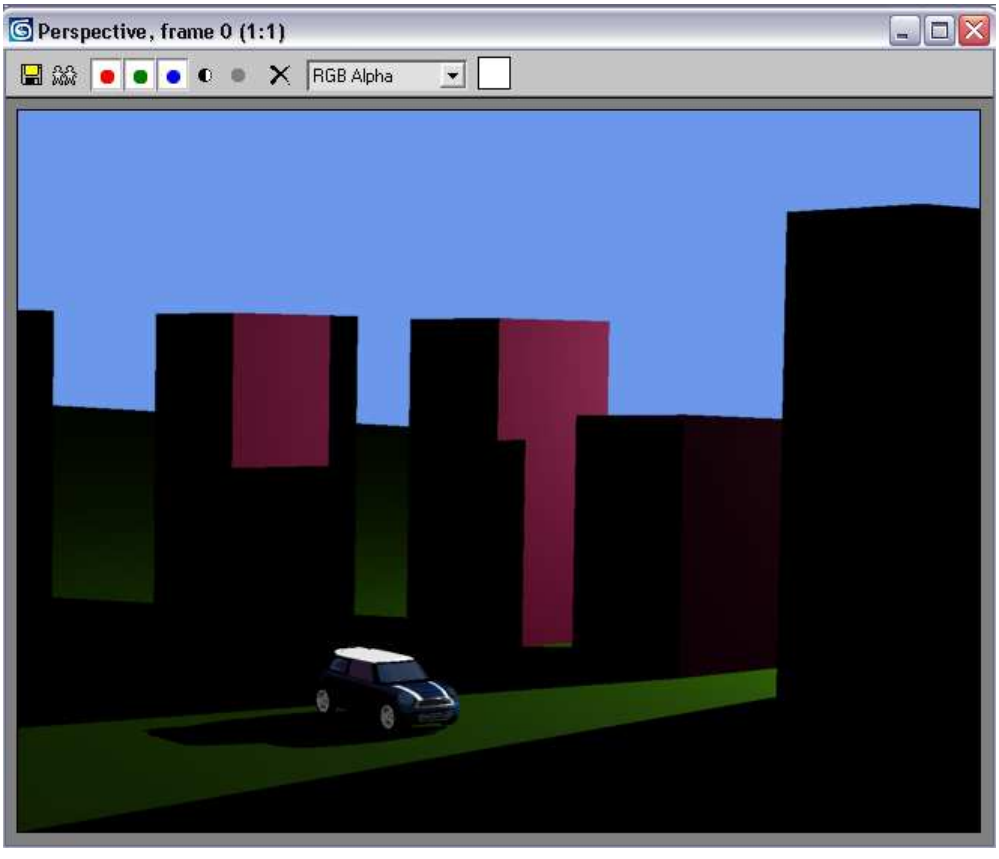
Στο 3ds max υπάρχει εξαρχής, ένας υποτυπώδης φωτισμός στη σκηνή. Ωστόσο, πατώντας F9 παρατηρούμε ότι δεν υπάρχουν σκιές (βλ. Εικόνα 30). Από το πάνελ «create» επιλέγουμε την υποκατηγορία «lights». Εκεί βλέπουμε όλες τις πιθανές πηγές φωτός που μπορούμε να προσθέσουμε και τις παραμέτρους που μπορούμε να ρυθμίσουμε. Πατώντας «omni» και κοιτώνας το «top view», τοποθετούμε ένα είδος φωτισμού, όπου επιθυμούμε στη σκηνή (βλ. Εικόνα 31). Μετακινώντας το Omni έχουμε διαφορετικά αποτελέσματα. Συγκρίνοντας την εικόνα 30 με την εικόνα 32, βλέπουμε τις διαφορές, ανάμεσα στον αρχικό φωτισμό, και στον φωτισμό της ίδιας σκηνής, με μια πηγή φωτός.



Εικόνα 30. Rendering με αρχικό φωτισμό του 3ds max

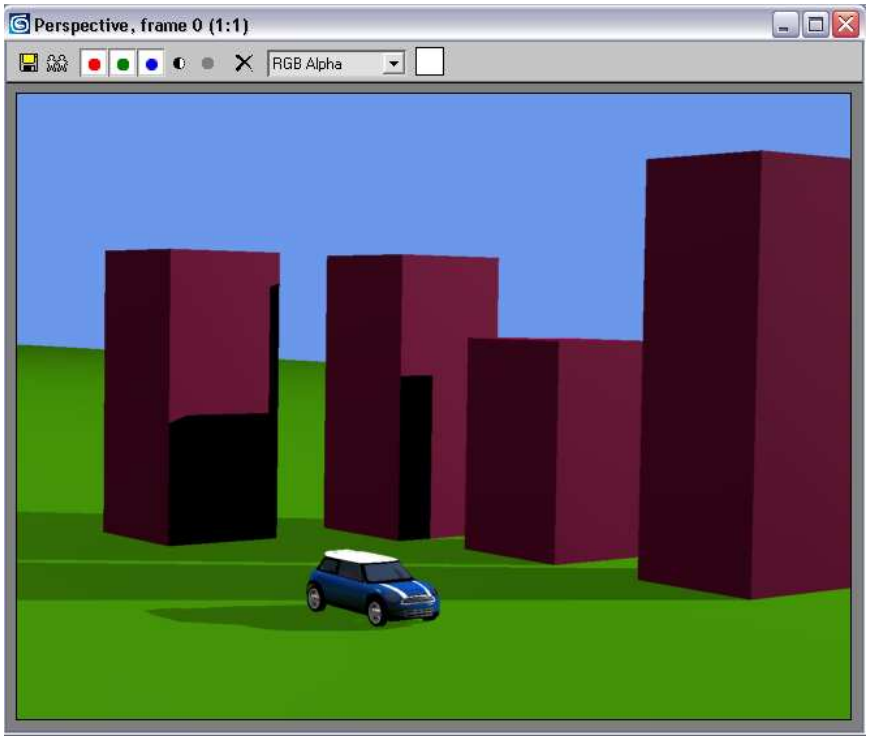


Εικόνα 31. Επιλογές φωτισμού

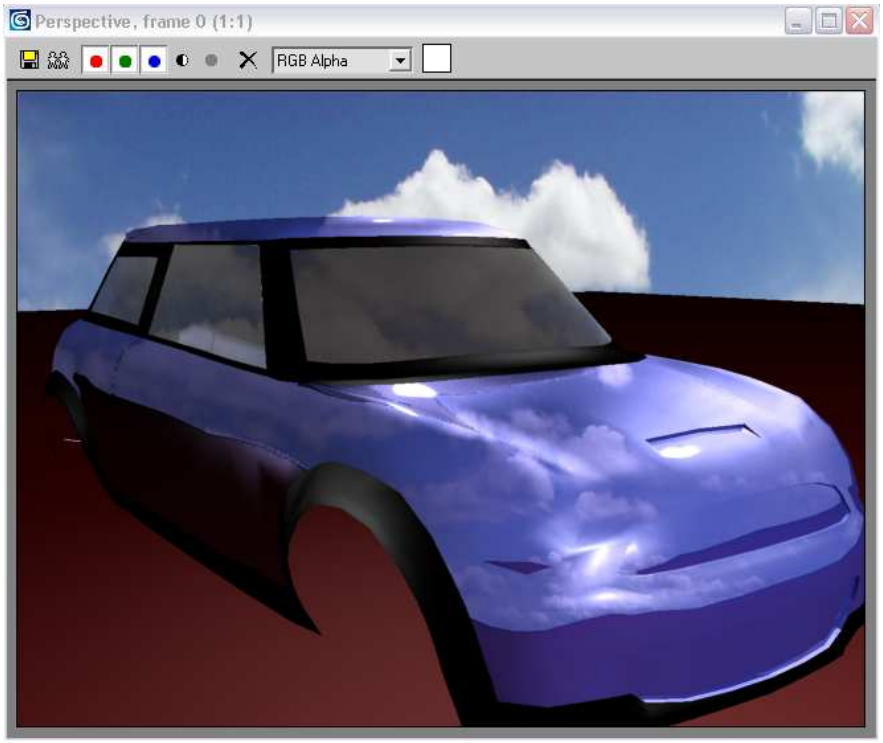


Εικόνα 32. Προσθήκη ενός Omni light

Στην παρακάτω εικόνα, παρατηρούμε ότι το σκηνικό αλλάζει, όταν προστεθεί παραπάνω από μια πηγή φωτός. Με τον συνδυασμό σωστών υλικών και φωτισμού και τις κατάλληλες ρυθμίσεις αυτών, το εικονικό μοντέλο προσεγγίζει αρκετά το πραγματικό (βλ. Εικόνα 34).

