ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

MULTITOUCH GESTURES FOR MANIPULATING 3D SHAPES

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΤΟΥ

ΑΘΑΝΑΣΙΟΥ ΨΑΛΤΗ

Βόλος, Φεβρουάριος 2011

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

MULTITOUCH GESTURES FOR MANIPULATING 3D SHAPES

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΤΟΥ

ΑΘΑΝΑΣΙΟΥ ΨΑΛΤΗ

Επιβλέπων : Ηλίας Χούστης
Καθηγητής Τ.Μ.Η.Υ.Τ.Δ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή τον Ιανουάριο 2011.

(Υπογραφή)	*(Υπογραφή)*
.....	.....
Ηλίας Χούστης	Μανόλης Βάβαλης
Καθηγητής Τ.Μ.Η.Υ.Τ.Δ	Αναπληρωτής Καθηγητής Τ.Μ.Η.Υ.Τ.Δ

Βόλος, Φεβρουάριος 2011

(Υπογραφή)

.....

ΑΘΑΝΑΣΙΟΣ ΨΑΛΤΗΣ

Διπλωματούχος Μηχανικός Ηλεκτρονικών Υπολογιστών, Τηλεπικοινωνιών και Δικτύων Πανεπιστημίου Θεσσαλίας

Περίληψη

Στις οθόνες αφής, το multi-touch[17] αναφέρεται στην δυνατότητα ταυτόχρονης εγγραφής διακριτών θέσεων (σημείων) ως αποτέλεσμα αγγιγμάτων εισόδου. Οι οθόνες multi-touch αποτελούν μια πολλά υποσχόμενη τεχνολογία για την εμφάνιση και τον χειρισμό τρισδιάστατων αντικειμένων. Μολονότι ο χειρισμός των 2-διάστατων αντικειμένων έχει σχεδόν καλυφθεί, ο χειρισμός 3-διάστατων σε multi-touch οθόνες παραμένει ανεξερεύνητος.

Για την ανάπτυξη εφαρμογών γραφικών 3d, η διάταξη των 3-διάστατων αντικειμένων είναι ένας σημαντικός παράγοντας. Ωστόσο, είναι πολύ επίπονη δουλειά και παίρνει πολύ χρόνο γιατί τα 3-διάστατα αντικείμενα έχουν έξι βαθμούς ελευθερίας (DOF) και δεν είναι εύκολο να σχεδιαστούν χρησιμοποιώντας μια 2d συσκευή εισόδου.

Στα πλαίσια της έρευνας για την εκπόνηση της διπλωματικής μου, αναζητώντας την κατάλληλη συσκευή εισόδου που θα χρησιμοποιούσαμε, παρατηρήσαμε πως οι συσκευές touch της APPLE inc. (iPod-touch[13], iPhone[12], iPad[15]) δίνουν την αίσθηση «πραγματικής» αφής και επιπλέον το hardware που διαθέτουν ευνοεί την ανάπτυξη εφαρμογών γραφικών 3d υψηλής ευκρίνειας.

Ο σκοπός της εργασίας είναι η αξιοποίηση της τεχνολογίας που προσφέρουν οι καινοτόμες αυτές συσκευές μέσω της ανάπτυξης μιας εφαρμογής που αλληλεπιδρά με τον χρήστη, για τον χειρισμό 3d αντικειμένων με multi-touch κινήσεις προκαθορισμένες είτε από την ίδια την Apple είτε από εμάς. Πιο συγκεκριμένα προσπαθήσαμε θεωρητικά να βρούμε τις κατάλληλες κινήσεις touch, που θα δίνουν την αίσθηση χειρισμού ενός φυσικού 3d αντικειμένου και κατά το δυνατόν να τις εφαρμόσουμε μέσα σε ένα 3d γραφικό περιβάλλον. Η εφαρμογή έχει κατατεθεί στην Apple και θα είναι διαθέσιμη στο apple store με τίτλο ModelVision3D.

Λέξεις κλειδιά: «multi-touch οθόνες, 3D χειρισμός, χειρονομίες,iPhone».

Πίνακας περιεχομένων

1

Εισαγωγή

1.1 Multi-touch displays, multi-touch gestures

Η χειρονομία (gesture) είναι μια μορφή μη-λεκτικής επικοινωνίας με την οποία ορατές σωματικές ενέργειες μεταφέρουν κυρίως μηνύματα, είτε σε αντικατάσταση του λόγου είτε μαζί και παράλληλα με τον προφορικό λόγο. Οι χειρονομίες μπορεί να περιλαμβάνουν την κίνηση των χεριών, του προσώπου, ή του σώματος γενικά.

Μέχρι πρότινος ο τρόπος χειρισμού αντικειμένων από υπολογιστικά συστήματα ήταν το ποντίκι, οπού ο χειρισμός ήταν αρκετά πιο αργός από τον πραγματικό χειρισμό ενός αντικειμένου. Σε σύγκριση λοιπόν με το ποντίκι, οι οθόνες αφής παρέχουν επιπλέον σημεία επαφής και δίνουν την δυνατότητα χειρισμού των αντικειμένων απευθείας (άμεσα).

Η οθόνη αφής είναι μια συσκευή εισόδου που μπορεί να ανιχνεύσει την παρουσία και τη θέση του αγγίγματος μέσα στην περιοχή της οθόνης. Ο όρος αναφέρεται γενικά στην αφή ή άγγιγμα της οθόνης της συσκευής με ένα δάχτυλο ή χέρι. Μέχρι τις αρχές της δεκαετίας του 1980, οι περισσότερες καταναλωτικές

οθόνες αφής μπορούσαν να ανιχνεύσουν μόνο ένα σημείο επαφής σε κάθε στιγμή, και λίγες είχαν τη δυνατότητα να ανιχνεύσουν πόσο δυνατή ήταν η επαφή. Αυτό έχει αρχίσει να αλλάζει με την εμπορευματοποίηση της τεχνολογίας πολυ-αφής.

Οι οθόνες αφής έχουν δύο βασικά χαρακτηριστικά. Πρώτον, επιτρέπουν την απευθείας αλληλεπίδραση με τα στοιχεία που εμφανίζονται στην οθόνη, όπου και αν αυτή εμφανίζεται, και όχι έμμεσα όπως με ένα ποντίκι ή touchpad (πινακίδιο αφής). Δεύτερον, επιτρέπει σε κάποιον να το κάνει χωρίς να απαιτείται οποιαδήποτε συσκευή ενδιάμεσα, όπως π.χ. μια γραφίδα που πρέπει να κρατείται με το χέρι. Τέτοιες οθόνες μπορούν να συνδεθούν με ηλεκτρονικούς υπολογιστές. Επίσης, διαδραματίζουν ουσιαστικό ρόλο στο σχεδιασμό των ψηφιακών συσκευών, όπως οι προσωπικοί ψηφιακοί βοηθοί (PDA), οι δορυφορικές συσκευές πλοήγησης και τα κινητά τηλέφωνα. [14]

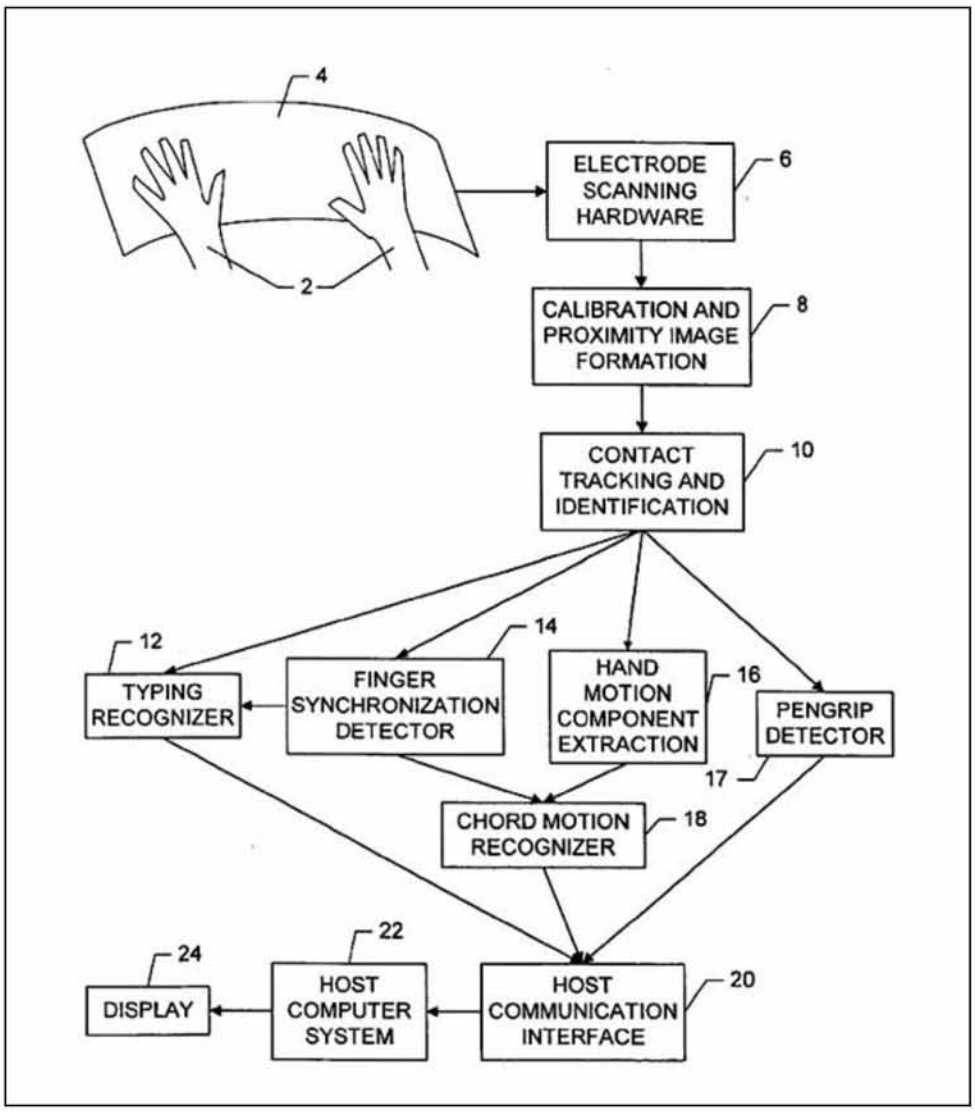
Εικόνα 1-1: παράδειγμα λειτουργίας multitouch.(http://en.wikipedia.org/wiki/Multi-touch)

Η τεχνολογία Multi-touch[17] έχει υλοποιηθεί με πολλούς διαφορετικούς τρόπους, εξαρτώμενους από τον τύπο και το μέγεθος της διεπαφής με τον χρήστη. Από φυσικής πλευράς υπάρχουν οι εξής τεχνολογίες :

- Resistive Touch (αντιστασιακή)
- Surface Capacitive Touch (χωρητική)
- Projected Capacitive Touch
- Bending ~Wave Touch
- Surface Acoustic Wave touch
- Infrared Touch
- Optical Touch

Η πιο δημοφιλής μορφή είναι οι πίνακες που προβάλουν μία εικόνα μέσα από ακρυλικό ή γυαλί και στην συνέχεια ρίχνουν black-light στην εικόνα μέσω led. Όταν

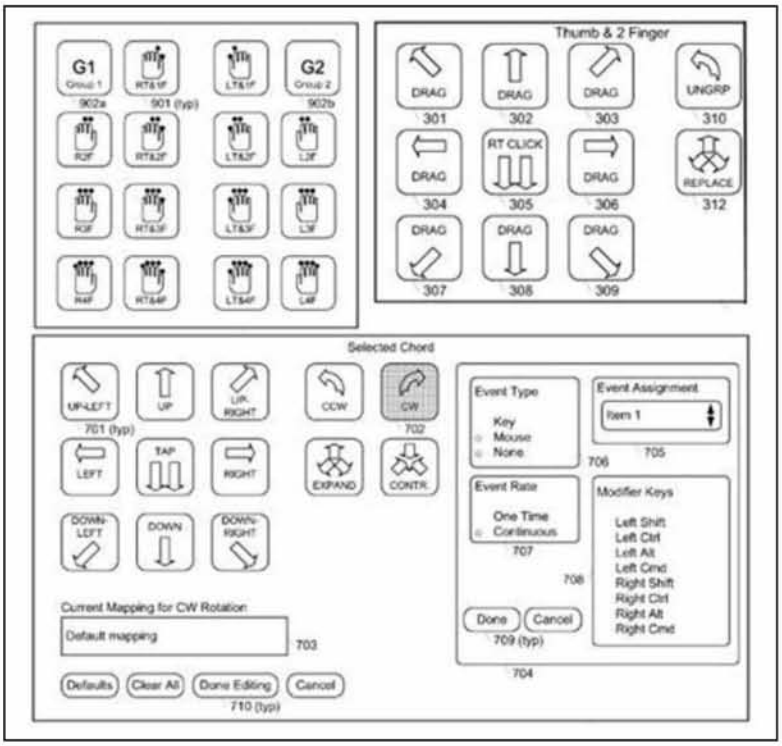
το δάχτυλο ή ένα αντικείμενο αγγίξει την επιφάνεια, προκαλώντας το φώς να σκορπίσει, εντοπίζεται η αντανάκλαση από αισθητήρες ή κάμερες και στέλνονται τα δεδομένα στο λογισμικό που υπαγορεύει απάντηση στο άγγιγμα, ανάλογα με την αντανάκλαση που μετρήθηκε. Επίσης επιφάνειες αφής μπορεί να γίνουν ευαίσθητες στην πίεση, με την προσθήκη επίστρωσης που λυγίζει διαφορετικά ανάλογα με το πόσο είναι πατημένο, αντικαθιστώντας την ανάκλαση. Οι φορητές τεχνολογίες χρησιμοποιούν ένα πάνελ που φέρει ηλεκτρικό φορτίο. Όταν ένα δάχτυλο αγγίξει την οθόνη, με το πάτημα διαταράσσει το ηλεκτρικό πεδίο του πίνακα. Η διακοπή αποθηκεύεται και εν συνεχεία αποστέλλεται στο λογισμικό, το οποίο εισάγει μια απάντηση στην κίνηση αυτή. [16]



Εικόνα 1-2: gesture recognition (http://www.appleinsider.com)

Κατά τα τελευταία χρόνια αρκετές εταιρίες έχουν κυκλοφορήσει προϊόντα που χρησιμοποιούν multi-touch σχεδιασμένα για τα πάντα που αφορούν από τον καθημερινό χρήστη μέχρι και τις πολυεθνικές εταιρίες. Οι κατασκευαστές λάπτοπ έχουν αρχίσει να συμπεριλαμβάνουν multitouch track pads στα λάπτοπ τους, οπως επίσης και την κατασκευή table-pc σαν το iPad[15] το οποίο ανταποκρίνεται στο άγγιγμα των δαχτύλων και όχι στην παραδοσιακή γραφίδα-αφής. Η εταιρία SYNAPTICS είναι ο κύριος προμηθευτής και κατασκευαστής των τεχνολογιών multi-touch στην αγορά.

Στον απόηχο του iPhone[12], αρκετές εταιρίες κινητής τηλεφωνίας έχουν ξεκινήσει να αντικαθιστούν τις μέχρι τώρα φορητές συσκευές που χρησιμοποιούσαν πλήκτρα με multi-touch διεπαφές. Έως τώρα τέτοιου είδους καινοτομίες έχουν εφαρμοστεί μόνο σε smart phones τελευταίας τεχνολογίας για σερφάρισμα στο διαδίκτυο και εφαρμογές εκτός βεβαίως από τις τηλεφωνικές υπηρεσίες. Κάθε κινητό με οθόνη αφής χρησιμοποιεί μια παρόμοια μέθοδο, αλλά αυτό που κάνει το iPhone ξεχωριστό είναι το πως το λογισμικό του iPhone ανταποκρίνεται τόσο γρήγορα σε κινήσεις των δακτύλων, αρκετά γρήγορα μάλιστα ώστε να δημιουργηθεί μια τεράστια αγορά για υψηλών προδιαγραφών παιχνίδια iPhone.



Εικόνα 1-3: the multi-touch gesture dictionary (http://www.patentlyapple.com/)

Αρκετές εταιρίες προσανατολίζονται σε μεγάλης κλίμακας επιφάνειες υπολογισμού και όχι σε προσωπικές ηλεκτρονικές συσκευές. Αυτά τα συστήματα χρησιμοποιούνται από κυβερνητικές οργανώσεις, μουσεία και εταιρίες για την παροχή πληροφοριών. Η εταιρεία Apple Inc. είναι η κάτοχος του εμπορικού ονόματος "Multitouch" και έχει κατοχυρωμένη πατέντα στα multi-touch gestures.

Να αναφέρουμε ότι οι τελευταίες εκδόσεις των περισσότερων λειτουργικών συστημάτων Mac OS X,Windows 7, Ubuntu όπως επίσης και λειτουργικά εφαρμοσμένα σε φορητές συσκευές Apple's iOS, Symbian, Google Android, υποστηρίζουν την τεχνολογία multi-touch. [10]

1.2 Αντικείμενο της Εργασίας

Ο 3-διάστατος χειρισμός είναι μια σημαντική πρόκληση για τους σχεδιαστές 3d user interface, εμπλέκοντας τον έλεγχο τον 6 βαθμών ελευθερίας: 3 για την θέση (μεταφορά στο x,y,z) και 3 για την περιστροφή στο x,y,z.

