Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Πτυχιακή Εργασία

Ανάπτυξη εφαρμογής με την χρήση του Asus Xtion Pro

Καραντινός Νικόλαος(ΑΜ:906)

Επιβλέπων καθηγητής: Τριανταφυλλίδης Γεώργιος

Επιτροπή Αξιολόγισης:

Ημερομηνία Παρουσίασης:

Ευχαριστίες

Ευχαριστώ τους γονείς μου και την οικογένεια μου που με στήριξαν σε αυτη την προσπάθεια μου. Επίσης τον κ. Τριανταφυλλίδη για την βοήθεια του και την υπομονή του.

Abstract

In this work we represent an interface of controlling and browsing webpages with the use of multimodal input device Asus Xtion Pro. With hand gestures we can browse the website we want simulating the movement of the mouse cursor and the selection of links. This can be achieved with the implementation of javascript code in the web page. For this purpose we will use the ZDK(Zigfu Development kit) to have access in depth and skeleton data from Asus Xtion Pro.

Σύνοψη

Σε αυτή την εργασία παρουσιάζεται μια διεπαφή χειρισμού και περιήγησης ιστοσελίδων με με βάση την συσκευή πολυτροπικής εισόδου Asus Xtion Pro. Με χειρονομίες του χεριού μπορούμε να περιηγηθούμε στην ιστοσελίδα που θέλουμε προσομοιόνωντας την κίνηση του δείκτη του ποντικιού και την επιλογή συνδέσμων. Αυτό επιτυγχάνεται με την ενσωμάτωση κώδικα javascript στην ιστοσελίδα. Για αυτό τον σκοπό θα χρησημοποιήσουμε το ZDK(Zigfu Development kit) για να έχουμε πρόσβαση στα δεδομένα βάθους και σκελετού απο το Asus Xtion Pro.

Περιεχόμενα

Εικόνες

Πίνακες και κώδικας

1. Εισαγωγή

Οσο υπαρχει το web και οι browser, ο χειρισμός και η πλοήγηση των ιστοσελίδων είναι συνηφασμένη με την χρήση πληκτρολογίου και ποντικιού. Αυτό μπορεί να προσίδει μεγάλη ακρίβεια στις κινήσεις μας, αλλά απέχει πολύ απο το να θεωρηθεί φυσική δεπαφή χρήστη.

Για να θεωρηθεί μια διεπαφή φυσική πρέπει να χρησημοποιήσουμε διάφορα μέσα έκφρασης όπως χειρονομίες, λόγο και στάση του σώματος. Με αυτό τον τρόπο πετυχαίνουμε να χρησημοποιηθούν ποιό φυσικά μέσα αλληλεπίδρασης ανθρώπου υπολογιστή.

Μερικά απο τα παραπάνω μπορούν να επιτευχθούν με την χρήση web κάμερας και μικροφώνου αλλά για να επιτύχουμε την ακρίβεια που θέλουμε στον χειρισμό των ιστοσελίδων θέλουμε ποιο εξειδικευμένο εξοπλισμό, ο οποίος δεν είναι εύκολα προσβάσιμος απο τον μέσο χρήστη.

Αυτό όμως άλλαξε το 2010 με την κυκλοφορία τον Νοέμβριο του MS Kinect ως περιφεριακό για το Xbox 360. Ακόμα το δεύτερο εξάμηνο του 2011 κυκλοφόρησαν το Asus Xtion Pro και το Xtion Pro Live. Αυτές οι συσκευές εκτός απο τον αισθητήρα βάθους έχουν και RGB κάμερα και μικρόφωνο. Με αυτό τον τρόπο οι χρήστες μπορουν να αποκτήσουν προσιτές συσκευές πολυτροπικής εισόδου.

Κίνητρο για την διεξαγωγή της εργασίας

Σκοπός αυτής της εργασίας είναι η προσαρμογή των ιστοσελίδων με χρήση javascript έτσι ώστε να μπορούν να χρησημοποιήσουν το Asus Xtion Pro σαν συσκευή εισόδου στον περιηγητή. Με αυτό τον τρόπο πετυχένουμε μια ποιό φυσιολογική εμπειρία για τον χρήστη, η οποία έχει μικρή καμπύλη εκμάθησης για τον αρχάριο χρήστη.

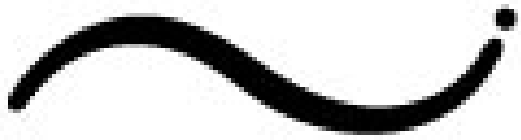
Δομή εργασίας

Στο επόμενο κεφάλαιο παρουσιάζονται γενικά τα NUI, η ιστορία, η εξέλιξη τους και μερικά παραδείγματα που πληρούν τις προϋποθέσεις για να χαρακτηριστούν NUI. Στο κεφάλαιο 3 βλέπουμε τις συσκευές πολυ-τροπικής εισόδου που θα χρησημοποιήσουμε και την σύγκριση τους. Στο κεφάλαιο 4 ρίχνουμε μια ματιά στην ιστορία και τα προϊόντα της Asus, κατασκευάστριας εταιρίας του Asus Xtion Pro. Στο κεφάλαιο 5 βλέπουμε μια σύντομη παρουσίαση εφαρμογών που είναι πέρα απο τον αρχικό σκοπό των συσκευών. Στο κεφάλαιο 6 βλέπουμε μιά ποιό αναλυτική παρουσίαση του API που χρησημοποιείται για κατασκευή εφαρμογών του OpenNI και NITE. Στο κεφάλαιο 7 βλέπουμε την ανάπτυξη της εφαρμογής μας μέσα απο το ZDK.

2. Natural User interface

Στους υπολογιστές, μια φυσική διεπαφή χρήστη, ή NUI, ή φυσική διεπαφή όπως είναι ο κοινός όρος που χρησημοποιείται απο σχεδιαστές και προγραμματιστές διεπαφών ανθρώπου μηχανής σε αναφορά σε μια διεπαφή χρήστη η οποία είναι (1) αποτελεσματικά αόρατη, ή γίνεται αόρατη με διαδοχικές αλληλεπιδράσεις, στους χρήστες της, και (2) είναι βασισμένο στην φύση ή σε φυσικά στοιχεία (π.χ. φυσική, γνωστή επίσης σαν Φυσική Φιλοσοφία). Η λέξη φυσικός χρησιμοποιείται γιατί οι περισσότερες διεπαφές υπολογιστή χρησημοποιούν τεχνητές συσκευές ελέγχου η λειτουργία των οποίων πρέπει να μαθευτεί. Ενα NUI βασίζεται στο ότι ο χρήστης να μπορεί να μεταβεί γρήγορα απο αρχάριο σε έμπειρο. Όσο η διεπαφή χρειάζεται μάθηση, αυτή η μάθηση διευκολύνεται μέσω σχεδιασμού, ο οποίος δίνει την αίσθηση στον χρήστη ότι είναι στιγμίαια και συνεχώς επιτυχειμένη. Έτσι, "φυσική" αναφαίρεται σε ένα στόχο στην εμπειρία του χρήστη – ότι η αλληλεπίδραση έρχεται φυσικά, όσο αλληλεπιδρά με την τεχνολογία, και η διεπαφή καθεαυτή ειναι φυσική. Διάφορες σχεδιαστικές στρατηγικές έχουν προταθεί οι οποίες έχουν επιτύχει τον στόχο σε διάφορα επίπεδα επιτυχίας. Μια στρατηγική ειναι η χρήση ενός "reality user interface" ("RUI"), επίσης γνωστού σαν "reality-based interfaces" (RBI) μεθόδοι.