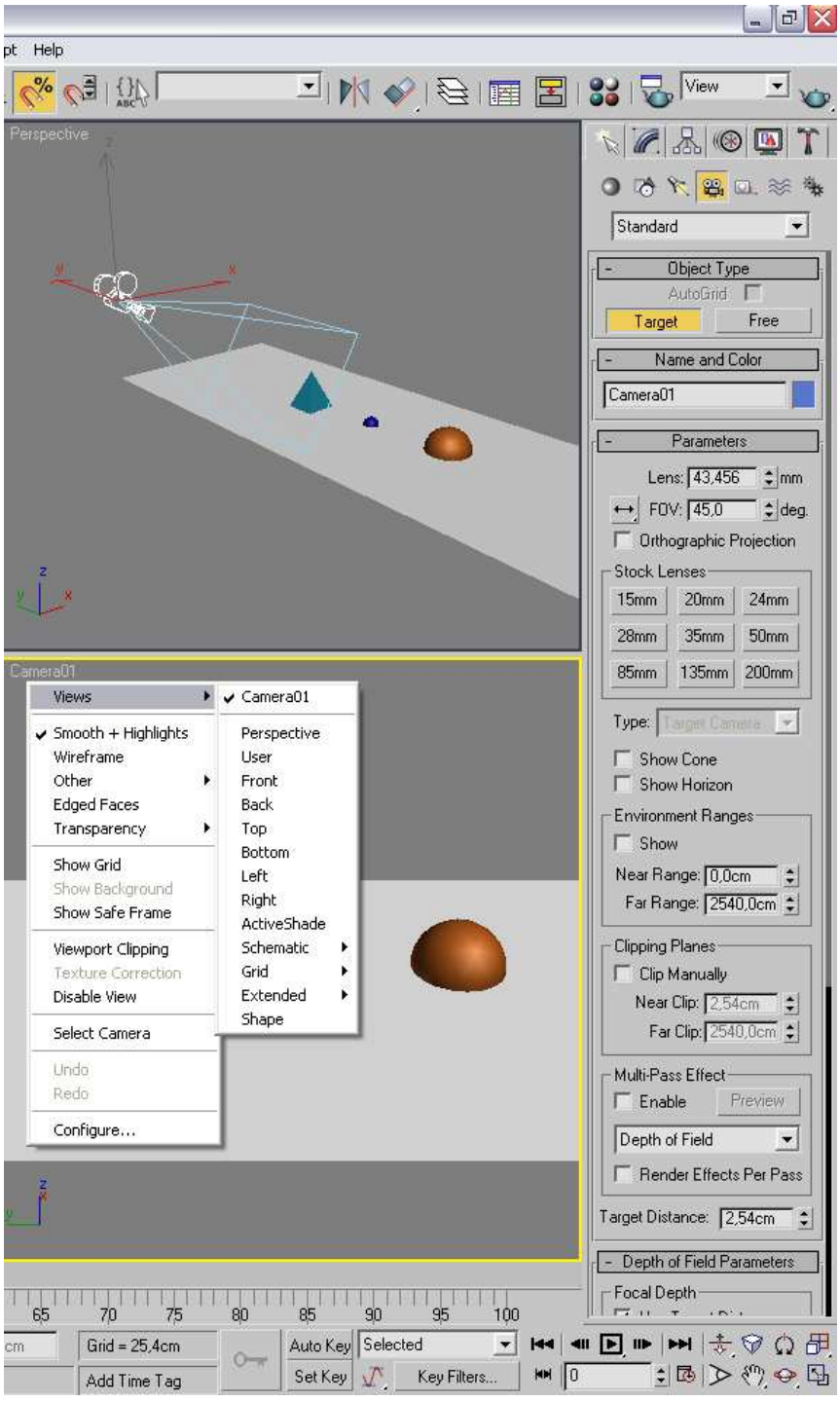


Εικόνα 33. Προσθήκη περισσότερων Omni lights στη σκηνή



Εικόνα 34. Τα σύννεφα αντανακλούν και στο σασί, αποδίδοντας καλύτερα την αίσθηση του γυαλιστερού μέταλλου

Τέλος, αφού ολοκληρώσουμε το σχεδιασμό του τρισδιάστατου κόσμου και των μοντέλων, προσθέτουμε κάμερες στη σκηνή (βλ. Εικόνα 35) και εξάγουμε το αρχείο από την οπτική μιας κάμερας. Από το πάνελ create επιλέγουμε cameras  free ή target και τοποθετούμε κάπου στη σκηνή την κάμερα. Σ' ένα viewport, πατώντας δεξί κλικ πάνω στη λέξη που βρίσκεται στην πάνω αριστερή γωνία (π.χ. Perspective) επιλέγοντας views και μετά camera01, κοιτάζουμε τον κόσμο από την οπτική της κάμερας. Το αποθηκεύουμε ως shockwave αρχείο με κατάληξη .w3d (shockwave 3d scene export), αφήνοντας τις άλλες ρυθμίσεις ως έχουν.



Εικόνα 35. Εισαγωγή κάμερας

4.3. Adobe director

Το Adobe Director (πρώην Macromedia Director) είναι ένα ισχυρό πρόγραμμα που χρησιμοποιείται στην ανάπτυξη πολυμεσικών εφαρμογών, εκπαιδευτικής ή ψυχαγωγικής φύσεως, animated films και παιχνίδια. Το όνομα director που σημαίνει σκηνοθέτης, παραπέμπει σε μια μεταφορά ταινίας, χρήζοντας στοιχεία (όπως εικόνες), ως ηθοποιούς και τον δημιουργό, ως σκηνοθέτη. Επιτρέπει στον προγραμματιστή, τη συναρμολόγηση ποικίλων διαφορετικών τύπων αρχείων, όπως γραφικά, ήχο, κείμενο, βίντεο, hypertext, Flash movies και 3d μοντέλων, για τη δημιουργία οποιασδήποτε εφαρμογής. Μέσο της scripting γλώσσας που διαθέτει, ο προγραμματιστής ελέγχει, όλα αυτά τα είδη και το πώς αυτά αντιδρούν στη «σκηνή». Για παράδειγμα το πότε θα εμφανιστεί ή θα κινηθεί μια εικόνα, στη σκηνή. Αυτό που έκανε το director να ξεχωρίσει από άλλες παρόμοιες εφαρμογές είναι οι απεριόριστες επιλογές που παρέχει στον χρήστη, για τη δημιουργία κι επεξεργασία τόσο δισδιάστατων όσο και τρισδιάστατων πολυμεσικών εφαρμογών.

Η προσθήκη διαφόρων extras, και συγκεκριμένα του «havok», έδωσε άλλη δυναμική και περισσότερες λειτουργίες. Το «havok» είναι μια μηχανή φυσικής, η οποία προσομοιώνει τις δυνάμεις φυσικής και τις ιδιότητες του πραγματικού κόσμου. Μια από τις ιδιότητές του είναι η ανίχνευση, σε πραγματικό χρόνο σύγκρουσης σωμάτων.

Το πρόγραμμα ξεκίνησε ως εφαρμογή της Apple Macintosh, το 1985 με το όνομα «videoworks». Το 1987 η ονομασία άλλαξε σε director και προστέθηκε η γλώσσα lingo. Το 2004 η macromedia παρουσίασε την έκδοση MX 2004. Έπειτα από 4 χρόνια η adobe αφού αγόρασε τα δικαιώματα , λάνσαρε την ανανεωμένη πλατφόρμα με όνομα director 11 και με τη προσθήκη αρκετών νέων χαρακτηριστικών (βλ. Πίνακα 6) εκδόθηκε από την ίδια εταιρία, λίγο αργότερα, το director 11.5.

Η εργασία εστιάζει και στη νέα μηχανή φυσικής Ageia Physics της Nvidia. Παρακάτω θα δούμε πως τα μοντέλα του εικονικού κόσμου αντιδρούν, ακολουθώντας τις αρχές φυσικής του πραγματικού κόσμου. Η παροχή φυσικής στα παιχνίδια δεν είναι εύκολη υπόθεση. Είναι ένα εξαιρετικής έντασης περιβάλλον υπολογισμού, το οποίο βασίζεται σ' ένα μοναδικό σύνολο αλγορίθμων φυσικής που απαιτούν τεράστιους μαθηματικούς και λογικούς υπολογισμούς, ταυτόχρονα. Στην πλειοψηφία, τα σύγχρονα βιντεοπαιχνίδια διαθέτουν μια μηχανή φυσικής που δίνει τη δυνατότητα καλύτερης αναπαράστασης του φυσικού κόσμου. Η ενσωμάτωση φυσικής σε 3d λογισμικά, επιτρέπει τη δημιουργία μιας δυναμικής σκηνής μέσω μιας στατικής 3d σκηνή. Ας υποθέσουμε ότι σ' ένα εικονικό κόσμο υπάρχουν δύο μοντέλα, μια μπάλα και το δάπεδο. Για να δημιουργήσουμε το εφέ της αναπήδησης, θα πρέπει να κινήσουμε τη μπάλα προς το δάπεδο και να ανιχνεύουμε κάθε frame, για να δούμε σε ποια χρονική στιγμή θα συγκρουστεί με το πάτωμα. Αντί αυτού, η μηχανή φυσικής μπορεί να χρησιμοποιηθεί ως μεσάζοντας ο οποίος θα ελέγχει τις συγκρούσεις, τις κινήσεις και όλες τις αλληλεπιδράσεις μεταξύ μοντέλων.