Δυσκολίες παρατηρούνται στην εύρεση και τυποποίηση του κατάλληλου χειρισμού στα 3d αντικείμενα και αυτό μπορεί να εξηγηθεί αν αναλογιστεί κανείς τις δυσκολίες χαρτογράφησης των σημείων επαφής εισόδου πάνω στα χαρακτηριστικά ενός 3d αντικειμένου, λόγω του δεδομένου της 2d φύσης της συσκευής εισόδου.

Η Apple με πατέντα έχει καθιερώσει κάποιες συγκεκριμένες κινήσεις (gestures) των δακτύλων, οι οποίες είναι ουσιαστικά ότι πιο σημαντικό για τα ανερχόμενα λειτουργικά συστήματα που θα στηρίζονται στην αφή, για εφαρμογή πάνω στα 2d αντικείμενα. Στόχος της διπλωματικής εργασίας είναι να προσαρμόσουμε τις περισσότερες από αυτές τις κινήσεις πάνω σε 3d σχήματα και να αποδώσουμε καινούριες κινήσεις δακτύλων, που θα μπορούσαν να χρησιμοποιηθούν για τον χειρισμό 3d σχημάτων, αξιολογώντας την ρεαλιστικότητα τους παράλληλα. [18]

Εικόνα 1-4: Apple multitouch gesture patent (http://www.patentlyapple.com)

2

Προδιαγραφές και αρχιτεκτονική της υλοποίησης

2.1 Απαιτήσεις χρηστών

Παρακάτω θα αναλύσουμε τις απαιτήσεις για μια εφαρμογή χειρισμού 3d αντικειμένων στο χώρο πάνω σε συσκευές multitouch της εταιρίας apple inc. Η στροφή μας προς αυτή τη συγκεκριμένη εταιρία υλικού και λογισμικού έχει να κάνει με το γεγονός της χρησιμοποίησης των κορυφαίων συσκευών multitouch τελευταίας τεχνολογίας αλλά και της τεράστιας αγοράς που διαθέτει ανά τον κόσμο. Έτσι για τους λόγους που αναλύσαμε και προηγουμένως αποκτά μεγαλύτερη αξία η δυνατότητα χειρισμού 3d γραφικών με multitouch κινήσεις.

2.1.1 Χειρισμός αντικειμένων

Ένα από τα κυρία χαρακτηριστικά μια εφαρμογής χειρισμού 3d αντικειμένων είναι η εύρεση των κατάλληλων κινήσεων(gestures). Η εφαρμογή θα πρέπει να παρέχει εύκολα αντιληπτές, κατανοητές και πλήρως εναρμονισμένες κινήσεις με την κίνηση ενός πραγματικού αντικειμένου στον έξω κόσμο. Ο χρήστης θα πρέπει να αναγνωρίζει την ρεαλιστικότατα της κάθε κίνησης και να την παραλληλίζει με την αντίστοιχη κίνηση πραγματικού κόσμου.

2.1.2 Ρεαλιστικά μοντέλα

Είναι θεμιτό ο χρήστης να θέλει να βλέπει στην οθόνη και να διαχειρίζεται μοντέλα υψηλής ανάλυσης. Θα πρέπει να του δίνεται η δυνατότητα να χρησιμοποιεί μοντέλα που έχει στην κατοχή του, που ο ίδιος έχει κατασκευάσει και θέλει να δει το

πως ανταποκρίνονται στις multitouch κινήσεις με ένα σύστημα εισαγωγής 3δ αντικειμένων.

2.1.3 Διαχείριση μοντέλων

Θέλουμε να ενσωματώσουμε μια βάση μοντέλων στην εφαρμογή ώστε να μπορεί να γίνει εύκολη η διαχείριση των μοντέλων που έχουν στην κατοχή τους οι χρήστες. Πρέπει να είναι δυνατή η άμεση επισκόπηση των μοντέλων έτσι ώστε με αυτόν τον τρόπο να μπορεί ο χρήστης να ενημερώνεται για τα χαρακτηριστικά των μοντέλων που διαθέτει, να εναλλάσσει το μοντέλο προβολής και να διαγραφεί τυχόν μοντέλα που δεν χρειάζεται.

2.1.4 Απεριόριστες επιλογές

Η δυνατότητα αναζήτησης μοντέλων είναι κάτι που έχει μεγάλη σημασία. Ο κάθε χρήστης θα πρέπει να μπορεί να έχει πρόσβαση σε μια καθολική βάση 3-διάστατων αντικειμένων. Έτσι θα μπορεί να αναζητά οτιδήποτε θα τον ενδιέφερε να προσομοιώσει στην συσκευή του και να το χειριστεί.

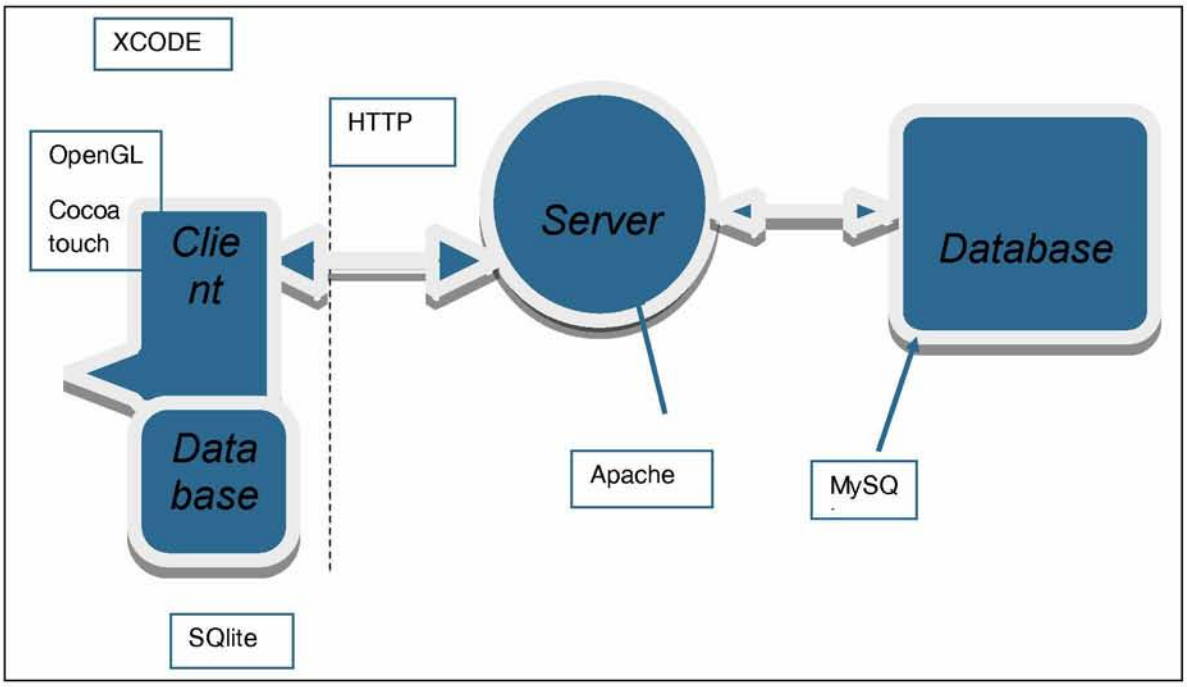
2.2 Αρχιτεκτονική του συστήματος

Αναλύοντας την αρχιτεκτονική του συστήματός μας αναφέρουμε ότι χρησιμοποιήθηκαν τεχνολογίες UI για να δημιουργηθεί μια εφαρμογή που να καλύπτει τις προαναφερθείσες απαιτήσεις.

Τα βασικά τμήματα της εφαρμογής είναι δυο:

1. Ένα UI τμήμα χειρισμού των αντικειμένων και
2. Ενα περιβάλλον διαχείρισης με μια βάση δεδομένων στην οποία ο σέρβερ ανακαλεί τα δεδομένα

Η δική μας υλοποίηση όπως φαίνεται και στην εικόνα 2.1 χρησιμοποιεί για το πρώτο κομμάτι OpenGL ES[2] και Cocoa touch[20], για την τοπική βάση δεδομένων χρησιμοποιείται SQlite ενώ για την απομακρυσμένη βάση χρησιμοποιείται MySQL. Για την μεταφορά των δεδομένων θα χρησιμοποιείται το απλό πρωτόκολλο HTTP.



Εικόνα 2-1

2.3 Περιγραφή λειτουργιών

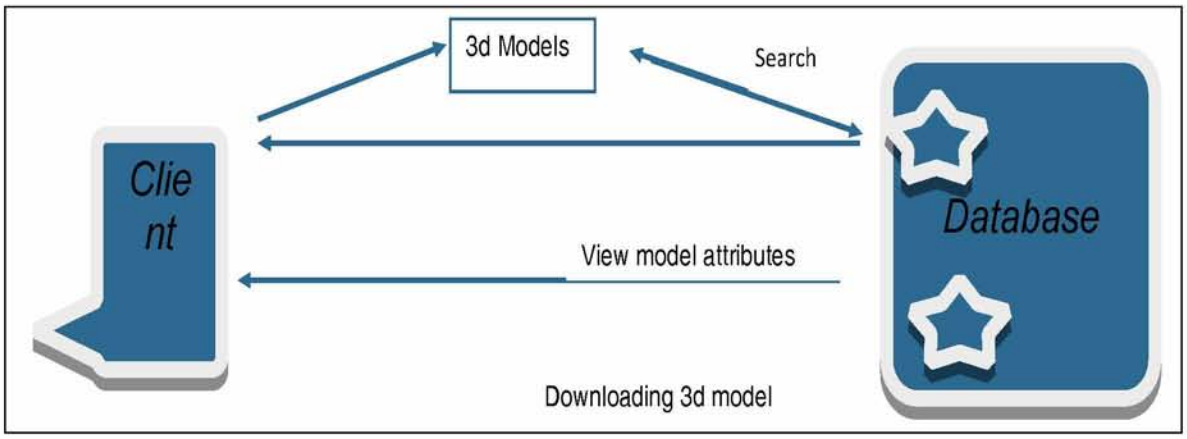
2.3.1 Διαχείριση βάσης μοντέλων

Η εφαρμογή αυτή θα μπορεί να τρέξει σε οποιαδήποτε συσκευή λειτουργεί με λογισμικό IOS iPhone[12], ο χρήστης θα μπορεί να διαθέτει μια προσωπική συλλογή από μοντέλα στα οποία μόνο αυτός θα έχει πρόσβαση. Η συλλογή αυτή θα βρίσκεται σε βάση δεδομένων μέσα στην συσκευή.

Ένας χρήστης θα μπορεί ακόμα να έχει πρόσβαση στην κεντρική βάση 3d αντικειμένων όπου και αν βρίσκεται. Αυτό γίνεται εφικτό, καθώς τα smart Phones, όπως το apple iPhone, iPad 3G , έχουν την δυνατότητα σύνδεσης στο Ιντερνέτ καθώς το δίκτυο 3g μπορεί πλέον να προσφέρει αξιοπρεπείς ταχύτητες. Πέραν αυτού η δυνατότητα σύνδεσης μέσω wi-fi επιτρέπει στους χρήστες να αξιοποιήσουν τυχόν υπάρχοντα σημεία πρόσβασης.

Εφόσον επιτευχθεί η πρόσβαση ο χρήστης θα έχει την δυνατότητα να εκτελέσει τις εξής βασικές λειτουργίες:

1. Να αναζητά εγγραφές 3d μοντέλων
2. Να εποπτεύει τα χαρακτηριστικά τους
3. Και ανάλογα με την επιλογή του να αποθηκεύει το μοντέλο της αρεσκείας από την κεντρική βάση στην τοπική ώστε να το διαχειρίζεται offline



Εικόνα 2-2: Λειτουργίες

Κατά την αναζήτηση του μοντέλου θα επιστραφεί ένα αρχείο XML από όπου θα ανασύρουμε τις απαραίτητες πληροφορίες και θα δημιουργήσουμε την λίστα με τα μοντέλα που είναι διαθέσιμα. Για κάθε μοντέλο που επιλέγουμε μας δίνεται η δυνατότητα να αντλήσουμε πληροφορίες για αυτό και αν επιθυμούμε να το κατεβάσουμε μέσω πρωτοκόλλου HTTP. Το αρχείο θα αποθηκευτεί στη συσκευή και δημιουργείται μια νέα θέση στον αντίστοιχο πίνακα στην τοπική βάση.

2.3.2 Χειρισμός μοντέλων

Έχοντας μιλήσει για το κομμάτι διαχείρισης των 3d μοντέλων και τις λειτουργίες που έχουν υλοποιηθεί, έμεινε να ασχοληθούμε με το σημαντικό το κομμάτι, αυτό της υλοποίησης της απεικόνισης και του χειρισμού των αντικειμένων με κινήσεις multitouch.

Όπως ήδη αναφέραμε υπάρχουν 2 ξεχωριστές λειτουργίες:

1. Η υλοποίηση της απεικόνισης
2. Και ο χειρισμός των αντικειμένων

Σε ένα πλήθος διαφορετικών τύπων αρχείων μοντέλων, προσπαθήσαμε να εντοπίσουμε εκείνο τον τύπο που επιτυγχάνει παράλληλα τους στόχους που έχουν τεθεί όσον αφορά τον ρεαλισμό των μοντέλων , την υψηλή ανάλυση αλλά και την δυνατότητα μεταφοράς σε συσκευές που λειτουργούν με iPhone iOS. Καταλήξαμε στο τύπο αρχείων .OBJ όπου ικανοποιεί τα παραπάνω, αλλά και είναι ένας τύπος αρχείων αρκετά διαδεδομένος σε σχεδιαστές 3d μοντέλων. Τέτοιου τύπο αρχεία μπορούν να εξαχθούν από σχεδόν όλα τα προγράμματα κατασκευής 3d μοντέλων που χρησιμοποιούν οι σχεδιαστές, όπως 3dsMax, Maya,blender.

Το αρχείο obj ουσιαστικά αποτελεί μια αναπαράσταση της επιφάνειας του μοντέλου. Αναλυτικότερα είναι μια απλή αναπαράσταση δεδομένων που παρουσιάζουν την 3d γεωμετρία, καταγράφοντας την θέση κάθε μιας κορυφής, την uv θέση κάθε μιας κορυφής υφής, τις νόρμες, αλλά και τις όψεις κάνοντας κάθε πολύγωνο να ορίζεται σαν μια λίστα από κορυφές, κορυφές υφών. Οι κορυφές αναγράφονται με φορά αυτή την αντίθετη του ρολογιού.

Εστιάζοντας στο αρχείο περισσότερο παρατηρούμε τα εξής:

#Λίστα κορυφών, με (x,y,z,[w]) συντεταγμένες	#συντεταγμένες υφής , σε (u,v[,w]) συντεταγμένες	# Νόρμες σε (x,y,z) ;	# όψεις
v 0.123 0.234 0.345 1.0 v ...	vt 0.500 -1.352 [0.234] vt	vn 0.707 0.000 0.707 vn ...	f 1 2 3 f 3/1 4/2 5/3 f 6/4/1 3/5/3 7/6/5 f ...

Σημαντική διαφορά της OpenGL ES από opengl[2] είναι ο περιορισμός στην χρήση διαφόρων πολυγώνων στην αναπαράσταση αντικειμένων και συγκεκριμένα η αποκλειστική χρήση στην αναπαράσταση με τρίγωνα σχήματα. Αναλύοντας τα δεδομένα του αρχείου συγκεντρώνουμε και χωρίζουμε τις κορυφές, τις νόρμες, τις υφές και τις όψεις σε πίνακες. Αντλώντας τα στοιχεία για τις κορυφές από τους πίνακες και χωρίζοντας τα σε τριάδες αρχίζουμε να απεικονίζουμε το μοντέλο τρίγωνο - τρίγωνο δημιουργώντας όψεις.

Από την στιγμή που καταφέραμε να εμφανίσουμε το αντικείμενο στην οθόνη επικεντρωθήκαμε στο κομμάτι της εύρεσης κατάλληλων κινήσεων για τον χειρισμό του αντικειμένου και φυσικά την ερμηνεία τους πάνω στην οθόνη με multitouch κινήσεις. Οι βασικές κινήσεις που έχουν υλοποιηθεί γύρω από τα 2διαστατα αντικείμενα έχουν να κάνουν με την περιστροφή την μετατόπιση και την εστίαση (μεγέθυνση, σμίκρυνση).