Εικόνα 1



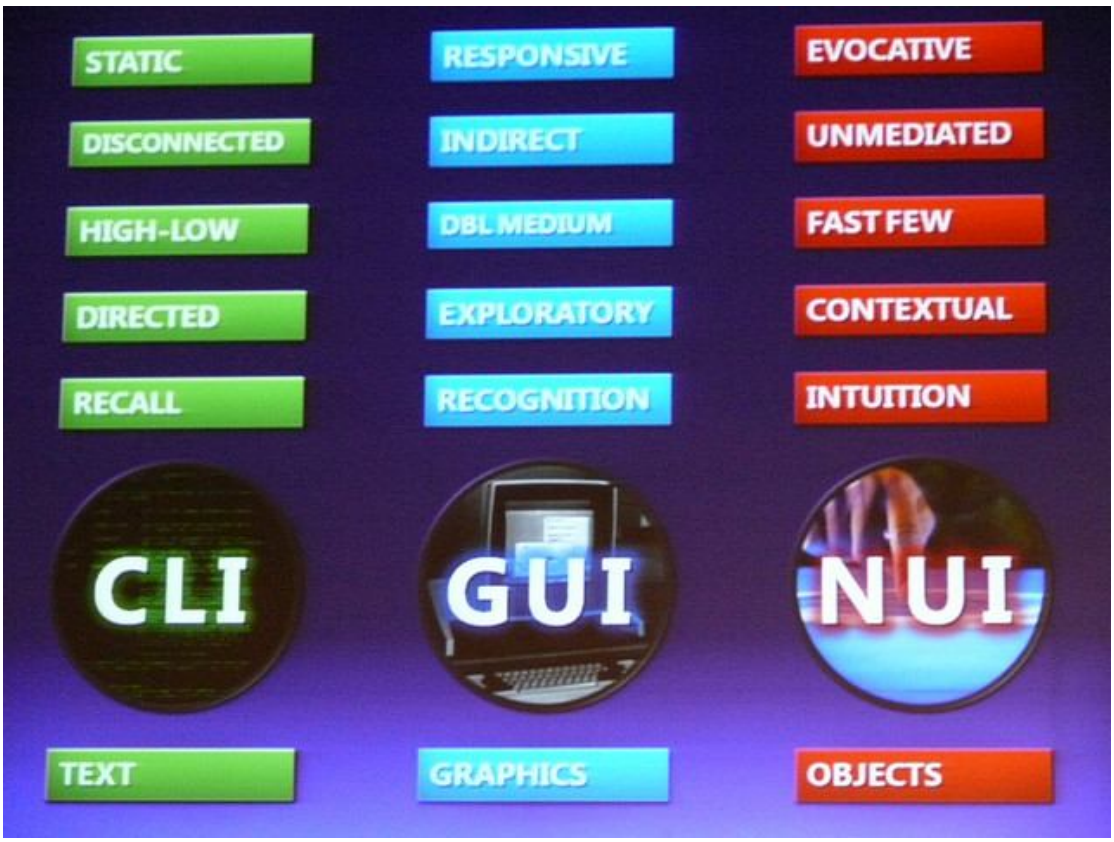
Natural User Interface

Ένα παράδειγμα απο μία RUI στρατηγική να χρησημοποιήσεις ένα φορετό υπολογιστή για να καταστήσεις αντικείμενα του πραγματικού κόσμου "επιλέξιμα", έτσι ώστε ο χρήστης να μπορεί να κάνει κλικ σε κάθε καθημερινό αντικείμενο έτσι ώστε να το κάνει να λειτουργεί σαν υπερσύνδεση, συγχώνεύοντας έτσι τον κυβερνοχώρο και τον πραγματικό κόσμο.

Ένα παράδειγμα μιάς στρατηγικής για τον σχεδιασμό ενός NUI που δεν βασίζεται σε RBI ο αυστηρός περιορισμός της λειτουργικότητας και προσαρμογής, έτσι ώστε οι χρήστες να έχουν πολύ λίγα να μάθουν για τον χειρισμό μιας συσκευής. Δεδομένου οτι οι βασικές δυνατότητες ταιριάζουν με τους στόχους του χρήστη , η διεπαφή είναι εύκολη στην χρήση. Αυτή είνα μια πρωταρχική σχεδιαστική στρατηγική στο iOS της Apple. Επειδή αυτός ο σχεδιασμός συμπίπτει με μια οθόνη αφής, οι μη-σχεδιαστές συχνά αποδίδουν λάθος την αβίαστη αλληλεπίδραση με την συσκευή στην οθόνη πολλαπλης αφής, και όχι στον σχεδιασμό του λογισμικύ όπου πραγματικά βασίζεται.

Ιστορια

Εικόνα 2



Εξέλιξη των διεπαφών χρήστη

Στα έτη 1970, 80 και 90 ο Steve Mann ανέπτυξε ένα αριθμό απο στρατηγικές για διεπαφές χρήστη που χρησημοποιούν την φυσική αλληλεπίδραση με τον πραγματικό κόσμο σαν μια ενναλακτική στην διεπαφή γραμμής εντολών (command-line interface (CLI)) ή γραφική διεπαφή χρήστη (graphical user interface (GUI)).Ο Mann αναφερόταν σε αυτήν την δουλειά σαν "Natural User Interfaces" (Φυσικές Διεπαφές Χρήστη), "Direct User Interfaces" (Άμεσες Διεπαφές Χρήστη, και "Metaphor-Free Computing". Η τεχνολογία EyeTap του Mann τυπικά ενσωματώνει ένα παράδειγμα μιας φυσικής διεπαφής χρήστη. Η χρήση της λέξης "Natural" (φυσικός) απο τον Mann αναφέρεται τόσο στην δράση η οποία έρχεται φυσικά στους ανθρώπινους χρήστες, όσο και στην χρήση της φύσης καθεαυτής, και του φυσικού περιβάλλοντος. Ένα καλό παράδειγμα ενός NUI και στις δυο αυτές περιπτώσεις είναι το υδραυλόφωνο, ιδιαίτερα αν χρησημοποιείται σαν συσκευή εισαγωγής, στην οποία αγγίζοντας ένα φυσικό στοιχείο (νερό) μετατρέπεται σε ένα τρόπο εισαγωγής δεδομένων. Ποιό γενικευμένα, μια κατηγορία μουσικών οργάνων τα οποία λέγονται "φυσίφωνα", βασισμένα στις ελληνικές λέξεις "φυσικά", "φυσικός" και "φώνος" (ήχος) έχουν επίσης προταθεί σαν "διεπαφές χρήστη με βάση την φύση".

Το 2006 ο Christian Moore ίδρυσε μια ανοιχτή ερευνητική κοινότητα με σκοπό να επεκτείνει την συζήτηση και την ανάπτυξη σχετικά με τις τεχνολογίες NUI . Το 2008 σε μια παρουσίαση του συνεδρίου "Προβλέποντας το Παρελθόν," ο August de los Reyes, ένας γενικός διευθυντής εμπειρίας χρήστη στο Surface Computing στη Microsoft περίγραψε το

NUI σαν την επόμενη εξελικτική φάση ακολουθόντας την αλλαγή απο το CLI στο GUI. Φυσικά, και αυτό ήταν μια υπεραπλούστευση, αφου τα NUIs αναγκαστικά περιλαμβάνουν οπτικά στοιχεία – και κατα συνέπεια, γραφικές διεπαφές χρήστη. Μια ποιο ακριβής περιγραφή αυτής της έννοιας θα ήταν να το περιγράψουμε σαν μια μετάβαση απο το WIMP στο NUI.

Στο CLI, οι χρήστες έπρεπε να μάθουν ένα τεχνητό μέσο εισαγωγής, το πληκτρολόγιο, και μια σειρά απο κώδικοποιημένες εισαγωγές , οι οποίες έχουν ένα περιορισμένο πεδίο απαντήσεων, όπου και η σύνταξη αυτών των εντολών είναι επίσης αυστηρή.

Τότε, όταν το ποντίκι επέτρεψε τα GUI, οι χρήστες μπορούσαν να μάθουν ποιο εύκολα τις κινήσεις και τις δράσεις, και είχαν την δυνατότητα να εξερευνήσουν την διεπαφή πολύ ποιο διεξοδικά. Το GUI βασίστηκε στις μεταφορές για την αλληλεπίδραση με περίεχόμενο η αντικείμενα στην οθόνη. Το 'desktop' (επιφάνεια εργασίας) και το 'drag' (συρσιμο) για παράδειγμα, σαν μεταφορές για μια οπτική διεπαφή η οποία μεταφράστηκε πίσω σε αυστηρά κωδικοποιημένη γλώσσα του υπολογιστή.

Ένα παράδειγμα της παρανόησης του όρου NUI επιδείχτηκε στην CES 2010. "Τώρα ένα νέο κύμα απο προιόντα είναι έτοιμο να φέρει τις φυσικές διεπαφές χρήστη, όπως ονομάζονται αυτοί οι μεθόδοι ελέγχου των ηλεκτρονικών συσκευων, σε ένα ακόμα ευρύτερο κοινό".

Το 2010 ο Bill Buxton της Microsoft επανέλαβε την σημασία των NUI μέσα στην Microsoft Corporation με ένα βίντεο στο οποίο συζητούνται τεχνολογίες, οι οποίες θα μπορούσαν να χρησημοποιηθούν στην δημιουργία ενός NUI, και τις μελλοντικές δυνατότητες του.

Πρώιμα παραδείγματα

Multi-Touch

Όταν ο Bill Buxton ρωτήθηκε για την διεπαφή του iPhone's, απάντησε "Οι τεχνολογίες πολλαπλής αφής έχουν μακρά ιστορία. Για να το θέσουμε σε μια προοπτική, η αρχική δουλειά η οποία ανέλαβε η ομάδα μου, η οποία έγινε το 1984, τον ίδιο χρόνο που ο πρώτος υπολογιστής Macintosh τέθηκε σε κυκλοφορία, και δεν είμασταν οι πρώτοι.". Πράγματι, η διεπαφή του iPhone περιλαμβάνει στοιχεία απο WIMP το οποίο συμβάλλει στην ταξινόμιση μερικών περιορισμένων παραδειγμάτων στα οποία ένα NUI επιτυγχάνεται στην σχεδίαση.

Εικόνα 3

Το Multi-Touch είναι μια τεχνολογία η οποία θα μπορούσε να επιτρέψει μια φυσική διεπαφή χρήστη. Όμως, τα περισσότερα UI toolkits χρησημοποιούνται στο να κατασκευάζουν διεπαφές που εκτελούνται με τέτοια τεχνολογία είναι παραδοσιακές διεπαφές GUI .