Σύγκριση χαρακτηριστικών διαφορετικών εκδόσεων του Director			
Χαρακτηριστικά προϊόντος	**Director MX 2004**	**Director 11**	**Director 11.5**
Υποστήριξη 5.1 surround ήχο	–	–	•
Audio mixing σε πραγματικό χρόνο	–	–	•

Σύγκριση χαρακτηριστικών διαφορετικών εκδόσεων του Director			
Χαρακτηριστικά προϊόντος	**Director MX 2004**	**Director 11**	**Director 11.5**
Ηχητικά εφέ και DSP filters	–	–	●
H.264 MPEG-4, FLV, και F4V video support	–	–	●
Υποστήριξη ροής ήχου και βίντεο με RTMP	–	–	●
Δυνατότητα εφαρμογής φίλτρων ήχου σε βίντεο	–	–	●
Δυνατότητα εφαρμογής φίλτρων εικόνας σε βίντεο	–	–	●
Εισαγωγή αρχείου Google SketchUp	–	–	●
Ενισχυμένη μηχανή φυσικής που υποστηρίζει dynamic concave στέρεα σώματα	–	–	●
ByteArray datatype για χειρισμό binary data	–	–	●
Πολλαπλό undo/redo για επεξεργαστές κειμένου	–	–	●
Text rendering και βελτιστοποίηση λειτουργίας	–	–	●
Υποστήριξη Cross-domain policy για Adobe Shockwave® Player	–	–	●
Συμβατότητα λογισμικού Mac OS X Leopard	–	–	●
Υποστήριξη Unicode	–	●	●

Σύγκριση χαρακτηριστικών διαφορετικών εκδόσεων του Director			
Χαρακτηριστικά προϊόντος	**Director MX 2004**	**Director 11**	**Director 11.5**
Υποστήριξη Microsoft DirectX 9	–	●	●
Ανεπτυγμένη μηχανή Φυσικής με υποστήριξη NVIDIA® PhysX™	–	●	●
JavaScript Λεξικό	–	●	●
Code snippets	–	●	●
Bitmap φίλτρα	–	●	●
Συμβατότητα με Microsoft® Windows Vista®	–	●	●
Support for Intel® based Macs	–	●	●
Cross-platform projector publishing	●	●	●
Δημοσίευση στο web με το Adobe Shockwave Player	●	●	●
Υποστήριξη πάνω από 40 τύπους βίντεο, ήχου, και εικόνας συμπεριλαμβανομένου και SWF	●	●	●

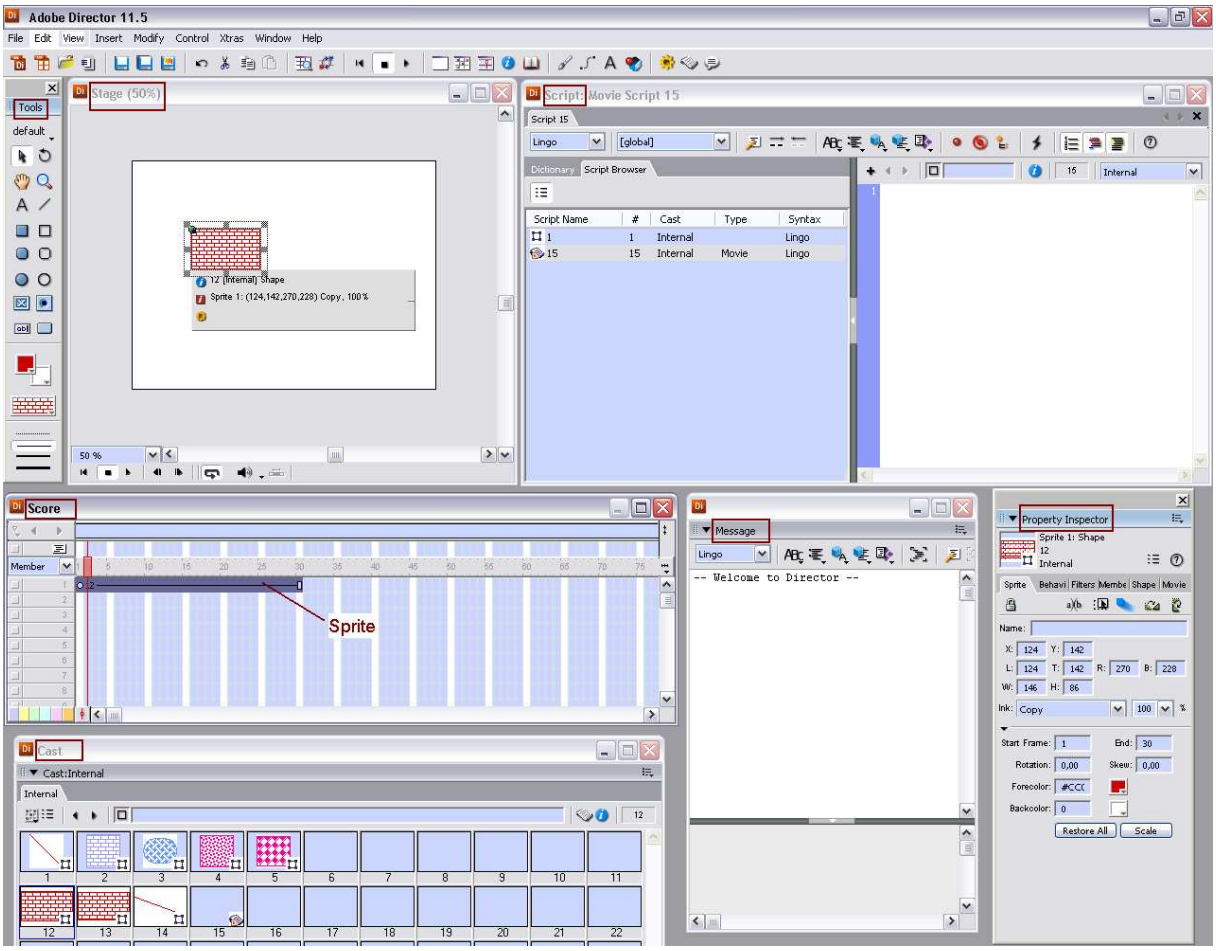
Πίνακας 6. Σύγκριση χαρακτηριστικών, μεταξύ διαφορετικών εκδόσεων του director. Τα πολλά και νέα χαρακτηριστικά, της τελευταίας έκδοσης, φανερώνουν το πέρασμα ενός αρκετά μεγάλου χρονικού διαστήματος, όπου δεν πραγματοποιήθηκε καμία αναβάθμιση στο πρόγραμμα.

Εικόνα 36. Προσομοίωση με χρήση Ageia Physics

Εικόνα 37. Προσομοίωση κίνησης υφάσματος

Παρακάτω (βλ. Εικόνα 38) διακρίνεται η νέα διεπαφή, η οποία δε διαφέρει ιδιαίτερα από την διεπαφή της παλαιότερης έκδοσης, όμως είναι αρκετά πιο εύχρηστη και προσφέρει νέες δυνατότητες που διευκολύνουν τον χρήστη. Στο παράθυρο με τίτλο «Stage» είναι η σκηνή όπου διαδραματίζονται όλα τα γεγονότα και αυτό που θα βλέπει ο τελικός χρήστης. Οτιδήποτε πέρα από το άσπρο τετράγωνο, δε θα είναι ορατό. Στο παράθυρο «script» γράφουμε τον κώδικα είτε σε lingo, είτε σε Javascript. Το «cast window» είναι η βιβλιοθήκη όλων των αρχείων που έχουν κάποιο ρόλο ή αυτών που πρόκειται ν' αποκτήσουν αργότερα

ρόλο. Αν για παράδειγμα θέλουμε να χρησιμοποιήσουμε κάποιο μουσικό κομμάτι, τότε πρέπει να το εισάγουμε πρώτα εκεί. Η εργαλειοθήκη «Tools» παρέχει διάφορα εργαλεία για δημιουργία και επεξεργασία αντικειμένων. Στο «score» μπαίνουν μόνο τα αρχεία που ήδη έχουν ρόλο και βρίσκονται στη σκηνή. Κάθε αρχείο, αντιπροσωπεύεται από ένα «sprite». Από αυτό καθορίζεται κατά πρωτίστως, ο χρόνος παραμονής του στη σκηνή. Με το sprite, ελέγχουμε τη διάρκεια δράσης του κάθε αρχείου, καθώς και το ποια χρονική στιγμή θα εισέρθει στη σκηνή και πότε θα εξέρθει από αυτήν. Η μπάρα αριθμών συμβολίζει τα frames. Το «message window» παρέχει ένα τρόπο δοκιμής και παρακολούθησης των εντολών του κώδικα. Ένα από τα πιο χρήσιμα εργαλεία του director είναι το «property inspector». Αυτό το πάνελ περιέχει πληροφορίες για κάθε επιλεγμένο sprite.