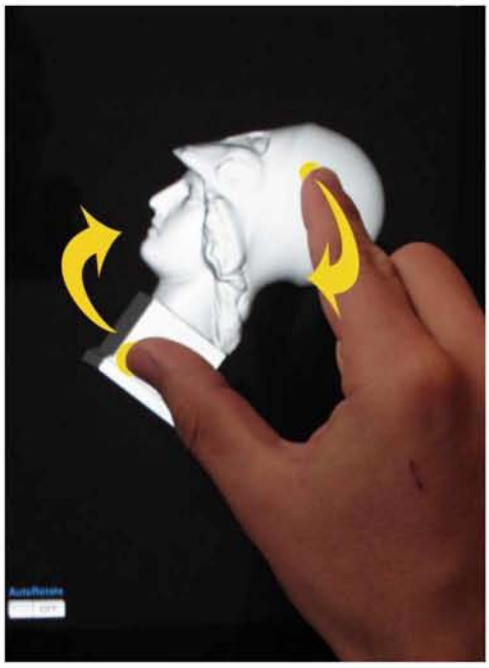
2.3.2.1 Περιστροφή αντικείμενου

Όσον αφορά την περιστροφή για 3-διαστατα αντικείμενα σκεφτήκαμε πως η κινήσεις που θα υλοποιηθούν πρέπει να υποδηλώνουν την περιστροφή στον πραγματικό κόσμο. Έτσι καταλήξαμε στην πραγματοποίηση των δυο παρακάτω κινήσεων.

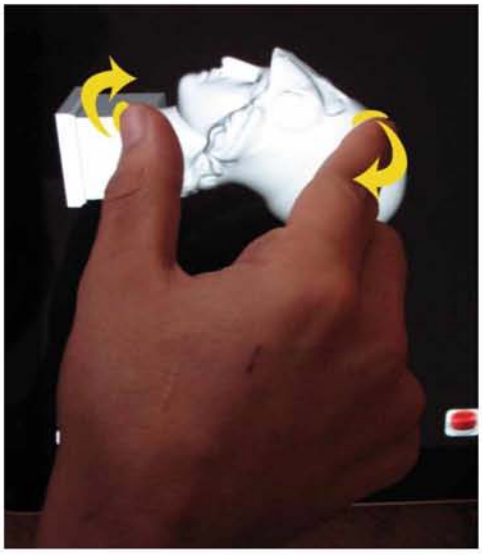
Περιστροφή ως προς τον άξονα z:

- Αγγίζοντας το αντικείμενο με δυο δάχτυλα πάνω στην οθόνη και κινώντας τα σε αντίθετη κατεύθυνση πάντα σε κυκλική τροχιά , το αντικείμενο κινείται περιστροφικά στην φορά των δαχτύλων με άξονα περιστροφής τον z. Όσο η ταχύτητα περιστροφής των δάχτυλων αυξάνεται τόσο αυξάνεται και η ταχύτητα περιστροφής του αντικείμενου. Αφήνει την αίσθηση στον χρήστη του κρατήματος του αντικείμενου από 2 σημεία και περιστροφής βασιζόμενο σε αυτά.

Εικόνα 2-3



Εικόνα 2-4

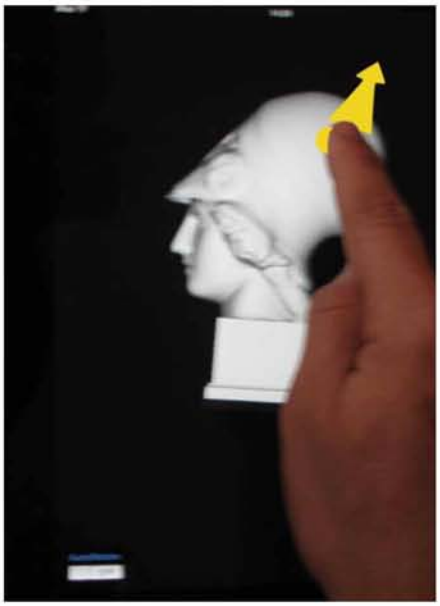


Εικόνα 2-5

Επειδή όμως η κίνηση αυτή δεν καλύπτει την πλήρη περιστροφή του αντικείμενου σε όλους τους άξονες θεωρήσαμε σωστό την εύρεση δεύτερης κίνησης, όπου η μετατόπιση ενός δαχτύλου πάνω στο αντικείμενο θα υποδηλώνει την κατεύθυνση προς την οποία θέλουμε να περιστραφεί και όχι τον άξονα της περιστροφής του.

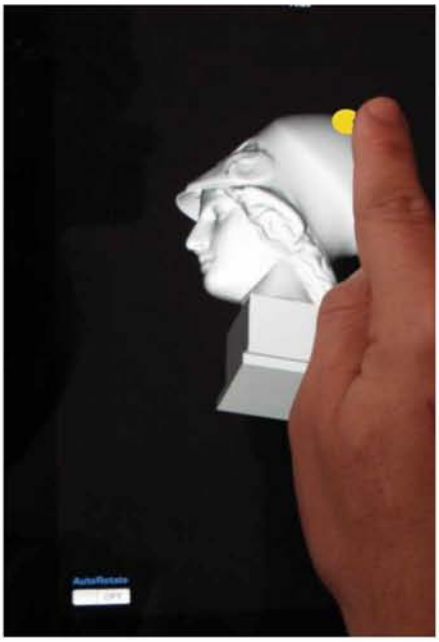
- Αγγίζοντας το αντικείμενο με ένα δάχτυλο και μετατοπίζοντας το, το αναγκάζουμε να κινηθεί και αυτό μαζί με το δάχτυλο διαγράφοντας περιστροφική κίνηση στους άξονες x ,y. Ουσιαστικά δίνει την εντύπωση του ώθησης του αντικειμένου προς της κατεύθυνση του δάχτυλου ώστε να εμφανιστούν στην οθόνη οι κρυμμένες όψεις.

Εικόνα 2-6

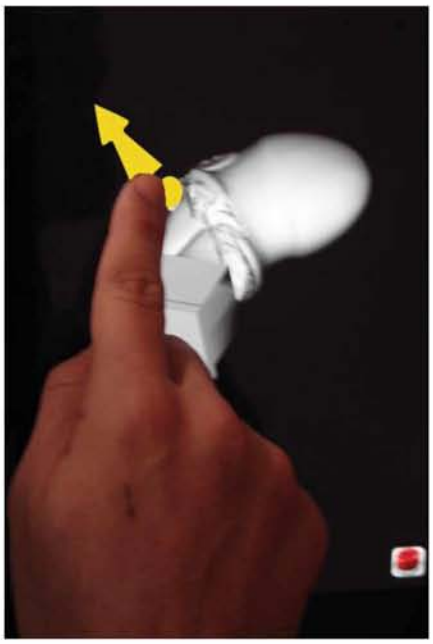


Εικόνα 2-7

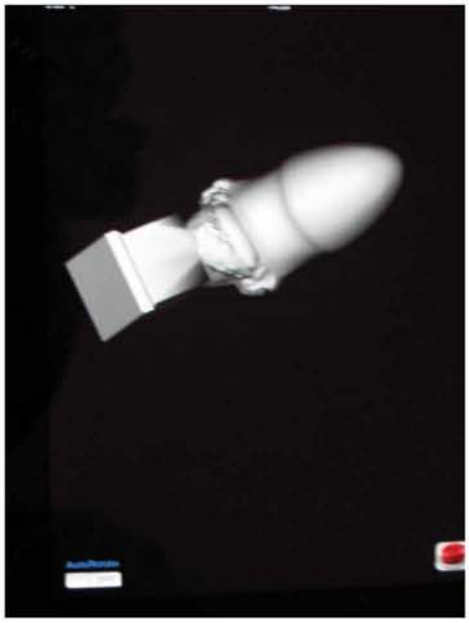
Στις εικόνες 2-6, 7,8,9,10 παρουσίαζεται η περιστροφη του αντικειμενου στο πατημα του δαχτυλου



Εικόνα 2-8



Εικόνα 2-9

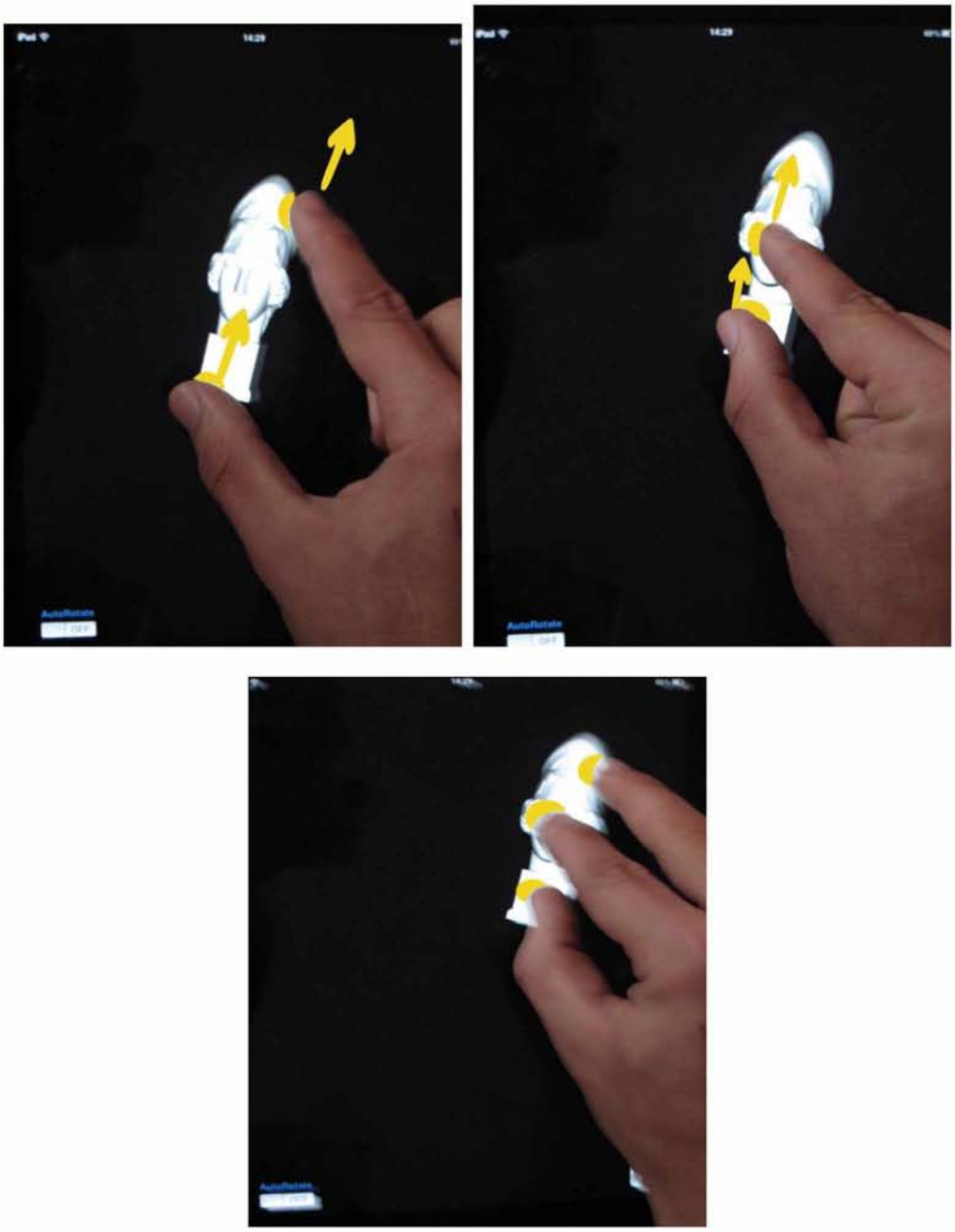


Εικόνα 2-10

2.3.2.2 Μετακίνηση αντικειμένου

Για την μετακίνηση ενός αντικειμένου στο πραγματικό κόσμο είναι απαραίτητο να το κρατήσουμε με 2 ή περισσότερα δάχτυλα, έτσι και στο κόσμο της εφαρμογής μας σκεφτήκαμε πως έπρεπε να «κρατάμε» το αντικείμενο με παρόμοιο τρόπο.

- Αγγίζοντας το αντικείμενο με δυο ή και περισσότερα δάχτυλα και μετακινώντας τα πάνω στην οθόνη δίνει την εντολή στο αντικείμενο να ακολουθήσει την κίνηση αυτή. Σηκώνοντας τα δάχτυλα από την οθόνη το αντικείμενο αφήνεται στην τελική του θέση.

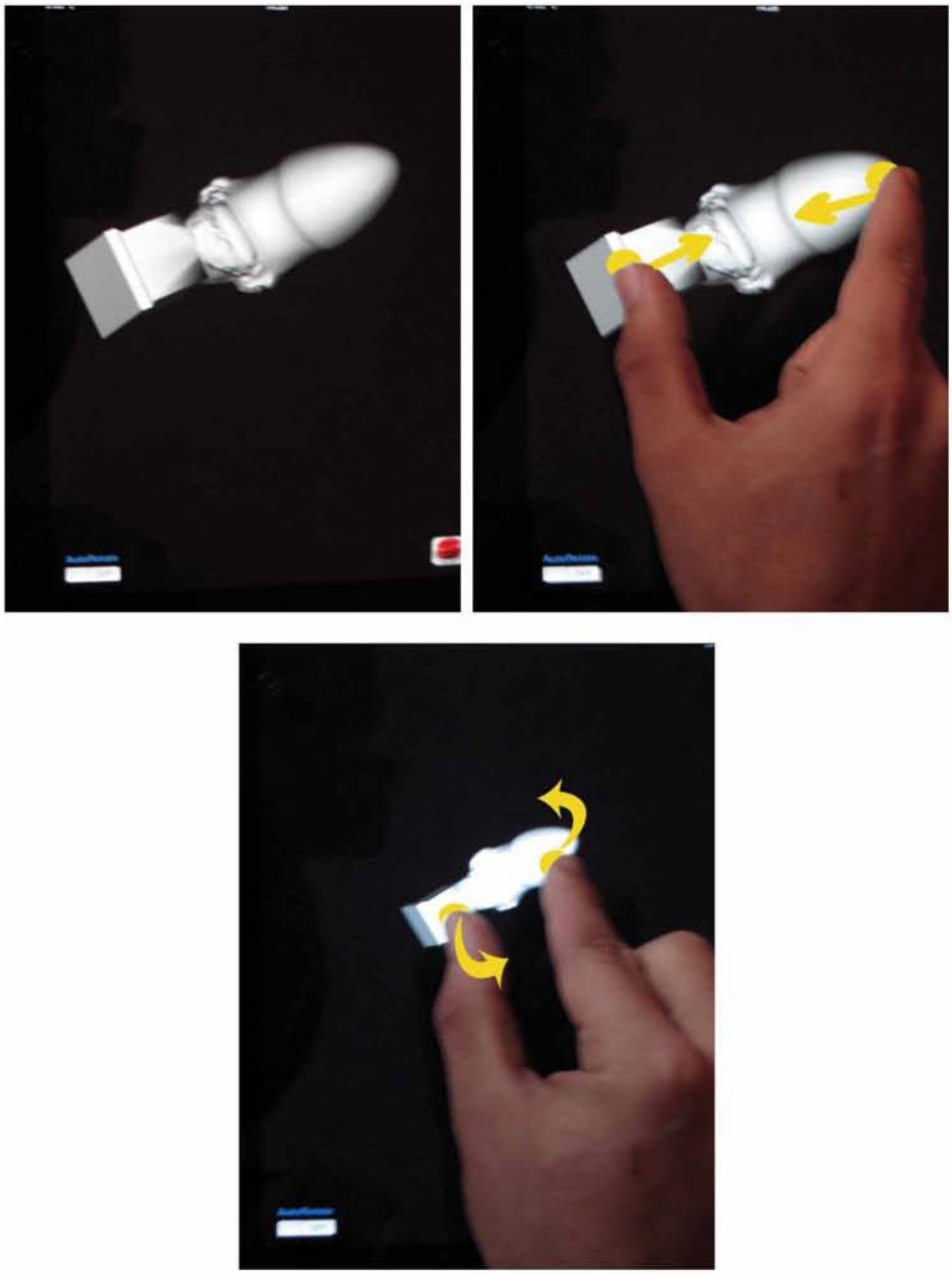


Εικονες 2-11, 12, 13 μετακινηση αντικειμένου στο χωρο.

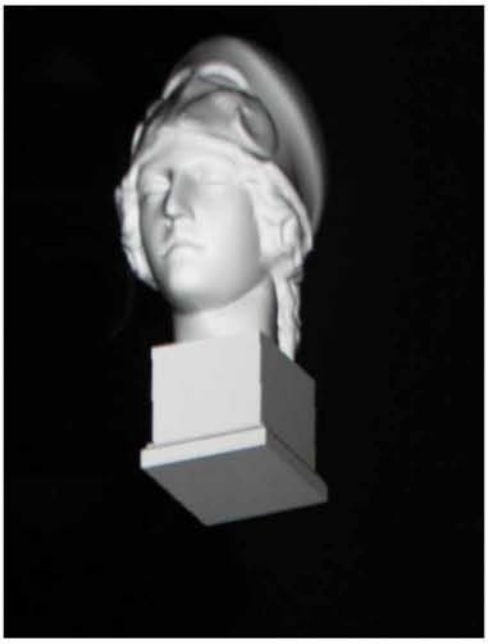
2.3.2.3 Εστίαση (μεγέθυνση, σμίκρυνση) αντικειμένου

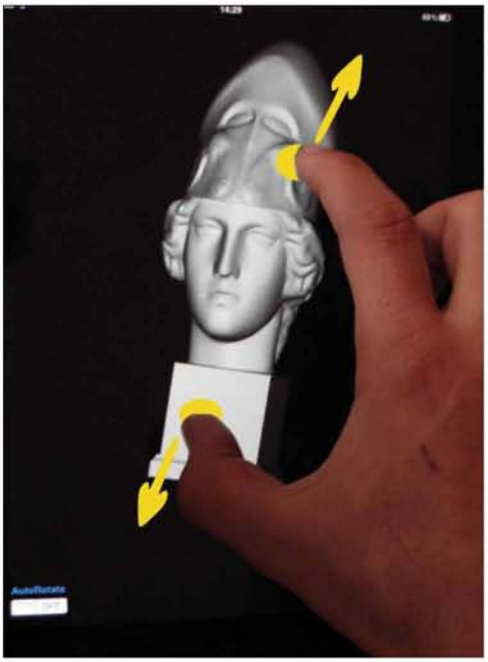
Υλοποιώντας την εστίαση χρησιμοποιήσαμε την προκαθορισμένη κίνηση που εφαρμόζεται και σε 2d αντικείμενα (pinch gesture).