Παραδείγματα των διεπαφών που συνήθως αναφέρονται ως NUI

Perceptive Pixel

Ένα παράδειγμα είναι η δουλειά που έγινε απο τον Jefferson Han σε διεπαφές πολλαπλής αφής. Σε μια επίδειξη στην έκθεση TED το 2006, έδειξε μια ποικιλία αλληλεπιδρόντων μέσων με υλικό επι της οθόνης, χρησημοποιόντας τόσο άμεσους χειρισμους, οσο και χειρονομίες. Για παράδειγμα, το να δώσεις σχήμα σε μια κολλώδη μάζα επι της οθόνης, ο Jeff πραγματικά 'τσιμπά' , την κεντρίζει και την σπρώχνει με τα δάχτυλα του. Σε μια διεπαφή GUI για μια εφαρμογή σχεδίασης για παράδειγμα, ένας χρήστης θα μπορούσε να χρησημοποιήσει την μεταφορά των 'εργαλείων' για να το κάνει αυτό, για παράδειγμα, διαλέγοντας ένα εργαλείο κεντρίσματος, ή διαλέγοντας 2 μέρη της μάζας στα οποία μετα θέλουν να εφαρμόσουν μια δράση 'τσιμπίματος' σε αυτά . Ο Han έδειξε οτι η διάδραση του χρήστη θα μπορούσε να είναι πολύ περισσότερο ενστικτώδης με την απομάκρυνση των συσκευών αλληλεπίδρασης που είμαστε συνηθησμένοι και την αντικατασταση τους με μία οθόνη η οποία έχει την δυνατότητα να εντοπίζει μια πολύ ευρύτερη γκάμα απο ανθρώπινες δράσεις και χειρονομίες. Φυσικά, αυτό επιτρέπει μόνο για ένα πολύ περιορισμένο σετ αλληλεπιδράσεων οι οποίες τοποθετούνται εύκολα σε φυσικούς χειρισμούς (RBI). Επεκτείνοντας τις δυνατότητες του λογισμικού πέρα απο τις σωματικές δράσεις απαιτεί αρκετά περισσότερη σχεδιαστική δουλειά.

Εικόνα 4

Microsoft PixelSense

Το Microsoft PixelSense χρησημοποιεί παρόμιες ιδέες στο πως οι χρήστες αλληλεπιδρούν με το περιεχόμενο, αλλα προσθέτει την δυνατότητα για την συσκευή να αναγνωρίσει οπτικά αντικείμενα τα οποία τοποθετούνται πάνω σε αυτό. Με αυτό τον τρόπο, μπορούν να πυροδοτήσουν δράσεις στον υπολογιστή μέσω των ίδιων χειρονομιών και κινήσεων όπως η οθόνη αφής του Jeff Han επιτρέπει, αλλά επίσης και αντικέιμενα γίνονται μέρος των μηχανισμών χειρισμού. Για παράδειγμα, όταν τοποθετείς ένα ποτήρι κρασιού στο τραπέζι, ο υπολογιστής το αναγνωρίζει σαν ποτήρι και προβάλλει περιεχόμενο σχετικό με αυτό. Τοποθετόντας ένα ποτήρι κρασιού σε ένα τραπέζει, τοποθετείται αποδεκτά σε δράσεις που παίρνονται με ποτήρια κρασιού και άλλα τραπέζια, και ετσι καθορίζεται αποδεκτά σε διεπαφές που βασίζονται στην αλήθεια. Έτσι, μπορεί να θεωρηθέι σαν εισαγωγή σε μια NUI εμπειρία.

Εικόνα 5

3D Immersive Touch

3D Immersive Touch ορίζεται σαν την απευθείας χειραγώγιση 3D εικονικών αντικειμένων περιβάλλοντος χρησημοποιόντας απλής ή πολλαπλής αφής επιφάνειες 3D σε τρισδιάστατα εικονικά περιβάλλοντα πολλαπλών χρηστών. Προχόρησε πρώτη το 2007, στην περιγραφή και τον ορισμό των αρχών μάθησης των 3D φυσικών διεπαφών χρήστη που σχετίζεται με το Edusim. Η Immersive Touch φυσική διεπαφή χρήστη τώρα διαφαίνεται να παίρνει μια ευρύτερη εστίαση και νόημα με την ευρύτερη προσαρμογή του υλικού επιφάνειας και επαφής όπως τα iPhone, iPod touch, iPad, και μια αναπτυσσόμενη λίστα απο άλλο υλικό. Η Apple δείχνει επίσης έντονο ενδιαφέρον στις "Immersive Touch" 3D φυσικές διεπαφές χρήστη τα

Εικόνα 6

τελευταία χρόνια. Αυτή η δουλειά χτίζεται πάνω στην ευρεία ακαδημαική βάση η οποία έχει διδαχτεί τον 3D χειρισμό σε περιβάλλοντα εικονικής πραγματικότητας.

Xbox Kinect

Xbox Kinect είναι ένα προιόν απο το Xbox το οποίο χρησημοποιεί χωρικές χειρονομίες για την αλληλεπίδραση αντι ενός χειριστηρίου παιχνιδιών. Σύμφώνα με την σελίδα της Microsoft, το Kinect έχει σχεδιαστεί για "ένα επαναστατικό νέο τρόπο για να παίζεις: δεν χρειάζεται χειριστήριο.". Πάλι, επειδή το Kinect επιτρέπει την αίσθηση του φυσικού κόσμου, δείχνει δυνατότητες για σχέδια RBI, και έτσι δυνητικά επίσης για NUI. Το ότι το Kinect έχει περιγραφεί σαν "NUI" είναι επίσης άλλο ένα παράδειγμα μιας παρανόησης του όρου, η τουλάχιστον συνέργεια του όρου για σκοπούς marketing.

Εικόνα 7

KINECT
for XBOX 360.

3. Συσκευές πολυ-τροπικής εισόδου

Στις αρχές του 2011 η εταιρία κατασκευής υπολογιστών και εξαρτημάτων αυτών, βλέποντας την επιτυχία και τους δρόμους που άνοιγε η καθιέρωση του Microsoft Kinect στην κονσόλα XBOX360 στην αλληλεπίδραση ανθρώπου υπολογιστή. Αποφάσισε να αναπτύξει και αυτή τον δικό της αισθητήρα βάθους και κίνησης για ανάπτυξη εφαρμογών στους προσωπικούς υπολογιστές. Η κίνηση αυτή έγινε για να καλυφθεί το κενό που είχε αφήσει η Microsoft με την μη-υποστήριξη developer kit μέχρι εκείνη την στιγμή.

Εικόνα 8

Σύντομη παρουσίαση του Asus Xtion Pro

Μια συναρπαστική νέα προσέγγιση για την ψυχαγωγία στο PC παρουσιάστηκε στη CeBIT2011. Το WAVI Xtion (προφέρεται "γουέι-βι ακσιον") αποτελείται από δύο συστατικά.Την WAVI ασύρματο media streaming συσκευή, και τον αισθητήρα κίνησης Xtion. Μεταφέρει υψηλής ανάλυσης πολυμέσα ασύρματα από ένα PC που βρίσκεται σε ένα δωμάτιο σε μια τηλεόραση που βρίσκεται ένα άλλο, ενώ η υπέρυθρη βασιζόμενη σε χειρονομίες αναπαραγωγή πολυμέσων, παιχνιδιών και εφαρμογών εισαγάγει για πρώτη φορά επίσημα υποστηρίζόμενο interface ανίχνευσης κίνησης με το PC.

Εικόνα 9

WAVI Xtion ενσωματώνει τεχνολογία ανίχνευσης κίνησης από την PrimeSense ® με το αποκλειστική διεπαφή χρήστη ASUS Xtion Portal να φέρνει πιο κοντά τους καταναλωτές με την τεχνολογία μέσω διαισθητικής αλληλεπίδρασης με εφαρμογές. Οι συνδέσεις υπολογιστών και τηλεοράσεων μέσω του ασύρματου HDMI(τεχνολογία WHDI) παρέχει ομαλή μεταφορά σε περιεχόμενο υψηλής ευκρίνειας,όπωση προβολή ταινιών,παιχνιδιών και φωτογραφιών σε 5GHz με εμβέλεια έως και 25 μέτρα.

Ποιο συγκεκριμένα στο παρόν κείμενο θα ασχολιθούμε με την αναλυτική παρουσίαση και ανάπτυξη εφαρμογών με βάση το Asus Xtion Pro Developer kit. Οι προγραμματιστές που ενδιαφέρονται να δημιουργήσουν με βαση χειρονομίες περιεχόμενο ψυχαγωγίας PC μπορούν να κάνουν χρήση του πακέτου αναπτυξης, την πρώτη επαγγελματική σουίτα εργαλείων ανίχνευσης κίνησης σε PC. Ανοίγει νέες ευκαιρίες για την ανάπτυξη των διαφόρων εφαρμογών αποτελεσματική χρήση του κόστους και του χρόνου, συμπεριλαμβανομένων των παιχνιδιών κάθε είδους.