Εικόνα 38. Η ανανεωμένη διεπαφή του director 11.5

4.3.1. Προγραμματισμός σε Lingo

Τα βήματα που πρέπει ν' ακολουθήσουμε για να επεξεργαστούμε τον εικονικό κόσμο με το director είναι τα εξής:

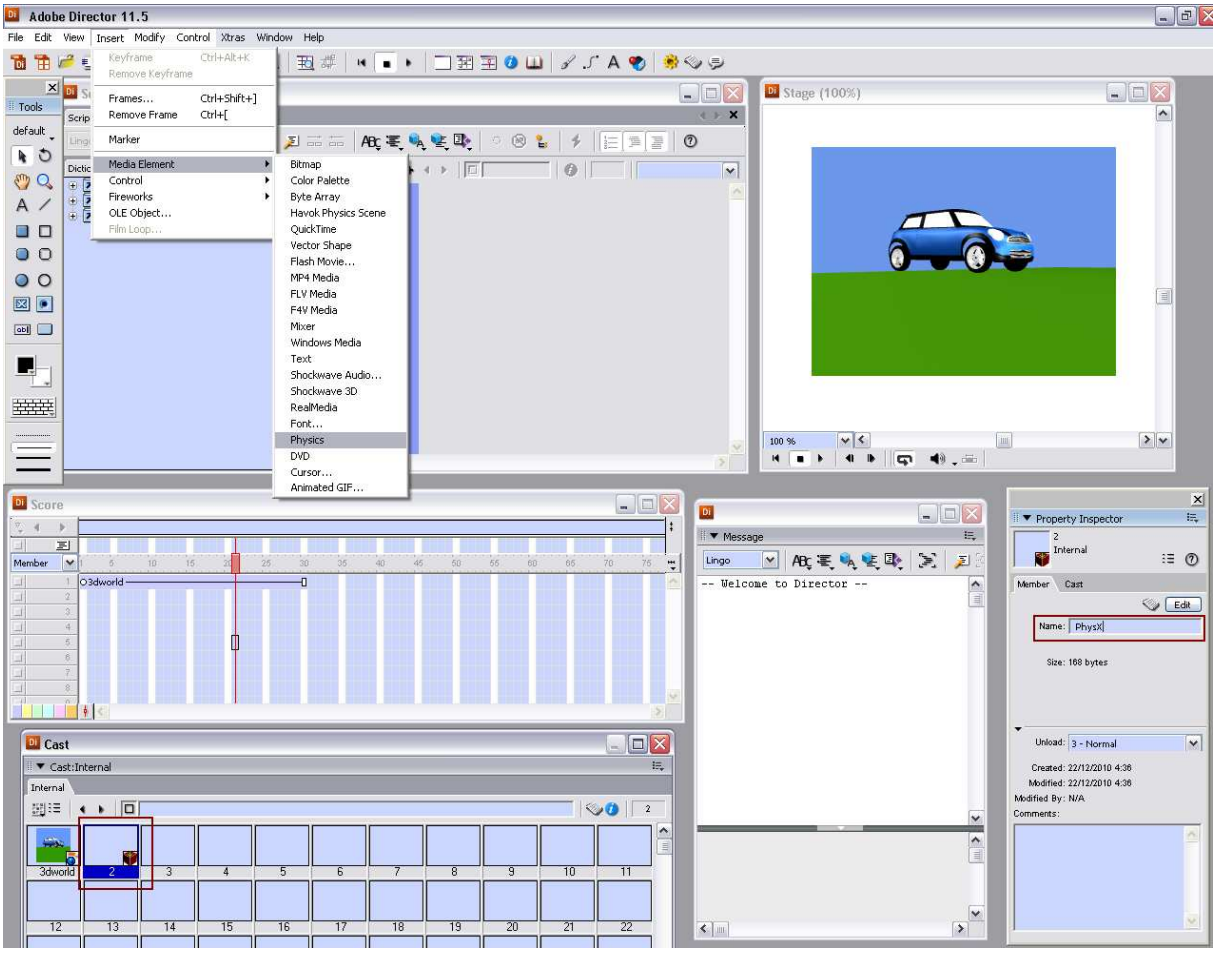
1. **Εισαγωγή ενός .w3d αρχείου στη βιβλιοθήκη αρχείων (cast member)**

Πρώτα εισάγουμε, στο director, τον εικονικό κόσμο που δημιουργήσαμε στο 3ds max. Πατάμε «File» επιλέγουμε «import» και βρίσκουμε το αρχείο με κατάληξη .w3d.

Παρατηρώντας το cast window βλέπουμε ότι το αρχείο υπάρχει πλέον εκεί. Άλλος τρόπος για να το εισάγουμε, είναι πατώντας δεξί κλικ, σε μια κενή θέση του cast window και επιλογή import. Για να το μεταφέρουμε στη σκηνή, απλά το επιλέγουμε και το «σέρνουμε» αφήνοντάς το στο μέσο περίπου της σκηνής, έτσι ώστε να εσωκλείεται ολόκληρο μέσα στο άσπρο τετράγωνο. Αφού το αφήσουμε στη σκηνή, βλέπουμε ότι δημιουργήθηκε αυτόματα ένα sprite, στο score window.

2. Εισαγωγή μηχανής φυσικής

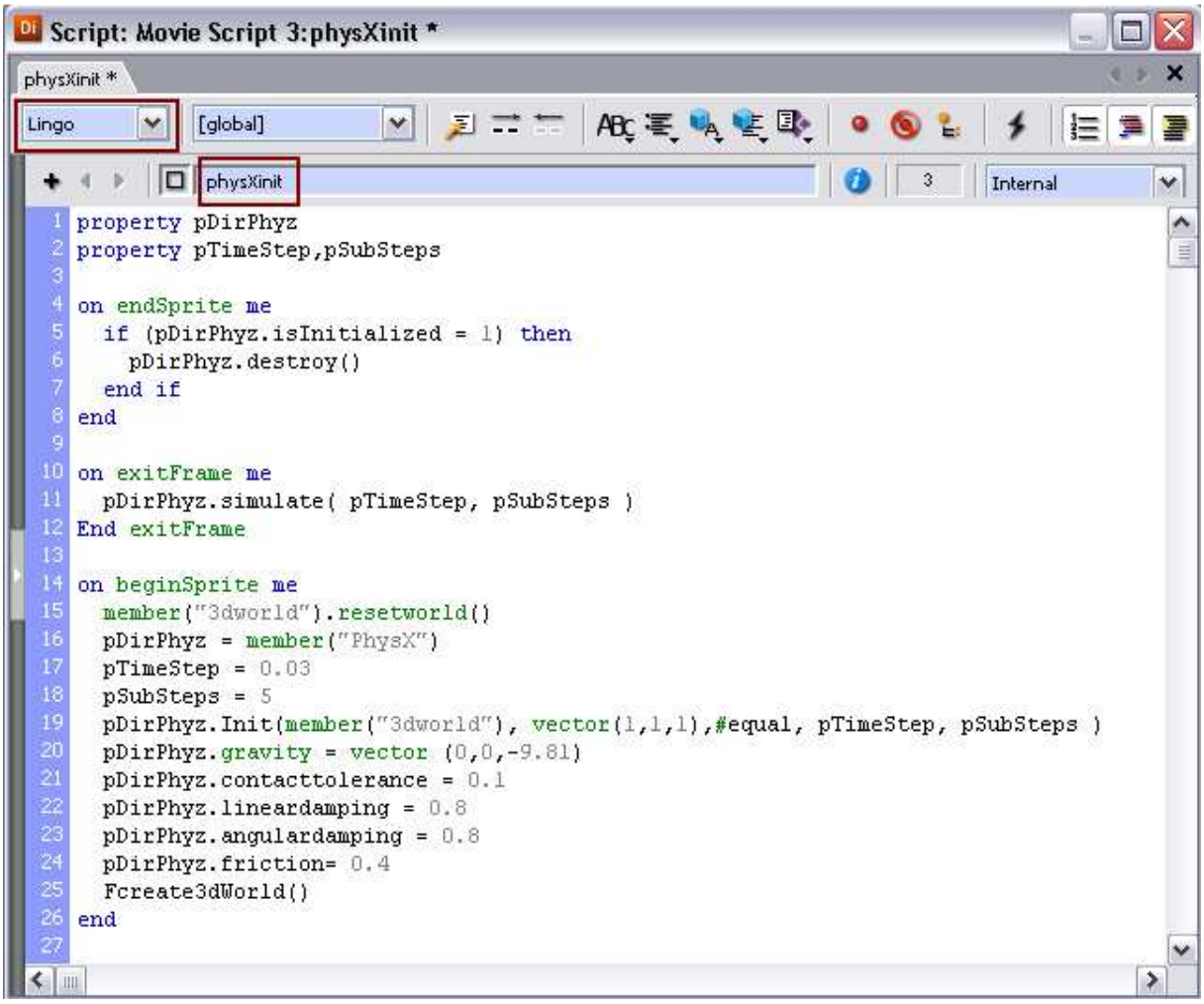
Ομοίως, από το κυρίως μενού επιλέγουμε «Insert» και από την κατηγορία «media element» πατάμε «physics». Στο παράθυρο cast, μαρκάρουμε το νέο στοιχείο, και από το πάνελ «property inspector», πληκτρολογούμε ένα όνομα, όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 39. Εισαγωγή στοιχείων