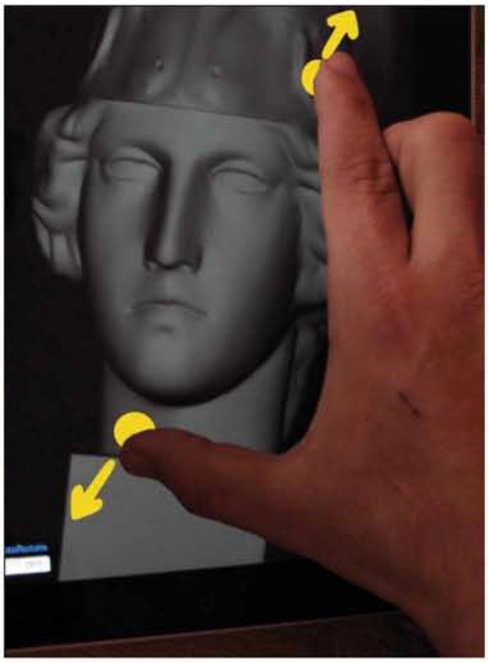
- Αγγίζοντας το αντικείμενο με δυο δάχτυλα και απομακρύνοντας το ένα από το άλλο, προσομοιώνοντας την αίσθηση της διείσδυσης, εστιάζουμε στο αντικείμενο. Με την ακριβώς αντίθετη κίνηση απομακρυνόμαστε από αυτό.



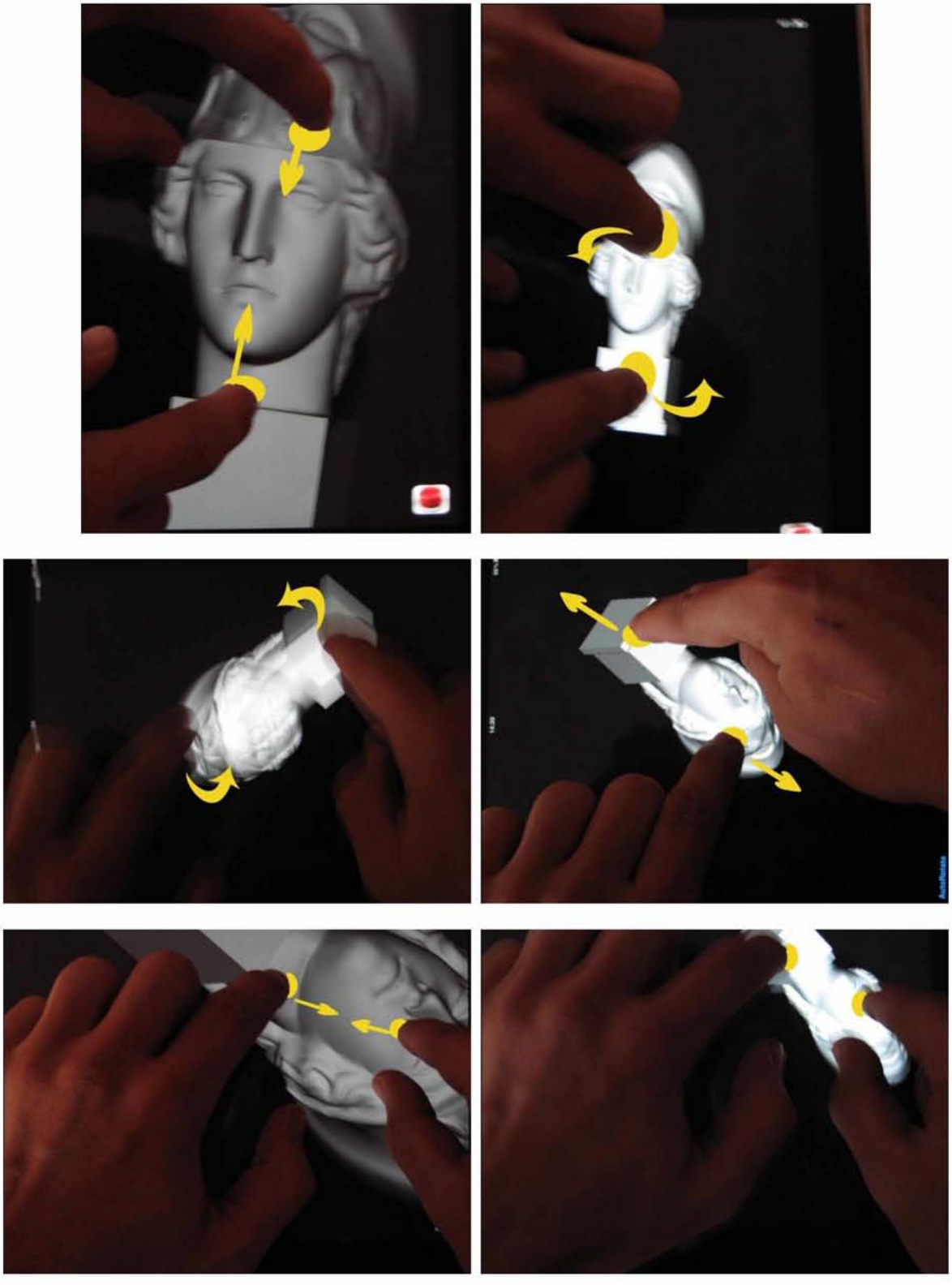
Εικονες 2-14, 15, 16 Εστίαση(σμίκρυνση) αντικειμένου.







Εικονες απο 2-17 εως 2-25 παρουσιάζουν ενα παράδειγμα σύνθετων κινήσεων(εστιαση,περιστροφή) ,με την χρησιμοποιηση και των δυο χεριων.

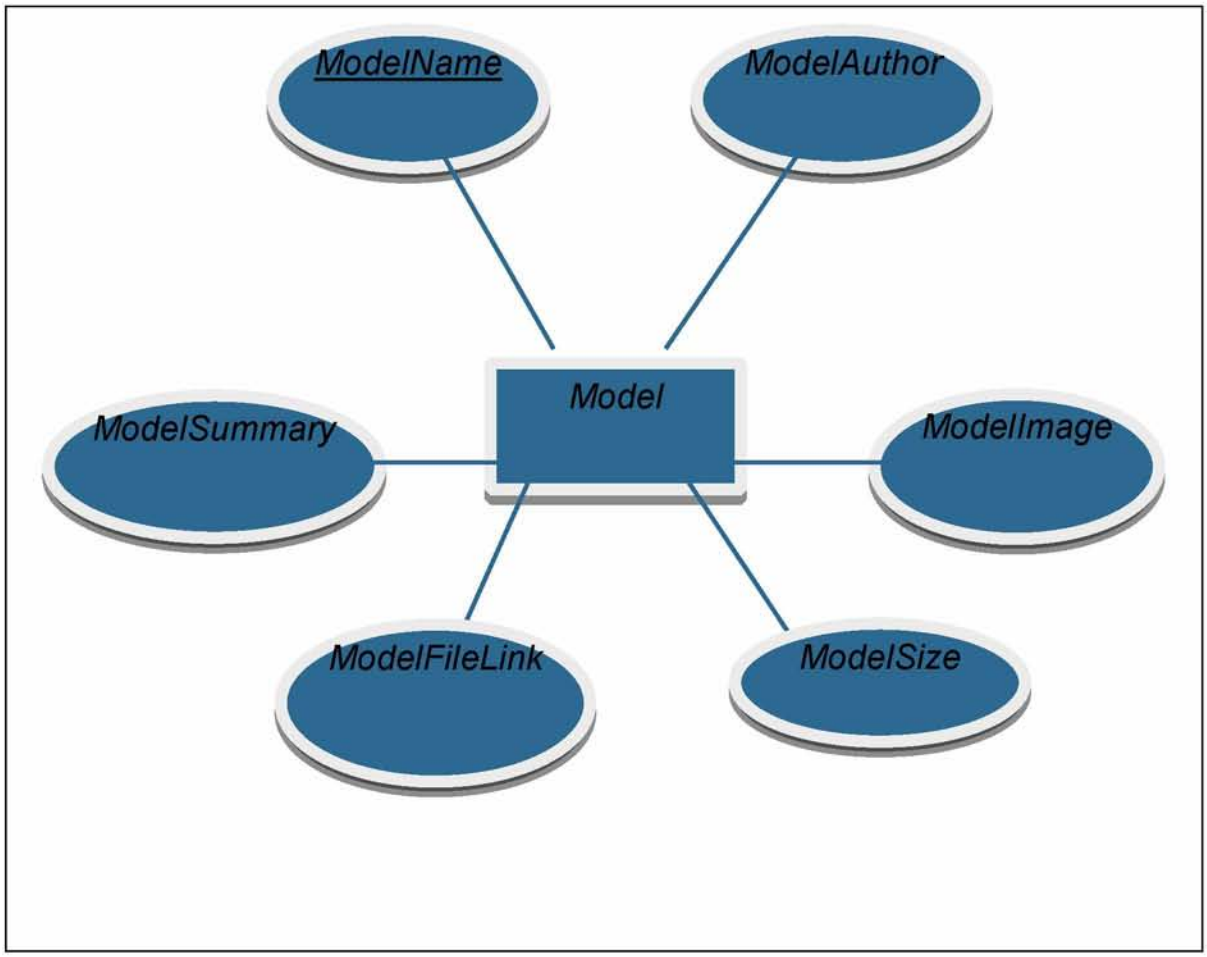


2.3.2.4 Άλλες κινήσεις

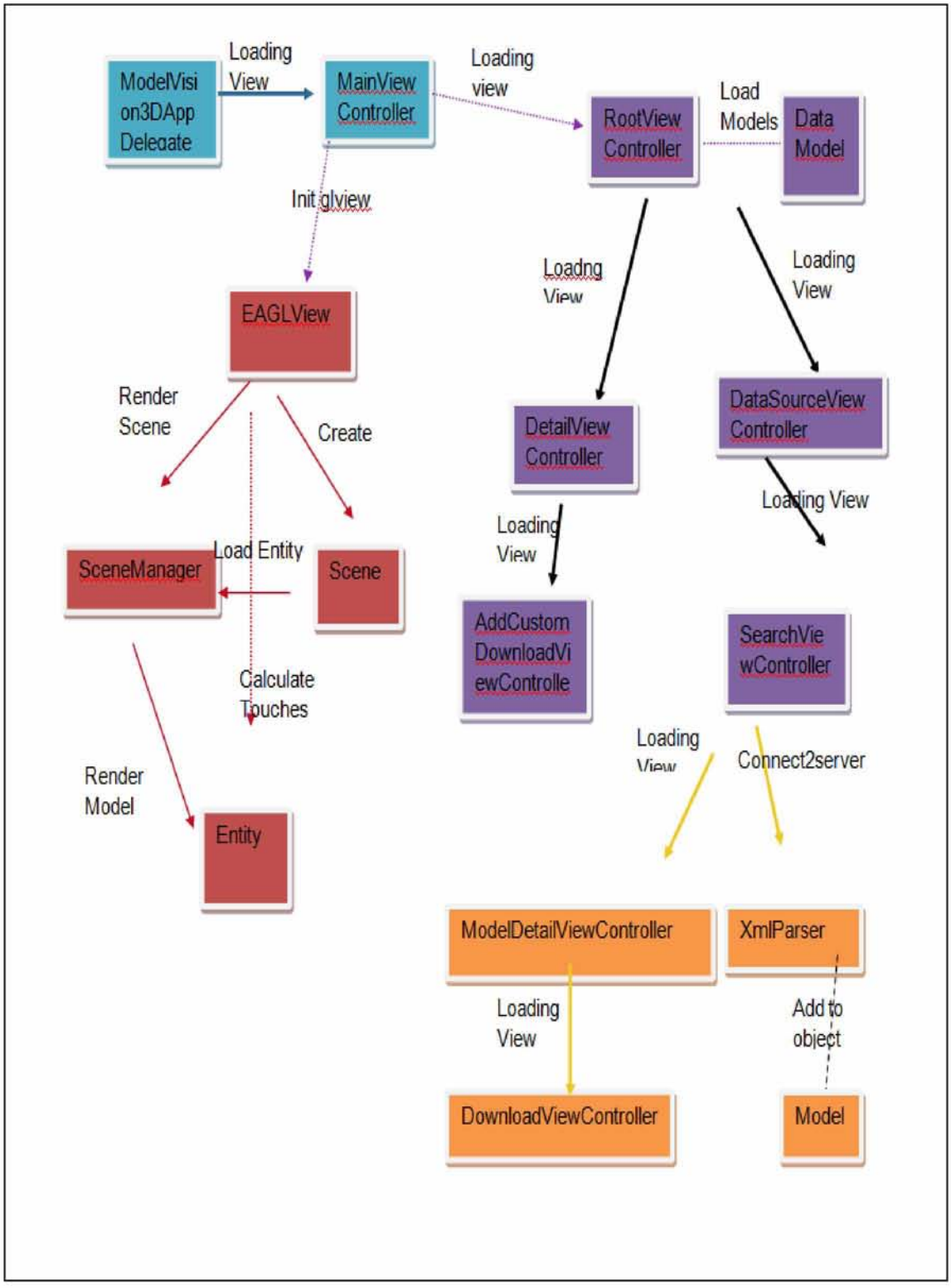
Για επίτευξη καλύτερης λειτουργικότητας του προγράμματος χρησιμοποιήθηκαν κινήσεις όπως double tap και triple tap, για επαναφορά στην αρχική θέση και προβολή ιστογράμματος του αντικειμένου αντίστοιχα.

2.4 Μοντέλο Οντοτήτων - Συσχετίσεων

Παρουσίαση μοντέλου Οντοτήτων -Συσχετίσεων της βάσης δεδομένων που χρησιμοποιεί η εφαρμογή μας τόσο στο σέρβερ όσο και στο mobile.



2.5 Σχέσεις κλάσεων



3

Θεωρητικό υπόβαθρο-Τεχνολογίες που χρησιμοποιήθηκαν

Στο κεφάλαιο αυτό θα ασχοληθούμε με τις τεχνολογίες που χρησιμοποιήθηκαν σε κάθε κομμάτι της εφαρμογής και είναι αναγκαία η κατανόηση τους, θα αναλύσουμε ειδικότερα την βιβλιοθήκη OpenGL ES[2],την γλώσσα προγραμματισμού objective-C[7],το Cocoa,Cocoa touch framework[20], την Ψηφιακή Φυσική, το Core Animation, συστήματα βάσεων δεδομένων SQlite και MySQL.

3.1 Κατανοώντας τα iPhone gestures

3.1.1 Digital physics

Το λογισμικό της Apple Inc. Είναι συντονισμένο έτσι ώστε να ανταποκρίνεται στις δικές μας κινήσεις χεριών με ένα τρόπο που δίνει την εντύπωση της επαφής με το φυσικό αντικείμενο. Ακόμα, οι αλληλεπιδράσεις είναι θεμελιώδης ψηφιακές. Τα πάντα έχουν την αίσθηση της ελαφρότητας και του απλού, επειδή δεν υπάρχει τριβή.

Είναι πιθανόν ότι το iPhone της Apple είναι μόνο η αρχή μιας σειράς πολλών συσκευών που θα χρησιμοποιήσουν την λεγόμενη ψηφιακή φυσική, μια εμπειρία

βασισμένη σε ένα πραγματικό παγκόσμιο φαινόμενο, αλλά ταυτόχρονα εκμεταλλεύεται τις απεριόριστες δυνατότητες του λογισμικού.

Για τους περισσότερους η ψηφιακή φυσική είναι μια νέα λέξη μια νέα εμπειρία. Κάπου κει στο 90΄ τα πρώτα windows desktop έφεραν μια πρωτόγνωρη εμπειρία αλληλεπίδρασης στην καθημερινή πραγματικότητα. Ανάγκασαν τον κόσμο να μάθει να χρησιμοποιεί νέες μεθόδους επικοινωνίας με τα «έξυπνα» μηχανήματα με την χρήση του ποντικιού και του πληκτρολογίου. Ήταν η επιθυμία μας να βάλουμε το χέρι μας μέσα στον υπολογιστή για δείξουμε κάτι και το ποντίκι έγινε η προέκταση του χεριού μας. Το iPhone μεταφέρει αυτή την εμπειρία σε άλλο επίπεδο, αφήνοντας εμάς να αλληλεπιδράσουμε με την ψηφιακή πληροφορία χρησιμοποιώντας τα χέρια μας.