Εικόνα 10

Ο αισθητήρας κινησης εχει τις εξης δυνατότητες:

Εικόνα 11

Specification

Effective Distance	Between 0.8 meters and 3.5 meters
Effective Angle	58° horizontal, 45° vertical, 70° diagonal
Interface/Power	USB 2.0
Sensor	Depth
Development Platform	Visual Studio .NET 2008, 2010
Framework	OpenNI
Programming Language	C++/C# (Windows) | C/C++ (Linux)
Supported OS	Win 32/64: XP, Vista, 7 | Linux 32/64: Ubuntu 10.10
Dimensions	1.5" x 7" x 1.9" | USB Cable Length: 5.5 feet

Γνωρίζωντας το τεράστιο οικονομικό μέγεθος της εταιρίας κατασκευής, κάποιος μπορεί εύκολα να συμπεράνει οτι θα υπάρχει και ανάλογη υποστήριξη στην ανάπτυξη και διανομή εφαρμογών που κατασκευάστηκαν μέσα απο το Xtion Pro. Μαζί με το αισθητήρα κίνησης έρχεται cd με την σουίτα ανάπτυξης εφαρμογών OpenNI NITE για την εύκολη ανάπτυξη motion-sensing εφαρμογών.

Εικόνα 12

Εικόνα 13

Και οταν ολοκληρωθεί η ανάπτυξη της εφαρμογής, ο developer μπορεί εύκολα να την πουλήσει στο ASUS@vibe διαδικτυακό κατάστημα πώλησης περιεχομένου διασκέδασης.

Εικόνα 14

Microsoft Kinect

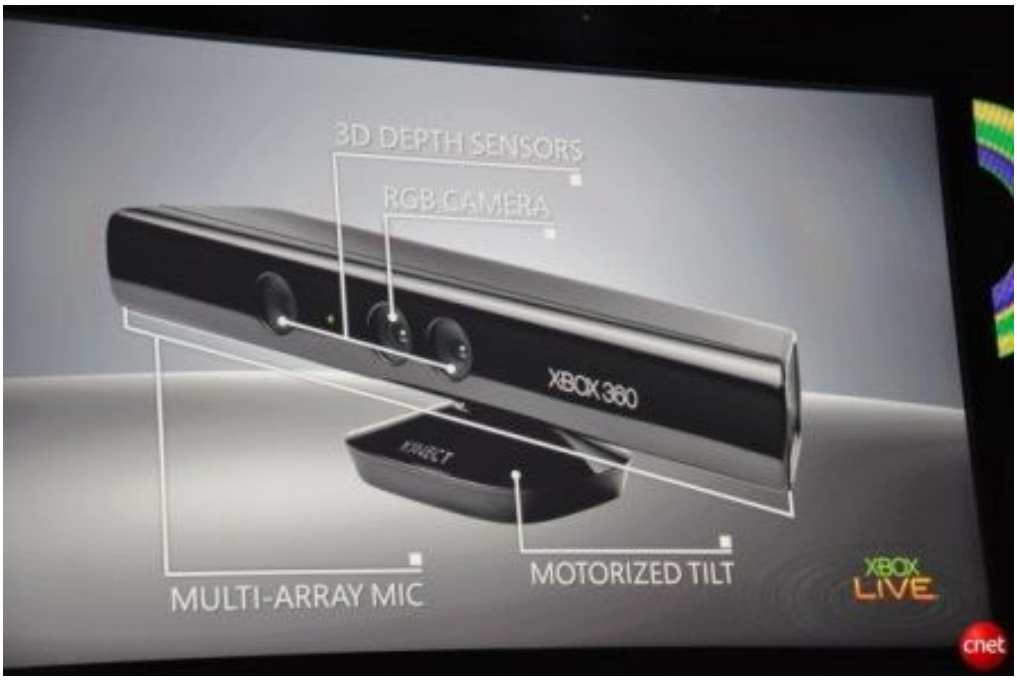
Το Kinect είναι μια συσκευή άισθησης της κίνησης απο την Microsoft για την κονσόλα ηλεκτρονικών παιχνιδιών Xbox 360 και για Windows PCs. Βασισμένο σε ένα webcam-style πρόσθετο περιφεριακό για την κοσόλα Xbox 360, επιτρέπει στους χρήστες να ελέγχουν και να αλληλεπιδρούν με Xbox 360 χωρίς να χρειάζεται να αγγίξουν χειριστήριο παιχνιδιών, μέσω μιας φυσικής διεπαφής χρήστη που χρησημοποιεί χειρονομίες και φωνητικές εντολές. Το το σχέδιο στοχεύει στην διεύρυνση του κοινού του Xbox 360 πέρα απο την τυπική βάση απο παίχτες παιχνιδιών. Το Kinect συναγωνίζεται με τα Wii Remote Plus και PlayStation Move με PlayStation Eye χειριστήρια κίνησης των Wii και PlayStation 3 οικιακών κονσολών αντίστοιχα. Μια έκδοση για Windows τέθηκε σε διάθεση στις 1Φεβρουαρίου 2012.

Το Kinect τέθηκε σε κυκλοφορία στην βόρεια Αμερική στις 4 Νοεμβρίου 2010, στην Ευρώπη στις 10 Νοεμβρίου 2010, στην Αυστραλία, Νέα Ζηλανδία και Σιγκαμπούρη στις 18 Νοεμβρίου 2010, κα στην Ιαπωνία στις 20 Νοεμβρίου 2010. Επιλογές παραγγελίας για το περιφεριακό αισθητήρα περιλαμβάνει στην συσκευασία το παιχνίδι Kinect Adventures και συσκευασίες μαζί με κονσόλα με είτε ένα 4 GB ή ένα 250 GB Xbox 360 μαζί με το Kinect Adventures.

Μετά απο πωλήσεις 8 εκατομμυρίων μονάδων στις πρώτες 60 μέρες, το Kinect κατέχει το ρεκόρ Guinness για την "ταχύτερη πώληση καταναλωτικών ηλεκτρονικών συσκευών ". 18 εκατομμύρια μονάδες του αισθητήρα Kinect έχουν ήδη πουληθεί μέχρι τον Ιανουάριο του 2012.

Η Microsoft διέθεσε πακέτο ανάπτυξης λογισμικού Kinect για Windows 7 στις 16 Ιουνίου 2011. Αυτό το SDK θα επιτρέπει στους προγραμματιστές να γράψουν εφαρμογές στο Kinect σε C++/CLI, C#, ή Visual Basic .NET.

Εικόνα 15



Xtion Pro(Live) VS Kinect

Τόσο οι αισθητήρες Microsoft Kinect και ο ASUS Xtion βασίζονται στην ίδια PrimeSense υπέρυθρη τεχνολογία. Έτσι όλα τα βασικά χαρακτιριστικά που είναι κρίσιμα για το full-body motion capture είναι γενικά τα ίδια. Αλλά υπάρχουν συγκεκριμένες διαφορές τις οποίες μπορείς να λάβεις υποψην:

Συσκευή	Συν	Μειόν
Asus Xtion	• Ποιό συμπαγές(7" x 2" x 1.5" έναντι 12" x 3" x 2.5") • Ποιό ελαφρύ(227g έναντι 1361g) • Δεν απαιτεί παροχή ηλεκτρικού ρευματος εκτός απο το USB. • Επιτρέπει 60 fps frame rate (μόνο στην ανάλυση 320 x 240)	• Λιγότερο δημοφιλής συσκευη • Χαμηλότερη ποιότητα Drivers • Δεν δουλέυε με μερικούς ελεγκτές USB • Όχι κινητήρας, επιτρέπει μόνο χειροκίνητη τοποθέτηση
MS Kinect	• Υψηλή ποιότητα οδηγών συσκευής • Σταθερή εργασία με διάφορα μοντέλα υλικού • Εχει κινητήρα που μπορεί να ελεγχθεί απομακρυσμένα, αυτό κάνει την τοποθέτηση της συσκευής ποιο βολική	• Μεγαλύτερο μέγεθος • Ποιό βαρύ • Χρειάζεται εξωτερική πηγη τροφοδοσίας • Μόνο 30 fps frame rate

Εικόνα 16

Leap Motion

Leap Motion

Βιομηχανία έλεγχος κίνησης

Ιδρυθηκέ 2010

Ιδρυτές Michael Buckwald, David Holz

Έδρα San Francisco, California, United States

Website LeapMotion.com

Το Leap Motion είναι μια νεοσύστατη ανάπτυξη προχωρημένης τεχνολογίας στην ανίχνευση της κίνησης για αλληλεπίδραση ανθρώπου-υπολογιστή. Αρχικά εμπνευσμένη απο την απο την απογοήτευση που περιβάλλει το 3D modeling με την χρήση ποντικιού και πληκτρολογίου, η Leap Motion ισχυρίζεται οτι η διαμόρφωση εικονικου πηλού θα πρέπει να είναι τόσο εύκολη όσο η διαμόρφωση πηλού στον πραγματικό κόσμο.

Εικόνα 17

Τεχνολογία

Το Leap είναι ένα μικρό USB περιφεριακό που σχεδιάστηκε να τοποθετηθεί στο γραφείο ενός χρήστη προς τα πάνω, έτσι δημιουργόντας ένα 3D χώρο αλληλεπίδρασης περίπου 8 κυβικών ποδιών. Μέσα σε αυτό τον χώρο, το Leap διαφημήστηκε μέσα απο ένα απόσπασμα video να ανιχνεύει χέρια και δάχτυλα όπως επίσης και εργαλεία όπως στυλούς, μολύβια, και chopsticks με πολύ μεγάλη ακρίβεια. Αυτο διαφοροποιεί το προιόν απο το Kinect, το οποίο είναι ποιό κατάλληλο για ανίχνευση ολόκληρου του σώματος σε ένα χώρο με το μέγεθος ενός σαλονιού.