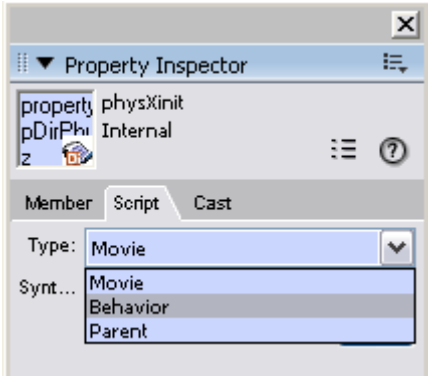
3. Αρχικοποίηση τιμών του κόσμου φυσικής

Στο παράθυρο script πληκτρολογούμε τα παρακάτω:



Εικόνα 40. Αρχικοποίηση τιμών

Όπου «3dworld», τοποθετούμε το όνομα του αρχείου με κατάληξη .w3d και όπου «physX» το όνομα που ορίσαμε για το στοιχείο Physics. Τα scripts μπορεί να είναι είτε behavior είτε movie. Τα behavior είναι κώδικας που εμείς αναθέτουμε σε ένα ή περισσότερα sprite και καθορίζουν τη συμπεριφορά τους στη σκηνή. Τα movie scripts είναι κώδικας ο οποίος «τρέχει» παρασκηνιακά, κατά τη διάρκεια εκτέλεσης του παιχνιδιού και ελέγχει όλα τα στοιχεία. Στο property inspector επιλέγουμε τύπο για αυτό το script, *behavior* και τέλος πληκτρολογούμε ένα νέο όνομα (βλ. Εικόνα 40).



Εικόνα 41. Property inspector

Με τη μέθοδο Init(), αρχικοποιούμε τον κόσμο. Δέχεται 5 ορίσματα, το όνομα του 3D κόσμου, την κλίμακα του κόσμου, το Timestep, το Timestepmode και το subStepcount. Το timestep μπορεί να είναι είτε #equal είτε #automatic. Με την επιλογή automatic, τα βήματα της προσομοίωσης του κόσμου φυσικής θα είναι με βάση τον πραγματικό χρόνο ροής, ενώ με την επιλογή equal, η εξομοίωση προχωράει με βάση τον χρόνο που ορίζουμε από την παράμετρο timestepmode. Το substepcount παρέχεται για περισσότερη ακρίβεια. Η μέθοδος simulate() καλείται στο τέλος κάθε frame. Χρησιμοποιείται για να εξομοιώνει τον κόσμο φυσικής, μέσο της timestep, η οποία ορίζεται πάντα κατά τη διάρκεια που καλείται η init(). Η μέθοδος destroy() πρέπει να καλείται πάντα, πριν από την επόμενη init() του ίδιου κόσμου φυσικής, ώστε να ελευθερώνει τους πόρους και να σταματά την προσομοίωση. Η resetworld() επαναφέρει τις αρχικές ρυθμίσεις του κόσμου. Δεν πρέπει ποτέ να καλείται ανάμεσα σε μια init() και μια destroy().

Στον πραγματικό κόσμο δυνάμεις ασκούνται τόσο στην κατάσταση ηρεμίας των σωμάτων, όσο και στην κατάσταση κίνησης. Μια από αυτές τις δυνάμεις είναι η βαρύτητα. Στον παραπάνω κώδικα, η βαρύτητα (gravity) είναι ορισμένη στον άξονα -Z. Οι ιδιότητες του 3d κόσμου φυσικής, που μπορούμε να ρυθμίσουμε και θα ισχύουν για όλα τα σώματα είναι οι εξής:

- angularDamping
- contactTolerance
- friction
- gravity
- IsInitialized
- linearDamping
- restitution
- scalingFactor
- sleepMode
- sleepThreshold

Αυτές οι ιδιότητες καθορίζουν το χρόνο διάρκειας της προσομοίωσης :

- subSteps
- timeStep
- timeStepMode

4. Δημιουργία και επεξεργασία rigid bodies

Στη συνέχεια δημιουργούμε rigid bodies από τα 3d μοντέλα. Ουσιαστικά ορίζουμε σε ποια σώματα του κόσμου θα ασκούνται δυνάμεις, όμως η μορφή, η μάζα και ο όγκος τους παραμένουν σταθερά. Εξίσου και η ονομασία rigid (= στερεό, άκαμπτο). Τα rigid bodies, είναι είτε στατικά είτε δυναμικά.

Στο script window πατάμε «+» για να δημιουργήσουμε νέο script και πληκτρολογούμε τα παρακάτω, αφού επιλέξουμε movie script από τον property inspector :

Script: Movie Script 4:Rigidbodies *!

physXinit * | Rigidbodies *!

Lingo | Fcreate3dWorld | Rigidbodies

```
1  global rb_chassis, rb_wheelfr, rb_wheelbr, rb_earth
2
3  on Fcreate3dWorld
4
5    physX = member("PhysX")
6    scene3d = member("3dworld")
7
8    member("3dworld").model("chassis").addChild(member("3dworld").camera[1])
9
10   ---------------------- RIGID BIODIES ------------------------------------
11   -- Έδαφος
12   earth = scene3d.model("terrain")
13   rb_earth = FcreateRigidBody_S(earth)
14   rb_earth.mass = 400
15   rb_earth.restitution = 0.7
16   rb_earth.contacttolerance = 0.1
17   -- Σασί αυτοκινήτου
18   fchassis = scene3d.model("chassis")
19   rb_chassis = FcreateRigidBody_D(fchassis)
20   rb_chassis.mass = 100
21   rb_chassis.restitution = 0.7
22   rb_chassis.contacttolerance = 0.1
23   -- Group μπροστινών τροχών
24   wfr = scene3d.model("FR")
25   rb_wheelfr=FcreateRigidBody_D(wfr)
26   rb_wheelfr.mass = 10
27   rb_wheelfr.restitution = 0.7
28   rb_wheelfr.contacttolerance = 0.1
29   -- Group πίσω τροχών
30   wbr = scene3d.model("BR")
31   rb_wheelbr=FcreateRigidBody_D(wbr)
32   rb_wheelbr.mass = 10
33   rb_wheelbr.restitution = 0.7
34   rb_wheelbr.contacttolerance = 0.1
35   -- 1η πλατφόρμα
36   platl = scene3d.model("Platform01")
37   rb_platl = FcreateRigidBody_S(platl)
38   rb_platl.mass = 80
39   rb_platl.restitution = 0.7
40   rb_platl.contacttolerance = 0.1
41   -- 2η πλατφόρμα
42   plat = scene3d.model("Platform")
43   rb_plat = FcreateRigidBody_S(plat)
44   rb_plat.mass = 80
45   rb_plat.restitution = 0.7
46   rb_plat.contacttolerance = 0.1
47   ------------------------------------------------------------------------
48   FcreateJoints()
49 End
```

Εικόνα 42. Δημιουργία κόσμου

Όταν καλείται η συνάρτηση Fcreate3dworld(), το στοιχείο PhysX θα ανατίθεται στη μεταβλητή physX, όπως και ο εικονικός κόσμος, στη μεταβλητή 3dScene.

Η camera χρησιμοποιείται για να καθορίσει το σημείο (vector) από το οποίο ο 3D κόσμος παρακολουθείται. Η εντολή addchild() προσθέτει ένα κόμβο, στη λίστα «παιδιών» ενός άλλου κόμβου, και τον αφαιρεί από τη λίστα «παιδιών» του προηγούμενου «γονέα». Αυτή η γραμμή κώδικα, προσθέτει το στοιχείο camera 1 στο μοντέλο chassis του κόσμου, με ονομασία «3dworld». Στην ουσία, με αυτό τον τρόπο όταν κινείται το μοντέλο «chassis» η camera1 θα το ακολουθεί. Μπορεί στον κόσμο να υπάρχουν πολλές κάμερες, αλλά αν δεν ορίσουμε, με κώδικά, από ποια οπτική θέλουμε να κοιτάμε, θα βλέπουμε πάντα από την οπτική της κάμερας που κάναμε εξ' αρχής export από το 3ds max.

Στη συνέχεια, τα βήματα είναι παρόμοια για όλα τα μοντέλα. Καλείται η συνάρτηση FcreateRigidBody_S (για στατικά rigid bodies) και η συνάρτηση FcreateRigidBody_D (για δυναμικά), δέχεται ως όρισμα μια συμβολοσειρά που θα αντιπροσωπεύει το όνομα του rigid body. Έπειτα, ορίζουμε μάζα (mass), αποκατάσταση θέσης (restitution) και ανεκτικότητα επαφής με άλλα σώματα, για κάθε rigid body ξεχωριστά.

Σ' ένα νέο movie script πληκτρολογούμε τις δύο αυτές συναρτήσεις:

o

Script: Movie Script 7:CreateRigidB...