Η εμπειρία αυτή που δημιούργησε η Apple στο iPhone είναι η ψηφιακή φυσική. Το κυριότερο χαρακτηριστικό του iPhone είναι ο ρεαλισμός, γιατί όλες οι αλληλεπιδράσεις χρήστη – συσκευής σου δίνουν την αίσθηση ότι είναι πραγματικές. Ωστόσο η εμπειρία αυτή δεν είναι τόσο κοντά σε μια εμπειρία πραγματικού κόσμου. Απεναντίας είναι ένα είδος αλληλεπίδρασης χωρίς τριβή, και γι αυτο μια απλή κίνηση αρκεί για να κινηθεί ένα ολόκληρο «βουνό» ψηφιακών δεδομένων.

Το σημείο-κλειδί σχετικά με την ψηφιακή φυσική είναι ότι δεν προσπαθεί να αναγκάσει τους χρήστες να εργαστούν σκληρά. Η επιτυχία είναι ότι έχει μια διεπαφή που είναι προφανής. Για δεκαετίες έχουμε παιδευτεί με το λογισμικό που μας εξέπληττε, επειδή συμπεριφερόταν με τρόπους που δεν το περιμέναμε να κάνει. Το λογισμικό που ακολουθεί τους νομούς της ψηφιακής φυσικής δεν έχει αυτό το πρόβλημα. Είναι τόσο απλό στην χρήση, επειδή εκμεταλλεύεται τις κινήσεις που χρησιμοποιούμε κάθε μέρα στην πραγματική ζωή μας. [26]

3.1.2 Gestures

Οι βασικές κινήσεις που πρέπει να γνωρίζουμε είναι οι εξής:

- Tap: Για να ξεκινήσει μια εφαρμογή, ένα απλό άγγιγμα στην οθόνη αφής με το δάχτυλο, στο σχετικό εικονίδιο της εφαρμογής αρκεί. Για παράδειγμα, το εικονίδιο του Safari στην Αρχική οθόνη.

- Double-Tap: Ένα γρήγορο διπλό άγγιγμα του δακτύλου έχει μια ειδική επίδραση σε διάφορους τύπους εφαρμογών. Ένα διπλό άγγιγμα σε βίντεο διευρύνει την εικόνα του βίντεο, ενώ ένα δεύτερο διπλό άγγιγμα θα επαναφέρει την εικόνα του βίντεο στο αρχικό του μέγεθος.

- Drag: Κατά την ανάγνωση ενός ηλεκτρονικού ταχυδρομείου ή την προβολή του περιεχομένου μιας ιστοσελίδας, μπορείτε να σύρετε αργά το δάχτυλό σας πάνω στην οθόνη είτε οριζόντια είτε κάθετα για να μετακινηθείτε στο κείμενο επί της οθόνης στην επιλεγμένη κατεύθυνση σας.

- Flick: Μια κίνηση που είναι παρόμοια με το drag, με την διάφορα ότι μετακινεί τον δείκτη γρηγορότερα προς μια κατεύθυνση και το αποτέλεσμα είναι η Γρηγόρη μετάβαση σε αυτή.

- Delete : ένας γρήγορος και εύκολος τρόπος για να διαγράφουν στοιχεία όπως ένα βίντεο, το τραγούδι ή το ηλεκτρονικό ταχυδρομείο, χρησιμοποιώντας μια κίνηση, έχοντας οριζόντια κατεύθυνση, πάνω από το συγκεκριμένο όνομα του στοιχείου. Αυτό θα αποκαλύψει ένα κόκκινο κουμπί delete που σας επιτρέπει να αφαιρέσετε το επιλεγμένο στοιχείο. Εάν πατήσετε το κουμπί Διαγραφή θα σας δοθεί η ευκαιρία για επιβεβαίωση. Εάν αλλάξετε γνώμη, είτε πατήστε το πλήκτρο ακύρωσης (όταν σας ζητηθεί να επιβεβαιώσετε), ή ακουμπήστε οπουδήποτε εκτός από το κόκκινο κουμπί delete για να ακυρώσετε.

- Pinch: εκτελώντας μια ενέργεια που ονομάζεται pinch είναι δυνατό να μεγεθύνουμε και να σμικρύνουμε μια συγκεκριμένη περιοχή της οθόνης (τοποθέτηση δύο δακτύλων πάνω στην οθόνη και απομάκρυνση τους (σε μεγέθυνση) ή τοποθέτηση των δάχτυλων και πλησίασμα τους (για σμίκρυνση).

Διαπιστώνεται ότι αυτό είναι παρόμοιο με ένα διπλό άγγιγμα μόνο η διαφορά είναι ότι το pinch δίνει μεγαλύτερο έλεγχο. [11]

3.2 OpenGL and OpenGL ES

Η OpenGL (Open Graphics Library) [2] αποτελεί μια βασική προδιαγραφή που καθορίζει τη διαγλωσσική, cross-platform API για τη σύνταξη των εφαρμογών που παράγουν 2D και 3D γραφικά υπολογιστών. Το interface αποτελείται από πάνω από 250 διαφορετικά καλέσματα μεθόδων το οποίο μπορεί να αξιοποιηθεί για να αναπαραχθούν πολύπλοκες τρισδιάστατες σκηνές από απλά αρχέτυπα. Η OpenGL αναπτύχθηκε από την Silicon Graphics Inc το 1992 και χρησιμοποιείται ευρέως σε εφαρμογές CAD, εικονική πραγματικότητα, την επιστημονική απεικόνιση, απεικόνιση πληροφοριών, και την προσομοίωση πτήσης. Χρησιμοποιείται επίσης σε video games, όπου ανταγωνίζεται με την Direct3D σε πλατφόρμα Microsoft Windows. Η OpenGL είναι υπό τη διαχείριση της μη κερδοσκοπική τεχνολογικής κοινοπραξίας Khronos Group.

Η OpenGL είναι μια χαμηλού επιπέδου, διαδικαστική API, απαιτώντας από τον προγραμματιστή να υπαγορεύσει τα ακριβή βήματα που απαιτούνται για να αναπαραχθεί μια σκηνή. Αυτό έρχεται σε αντίθεση με την περιγραφική API, όπου ένας προγραμματιστής πρέπει μόνο να περιγράψει μια σκηνή και μπορεί να αφήσει την βιβλιοθήκη να διαχειριστεί τις λεπτομέρειες της αναπαραγωγής της. Η χαμηλού επιπέδου σχεδιασμού OpenGL απαιτεί οι προγραμματιστές να έχουν καλή γνώση του graphics pipeline, αλλά και δίνει μια κάποια ελευθερία να εφαρμόσουν αλγόριθμους απόδοσης. [1]

3.2.1 OpenGL ES

Η OpenGL για ενσωματωμένα συστήματα (OpenGL ES) είναι ένα υποσύνολο της OpenGL API γραφικων3D, σχεδιασμένη για φορητές συσκευές όπως τα iPod και iPhone.

Η OpenGL ES ® είναι μια δωρεάν, cross -platform API για την πλήρη λειτουργία 2D και 3D γραφικών για τα ενσωματωμένα συστήματα, συμπεριλαμβανομένων τις κονσόλες, κινητά τηλέφωνα, συσκευές και οχήματα Αποτελείται από σαφώς καθορισμένα υποσύνολα του desktop OpenGL, δημιουργώντας μια ευέλικτη και ισχυρή διασύνδεση χαμηλού επιπέδου μεταξύ του λογισμικού και του graphic acceleration. Η OpenGL ES περιλαμβάνει προφίλ για floating-point και fix-point συστήματα και η EGL ™ προδιαγραφή δεσμευτική για τα συστήματα παραθύρων. OpenGL ES 1.X είναι για σταθερό λειτουργικότητας υλικό και προσφέρει επιτάχυνση, ποιότητα εικόνας και επιδόσεις. Η OpenGL ES 2.x, επιτρέπει πλήρη προγραμματιζόμενα 3D γραφικά.

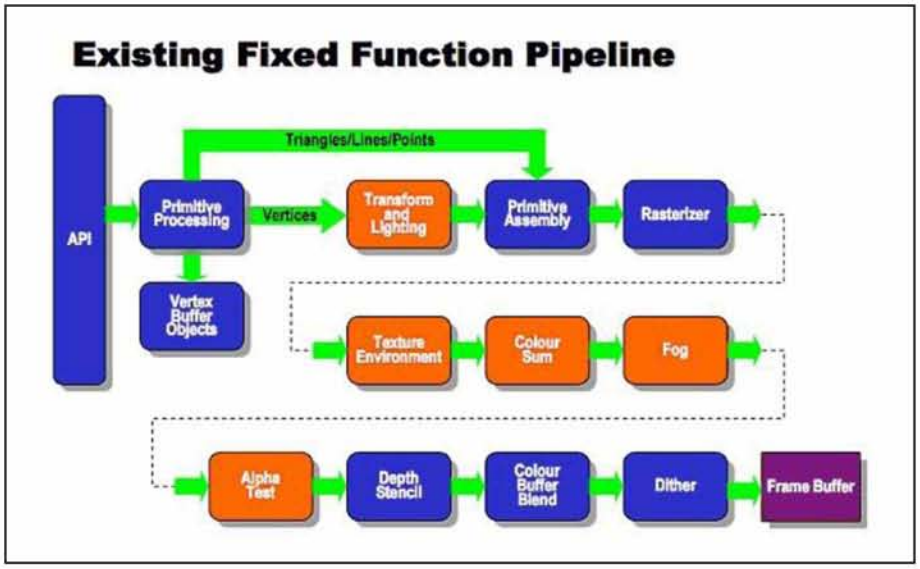
Διάφορες εκδόσεις της OpenGL ES υπάρχουν τώρα. OpenGL ES 1,0 ορίζεται σε σχέση με την OpenGL 1.3, η OpenGL ES 1,1 ορίζεται σε σχέση με την OpenGL 1.5 και η OpenGL ES 2.0 ορίζεται σε σχέση με τη OpenGL 2.0 (που σημαίνει ότι για παράδειγμα, αν ένα κομμάτι κώδικα εκτελείται με OpenGL 1.3 σε ένα desktop σύστημα, θα πρέπει να εκτελείται επίσης από συσκευές που υποστηρίζουν OpenGL ES 1,0.

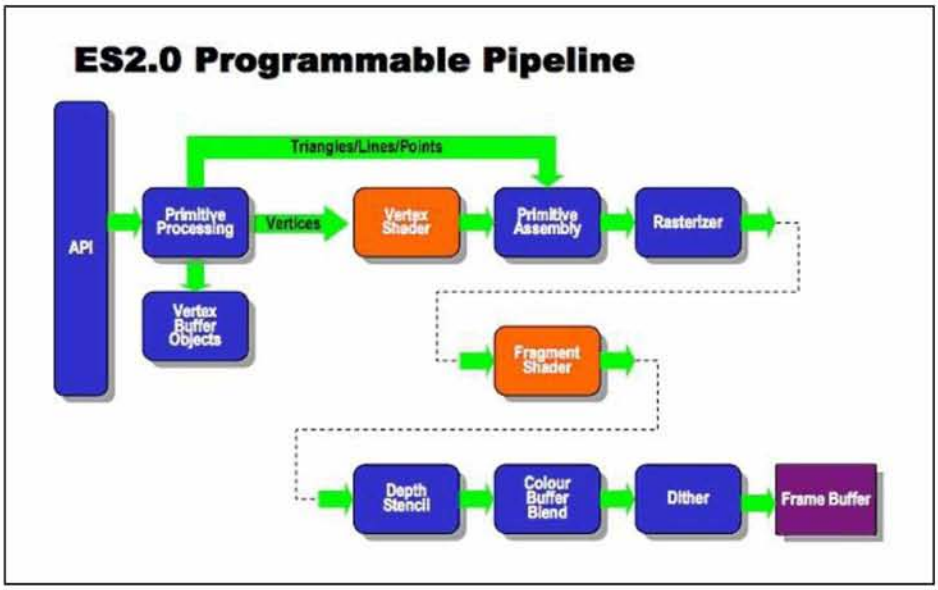
Η λειτουργικότητα προέρχεται από το αρχικό OpenGL API με κάποιες προσθέσεις. Δύο από τις πιο σημαντικές διαφορές μεταξύ των OpenGL ES και OpenGL είναι η αφαίρεση των glBegin ... glEnd (λόγω των πινάκων κορυφών) και η εισαγωγή τύπων δεδομένων fix-point για συντεταγμένες κορυφών και χαρακτηριστικών για να υποστηρίξουν καλύτερα τις υπολογιστικές ικανότητες του επεξεργαστή.

Πολλές άλλες περιοχές της λειτουργικότητας έχουν αφαιρεθεί στην έκδοση 1.0 για να παράγει ένα ελαφρύ interface.

Opengl ES 2.0 Υποστηρίζεται από το iPhone (3GS ή αργότερα), το iPod Touch (3ης γενιάς και αργότερα) και iPad. συνδυάζει μια έκδοση της OpenGL

shader language for programming vertex and fragment shaders που έχει προσαρμοστεί για ενσωματωμένες πλατφόρμες, από κοινού με ένα βελτιωμένο API από το OpenGL ES 1.1 το οποίο έχει αφαιρέσει όλα τη σταθερή λειτουργικότητα που μπορεί εύκολα να αντικατασταθεί από προγράμματα shader, με σκοπό την ελαχιστοποίηση του κόστους και της κατανάλωσης ενέργειας των προηγμένων προγραμματιζόμενων υποσυστημάτων γραφικών. [1]





Εικόνα 3-1,3-2: Programmable Pipeline of OpenGL 2.0 replaces the fixed function transformation and fragment pipeline of OpenGL 1.x. (http://www.khronos.org/opengles/2_X/)

3.3 Objective-C

Η Objective-C είναι μια ισχυρή, αντικειμενοστραφής γλώσσα με ένα δυναμικό σύστημα κλάσεων που είναι εξαιρετικά ευέλικτο και γρήγορο, σχεδιασμένο ως ένα υπερσύνολο πάνω από την τυποποιημένη γλώσσα C.

Η Objective-C υλοποιεί το cocoa και Cocoa Touch framework, παρέχοντας υψηλού επιπέδου APIs δίνοντας αυξημένη λειτουργικότητα μέσα από λίγες σαφείς, κατανοητές γραμμές κώδικα. Η Objective-C 2 περιλαμβάνει τέτοια μοντέρνα χαρακτηριστικά, όπως πολύ γρήγορη επανάληψη, άλλα και η συλλογή σκουπιδιών (garbage collector) εναρμονισμένη για Macs πολλαπλών πυρήνων τελευταίας τεχνολογίας. [19]

3.3.1 Σύνταξη

Η objective-C είναι ένα λεπτό στρώμα πάνω από C, και επιπλέον είναι ένα απόλυτο υπερσύνολο της C. Είναι εφικτό να κάνει compile σε οποιοδήποτε πρόγραμμα γραμμένο σε C με το μεταγλωττιστή της Objective-C, και να συμπεριλάβει ελεύθερα τον κωδικό C σε μια κλάση Objective-C.

Η Objective-C οφείλει το αντικειμενοστραφές συντακτικό του στο Smalltalk. Όλο το συντακτικό για μη αντικειμενοστραφείς πράξεις (περιλαμβανομένων των πρωτόγονων μεταβλητών, της προεπεξεργασίας, τις εκφράσεις, τις δηλώσεις μεθόδων, και κλήσεις συνάρτησης) είναι ίδιο με αυτό της C, ενώ το συντακτικό για αντικειμενοστραφή χαρακτηριστικά είναι μια εφαρμογή μηνυμάτων του στυλ της Smalltalk.

3.3.2 Υλοποίηση και διεπαφή

Η Objective-C απαιτεί η διεπαφή και η υλοποίηση μιας κλάσης να είναι δηλωμένα σε χωριστά μπλοκ κώδικα. Κατά συνθήκη, η διεπαφή μπαίνει σε ένα αρχείο header καθώς και την υλοποίηση σε ένα αρχείο κώδικα. Τα αρχεία header, χρησιμοποιούν ως συνήθως το επίθεμα .h, είναι παρόμοια με αρχεία header της C,

ενώ τα αρχεία υλοποίησης (μεθόδων) , συνήθως το επίθεμα .m, τα οποία μπορεί να είναι πολύ παρόμοια με τα αρχεία κώδικα C.

3.3.3 Μηνύματα

Το μοντέλο Objective-C του αντικειμενοστραφούς προγραμματισμού βασίζεται στην ανταλλαγή μηνυμάτων με αναπαραστάσεις αντικείμενων. Στην Objective-C δεν γίνονται κλήσεις μεθόδων. Γίνονται κλήσεις μηνυμάτων.