Σε μια επίδειξη στην CNET, Το Leap δείχθηκε να κάνει εκτέλεση καθηκόντων όπως η περιήγηση μιάς ιστοσελίδας, με την χρήση pinch-to-zoom χειρονομιών σε χάρτες, υψηλής ακρίβειας σχεδίαση, και χειρισμός περίπλοκων 3D οπτικοποιήσεων δεδομένων.Ο CEO της Leap Motion Michael Buckwald είπε στη CNET:

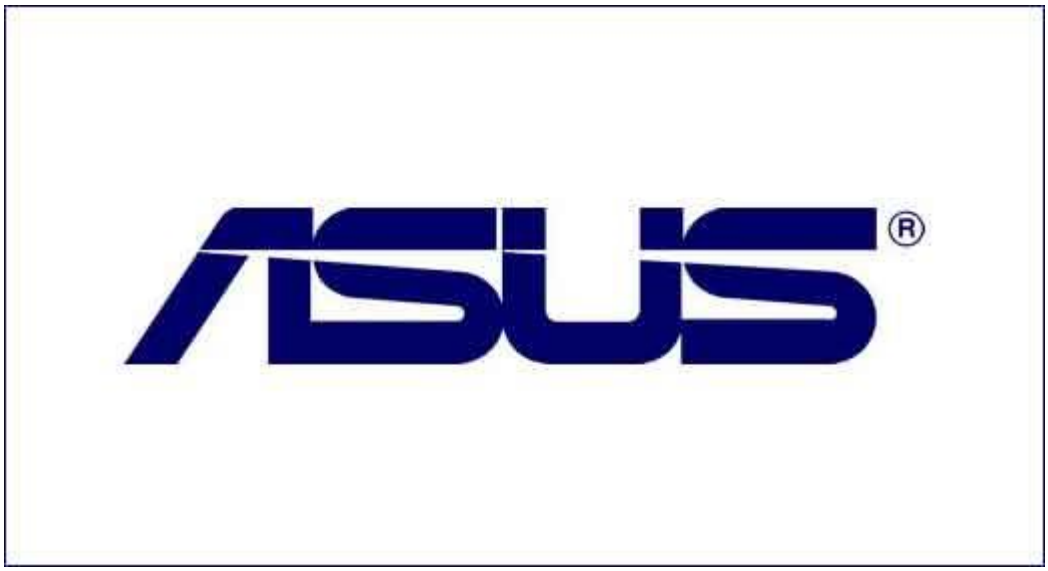
"Θέλουμε να υπάρχουν εφαρμογές που αλλάζουν τον κόσμο και ουσιαστικά αλλάζουν πως οι άνθρωποι αλληλεπιδρούν με το λειτουργικό σύστημα ή περιηγούνται Web.... Ο σκοπός είναι να αλλάξουμε ουσιαστικά το πως οι ανθρώποι αλληλεπιδρουν με τους υπολογιστές και πως να το κάνουν με τον ίδιο τρόπο όπως το έκαναν με το ποντίκι, το οποίο σημαίνει ότι τους επιρεάζει όλους, απο την ποιο βασική περίπτωση χρήσης μέχρι τις ποιο προχωρημένες περιπτώσεις χρήσης που μπορείς να φανταστείς για την τεχνολογία υπολογιστών."

Η Leap Motion σχεδιάζει να διαμοιράσει χιλάδες μονάδες δωρεάν στους προγραμματιστές που ενδιαφέρονται για την δημουργία εφαρμογών για την συσκευή, ακολουθούμενη απο την κυκλοφορία στο κοινό με τιμή $69.99.

Εικόνα 18

4. Ποιά είναι η ASUS

Εικόνα 19



Asus επίσημα Η ASUSTeK Computer Inc) (αγγλική προφορά: / e□s□s) είναι μια ταϊβανέζικη πολυεθνική ηλεκτρονικών υπολογιστών και ηλεκτρονικών ειδών της εταιρείας που εδρεύει στην Ταϊπέι, Ταϊβάν. Τα προϊόντα της περιλαμβάνουν μητρικές κάρτες, υπολογιστές γραφείου, φορητούς υπολογιστές, οθόνες, tablet PCs, servers, κάρτες βίντεο και κινητά τηλέφωνα. Πωλεί κυρίως προϊόντα με το δικό της σήμα, αλλά και παράγει εξαρτήματα για τους υπόλοιπους κατασκευαστές, συμπεριλαμβανομένης της Apple, η Dell και η HP.

Asus είναι η πέμπτη σε μέγεθος προμηθευτής στον κόσμο PC μέχρι το 2011 σε πωλήσεις (μετά την HP, η Lenovo, η Dell και η Acer). Asus εμφανίζεται στο BusinessWeek "InfoTech 100" και "Top 10 IT εταιρείες της Ασίας" rankings. Η Wall Street Journal της Ασίας την κατατάσσει το νούμερο ένα στην ποιότητα και την εξυπηρέτηση, και κατέλαβε την πρώτη θέση στην κατηγορία υλικού πληροφορικής της Ταϊβάν 2008 Top 10 Παγκόσμιες Μάρκες σε μια έρευνα, με συνολική αξία του εμπορικού σήματος των ΗΠΑ 1,324 δισεκατομμύρια δολάρια.

Asus είναι πρωτίστως εισηγμένη στο Χρηματιστήριο της Ταϊβάν (TWSE: 2357) και παράλληλη εισαγωγή στο Χρηματιστήριο του Λονδίνου (LSE: ASKD).

Ονομα

Το όνομα Asus προέρχεται από την λέξη Πήγασος, το φτερωτό άλογο της ελληνικής μυθολογίας. Μόνο τα τελευταία τέσσερα γράμματα της λέξης χρησιμοποιήθηκαν προκειμένου να δοθεί στο όνομα υψηλή θέση στις αλφαβητικές καταχωρήσεις.

Ιστορία

Asus ιδρύθηκε στην Ταϊπέι το 1989 από T.H. Tung, Τεντ Hsu, Γουέιν Hsieh και M.T. Liao, [και οι τέσσερις έχουν εργαστεί προηγουμένως στην Acer, ως μηχανικοί ηλεκτρονικών υπολογιστών.

Στις αρχές της δεκαετίας του 2000, κατασκευαστές motherboard με βάση την Ταϊβάν δεν είχαν πάρει ακόμη την ηγετική θέση τους στην επιχείρηση υπολογιστών-υλικού. Η Intel Corporation θα παρείχε όλους τους νέους επεξεργαστές σε πιο καθιερωμένες εταιρείες όπως η IBM πρώτα, και οι εταιρείες της Ταϊβάν θα πρέπει να περιμένουν για περίπου έξι μήνες μετά απο την IBM που έλαβε την μηχανική πρωτοτύπων τους. Από το 2009, η Asus λαμβάνει δείγματα μηχανικής Intel ποιο μπροστά από τους ανταγωνιστές της.

Τον Ιανουάριο του 2007, η Asus ξεκίνησε την αναδιάρθρωση των δραστηριοτήτων της. Η εταιρεία χωρίζεται σε τρεις ξεχωριστές επιχειρησιακές μονάδες: Asus, η Pegatron και η Unihan Corporation. Η μάρκα ASUS εφαρμόστηκε μόνο στο πρώτο τμήμα επώνυμων υπολογιστων. Η Pegatron χειρίζεται την κατασκευή των OEM μητρικών και εξαρτημάτων, και η Unihan Corporation επικεντρώθηκε σε μη-PC κατασκευής, όπως τα cases και καλουπιών. Τον Ιανουάριο του 2008, Pegatron απέκτησε την Unihan Corporation ως θυγατρική απο την Asus.

Στις 9 Δεκεμβρίου 2008, την Open Handset Alliance ανακοίνωσε ότι η Asus είχε γίνει ένα από τα 14 νέα μέλη της οργάνωσης. Αυτά τα νέα μέλη θα αναπτύξουν, είτε συμβατές συσκευές Android, συμβάλλουν σημαντικά κώδικα του Android έργο ανοικτού πηγαίου κώδικα, ή να υποστηρίξει το οικοσύστημα μέσω των προϊόντων και υπηρεσιών που θα επιταχύνει τη διαθεσιμότητα απο Android-based συσκευές.

Τον Οκτώβριο του 2010 η Asus και η Garmin ανακοίνωσαν ότι θα τελειώνουν την εταιρική σχέση τους στα smartphone, ως αποτέλεσμα της Garmin να αποφασίσει για έξοδο από αυτή την κατηγορία προϊόντων. Οι δύο εταιρείες είχαν παράγει έξι επώνυμα Garmin-Asus smartphones προηγούμενη διετία.