CreateRigidB... | Fcreate3dWor... | InitPhysics

Lingo | [global] | CreateRigidBody

```
1
2  -- Dynamic RigidBody
3  on FcreateRigidBody_D modelname
4    physX = member("physX")
5    modelname.addmodifier(#meshdeform)
6    rbname = physX.createRigidBody("rb_" & modelname.name, modelname.name,#convexshape,#dynamic)
7    return rbname
8  end
9
10 -- Static rigidBody
11 on FcreateRigidBody_S modelname
12   physX = member("physX")
13   modelname.addmodifier(#meshdeform)
14   rbname = physX.createRigidBody("rb_" & modelname.name, modelname.name,#concaveshape,#static)
15   return rbname
16 end
```

Εικόνα 43. Δημιουργία rigid bodies

Για τη δημιουργία rigid body χρειάζεται η μέθοδος createRigidBody() και η προσθήκη του τροποποιητή, meshdeform (σφάλμα "-1" αν τον παραλείψουμε). Το πρώτο όρισμα της συνάρτησης προσδιορίζει το 3d μοντέλο βάση του οποίου θα δημιουργηθεί το rigid body. Στο δεύτερο όρισμα, προσδιορίζουμε το όνομα του νεοσύστατου rigid body. Στο τρίτο όρισμα μπορούμε να επιλέξουμε ανάμεσα σε #box, #sphere, #convexshape και #concaveshape. Αυτό το όρισμα, πρέπει να είναι ανάλογο του σχήματος του μοντέλου, διότι αυτό το «κάλυμμα» θα περικλείει το rigid body και μέσο αυτού θ' ανιχνεύονται οι συγκρούσεις (collisions). Στο τέταρτο όρισμα επιλέγουμε #static αν θέλουμε το rigid body να είναι στάσιμο ή #dynamic αν θέλουμε να έχει τη δυνατότητα κίνησης. Το όρισμα

#flipnormals είναι προαιρετικό. Αν δηλωθεί, τότε η εσωτερική επιφάνεια θα πάρει τη θέση της εξωτερικής επιφάνειας, ενώ ταυτόχρονα η εξωτερική θα πάρει τη θέση της εσωτερικής.

Restitution, mass, friction είναι μερικές από τις ιδιότητες των rigid bodies, τις οποίες μπορούμε να καθορίσουμε. Για παράδειγμα το contacttolerance ορίζει το πόσο θα επιτρέπεται η επιφάνεια, ενός σώματος να εισχωρεί στην επιφάνεια ενός άλλου σώματος, πριν τη μετατόπιση θέσης τους, κατά τη σύγκρουση. Το restitution, ορίζει κατά πόσο και αν θα αναπηδά ένα σώμα αμέσως μετά τη σύγκρουσή του μ' ένα άλλο. Οι παράμετροι των rigid bodies, που μπορούμε να ρυθμίσουμε είναι οι εξής:

- angularDamping
- angularMomemtum
- angularVelocity
- centerOfMass
- contactTolerance
- friction
- isPinned
- isSleeping
- linearDamping
- linearMomemtum
- linearVelocity
- mass
- model
- name
- orientation
- position
- restitution
- properties
- shape
- sleepMode
- sleepThreshold
- type
- userData

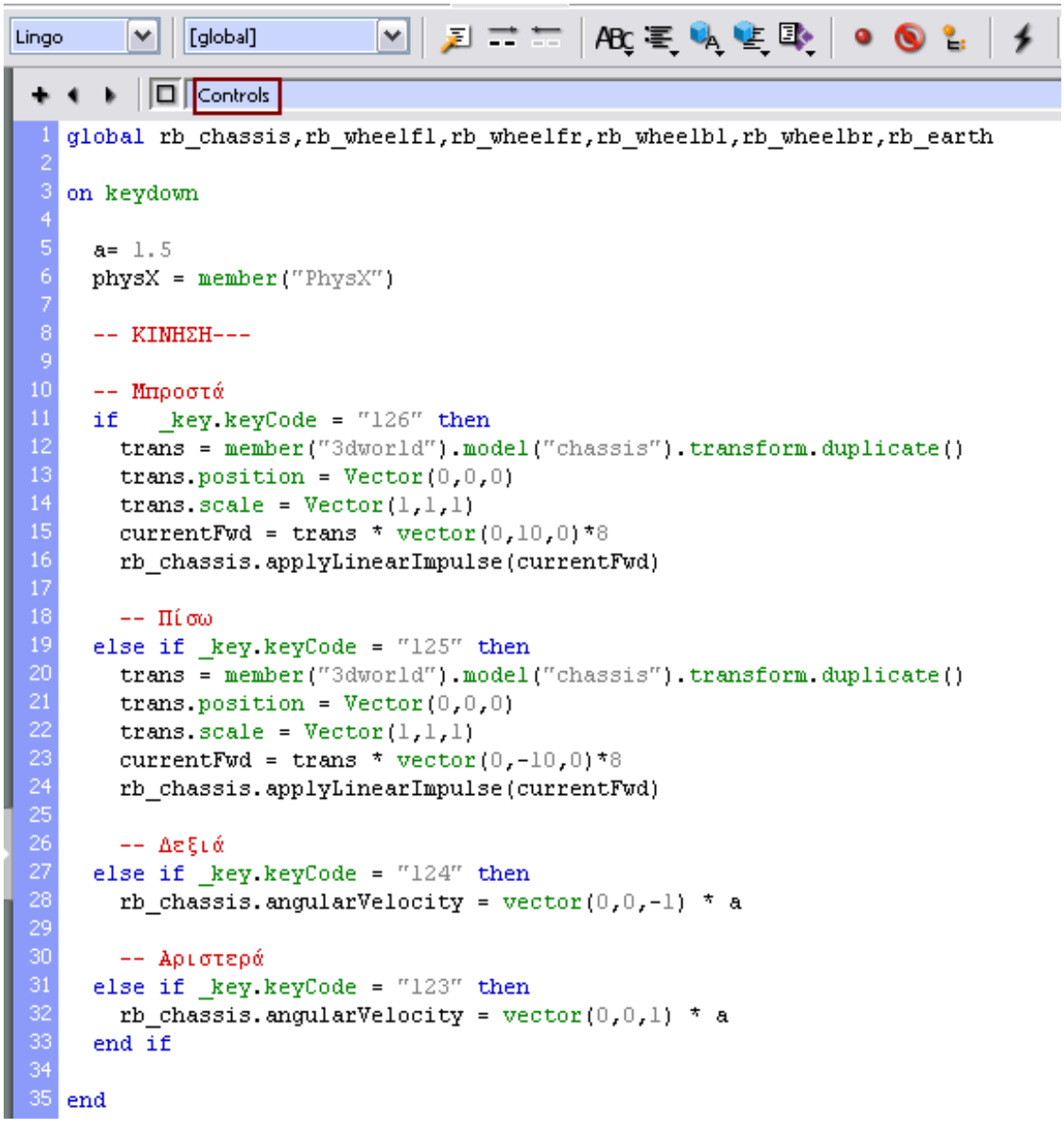
Μόλις δημιουργηθούν όλα τα rigid bodies, χρειάζεται να προσθέσουμε συναρτήσεις οι οποίες θα κινούν αυτά τα σώματα, στη σκηνή. Στον εικονικό κόσμο, τα 3d μοντέλα κινούνται με συναρτήσεις *transform*, δεν ισχύει όμως το ίδιο και για τα rigid bodies. Στον κόσμο φυσικής πρέπει να προσθέσουμε διαφορετικές συναρτήσεις ώστε να γίνεται σωστά η προσομοίωση κίνησης των σωμάτων. Οι κυριότερες από αυτές είναι:

- applyForce()
- applyLinearImpulse()
- applyAngularImpulse()
- applyTorque()
- attemptMoveTo()

Εν συντομία, με την applyforce() μέθοδο ασκείται, μια δύναμη σ' ένα προκαθορισμένο σημείο, στο σώμα ενός αντικειμένου με αποτέλεσμα το αντικείμενο να μετατοπιστεί. Εάν δεν καθοριστεί ένα σημείο, τότε η δύναμη ασκείται στο κέντρο μάζας του σώματος. Η applylinearImpulse() ωθεί ένα αντικείμενο γραμμικά, ασκώντας πίεση στο κέντρο μάζας του, ενώ η applyangularimpulse() ωθεί ένα αντικείμενο υπό γωνία. Η applytorque() προκαλεί ροπή, και το αντικείμενο κινείται περιστροφικά, με χρήση του κανόνα του δεξιού χεριού. Τέλος, με την κλήση της attemptMoveTo() γίνεται προσπάθεια

μετακίνησης του αντικειμένου σε μια συγκεκριμένη τοποθεσία, με την προϋπόθεση ότι δε υπάρχει άλλο rigid body σε αυτή. Αν ανιχνευτεί σύγκρουση στη θέση εκείνη, τότε το αντικείμενο δε θα κουνηθεί καθόλου.