Αυτό έρχεται σε αντίθεση με το Simula-style μοντέλο προγραμματισμού που χρησιμοποιείται από C + +. Η διαφορά μεταξύ των δύο αυτών εννοιών είναι στο πώς ο κώδικας που αναφέρεται με το όνομα μεθόδου ή μηνύματος εκτελείται. Σε μια γλώσσα Simula-style, το όνομα της μεθόδου είναι στις περισσότερες περιπτώσεις συνδεδεμένο με ένα τμήμα του κώδικα στην κλάση-στόχο από τον compiler. Στο Smalltalk και στην Objective-C, ο στόχος του μηνύματος έχει επιλυθεί κατά το χρόνο εκτέλεσης, με το ίδιο το αντικείμενο που παραλήφτηκε να ερμηνεύει το μήνυμα.

Το Smalltalk -style προγραμματισμού επιτρέπει σε μηνύματα να ταξιδεύουν μη-υλοποιημένα. Για παράδειγμα, ένα μήνυμα μπορεί να αποστέλλεται σε μια συλλογή αντικειμένων, στο οποίο μόνο ορισμένα από αυτά θα αναμένεται να ανταποκριθούν, χωρίς το φόβο της παραγωγής σφαλμάτων χρόνου εκτέλεσης. Επίσης το μήνυμα που διέρχεται δεν απαιτεί ένα αντικείμενο να καθοριστεί κατά τη μεταγλώττιση.

3.4 Cocoa frameworks

Τα Cocoa frameworks αποτελούνται από βιβλιοθήκες, API, και runtimes που αποτελούν το επίπεδο ανάπτυξης για το σύνολο των Mac OS X. Χρησιμοποιώντας Cocoa, δημιουργούνται εφαρμογές με τον ίδιο τρόπο που δημιουργήθηκε το Mac OS X. Η εφαρμογή θα κληρονομήσει αυτόματα τις συμπεριφορές και τις εμφανίσεις του Mac OS X, με πλήρη πρόσβαση στις δυνατότητες του λειτουργικού συστήματος UNIX. Χρησιμοποιώντας το Cocoa με το Xcode IDE είναι απλά ο καλύτερος τρόπος για να δημιουργηθούν εφαρμογές Mac.

Το cocoa χρησιμοποιεί το πρότυπο σχεδιασμού Model-View-Controller (MVC) . Τα Μοντέλα ενσωματώνουν τα στοιχεία της εφαρμογής, οι όψεις εμφανίζουν οθόνη και επεξεργάζονται τα δεδομένα, και οι Controllers μεσολαβούν στην λογική σύνδεση μεταξύ των δύο. Με τον διαχωρισμό των ευθυνών με αυτόν τον τρόπο, θα καταλήξουμε με μια εφαρμογή που είναι ευκολότερο να σχεδιάσει, να υλοποιηθεί και να διατηρηθεί.

Το μοτίβο MVC σημαίνει ότι Interface Builder δεν απαιτεί κώδικα να γραφτεί ή να δημιουργηθεί, ενώ επικεντρώνεται αποκλειστικά στην όψη της εφαρμογής. Το Cocoa που προσανατολίζεται για το Mac εξαλείφει την χρήση κώδικα σύνδεσης, με αποτέλεσμα η σχέση μεταξύ Controllers που γράφονται σε Xcode και των όψεων που σχεδιάζονται εντός Interface Builder να αποτελεί ένα απλό θέμα της γραφικής "καλωδίωσης" και των δύο μαζί. Το Interface Builder συνεργάζεται με το Cocoa ώστε να προσαρμόσουν την εφαρμογή εύκολα έτοιμη για την είσοδο σε εντελώς νέες αγορές. [20]

Εικόνα 3-3: Model-View-Controller (MVC) (http://developer.apple.com)

3.5 Βάσεις Δεδομένων

Όπως έχουμε ήδη αναφέρει η τεχνολογία που χρησιμοποιήθηκε για την υλοποίηση των βάσεων δεδομένων ήταν η εξής:

1. SQlite από πλευράς mobile app
2. Και MySQL από την πλευρά του server

3.5.1 SQlite

Η SQlite είναι μια βιβλιοθήκη λογισμικού που υλοποιεί μια αυτόνομη, χωρίς server, δίχως διαμόρφωση, βάση δεδομένων SQL με συναλλασσόμενο μηχανισμό. Η SQlite είναι η ευρύτερα διαδεδομένη μηχανή βάσης δεδομένων SQL στον κόσμο. Λόγω του μικρού μεγέθους, η SQlite είναι κατάλληλη για τα ενσωματωμένα συστήματα, και έχει επίσης ενσωματωθεί στο iOS της Apple (όπου χρησιμοποιείται για τα SMS / MMS, Ημερολόγιο, την ιστορία κλήσεων και στις επαφές αποθήκευσης), στο λειτουργικό σύστημα Symbian OS, στο Maemo της Nokia, το Android της Google, το BlackBerry της RIM, το MeeGo Linux Ιδρύματος και webOS της Palm. Ωστόσο, είναι επίσης κατάλληλο για desktop λειτουργικά συστήματα όπως στην Apple, όπου ενέκρινε ως επιλογή στο πυρήνα δεδομένων του Mac OS X API από την αρχική υλοποίηση του Mac OS X 10.4 και μετά, καθώς και για τη διαχείριση των βίντεο και τραγούδια για το iPhone.

3.5.2 MySQL

Το MySQL είναι ένα σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων (RDBMS) που τρέχει ως server που παρέχει την πρόσβαση πολλών χρηστών σε πολλές βάσεις δεδομένων. Είναι μια δημοφιλής επιλογή για χρήστες που αναζητούν βάση δεδομένων σε δικτυακές εφαρμογές, και αποτελεί κεντρική συνιστώσα των ευρέως διαδεδομένων εφαρμογών λογισμικού web LAMP,WAMP,MAMP. Η δημοτικότητά της είναι στενά συνδεδεμένη με τη δημοτικότητα της PHP μιας γλώσσας που σχεδιάστηκε για την ανάπτυξη εφαρμογών web με σκοπό την υλοποίηση δυναμικών ιστοσελίδων.

Για τους σκοπούς της παρούσας διπλωματικής, στην πλευρά του server χρησιμοποιείται εφαρμογή λογισμικού web WAMP και php scripting language.

3.6 Core Animation

Το σύστημα συντεταγμένων για τα επίπεδα διαφέρει ανάλογα με την τρέχουσα πλατφόρμα. Στα iOS, η αρχή του συστήματος συντεταγμένων από

προεπιλογή είναι στην επάνω αριστερή γωνία και οι θετικές τιμές εκτείνονται προς τα κάτω και στα δεξιά από την αρχή. Στα Mac OSX η αρχή συντεταγμένων βρίσκεται στην κάτω αριστερή γωνία και οι θετικές τιμές εκτείνονται προς τα άνω και προς τα δεξιά από το σημείο αυτό. Όλες οι τιμές των συντεταγμένων καθορίζονται σαν αριθμοί κινητής υποδιαστολής. Και οποιαδήποτε στρώση (Layer) δημιουργείται σε μια δεδομένη πλατφόρμα χρησιμοποιεί το σύστημα συντεταγμένων που σχετίζεται με την εν λόγο πλατφόρμα. Κάθε αντικείμενο - στρώμα ορίζει και διατηρεί το δικό του σύστημα συντεταγμένων, και όλο το περιεχόμενο του επιπέδου αυτού είναι συσχετισμένο με το σύστημα αυτό. Αυτό ισχύει τόσο για τα περιεχόμενα της στρώσης όσο και για τις υποκείμενες στρώσεις. Επειδή κάθε στρώση ορίζει αυτό το δικό του σύστημα συντεταγμένων, η κλάση CALayer παρέχει τις μεθόδους για να μετατρέψει το σημείο, το ορθογώνιο και τις τιμές των μεγεθών του από το σύστημα του ενός επιπέδου στο άλλο.

```
struct CATransform3D
{
  CGFloat m11, m12, m13, m14;
  CGFloat m21, m22, m23, m24;
  CGFloat m31, m32, m33, m34;
  CGFloat m41, m42, m43, m44;
};

typedef struct CATransform3D CATransform3D;
```

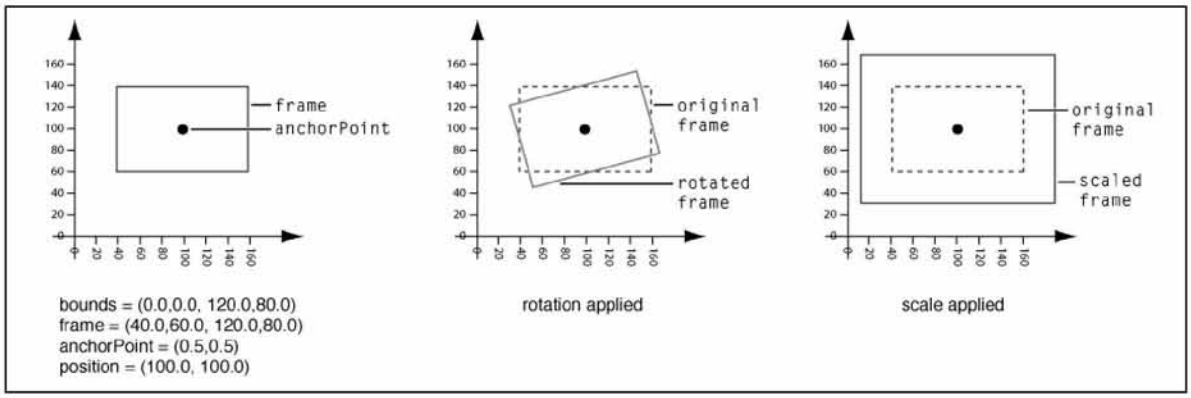
Εικόνα 3-4: Δομή CATransform3D (http://developer.apple.com)

3.6.1 Μετασχηματίζοντας την γεωμετρία ενός Επιπέδου

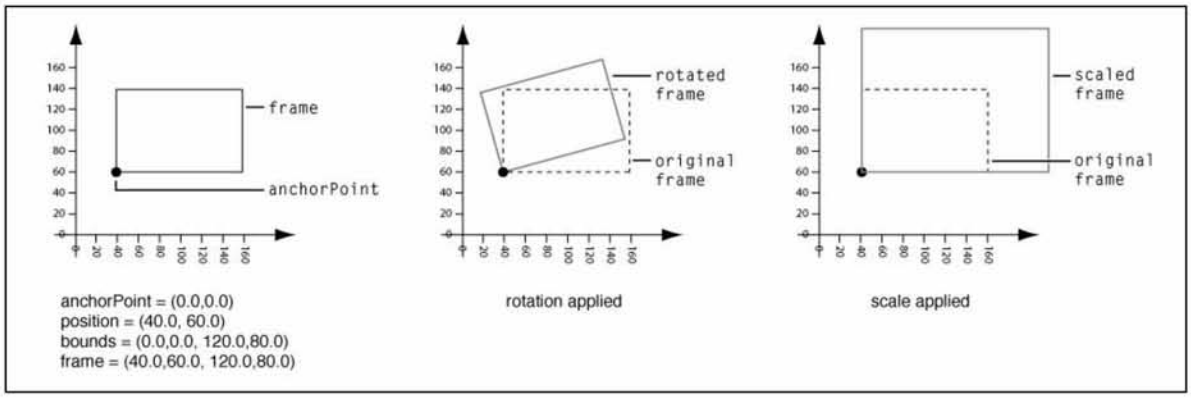
Είναι δυνατή η μετατροπή της γεωμετρίας ενός επιπέδου χρησιμοποιώντας πίνακα μετασχηματισμών.

Η CATransform3D δομή δεδομένων ορίζει ένα ομογενή τρισδιάστατο μετασχηματισμό (έναν 4 X 4 πίνακα CGFloat τιμών) ο οποίος χρησιμοποιείται για την περιστροφή, την κλιμάκωση, την μετατόπιση, την κλίση και την εφαρμογή μετασχηματισμών προοπτικής σε ένα επίπεδο.

Δυο ιδιότητες επιπέδων καθορίζουν την μετατροπή των πινάκων: ο μετασχηματισμός και ο μετασχηματισμός χαμηλότερων επιπέδων. Ο Πίνακας που ορίζεται από την ιδιότητα μετασχηματισμού εφαρμόζεται στο επίπεδο και στα χαμηλότερα επίπεδα σε σχέση με το anchor Point του επιπέδου. Στην εικόνα 3-5 φαίνεται πως η περιστροφή και η κλιμάκωση επιδρούν σε ένα επίπεδο όταν χρησιμοποιείται ένα σημείο αναφοράς (anchor Point) (0.5,0.5),η προεπιλεγμένη τιμή. Ενώ στην επόμενη εικόνα 3-6 φαίνεται πως η ίδια μετατροπή επιδρά στο επίπεδο όταν το anchor Point είναι στο 0,0. Ο Πίνακας που ορίζεται από τον μετασχηματισμό χαμηλότερων επιπέδων εφαρμόζεται μόνο σε χαμηλότερα επίπεδα του επιπέδου και όχι στο Ίδιο το επίπεδο. [21]



Εικονα 3-5



Εικόνα 3-6 (http://developer.apple.com)

Με τους ακόλουθους τρόπους μπορούν να δημιουργηθούν και να τροποποιηθούν δομές CATransform3D:

- Χρησιμοποιώντας τις συναρτήσεις CATransform3D,

- Τροποποιώντας τα μέλη των δεδομένων τις δομής άμεσα
- Χρησιμοποιώντας key-value κωδικοποίηση και key paths.

Ο CATransform3DIdentity είναι ο μοναδιαίος πίνακας, Πίνακας που δεν έχει εφαρμοστεί κανένας μετασχηματισμός. Εφαρμόζοντας τον, επαναφέρουμε το επίπεδο στην αρχική του γεωμετρία.

3.6.2 Συναρτήσεις μετασχηματισμού

Οι συναρτήσεις μετασχηματισμών που είναι διαθέσιμες από το Core Animation λειτουργούν πάνω σε πίνακες. Μπορούν να χρησιμοποιηθούν αυτές για την κατασκευή πίνακα που αργότερα θα εφαρμοστεί σε ένα επίπεδο μετατρέποντας την ιδιότητα μετασχηματισμού του. Οι συναρτήσεις μετασχηματισμών είτε ενεργούν πάνω σε, είτε επιστρέφουν, μια δομή δεδομένων CATransform3D. Δίνει έτσι την δυνατότητα κατασκευής απλών η και πολύπλοκων μετασχηματισμών που μπορούν εύκολα να επαναχρησιμοποιηθούν. [21]

Function	Use
CATransform3DMakeTranslation	Returns a transform that translates by '(tx, ty, tz)'. t' = [1 0 0 0; 0 1 0 0; 0 0 1 0; tx ty tz 1].
CATransform3DTranslate	Translate 't' by '(tx, ty, tz)' and return the result: * t' = translate(tx, ty, tz) * t.
CATransform3DMakeScale	Returns a transform that scales by `(sx, sy, sz)': * t' = [sx 0 0 0; 0 sy 0 0; 0 0 sz 0; 0 0 0 1].
CATransform3DScale	Scale 't' by '(sx, sy, sz)' and return the result: * t' = scale(sx, sy, sz) * t.
CATransform3DMakeRotation	Returns a transform that rotates by 'angle' radians about the vector '(x, y, z)'. If the vector has length zero the identity transform is returned.
CATransform3DRotate	Rotate 't' by 'angle' radians about the vector '(x, y, z)' and return the result. t' = rotation(angle, x, y, z) * t.

Εικόνα 3-7: Συναρτήσεις μετασχηματισμού (http://developer.apple.com**)**

4

Υλοποίηση

4.1 Λεπτομέρειες Υλοποίησης

Σε αυτό το κεφάλαιο θα περιγράψουμε λεπτομερώς θέματα της διπλωματικής που έχουν τεχνικό ή αλγοριθμικό ενδιαφέρον. Θα δούμε ορισμένες συναρτήσεις που υλοποιούν τυχόν πολύπλοκες αλγοριθμικές πράξεις αλλά και τις μεθόδους εκείνες που έχουν να κάνουν με την κίνηση του αντικειμένου στο άγγιγμα της οθόνης.