Timeline

- Σεπτέμβριος 2005: Η Asus διέθεσε την πρώτη κάρτα επιταχυντή PhysX
- Δεκέμβριος 2005:Η Asus μπήκε στην αγορα LCD με το μοντέλο TLW32001, αρχικά μόνο διαθέσιμο στην Ταιβανέζικη αγορά.
- Ιανουάριος 2006: Η Asus ανακοίνωσε οτι θα συνεργαστεί με την Lamborgini για να αναπτύχει την σειρά VX
- 9 Μαρτίου 2006: Η Asus επιβεβαιώθηκε ότι ήταν απο τους κατασκευαστές του πρώτου μοντέλου Microsoft Origami, μαζί με την Samsung και την Founder Technology
- 8 Αυγούστου 2006:Η Asus ανακοίνωσε μια κοινή δράση με την Gigabyte Technology
- 5 Ιουνίου 2007: Η Asus ανακοίνωσε το Eee Pc στην Computex Taipei 2007.
- 9 Σεπτεμβρίου 2007: Η Asus δήλωσε υποστήρηξη για το BlueRay, ανακοινόνοντας την διάθεση ενώς BD-ROM/DVD writer PC drive, BC-1205PT. Η διάθεση διάφορων blueray based notebooks ακολουθει.
- 31 Οκτωμβρίου 2007: Η Asus διαθέτει ενα PDA/smartphone συσκευή στην Αγγλική αγορά.

- 3 Ιανουαρίου 2008: Η Asus διασπάται επίσημα σε 3 εταρίες: ASUSTeK, Pegatron και Unihan.

- 22 Αυγούστου 2008: Λεπτομέριες απο το Asus N10 διέρευσαν στο διαδίκτυο.

- Δεκέμβριος 2008: Η Asus διέθεσε μια beta BIOS η οποία διορθώνει την ασυμβατότητα των Windows XP SP3 με την μητρική ASUS A8N32-SLI Deluxe.

- 1 Ιουνίου 2010: Η Asus αποσπάται απο την Pegatron Corp.

- Δεκέμβριος 2010: Η Asus κυκλοφορεί το πιο λεπτό laptop του κόσμου, το Asus U36, με Intel επεξεργαστή στανταρτ τάσεως (όχι χαμηλής τάσης) Intel core i3 ή i5 με πάχος μόνο 19 mm.

- Οκτώβριος 2011: Η Asus σχεδίασε το UX/21E/31E ZENBOOK™, χρησημοποιεί ένα σχέδιο κατασκευής ακριβείας, με διαστάσεις 3mm μπροστά και 9mm πίσω. Η μορφή αυτή περιλαμβάνει λεπτές σαν ξυράφι άκρες.

- Μάρτιος, 2012: Η Asus διαθέτει στην Ευρωπαική αγορα το Asus K93SM notebook με οθόνη 18.4-ιντζών με ανάλυση 1920 x 1080 pixels (Full HD), μια GeForce GTX 630M ανεξαρτητη κάρτα γραφικών με μνήμη 1GB καιWindows 7 Home Premium 64bit προεγκατεστημένα.

Δραστηριότητες και εγκαταστάσεις

Η Asus έχει το αρχηγείο της στην επαρχία Beitou , Taipei, Taiwan.

Στο 2009 η Asus εχει κατασκευαστικές εγκαταστάσεις στην Ταιβάν (Taipei, Luzhu, Nangan, Guishan), Κίνα (Suzhou), Μεξικό (Ciudad Juárez) και την Τσέχικη Δημοκρατία (Ostrava). Το πάρκο υψηλής τεχνολογίας της Asus, που βρίσκεται στην Suzhou, Κίνα, καλύπτει έκταση 540,000τμ , σχεδόν το μέγεθος 82 γηπέδων ποδοσφαίρου.

Η Asus ισχυρίζεται οτι έχει δυνατότητα μηνιαίας παραγωγής 2,000,000 μητρικών καρτών και 150,000 φορητών υπολογιστών.

Η Asus διαχειρίζεται περίπου στα 50 service sites σε 32 χώρες και εχει πάνω απο 400 service partners παγκοσμίως.

Προϊόντα

Τα προϊόντα της Asus περιλαμβάνουν φορητούς υπολογιστές, tablet υπολογιστές, κινητά τηλέφωνα, PDAs, διακομιστές, οθόνες υπολογιστών, μητρικές κάρτες, κάρτες γραφικών και ήχου, συσκευές οπτικών δίσκων, συσκευές δικτύων υπολογιστών, θήκες υπολογιστών, εξαρτήματα υπολογιστών και ψυκτικά συστήματα.

Eee range

Απο την πρεμιέρα του το 2007, το Eee PC netbook έχει μαζέψει πολλά βραβεία, συμπεριλαμβανομένου του Forbes Asia's Προϊόν της Χρονιάς και άλλα πολλά.

Η Asus στην συνέχεια πρόσθεσε διάφορα προϊόντα στην σειρά Eee , συμπεριλαμβανομένων:

EeeBox PC, ένα συμπαγές nettop

Eee Top, ένα όλα σε ένα υπολογιστή με οθόνη αφής που στεγάζεται σε μια θήκη για LCD οθόνη,

Eee Stick, ένα plug-and-play ασύρματο χειριστήριο για το PC το οποίο μεταφράζει τις φυσικές κινήσεις του χεριού του χρήστη σε αντίστοιχες στην οθόνη.

Eee Pad Transformer, ειναι ενας υπολογιστής tablet, ο οποίος τρέχει σε λειτουργικό σύστημα Android 3.0.

Eee Pad Transformer Prime, ο διάδοχος του αρχικού Transformer.

Στις 6 Μαρτίου 2009, Η Asus διέθεσε το Eee Box B202, το οποίο είδε το PCMag σαν "το αντίστοιχο desktop Asus EeePC", (Η σειρά υπολογιστών "Asus Eee Box" αργότερα το 2010 μετονομάστηκε σε "Asus EeeBox PC").

Με την τιμή του μεταξύ US$269 και US$299, αυτο το desktop συναγωνίζεται απευθείας με το Mac Mini.

Σειρά Essentio

Essentio είναι μια σειρά απο επιτραπέζια PCs. Απο τον Δεκέμβριο του 2011 η σειρά αποτελείται απο τις CG Series (σχεδιασμένη για παιχνίδια), την CM series (για διασκέδαση και οικιακή χρήση) και τις CS and CP slimline series.

Ψηφιακοί δέκτες πολυμέσων

Η Asus πουλά ψηφιακούς δέκτες πολυμέσων με όνομα Asus O!Play.

Συσκευές GPS

Η Asus παράγει την συσκευή GPS R700T , η οποία ενσωματώνει το Traffic Message Channel.

Republic of Gamers

Republic of Gamers είναι μια σειρά απο προιόντα που διατέθηκαν απο την Asus το 2006 "με σκοπό την διάθεση του ποιό σκληροπυρηνικού υλικού για αληθηνά αφοσιωμένους παίχτες." Τα προιόντα Republic of Gamers περιλαμβάνουν μητρικές, κάρτες γραφικών, φορητούς υπολογιστές, επιτραπέζιους, περιφεριακά και αξεσουάρ.

5. Εφαρμογές τρίτων

Τα Asus Xtion Pro/Kinect μπορουν να χρησημοποιηθούν σε πολλούς κλάδους πέρα απο την βασική τους χρήση σε παιχνίδια και εφαρμογές που χρησημοποιούνται στο σαλόνι μας.

Ανάπτυξη εφαρμογών τρίτων

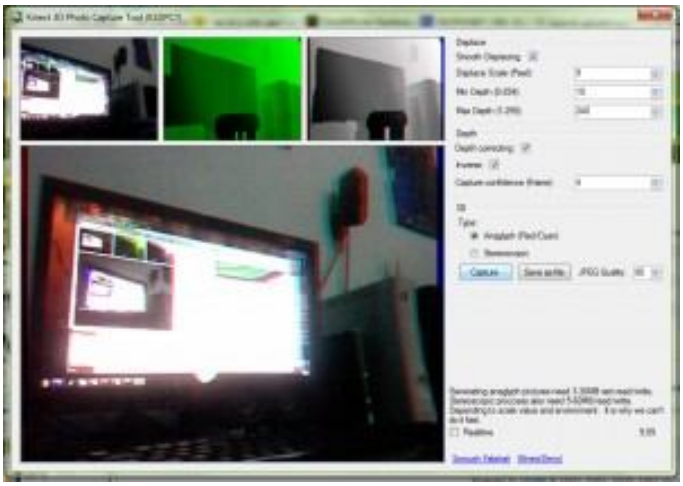
Μια επίδειξη για χρήση απο τρίτους του Kinect στο Maker Faire. Η οπτικοποίηση στα αριστερά, παρέχεται μέσω Kinect, είναι ενός χρήστη μπουφάν που έχει φορετό ηλεκτρονικό χειρισμό για τοVJing.