Σ' ένα νέο behavior script, αφού το ονομάσουμε *controls*, πληκτρολογούμε:



Εικόνα 44. Κίνηση

Με την _key.keycode δίνουμε τον κωδικό που αντιστοιχεί σε κάθε γράμμα ή σύμβολο του πληκτρολογίου. Έτσι, ο κωδικός «126» αντιστοιχεί στο πλήκτρο με την ένδειξη βέλους που δείχνει προς τα πάνω, οι κωδικοί «125», «124» και «123» αντιστοιχούν στα υπόλοιπα βέλη. Το αυτοκίνητο θα κινείται μπρος, πίσω, δεξιά και αριστερά ανάλογα την επιλογή του χρήστη. Όταν θέλουμε το όχημα να προχωρήσει μπροστά ή πίσω, αναθέτουμε στη μεταβλητή *trans*, ένα αντίγραφο της τρέχουσας θέσης του οχήματος. Για το

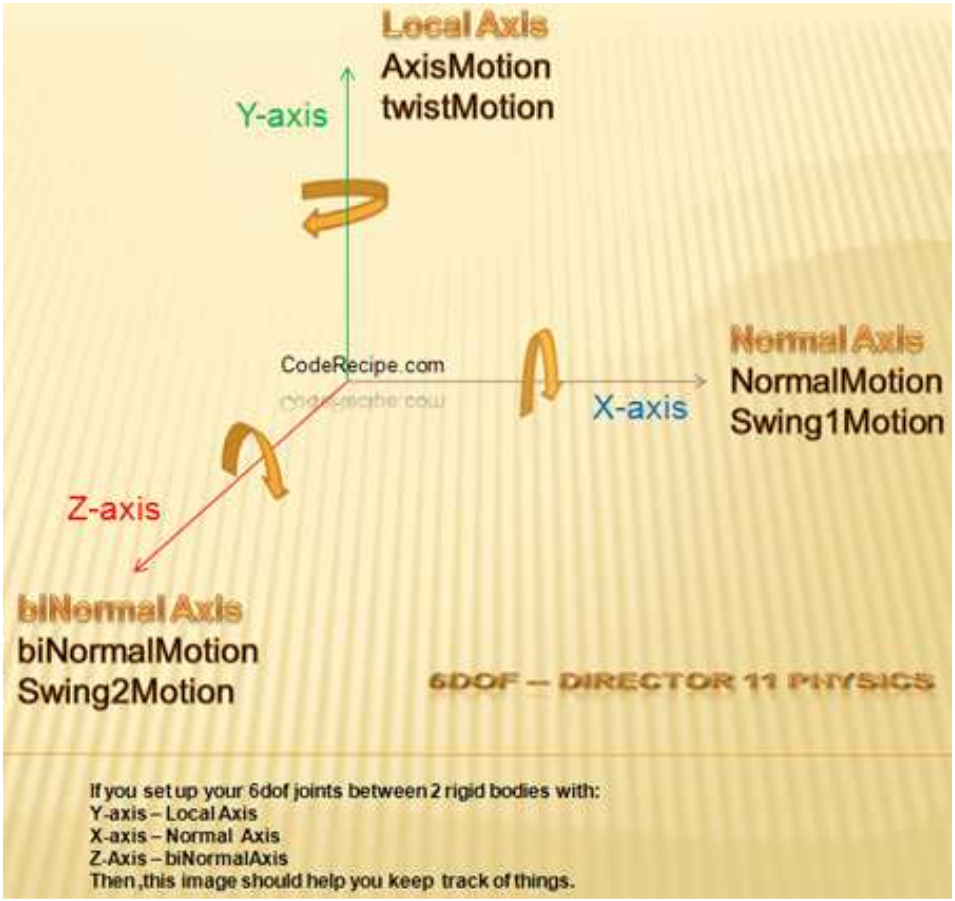
μετασχηματισμό είναι απαραίτητο να ορίσουμε position και scale, τα οποία αφήνουμε ως έχουν. Η currentFwd είναι το αποτέλεσμα ενός πολλαπλασιασμού διανύσματος που θα προσδιορίζει την ισχύ της δύναμης και την κατεύθυνση, στην οποία θα κινηθεί το όχημα όταν ασκηθεί πάνω του μια δύναμη. Η applyLinearImpulse() θα ωθήσει το σώμα γραμμικά, ενώ η angularvelocity() θα κάνει το όχημα να στρίψει με κέντρο τον άξονα κέντρου μάζας του. Ο συντελεστής a δίνει περισσότερο ισχύ στη δύναμη.

Κοιτώντας το stage και πατώντας play, παρατηρούμε πως το σασί κινείται αλλά οι ρόδες παραμένουν στην αρχική τους θέση. Μια λύση σε αυτό το πρόβλημα είναι να χρησιμοποιήσουμε αόρατες αρθρώσεις, όπου θα ακολουθούν το όχημα και παράλληλα θα κινούνται φυσικά, κυλώντας στην ίδια κατεύθυνση με το σασί.

Ολοκληρώνοντας τα παραπάνω βήματα, στον εικονικό κόσμο πλέον, ισχύουν οι νόμοι τις φυσικής και ασκούνται σε όλα τα rigid bodies. Το παιχνίδι όμως, δέχεται ακόμα πολλές βελτιώσεις για να φτάσουμε σ' ένα ικανοποιητικό αποτέλεσμα.

4.3.2. 6 Degree of Freedom Joints

Ένα νέο χαρακτηριστικό του director είναι η μηχανή Ageia Physics η οποία υποστηρίζει λειτουργία αρθρώσεων μεταξύ δύο σωμάτων, ή μεταξύ σημείου του κόσμου μ' ένα σώμα, με περισσότερη ελευθερία κινήσεων, όπως φαίνεται στο παρακάτω σχήμα:



Εικόνα 45. Ελευθερία κινήσεων στους 3 άξονες - 6DOF

Σ' ένα νέο movie script προσθέτουμε τα παρακάτω :

Script: Movie Script 18:6DOF-JOINTS *

6DOF-JOINTS *

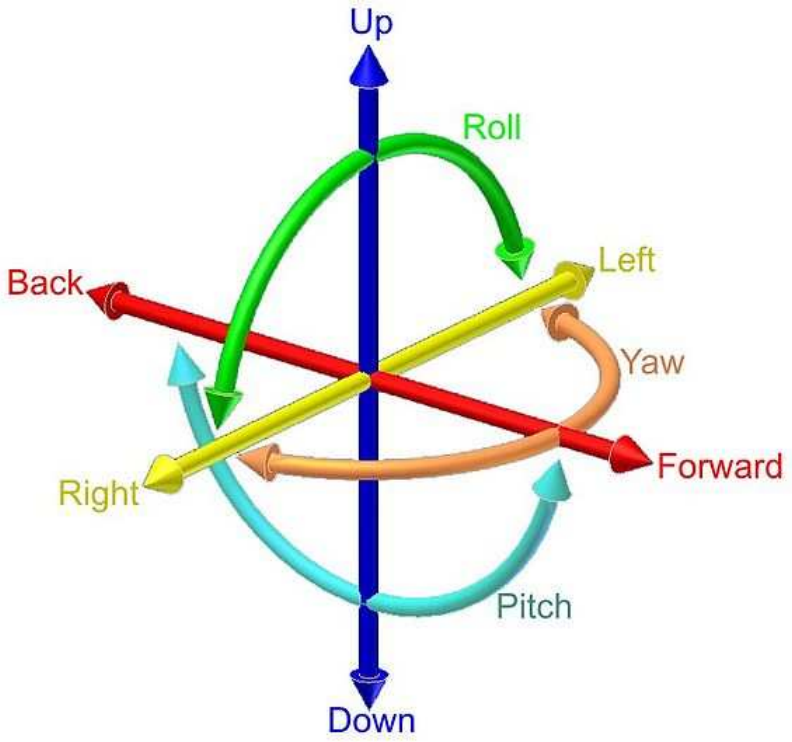
Lingo | [global]

6DOF-JOINTS

```
1  global rb_chassis,rb_wheelfr,rb_wheelbr
2  on FcreateJoints
3
4    physX = member("PhysX")
5    rb_car = rb_chassis
6    rb_tyre_front = rb_wheelfr
7    rb_tyre_rear = rb_wheelbr
8
9    -----------------6DOF ένωση μπροστινών τροχών με σασί-------------
10   D6jointl = physX.createD6Joint("myJointl",rb_car,rb_tyre_front,vector(1,1,1))
11   D6jointl.localAnchorA = vector(0,5,-2)
12   D6jointl.localAnchorB = vector(0,0,0)
13
14   D6jointl.localAxisA = vector(0,0,1)
15   D6jointl.localNormalA = vector(1,0,0)
16
17   D6jointl.localAxisB = vector(0,1,0)
18   D6jointl.localNormalB = vector(1,0,0)
19
20
21   D6jointl.axisMotion = #limited
22   D6jointl.normalMotion = #locked
23   D6jointl.BiNormalMotion = #locked
24
25
26   D6jointl.twistMotion = #limited
27   D6jointl.swinglMotion = #free
28   D6jointl.swing2Motion = #locked
29
30   ----------------------6DOF ένωση πίσω τροχώνν με σασί--------------------
31   D6joint2 = physX.createD6Joint("myJoint2",rb_car,rb_tyre_rear,vector(1,1,1))
32   D6joint2.localAnchorA = vector(0,-5.9,-2)
33   D6joint2.localAnchorB = vector(0,0,0)
34
35   D6joint2.localAxisA = vector(0,0,1)
36   D6joint2.localNormalA = vector(1,0,0)
37
38   D6joint2.localAxisB = vector(0,1,0)
39   D6joint2.localNormalB = vector(1,0,0)
40
41   D6joint2.axisMotion = #limited
42   D6joint2.normalMotion = #locked
43   D6joint2.BiNormalMotion = #locked
44
45
46   D6joint2.twistMotion = #limited
47   D6joint2.swinglMotion = #free
48   D6joint2.swing2Motion = #locked
49
50   --- Εφέ ανάρτησης ---
51   D6jointl.linearLimit =  [-5, 100, 0.01, 0.4]
52   D6joint2.linearLimit =  [-5, 100, 0.01, 0.4]
53 end
54
```