4.1.1 Εντοπισμός της απόστασης δυο σημείων

Η συνάρτηση distanceBetweenTouches υπολογίζει την απόσταση δυο σημείων που δημιουργούνται από το άγγιγμα της οθόνης. Αρχικοποιεί δυο μεταβλητές σημείων και βρίσκει την θέση στο x,y σύστημα συντεταγμένων, αφού ολοκληρωθούν δυο αγγίγματα γίνεται υπολογισμός της υποτείνουσας του τριγώνου που σχηματίζουν οι διάφορες των συντεταγμένων τους, δηλαδή υπολογίζεται η απόσταση τους και επιστρέφεται.

```
#pragma mark Touch handling
- (float)distanceBetweenTouches:(NSSet *)touches;
{
	int currentStage = 0;
	CGPoint point1, point2;


	for (UITouch *currentTouch in touches)
	{
		if (currentStage == 0)
		{
			point1 = [currentTouch locationInView:self];
			currentStage++;
		}
		else if (currentStage == 1)
		{
			point2 = [currentTouch locationInView:self];
			currentStage++;
		}
		else
		{
		}
	}
	return (sqrt((point1.x - point2.x) * (point1.x - point2.x) + (point1.y - point2.y) *
(point1.y - point2.y)));
}
```

4.1.2 Εντοπισμός κοινής κατεύθυνσης

Η συνάρτηση commonDirectionOfTouches ελέγχει κατά πόσο και τα δυο δάχτυλα κινούνται προς την ίδια κατεύθυνση. Για κάθε πάτημα κρατάει σε μεταβλητές την προηγούμενη θέση στον x,y και την τωρινή θέση. Υπολογίζει την κατεύθυνση στο χ και στο y για κάθε πάτημα υπολογίζοντας την διαφορά της τωρινής από την παλιά θέση, μόνο που στο y η πράξη γίνεται αντίθετα λόγω του ανεστραμμένου συστήματος συντεταγμένων του iPhone. Εάν και τα δυο δάχτυλα έχουν κατεύθυνση κίνησης ταυτόχρονα θετική ή αρνητική στον x και y τότε θεωρούμε πως κινούνται προς την ίδια κατεύθυνση υπολογίζοντας την κοινή αυτή πορεία στον x, y.

```
- (CGPoint)commonDirectionOfTouches:(NSSet *)touches;
{
	// Check to make sure that both fingers are moving in the same direction
	int currentStage = 0;
	CGPoint currentLocationOfTouch1, currentLocationOfTouch2,
previousLocationOfTouch1, previousLocationOfTouch2;
	//NSLog(@"common:");


	for (UITouch *currentTouch in touches)
	{

		if (currentStage == 0)
		{

			previousLocationOfTouch1 = [currentTouch
previousLocationInView:self];
			currentLocationOfTouch1 = [currentTouch locationInView:self];
			currentStage++;
		}
		else if (currentStage == 1)
		{

			previousLocationOfTouch2 = [currentTouch
previousLocationInView:self];
			currentLocationOfTouch2 = [currentTouch locationInView:self];
			currentStage++;
		}
		else
		{
		}
	}

	CGPoint directionOfTouch1, directionOfTouch2, commonDirection;
	// The sign of the Y touches is inverted, due to the inverted coordinate system
of the iPhone
	directionOfTouch1.x = currentLocationOfTouch1.x -
previousLocationOfTouch1.x;
	directionOfTouch1.y = previousLocationOfTouch1.y -
currentLocationOfTouch1.y;
	directionOfTouch2.x = currentLocationOfTouch2.x -
previousLocationOfTouch2.x;
	directionOfTouch2.y = previousLocationOfTouch2.y -
currentLocationOfTouch2.y;
	//NSLog(@"common  2:: %f , %f / %f , %f:", directionOfTouch1.x,
directionOfTouch1.y, directionOfTouch2.x, directionOfTouch2.y);

	// A two-finger movement should result in the direction of both touches being
positive or negative at the same time in X and Y
```

```
        if (!( ((directionOfTouch1.x <= 0) && (directionOfTouch2.x <= 0)) ||
((directionOfTouch1.x >= 0) && (directionOfTouch2.x >= 0)) ))
                return CGPointZero;
        if (!( ((directionOfTouch1.y <= 0) && (directionOfTouch2.y <= 0)) ||
((directionOfTouch1.y >= 0) && (directionOfTouch2.y >= 0)) ))
                return CGPointZero;

        // The movement ranges are averaged out
        commonDirection.x = ((directionOfTouch1.x + directionOfTouch1.x) / 2.0f) *
scalingForMovement;
        commonDirection.y = ((directionOfTouch1.y + directionOfTouch1.y) / 2.0f) *
scalingForMovement;


        return commonDirection;
}
```

4.1.3 Έναρξη χειρισμών

Η συνάρτηση touchesBegan ανταποκρίνεται στην επαφή των δάχτυλων με την οθόνη. Είναι η συνάρτηση που ουσιαστικά ελέγχει τον αριθμό των δάχτυλων που αγγίζουν την οθόνη. Ελέγχει επίσης εάν το αντικείμενο βρίσκεται σε κατάσταση κίνησης στέλνοντας σήμα τερματισμού προηγούμενης λειτουργίας και αρχικοποίησης χειρισμού νέων κινήσεων.

Σε περίπτωση που τα πατήματα είναι περισσότερα του ενός, υπολογίζεται η αρχική απόσταση τους καλώντας την συνάρτηση που παρουσιάσαμε προηγουμένως,distanceBetweenTouches,ενώ στην αντίθετη περίπτωση ελέγχεται ο αριθμός των συνεχόμενων μεμονωμένων touch

- εάν είναι single-tap κρατείται απλά η θέση σε μια μεταβλητή,
- αν είναι double-tap επαναφέρεται το αντικείμενο στην αρχική του θέση,
- και για triple-tap εμφανίζεται το ιστόγραμμα του αντικειμένου.

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
        //if(animating){
```

```
            ExampleAppDataObject* theDataObject = [self theAppDataObject];
            theDataObject.switch1=NO;

        [self performSelectorOnMainThread:@selector(showSwitch) withObject:nil
waitUntilDone:NO];


        //NSLog(@"Set togleswitch to NO");



        if (!animating)
        {
                [self startAnimation];
                Entity *e = [sceneManager.entities objectAtIndex:0];
                e.rx=0.0;
                e.ry=0.0;
                        [renderer rotation3DCube];

        }



        NSMutableSet *currentTouches = [[[event touchesForView:self] mutableCopy]
autorelease];
   [currentTouches minusSet:touches];
        //NSSet *totalTouches = [event allTouches];

        // New touches are not yet included in the current touches for the view
        NSSet *totalTouches = [touches setByAddingObjectsFromSet:[event
touchesForView:self]];
        if ([totalTouches count] > 1)
        {
                //NSLog(@"began touches");


                startingTouchDistance = [self distanceBetweenTouches:totalTouches];
                //NSLog(@"starting:%f",startingTouchDistance);

                previousScale = 1.0f;
                twoFingersAreMoving = NO;
                pinchGestureUnderway = NO;
                previousDirectionOfPanning = CGPointZero;
        }
        else
        {
                UITouch *touch = [[touches allObjects] objectAtIndex:0];


                switch ([touch tapCount])

                {
```

```
            //---single tap---

                        case 1: {

                                lastMovementPosition = [[touches anyObject]
locationInView:self];

                        } break;


            //---double tap---

                        case 2: {
                                ExampleAppDataObject* theDataObject = [self
theAppDataObject];
                                theDataObject.switch2=YES;

                                //NSLog(@"Double touch");

                        } break;
                        case 3: {
                                ExampleAppDataObject* theDataObject = [self
theAppDataObject];
                                if (theDataObject.switch3==NO) {
                                        theDataObject.switch3=YES;

                                }else {
                                        theDataObject.switch3=NO;

                                }


                                //NSLog(@"tre touch");

                        } break;
                }

                //lastMovementPosition = [[touches anyObject] locationInView:self];
        }

}
```

4.1.4 Υλοποίηση χειρισμών

Η συνάρτηση touchesMoved επαναφέρει το αντικείμενο σε κατάσταση κίνησης ανεβάζοντας το frame rate. Καλείται όταν αναγνωριστεί κάποια κίνηση των δαχτύλων πάνω στην οθόνη και από ένα πλήθος περιπτώσεων αντιδρά σε αυτές τις κινήσεις. Συγκεκριμένα στην περίπτωση κίνησης δυο δάχτυλων ελέγχεται το κατά πόσο αυτά κινούνται σε κοινή κατεύθυνση μέσω της συνάρτηση commonDirectionOfTouches που αναφέραμε προηγουμένως αλλά και σε συνδυασμό με την προηγούμενη κίνηση που πραγματοποιήθηκε. Αν ισχύουν οι περιορισμοί τότε το αντικείμενο εκτελεί μεταφορά στο χώρο, με κατεύθυνση αυτή που υπολογίστηκε χωρίς να περιστρέφεται και να αλλάζει το σχήμα του.

```
- (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event
{
	if (!animating)
	{
		[self startAnimation];


	}
	//if (isAutorotating)
		//[self startOrStopAutorotation:nil];
	Entity *e = [sceneManager.entities objectAtIndex:0];
	NSSet *allTouches = [event allTouches];


	if ([[event touchesForView:self] count] > 1) // Pinch gesture, possibly two-
finger movement
	{
		CGPoint directionOfPanning = CGPointZero;
	//	NSLog(@"two");

		// Two finger panning
		if ([allTouches count] > 1) // Check to make sure that both fingers are
moving
		{
```

```
                directionOfPanning = [self
commonDirectionOfTouches:touches];
        }
        //NSLog(@"common b:");
        //NSLog(@"panning return %f , %f:",
directionOfPanning.x,directionOfPanning.y );
```

Για την εύρεση διαφορετικής κίνησης υπολογίζεται η νέα απόσταση των δάχτυλων και ο λόγος με την παλιά απόσταση, και εάν τα δάχτυλα έχουν απομακρυνθεί περισσότερο από 10% ξεκινά το pinch gesture. Η κίνηση αυτή χρησιμοποιείται για την μεγέθυνση και την σμίκρυνση του αντικειμένου στέλνοντας την αλλαγμένη τιμή της μεταβλητής scale στον πίνακα προβολής.

Η συνάρτηση επίσης αντιλαμβάνεται την κίνηση ενός δάχτυλου που περιπλανιέται στο χώρο, υπολογίζοντας έτσι την διαφορά της νέας θέσης από την προηγούμενη και μεταβάλλοντας τον πίνακα προβολής ώστε να πραγματοποιηθεί η κίνηση περιστροφής προς την σωστή κατεύθυνση.

```
if ( (directionOfPanning.x != 0) || (directionOfPanning.y != 0) ) // Don't scale while
doing the two-finger panning
        {       //NSLog(@"common b:");

                if (pinchGestureUnderway)
                {       //NSLog(@"common c:");

                        if (sqrt(previousDirectionOfPanning.x *
previousDirectionOfPanning.x + previousDirectionOfPanning.y *
previousDirectionOfPanning.y) > 0.1 )
                        {
                                pinchGestureUnderway = NO;
```

```
                }
                previousDirectionOfPanning.x += directionOfPanning.x;
                previousDirectionOfPanning.y += directionOfPanning.y;
            }
        if (!pinchGestureUnderway)
            {                                   //NSLog(@"trans");

            //      NSLog(@"pre panning return %f , %f:",
previousDirectionOfPanning.x,previousDirectionOfPanning.y );
                //NSLog(@"common recieve %f , %f:",
directionOfPanning.x,directionOfPanning.y );

                twoFingersAreMoving = YES;
                e.rx=0.0;
                e.ry=0.0;
                e.sx=1.0;
                e.tx=directionOfPanning.x;
                e.ty=directionOfPanning.y;

                [renderer rotation3DCube];

                previousDirectionOfPanning = CGPointZero;
            }
        }
        else
        {

            float newTouchDistance = [self distanceBetweenTouches:[event
touchesForView:self]];
            //NSLog(@"distane: %f, %f, %f",newTouchDistance,
startingTouchDistance,previousScale);
            if (twoFingersAreMoving)
            {
            //      float A=newTouchDistance / startingTouchDistance;
            //      float B= A / previousScale;
            //      NSLog(@"%f",B);
                // If fingers have moved more than 10% apart, start pinch
gesture again
                if ( fabs(1 - (newTouchDistance / startingTouchDistance)
/ previousScale) > 0.3 )
                {
                    twoFingersAreMoving = NO;

                }

            }
            if (!twoFingersAreMoving)
            {
                //NSLog(@"scaling");
                // Scale using pinch gesture
                //e.sx=1.0;
                e.sx = ((newTouchDistance / startingTouchDistance) /
previousScale);
                e.x=directionOfPanning.x;
                e.y=directionOfPanning.y;
```

```
                        e.rx=0.0;
                        e.ry=0.0;
                        [renderer rotation3DCube];


                        previousScale = (newTouchDistance /
startingTouchDistance);
                        pinchGestureUnderway = YES;


                    }
                }
            }
            else // Single-touch rotation of object
            {
                CGPoint currentMovementPosition = [[touches anyObject]
locationInView:self];
                e.rx = (currentMovementPosition.x - lastMovementPosition.x);

                e.ry = (currentMovementPosition.y-lastMovementPosition.y);
                e.x=0.0;
                e.y=0.0;
                e.sx=1.0;

                [renderer rotation3DCube];

                lastMovementPosition = currentMovementPosition;
            }

}
```

4.1.5 Εντοπισμός Περιστροφής ως προς άξονα ζ

Δημιουργούμε ένα gesture recognized το οποίο ψάχνει για περιστροφές που γίνονται που σχετίζονται με κινήσεις δάχτυλων. Όταν ο χρήστης κινεί τα δάχτυλα σε αντίθετη κατεύθυνση μεταξύ τους σε κυκλική τροχιά, το αντικείμενο που βρίσκεται υπό τον έλεγχο του περιστρέφεται ως προς τον άξονα ζ.

Η κίνηση αυτή ουσιαστικά ενεργοποιεί την συνάρτηση Perform_Rotated η οποία υπολογίζει την γωνία περιστροφής και γίνονται οι απαραίτητες αλλαγές στον πίνακα προβολής για το μοντέλο που εμφανίζεται.

```
/* Create the Gesture Recognizer */
        UIRotationGestureRecognizer *Recoginizer = [[UIRotationGestureRecognizer
alloc]

        initWithTarget:self

        action:@selector(Perform_Rotated:)];

        /* Add it to the array of gesture recognizers */
        [self addGestureRecognizer:Recoginizer];
        //[self removeGestureRecognizer:Recoginizer];
        /* Decrease the retain count */
        [Recoginizer release];


- (void) Perform_Rotated:(UIRotationGestureRecognizer*)Sender{

        /* Get the rotation angle in degrees */
        float RotationinDegrees = Sender.rotation * (180/M_PI);

        /* Print it to the debug console */
        //NSLog(@"rotates:%f", RotationinDegrees);

        //CGPoint location = [Sender locationInView:self];
        //printf("Rotation detected @ %f %f, %f deg\n", location.x, location.y, [Sender
rotation]);
        Entity *e = [sceneManager.entities objectAtIndex:0];
        //e.rx=RotationinDegrees/10;
        e.rz=0.0f;
        //e.ry=RotationinDegrees;
        e.rx=RotationinDegrees/10;
        }
```

4.1.6 Τερματισμός χειρισμών

Όταν η εφαρμογή αντιληφθεί ότι κανένα δάχτυλο δεν ακουμπά την οθόνη καλεί την συνάρτηση touchesEnded, η οποία αναθέτει τιμές σε διάφορες μεταβλητές αρχικοποιώντας έτσι το σύστημα χειρισμών.

```
- (void)touchesEnded:(NSSet*)touches withEvent:(UIEvent*)event
{
        NSMutableSet *remainingTouches = [[[event touchesForView:self]
mutableCopy] autorelease];
   [remainingTouches minusSet:touches];
        if ([remainingTouches count] < 2)
        {
                twoFingersAreMoving = NO;
                pinchGestureUnderway = NO;
                previousDirectionOfPanning = CGPointZero;

                lastMovementPosition = [[remainingTouches anyObject]
locationInView:self];
        }

        if (animating)
        {
                //Entity *e = [sceneManager.entities objectAtIndex:0];
                //e.rx=0.0;
                //e.ry=0.0;
                //e.sx=1.0;
                Entity *e = [sceneManager.entities objectAtIndex:0];
                e.rz=1.0f;

                [self stopAnimation];

        }

}
```

4.1.7 Πίνακας προβολής

Η δομή CATransform3D που παρέχεται από το Core Animation για τον χειρισμό των CALayers εμφανίζει την ίδια δομή με τον πίνακα προβολής του μοντέλου στην OpenGL ES. Στην πραγματικότητα η apple παρέχει μια σειρά από λειτουργίες για τον χειρισμό της δομής CATransform3D με τρόπους που αντιστοιχούν στις κλήσεις των συναρτήσεων glTranslate,glRotate,glScale.

```
- (void)rotation3DCube
{       glPushMatrix();
        // position & orientation
        //translate to the center of the 3d object
        glTranslatef(midx, midy, midz);
        glMatrixMode(GL_MODELVIEW);

        if (isFirstDrawingOfModel)
        {
        glLoadIdentity();
                isFirstDrawingOfModel = NO;
        }
        currentMatrix = CATransform3DScale(currentMatrix, sx, sx, sx);
```

Αυτό λύνει το μεγαλύτερο πρόβλημα για την αναπαράσταση των μοντέλων, δεν χρειάζεται πλέον για να γίνει μια περιστροφή ή μια μεταφορά, να γνωρίζουμε συνεχώς την κατάσταση του πίνακα προβολής του μοντέλου. Δίνει την δυνατότητα παρακολούθησης της κατάστασης του πίνακα σε ένα CATransform3D, εκτέλεσης

χειρισμών στην δομή, και απλή αντικατάσταση του πίνακα προβολής της OpenGL με αυτή τη δομή κάθε φορά που υφίσταται μια αλλαγή.

Αφού μεταφέρω το κέντρο του συστήματος στο κέντρο του μοντέλου που πρόκειται να εμφανίσω κάνω το πρώτο βήμα για την αναπαράσταση εκτελώντας scale στο αντικείμενο με βάση την κίνηση pinch. Παρατηρούμε πως δεν χρησιμοποιήθηκαν OpenGL εντολές μα η εντολή CATransform3DScale.

Στην συνέχεια θα εκτελεστούν οι εντολές για τον υπολογισμό της περιστροφής. Υπάρχουν δυο διαφορετικές περιπτώσεις περιστροφής, αυτή που γίνεται με την κίνηση των δάχτυλων σε αντίθετη κατεύθυνση και μια δεύτερη που γίνεται με την κίνηση ενός δάχτυλου στην οθόνη.

Για την πρώτη η περιστροφή γίνεται ως προς άξονα z και υπολογισμός της είναι αποτέλεσμα του πολλαπλασιασμού του πίνακα προβολής σε συγκεκριμένες θέσεις με τον παράγοντα περιστροφής. Παρόμοια υπολογίζεται και η περιστροφή της δεύτερης κίνησης με την διαφορά ότι οι άξονες περιστροφής είναι ταυτόχρονα x,y και ο Πίνακας προβολής πολλαπλασιάζεται και με τους δυο παράγοντες σε αντίστοιχες θέσεις.