Εικόνα 20

Πολλοί προγραμματιστές διερευνούν πιθανές εφαρμογές του Kinect οι οποίες πάνε πέρα απο τον βασικό σκοπό του συστήματος, δηλαδή το παίξιμο παιχνιδιών. Για παράδειγμα, ο Philipp Robbel του MIT συνδύασε το Kinect με το iRobot Create για να χαρτογραφίσει ένα δωμάτιο σε 3D και να κάνει το ρομπότ να ανταποκρίνεται σε ανθρώπινες χειρονομίες, παράληλα μια ομάδα του MIT Media Lab εργάζεται πάνω σε ένα JavaScript extension για τον Google Chrome που ονομάζεται depthJS το οποίο επιτρέπει στους χρήστες να ελέγχουν τον περιηγητή με χειρονομίες. Άλλοι προγραμματιστές, περιλαμβανομένου του Robot Locomotion Group στο MIT, χρησημοποιούν τους οδηγούς να αναπτύξουν μια διεπαφή χρήστη ελεγχόμενη απο την κίνηση παρόμοια με αυτή που παρουσιάζεται στην ταινία Minority Report. Οι προγραμματιστές του MRPT ενσωμάτωσαν οδηγούς ανοιχτού λογισμικού στις βιβλιοθήκες και τα παρεχόμενα δείγματα του live 3D rendering και basic 3D visual SLAM. Αλλη μιά ομάδα έχει δείξει μια εφαρμογή η οποία επιτρέπει στους χρήστες του Kinect να παίξουν ένα εικονικό πιάνο με το να χτυπάνε ελαφρά τα δάχτυλα τους σε ένα άδειο θρανίο. Ο Oliver Kreylos, ένας ερευνητής στο πανεπιστήμιο της Καλιφόρνια, Davis, υιοθέτησε την τεχνολογία για να βελτιώσει ζωντανή τρισδιάστατη τηλεδιάσκεψη, στην οποία έδειξε ενδιαφέρον η NASA.

Ο Alexandre Alahi απο το EPFL παρουσίασε ένα σύστημα παρακολούθησης μέσω βίντεο το οποίο συνδιάζει πολλαπλές συσκευές Kinect για να εντοπίζει ομάδες ανθρώπων ακόμα και στο απόλυτο σκοτάδι. Οι εταιρίες So touch και Evoluce δημιούργησαν ένα λογισμικό παρουσίασης για το Kinect το οποίο μπορεί να ελεγχθεί απο χειρονομίες; ανάμεσα στα χαρακτηριστκά του είναι πολλαπλής αφής zoom mode. Τον Δεκέμβριο του 2010, η δωρεάν δημόσια beta του λογισμικού της HTPC, KinEmote τέθηκε σε κυκλοφορία, επιτρέπει την περιήγηση των μενού Boxee και XBMC χρησημοποιόντας ένα αισθητήρα Kinect. Ο Soroush Falahati έγραψε μία εφαρμογή η οποία μπορεί να χρησημοποιηθέι να δημιουργήσει στερεοσκοπικές εικόνες 3D με ένα αισθητήρα Kinect.

Εικόνα 21



Για περιορισμένο χρόνο το Μάη του 2011, ένα μαγαζί Topshop στη Μόσχα εγκατέστησε ένα Kinect kiosk το θα μπορούσε να επικαλύψει μαι συλλογή απο φορέματα πάνω στην ζωντανή μετάδωση βίντεο απο πελάτες. Μέσω αυτόματου εντοπισμού, η θέση και περιστροφή του εικονικού φορέματος ενημερώνονταν ακόμα και όταν οι πελάτες γυρνούσαν πλάτη για να δουν το πίσω μέρος του φορέματος.

Εικόνα 22

Το Kinect επίσης δείχνει συναρπαστική δυναμική για χρήση στην ιατρική. Ερευνητές στο πανεπιστήμιο της Μινεσότα χρησημοποίησαν το Kinect για να μετρήσηουν ένα εύρος απο συμπτώματα διαταραχών σε παιδιά, δημιουργώντας νέους τρόπους αντικειμενικής αξιολόγισης στον εντοπισμό τέτοιων καταστάσεων όπως ο αυτισμός, διαταραχή ελλειματικής προσοχής και ψυχαναγκαστική-καταναγκαστική διαταραχή. Διάφορες ομάδες ανέφεραν χρήση του Kinect σε διεγχειρητική, ανασκόπηση της ιατρικής απεικόνισης, επιτρέποντας στον χειρουργό να έχει πρόσβαση σε πληροφορίες χωρίς μόλυνση. Αυτή η τεχνική είναι ήδη σε χρήση στο Sunnybrook Health Sciences Centre στο Τορόντο, όπου οι γιατροί το χρησημοποιούν για απεικόνιση καθοδήγησης κατα την διάρκεια χειρουργείου καρκίνου. Τουλάχιστο μια εταιρία, η GestSure Technologies, επιδιώκει την εμπορευματοποίηση ενός τέτοιου συστήματος.

Εικόνα 23

6. OpenNi και Nite

Εισαγωγή

Κίνητρο

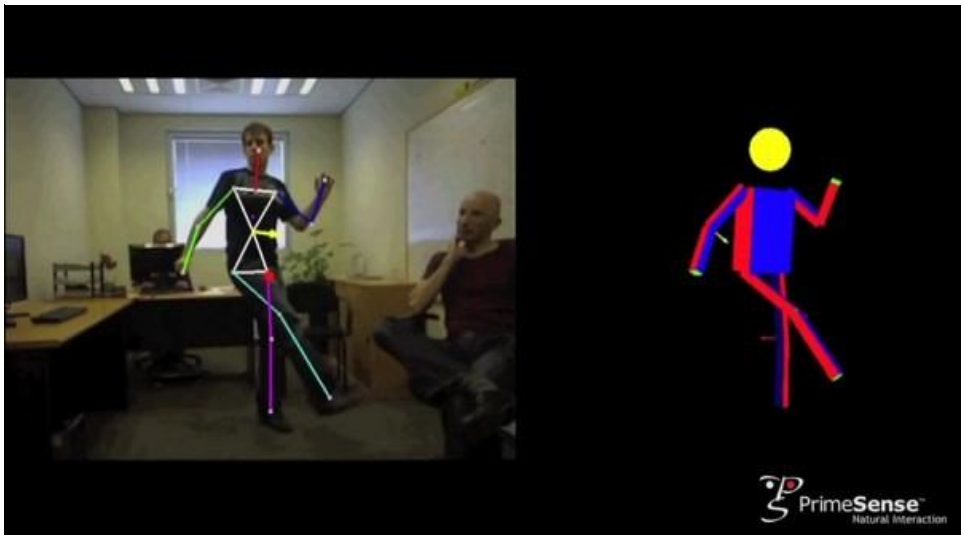
Το OpenNI έχει σκοπό να βοηθήσει στην ανάπτυξη εφαρμογών οι οποίες χρησημοποιούν 3D vision εισαγωγές δεδομένων (όπως έλεγχος με ολόκληρο το σώμα), σπάζοντας την άκαμπτη σύνδεση μεταξύ της εφαρμογής, του αισθητήρα και των αλγορίθμων όρασης. Ο σκοπός είναι να μικρύνει τον χρόνο διάθεσης τέτοιων εφαρμογών όταν θέλουμε να τις μεταφέρουμε για χρήση με άλλους αλγόριθμους, ή άλλο αισθητήρα.

Εικόνα 24

Γενική περιγραφή

Το OpenNI είναι μια στάνταρτ διεπαφή για αλγόριθμους επεξεργασίας δεδομένων 3D αισθητήρα. Είναι ανοιχτή για όλους, (δημοσιεύτικε στην ιστοσελίδα) και ανοιχτού κώδικα. Ο σκοπός είναι να οριστούν τύποι δεδομένων (depth map, color map, user pose, etc.) και μια διεπαφή μια μονάδα που μπορεί να τους παράγει (sensor, skeleton algorithm, etc.), για τρίτους προγραμματιστές.

Εικόνα 25



Οι προγραμματιστές εφαρμογών/παιχνιδιών μπορούν να γράψουν τις εφαρμογές τους ανεξάρτητα απο τον πραγματικό προμηθευτή, ο οποίος δημιουργεί τα προιόντα της 3D όρασης (skeleton, hand points, etc.) Προγραμματιστές ενδιάμεσου λογισμικού μπορούν να γράψουν αλγόριθμους πάνω απο τις ακατέργαστες μορφές δεδομένων, ανεξάρτητα απο την απο την πραγματική συσκευή που τα παράγει.

Οι κατασκευαστές αισθητήρων μπορούν να ενσωματώσουν την διεπαφή της συσκευής για τον αισθητήρα τους, έτσι ώστε οι εφαρμογές που είναι γραμμένες πάνω απο το OpenNI να μπορούν να δουλέψουν με κάθε αισθητήρα.

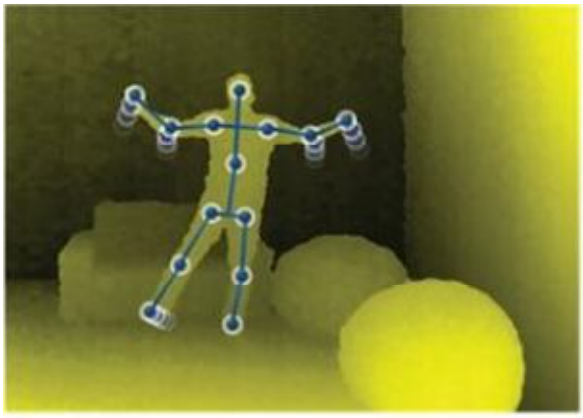
Ο κύριος σκοπός του OpenNI είναι να παράσχει μια διεπαφή για εφαρμογές οι οποίες χρησημοποιούν φυσική διεπαφή (χειρονομίες /στάσεις) σαν εισαγωγή τους. Η εφαρμογή μπορεί να γραφτεί μια φορά, και μετά μπορεί να εκτελεστεί ανεξάρτητα απο τον πωλητή ή την έκδοση του παροχέα φυσικής διεπαφής.