Εικόνα 46. Αρθρώσεις τροχών με σασί

Στο σχήμα 47 διακρίνονται οι κινήσεις που μπορεί να κάνει ένα τρισδιάστατο αντικείμενο. Ουσιαστικά γίνεται μετατόπιση και περιστροφή στους άξονες X,Y, Z καθώς και αντίθετα, δηλαδή -X, -Y, και -Z.



Εικόνα 47. Κινήσεις τρισδιάστατων σωμάτων

Με την createD6Joint() δημιουργείται μια άρθρωση, ελευθερίας 6 βαθμών, ανάμεσα σε δύο σώματα ή ανάμεσα σ' ένα σημείο του κόσμου και ενός σώματος. Δέχεται τέσσερα ορίσματα, ένα όνομα για την άρθρωση, το σώμα A, το σώμα B, και ένα διάνυσμα όπου θα υποδεικνύει το σημείο όπου θα κρατάει την άρθρωση αγκυροβολημένη. Στην προκειμένη περίπτωση το σώμα A, είναι το σασί ενώ το σώμα B είναι το group των δύο μπροστινών τροχών. LocalAnchorA συμβολίζει ένα σημείο στο χώρο του σώματος A, απ' όπου θα ξεκινά η άρθρωση. Τα LocalAxisA και localNormalA καθορίζουν τον άξονα κατεύθυνσης του σώματος A. Για το σώμα B (group τροχών) ισχύουν τα ίδια. Τα επόμενα ορίζουν την ελευθερία κινήσεων των δύο σωμάτων. AxisMotion, normalMotion, biNornalMotion, twistMotion, swing1Motion και swing2Motion ορίζονται ως limited – περιορισμένη κίνηση, locked – απαγορευμένη κίνηση και free – ελεύθερη. Για να βρούμε τους σωστούς άξονας πρέπει να ελέγξουμε τους τοπικούς άξονες κάθε σώματος, από το 3ds max, και να πειραματιστούμε επιλέγοντας ανάμεσα στις παραπάνω τιμές, για να έχουμε σωστά αποτελέσματα. Το linearlimit καθορίζει τη συμπεριφορά της άρθρωσης όταν η γραμμική κίνηση είναι περιορισμένη. Το rigid body θα αιωρηθεί όταν φτάσει την τιμή του ορίου. Τα ορίσματα που δέχεται είναι τέσσερα, τιμή ορίου, τιμή δυσκαμψίας, τιμή απόσβεσης ταλάντωσης και τιμή επιστροφής. Με αυτόν τον τρόπο, δημιουργείται η ψευδαίσθηση ότι το αυτοκίνητο διαθέτει αναρτήσεις.

Στο «score window» κάνουμε διπλό κλικ στο τελευταίο frame της ταινίας και προθέτουμε ένα behavior script ακόμα:

```
on exitFrame me
  go the frame
end
```

Με αυτό τον τρόπο ο playhead θα σταματήσει στο τελευταίο frame. Τέλος αφού ολοκληρώσουμε τη συγγραφή των script, ελέγχουμε τον κώδικα πατώντας  , μετά αναθέτουμε τα scripts “InitPhysics” και “controls” στο shockwave αρχείο (3dworld) που βρίσκεται στο stage. Είτε επιλέγουμε τα scripts, από το cast window και με drag-and-drop τα ελευθερώνουμε πάνω σε αυτό, είτε από τον property inspector πατώντας «+» προσθέτουμε τα behavior. Πατώντας ξανά *play*, παρατηρούμε ότι το όχημα κινείται ομαλά σε σωστό συγχρονισμό με τις ρόδες, όταν πατήσουμε τα πλήκτρα κίνησης.

5. Συμπεράσματα

Καθ' όλη τη διάρκεια έρευνας και μελέτης που πραγματοποιήθηκε για την πτυχιακή εργασία, αποκόμισα αρκετές γνώσεις για τα βιντεοπαιχνίδια γενικώς, αλλά και ειδικώς όσον αφορά τον τρόπο ανάπτυξής τους. Γεννήθηκε η επιθυμία και το ενδιαφέρον ν' ασχοληθώ περαιτέρω με αυτόν τον κλάδο και να προσανατολιστώ εκεί για την επαγγελματική μου αποκατάσταση. Από την πλευρά του σχεδιαστή, θεωρώ πως είναι άκρως δημιουργικό και από την πλευρά του προγραμματιστή είναι μια ευχάριστη αλλά δύσκολη πρόκληση.

Τέθηκαν όμως και αρκετά ερωτήματα, όπως το κατά πόσο συμβάλλουν τα βιντεοπαιχνίδια στην εξέλιξη της επικοινωνίας, του τρόπου εκπαίδευσης και τι αντίκτυπο επιστρέφουν σε μια σύγχρονη κοινωνία, όπου η πλειοψηφία των ανθρώπων καθημερινά ψυχαγωγείται από αυτά είτε περιστασιακά, είτε συστηματικά.

Αρνητικό γεγονός είναι ότι έως και σήμερα η adobe, επισήμως, δεν έχει δημοσιεύσει κανένα αξιόλογο βοήθημα ή παραδείγματα χρήσης των νέων ιδιοτήτων του director. Το ηλεκτρονικό λεξικό εντολών και συναρτήσεων που διατίθεται on-line, δεν επαρκή για την κατανόηση της γλώσσας lingo. Αυτό δυσκόλευσε την ερευνά μου σε μεγάλο βαθμό.

Παρά τις αντιξοότητες, το αποτέλεσμα είναι αξιόλογο και ευελπιστώ, αυτή η πτυχιακή να διδάξει τους νέους ενδιαφερόμενους, όπως δίδαξε κι εμένα.

Βιβλιογραφία

[1] Director MX 2004 Games - game Development with Director by *Nik Lever*
[2] Director in a Nutshell 1999 *Tim O'Reilly*
[3] Macromedia director MX 2004 - getting started with director
[4] Game Physics - engine development - series in interactive 3d technology by *Ian Millington*
[5] 3ds max 9 bible by *Kelly L. Murdock*
[6] New features – Autodesk 3d studio max 8 *ενσωματωμένος ηλεκτρονικός οδηγός, στο πρόγραμμα*

Πηγές

Adobe Director και Lingo scripting

- http://coderecipe.com/
- http://www.adobedirectoronline.com/
- http://www.adobe.com/products/director/
- http://www.havok.com
- http://www.director-online.com/
- http://www.deansdirectortutorials.com/
- http://userwww.sfsu.edu/~infoarts/technical/director/wilson.director.tutorial.html
- http://www.softwaretrainingtutorials.com/director-mx-2004-lingo.php
- http://www.cs.cf.ac.uk/Dave/Multimedia/node48.html
- http://www.agocg.ac.uk/train/lingo/index.htm
- http://www.furrypants.com/loope/index.htm
- http://www.noisecrime.com/
- http://www.3dpi-director.com/
- http://forums.techarena.in/reviews/1151022.htm
- http://www.gamedev.net/features/reviews/productreview.asp?categoryid=35&productid=712

Autodesk 3d studio max

- http://greece.autodesk.com/
- http://www.traptcg.com/
- http://en.wikipedia.org/wiki/3D_modeling
- http://ezinearticles.com/?An-Introduction-to-Spline-Modeling&id=1560349
- http://en.wikipedia.org/wiki/Utah_teapot
- http://en.wikipedia.org/wiki/3d_modeling

- http://en.wikipedia.org/wiki/UV_coordinates
- http://www.freeitsolutions.com/
- http://www.polygonblog.com/
- http://endlessquests.com/eqblogs/ice-child/?p=120

Extra πληροφορίες

- http://the-blueprints.com/
- http://www.adobe.com/products/shockwaveplayer/
- http://www.shockwave.com/home.jsp
- http://www.nvidia.com/object/physx_new.html
- http://www.warpedspace.org/lightingT/part1.htm
- http://www.gamedev.net/
- http://www.gamasutra.com
- http://www.gameover.gr/
- http://videogameslab.wordpress.com/