```
CATransform3D tempMatrix;
GLfloat totalRotation = sqrt(rx*rx + ry*ry);
if (rz==0.0f) {
        rz=1.0f;
        //NSLog(@"ROTATION!!!!");

tempMatrix = CATransform3DRotate(currentMatrix, -totalRotation * M_PI / 180.0,
                                 (rx/totalRotation) *
currentMatrix.m13,
                                 (rx/totalRotation) *
currentMatrix.m23,
                                 (rx/totalRotation) *
currentMatrix.m33);
```

```
            if ((tempMatrix.m11 >= -100.0) && (tempMatrix.m11 <= 100.0))

                currentMatrix = tempMatrix;

        }

        else{

        tempMatrix = CATransform3DRotate(currentMatrix, totalRotation * M_PI /
180.0,

                                         ((rx/totalRotation) *
                                         currentMatrix.m12 + (ry/totalRotation) *
                                         currentMatrix.m11),

                                         ((rx/totalRotation) *
                                         currentMatrix.m22 + (ry/totalRotation) *
                                         currentMatrix.m21),

                                         ((rx/totalRotation) *
                                         currentMatrix.m32 + (ry/totalRotation) *
                                         currentMatrix.m31));


            if ((tempMatrix.m11 >= -100.0) && (tempMatrix.m11 <= 100.0))

                currentMatrix = tempMatrix;

        }
```

Τελευταία ενέργεια που γίνεται είναι η μεταφορά στο χώρο, πολλαπλασιάζοντας και πάλι τον πίνακα σε συγκεκριμένες θέσεις με τον παράγοντα της χ και y μετατόπισης.

```
        float currentScaleFactor = sqrt(pow(currentMatrix.m11, 2.0f) +
pow(currentMatrix.m12, 2.0f) + pow(currentMatrix.m13, 2.0f));


        //NSLog(@"scalefactor:%f",currentScaleFactor);

        float tt=(currentScaleFactor * currentScaleFactor);


        tx = tx / (tt*30000);

        ty = ty / (tt*30000);
```

```
//NSLog(@"final: x:%f, y:%f",tx,ty);


tempMatrix = CATransform3DTranslate(currentMatrix, tx * currentMatrix.m11,
tx * currentMatrix.m21, tx * currentMatrix.m31);

tempMatrix = CATransform3DTranslate(tempMatrix, ty *
currentMatrix.m12, ty * currentMatrix.m22, ty * currentMatrix.m32);


if ((tempMatrix.m11 >= -100.0) && (tempMatrix.m11 <= 100.0))

currentMatrix = tempMatrix;

GLfloat ModelVMatrix[16];


// Finally, set the new matrix that has been calculated from the Core
Animation transform

[self convert3DTransform:&currentMatrix toMatrix: ModelVMatrix];


glLoadMatrixf(ModelVMatrix);

//[self printMatrix: ModelVMatrix];


//retranslate to 0.0.0

glTranslatef(-midx, -midy, -midz);
```

Αμέσως μετά την εντολή αυτή θα πρέπει να ενημερώσουμε τον πίνακα προβολής του μοντέλου για τις αλλαγές που έχουν επέλθει στη δομή CATransform3D.

4.2 Πλατφόρμες και Προγραμματιστικά Εργαλεία

4.2.1 Developer tools Xcode

4.2.1.1 Xcode IDE

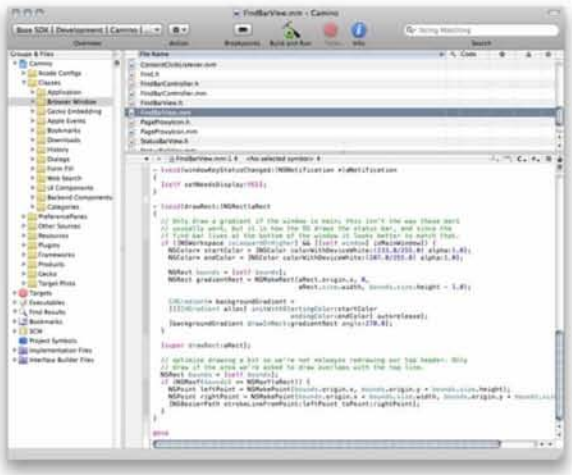
Το Xcode είναι μια σουίτα εργαλείων για την ανάπτυξη λογισμικού για Mac OS X, που αναπτύχθηκε από την Apple. Το Xcode 3.2, η τελευταία κύρια έκδοση, είναι συνδυασμένη με το Mac OS X v10.6, αλλά δεν εγκαθίσταται από προεπιλογή. Αντίθετα, πρέπει να εγκατασταθεί από το Mac OS X DVD, ή να κατεβάσετε δωρεάν από την ιστοσελίδα της Apple. Η έκδοση που είναι διαθέσιμες σήμερα είναι 3.2.5 που έρχεται με το IOS SDK 4.2.

Η κύρια εφαρμογή της σουίτας είναι το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE), που ονομάζεται επίσης και Xcode. Η σουίτα Xcode περιλαμβάνει επίσης τα περισσότερα από τα βοηθητικά έγγραφα για ανάπτυξη εφαρμογών της Apple, και το Interface Builder, μια εφαρμογή που χρησιμοποιείται για την κατασκευή γραφικών διεπαφών.

Η σουίτα Xcode περιλαμβάνει μια τροποποιημένη έκδοση του ελεύθερου λογισμικού GNU Compiler Collection (GCC, apple-darwin9-gcc-4.2.1, καθώς και apple-darwin9-gcc-4.0.1, με το πρώτο να είναι η προεπιλογή) και υποστηρίζει C, C + +, Objective-C, Objective-C + +, Java, AppleScript, Python και Ruby πηγαίου κώδικα με μια ποικιλία μοντέλων προγραμματισμού, συμπεριλαμβανομένων, αλλά δεν περιορίζονται σε cocoa, σε carbon και Java.

Ενσωματώνει την επεξεργασία του πηγαίου κώδικα, δίνοντας την δυνατότητα build και compile, μέσα σε μια γραφική εμπειρία εντοπισμού σφαλμάτων, όλα χωρίς την έξοδο από την προβολή του πηγαίου κώδικα.

Το IDE κάνει περισσότερα από αυτά τα παραδοσιακά καθήκοντα, ωστόσο με την έλευση του IOS SDK, το Xcode μπορεί να διαχειριστεί πλέον όλες τις συσκευές εκτελώντας έλεγχους, το αυτόματο πακετάρισμα εφαρμογών για iPhone με την κατάλληλη πιστοποίηση, καθώς και εγκατάσταση εφαρμογών στο iPhone. Ο απομακρυσμένος εντοπισμός σφαλμάτων συνδέεται με τη συσκευή σε πραγματικό χρόνο, γίνεται η διαχείριση των breakpoints καθώς η εφαρμογή ελέγχεται από τη συσκευή. [6]



4.2.1.2 Interface builder

Το Interface Builder είναι ένας εύκολος στη χρήση γραφικός συντάκτης για το σχεδιασμό κάθε πτυχής της IOS εφαρμογής. Το Interface Builder αποθηκεύει την σχεδίαση διεπαφών σε ένα ή περισσότερα αρχεία, ως σύνολο διεπαφής αντικείμενων και τις σχέσεις τους. Οι αλλαγές που γίνονται στο περιβάλλον συγχρονίζονται αυτόματα με το Xcode.

4.2.1.3 iPhone simulator

Το iPhone Simulator τρέχει την εφαρμογή με τον ίδιο τρόπο όπως και μια πραγματική συσκευή iPhone. Επειδή είναι γρήγορο στην εκκίνηση αλλά και στις διορθώσεις(debug), ο προσομοιωτής αποτελεί ιδανικό πεδίο δοκιμών για να βεβαιωθεί ότι η διεπαφή χρήστη λειτουργεί με ορθό τρόπο, οι κλήσεις δικτύου είναι σωστές, και οι όψεις αλλάζουν σωστά, όταν το τηλέφωνο περιστρέφεται. Μπορεί να προσομοιωθούν ακόμη και χειρονομίες επαφής με το ποντίκι.

4.2.2 Διαδικασία Εγκατάστασης

Παραθέτω την διαδικασία εγκατάστασης της εφαρμογής σε περιβάλλον Mac OS. Θα χρειαστούμε την σουίτα εργαλείων Xcode για μπορέσουμε να κάνουμε build την εφαρμογή. Αν δεν προκύψουν σφάλματα μας δίνεται η δυνατότητα να την τρέξουμε είτε σε περιβάλλον εξομοίωσης είτε στην ίδια την συσκευή. Να σημειώσουμε ότι για να τρέξει η εφαρμογή στην συσκευή πρέπει κάποιος να είναι μέλος του προγράμματος developer iOS. [22]

Τα βήματα που ακολουθεί κάποιος ούτως ώστε να τρέξει την εφαρμογή είναι τα εξής:

- Να ορίσει το περιβάλλον που θα γίνει build
- Να ορίσει το περιβάλλον τρεξίματος
- Να ορίσει την τοποθεσίας εκτέλεσης της εφαρμογής(συσκευή η εξομοιωτής)
- Να κάνει build την εφαρμογή
- Να τρέξει την εφαρμογή

5

Επίλογος

Στο κεφάλαιο αυτό θα γίνει η σύνοψη των αποτελεσμάτων της διπλωματικής εργασίας, όπως επίσης θα δοθούν κάποιες ιδέες για μελλοντική επέκταση της.

5.1 Σύνοψη και συμπεράσματα

Αναπτύξαμε μια εφαρμογή που αλληλεπιδρά με τον χρήστη, για τον χειρισμό 3d αντικειμένων με multi-touch κινήσεις. Αναζητήσαμε τις κινήσεις αυτές που θα χρησιμοποιούν την ψηφιακή φυσική και θα αξιοποιούν με τον καλύτερο τρόπο το λογισμικό που είχαμε στην διάθεση μας. Η μελέτη του Core Animation της Apple σε συνδυασμό με τους χειρισμούς πάνω σε multi-touch οθόνες μας οδήγησε σε μια τεχνική βελτίωσης του OpenGL ES τρόπου αναπαράστασης των αντικειμένων.

Καταφέραμε να αποδώσουμε με όσο το δυνατόν με μεγαλύτερη ακρίβεια αντικείμενα 3-διαστασεων, μα δυστυχώς δεν κατέστη δυνατή η αποτύπωση υφής σε αυτά παρότι έγινε αρκετή προσπάθεια σε αυτό το κομμάτι.
Δεν μας δόθηκε η δυνατότητα να ενσωματώσουμε στην εφαρμογή περισσότερες κινήσεις στο χειρισμό των αντικειμένων, καθώς η πλατφόρμα εξομοίωσης της συσκευής παρείχε περιορισμένο αριθμό κινήσεων. Η εφαρμογή έχει κατατεθεί στο apple store με τίτλο ModelVision3D και στην ιστοσελίδα υποστήριξης της εφαρμογής [18] παρέχονται αρκετές πληροφορίες για διάφορα κομμάτια της υλοποίησης όσο και τα τελευταία νέα για τις ενημερώσεις που πρόκειται να συμβούν.

Ως επέκταση της εργασίας αυτής, έχουμε σκοπό την αναζήτηση νέων κινήσεων, που θα εμπλουτίσουν τις προϋπάρχουσες multi-touch κινήσεις, και όχι μόνο θα χειρίζονται το αντικείμενο στις 3 διαστάσεις αλλά και θα αναδιαμορφώνουν την επιφάνεια του συμπιέζοντας, διασπώντας και τεντώνοντας αυτό. Μια ενδιαφέρουσα ιδέα είναι η χρήση του accelerometer στον εντοπισμό της δύναμης

της πίεσης που ασκείται στην επιφάνεια της οθόνης. Μετρώντας με το ενσωματωμένο accelerometer τις δονήσεις που δημιουργούνται σε κάθε πάτημα και εντοπίζοντας τις θέσεις των σημείων επαφής θα προσπαθήσουμε να δημιουργήσουμε την αίσθηση της πίεσης στα σημεία αυτά με σκοπό την μεταβολή της κατάστασης του σχήματος. Επιπλέον αναζητώντας νέες κινήσεις θα μπορούσαμε να χρησιμοποιούσαμε τις δυνατότητες που προσφέρει το ενσωματωμένο γυροσκόπιο, το οποίο ανιχνεύει την περιστροφή και ενημερώνει την συσκευή για τις τυχόν αλλαγές στο περασμα του χρόνου. Εφαρμόζωντας το πανω σε 3Δ αντικείμενα θα μπορούσαμε να δώσουμε την αίσθηση της μετακίνησης του παρατηρητή γύρω απο το αντικείμενο.

Επίσης επιθυμούμε την ενσωμάτωση περισσότερων τύπων 3d αντικειμένων ώστε να είμαστε σε θέση να χειρίστουμε οποιοδήποτε μοντέλο κατασκευασμένο απο οποιοδήποτε λογισμικό σχεδιασμού μοντέλων. Στην συνέχεια θεωρούμε αρκετά σημαντική την διασύνδεση της εφαρμογής με μεγάλες «αποθήκες» 3d αντικειμένων. Μια τέτοια «αποθήκη» είναι η Aim@Shape, ένα κοινό repository με συλλογή ψηφιακών σχημάτων. Αποτελεί ενα αναπόσπαστο κομμάτι του e-Science framework των εργαλείων και των υπηρεσιών για την μοντελοποίηση, την επεξεργασία και την ερμηνεία ψηφιακών σχημάτων, που αναπτύχθηκαν στο πλαίσιο του AIM@SHAPE project. Οι βασικοί τύποι σχημάτων που χρησιμοποιούνται απο τους σχεδιαστές είναι OFF,OBJ,VRML,PLY, τύποι που η ενσωμάτωση τους στην εφαρμογή προβλέπεται να μην παρουσιάσει σημαντικά προβλήματα. Μια μεγάλη κοινότητα αποτελούμενη απο χιλιάδες χρήστες που αλληλεπιδρούν καθημερινά ανεβάζοντας και κατεβάζοντας 3Δ μοντέλα αξιολογώντας παράλληλα τις δυνατότητες τους, θα δώσει στην εφαρμογή μας σημαντική ώθηση καταστώντας την ως ενα βασικό εργαλείο σχεδιαστών 3Δ αντικειμένων.

6

Αναφορές

1. http://en.wikipedia.org/wiki/OpenGL
2. http://www.opengl.org/
3. http://www.iphonedevsdk.com/forum/iphone-sdk-game-development/
4. http://iphonedevelopment.blogspot.com/
5. http://www.gamedev.net
6. http://en.wikipedia.org/wiki/Xcode
7. http://developer.apple.com/technologies/tools/xcode.html
8. http://www.engadget.com/
9. http://www.appleinsider.com/articles/08/02/21/mega_apple_filing_details_next_gen_multi_touch_input_surface.html
10. http://www.iphonebuzz.com/apple-files-patent-for-multi-touch-plus-054630.php
11. http://en.wikipedia.org/wiki/Multi-touch_gestures
12. http://en.wikipedia.org/wiki/IPhone_4#Display
13. http://en.wikipedia.org/wiki/IPod_Touch
14. http://en.wikipedia.org/wiki/Touchpad
15. http://en.wikipedia.org/wiki/IPad
16. http://www.computerworld.com/s/article/9138644/How_the_iPhone_works
17. http://en.wikipedia.org/wiki/Multi-touch
18. http://itouch3project.wordpress.com/
19. http://en.wikipedia.org/wiki/Cocoa_Touch
20. http://developer.apple.com/technologies/mac/cocoa.html
21. http://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/CoreAnimation_guide/Articles/Layers.html
22. http://developer.apple.com/library/ios/#documentation/Xcode/Conceptual/iphone_development/120-Building_and_Running_Applications/building_and_running_applications.html#//apple_ref/doc/uid/TP40007959-CH6-SW2
23. http://www.stanford.edu/class/cs193p/cgi-bin/drupal/
24. http://www.mobileorchard.com/getting-started-with-opengl-es-20-on-the-iphone-3gs/
25. http://www.computerworld.com/
26. http://www.readwriteweb.com/