Το OpenNI εγγυήται πλήρη συμβατότητα προς τα πίσω, έτσι ώστε οι οδηγοί ποι γράφτηκαν για παλιότερες εκδόσεις να μπορούν να δουλεύουν και στις νέες. Κάθε οδηγός/εφαρμογή που εγκαθιστά τον εαυτο του σε μια μηχανή πρέπει επίσης να εγκαθιστά και την νεώτερη έκδοση του OpenNI.

Production Nodes

Δημιουργόντας ένα προιόν 3D όρασης είναι συνήθως ποιό περίπλοκο απο το να παίρνεις απλά την έξοδο ενός συγκεκριμένου αισθητήρα. Συνήθως αρχίζει με μια πραγματική συσκευή (τον αισθητήρα) να παράγει ένα είδος εξόδου (η ποιό κοινή περίπτωση είναι ένας χάρτης βάθους, όπου κάθε πίξελ έχει την απόσταση του απο τον αισθητήρα). Κάποιο είδος ενδιάμεσου λογισμικού τότε χρησημοποιήται να επεξεργαστεί αυτή την έξοδο, και να παράγει μια υψηλότερου επιπέδου, όπως την τοποθεσία ενώς χρήστη, ή την τρέχουσα στάση του.

Εικόνα 26



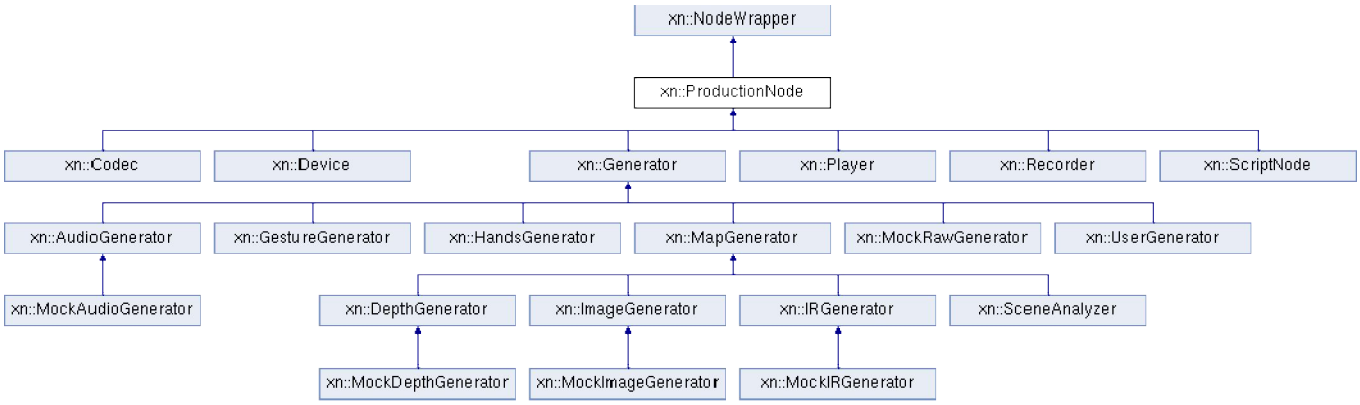
Το OpenNI ορίζει "μονάδες παραγωγής", όπου κάθε τέτοια μονάδα μπορεί να λάβει δεδομένα απο άλλες μονάδες, και προαιρετικά, να παράγει δεδομένα τα οποία μπορουν να χρησημοποιηθούν απο άλλες μονάδες ή απο την ίδια την εφαρμογή. Για να υποστηρίξει αυτή την ροή, κάθε τέτοια μονάδα ονομάζεται Production Node(κόμβος παραγωγής),και όλοι αυτοί οι κόμβοι είναι συνδεδεμένοι σε ένα γράφημα παραγωγής.

Στις περισσότερες περιπτώσεις, μια εφαρμογή ενδιαφέρεται μόνο στο κορυφαίο προιόν ενός τέτοιου γραφήματος. Το OpenNI επιτρέπει στην εφαρμογή να χρησημοποιήσει

ενα ενιαίο κόμβο, χωρίς να ξέρει τίποτα για ολόκληρο αυτό το γράφημα που είναι πίσω απο τον κόμβο. Φυσικά, για προχωριμένο tweaking, υπάρχει πάντα η επιλογή να έχεις πρόσβαση στο γράφημα, και μπορείς να ρυθμίσεις αυτούς τους κόμβους.

Εξαρτώμενα απο την μηχανή στην οποία τρέχει η εφαρμογή, και τα εξαρτήματα που το συνοδεύουν σε μια συγκεκρίμένη στιγμή, διάφορα γραφήματα μπορούν να δημιουργηθούν για να παράγουν τον ίδιο τύπο δεδομένων. Το OpenNI παρέχει μια διεπαφή για την απαρίθμηση πιθανών δέντρων παραγωγής για την υποδοχή ενός συγκεκριμένου προιόντος. Η εφαρμογή τότε μπορεί να επιλέξει ένα απο αυτά τα γραφήματα και να το δημιουργήσει. Είναι ολοκληρωτικά στην ευθύνη της εφαρμογής να επιλέξει ένα δέντρο απο τις πιθανότητες που επιστρέφονται απο το OpenNI framework. Μπορεί να το κάνει επίσης επιλέγοντας απο συγκεκριμένους πωλητές, εξαρτήματα, ή εκδόσεις.

Εικόνα 27



Τύποι Nodes

Κάθε κόμβος παραγωγής έχει ενα συγκεκριμένο τύπο. Προς το παρόν οι υποστηριζόμενοι τύποι είναι:

Συσκευή – εκπροσωπεί μια φυσική συσκευή

Βάθος – παράγει χάρτες-βάθους

Εικόνα – παράγει έγχρωμους χάρτες-εικόνων

IR - παράγει IR χάρτες-εικόνων

Ήχος – παράγει μια ροή ήχου

Χειρονομίες – παράγει επανακλήσεις όταν αναγνωρίσει συγκεκριμένες χειρονομίες

Αναλυτής σκηνής – αναλύει μια σκηνή (ξεχωρίζει το παρασκήνιο απο το προσκήνιο, κτλ.)

Χέρια – παράγει επανακλήσεις όταν δημιουργούνται σημεία χεριών, αλλάζει η θέση τους , και επίσης όταν καταστρέφονται.

Χρήστης – παράγει μια αντιπροσώπευση ενός χρήστη στον 3D χώρο.

Επίσης, για την υλοποίηση, οι ακόλουθοι τύποι κόμβων υποστηρίζονται:

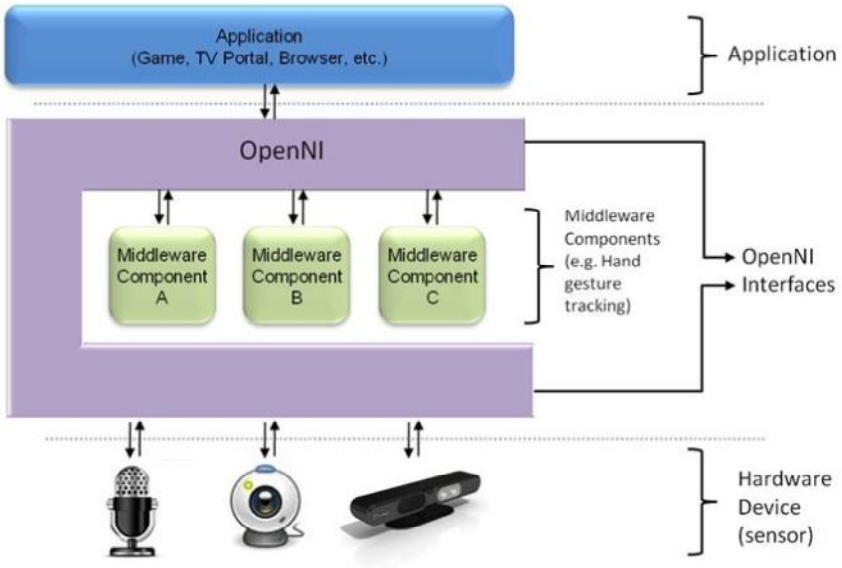
Recorder – υλοποιεί μια εγγραφή των δεδομένων.

Player – μπορεί να διαβάσει δεδομένα απο μια εγγραφή και να τα αναπαράγει.

Codec - χρησημοποιείται για την συμπίεση και αποσυμπίεση δεδομένων στις εγγραφές.

Κάθε τύπος ενός κόμβου παραγωγής παρέχει διαφορετική λειτουργικότητα για διαμόρφωση. Μερικές εξισώσεις έιναι κοινές για όλους παραγωγούς (να εξερέσουμε την φυσική συσκευή, η οποία δεν παράγει τίποτα πραγματικό). Μερικά ειναι κοινά για τους παραγωγούς χαρτών (βάθους,εικόνα, IR), etc.

Εικόνα 28



Δυνατότητες

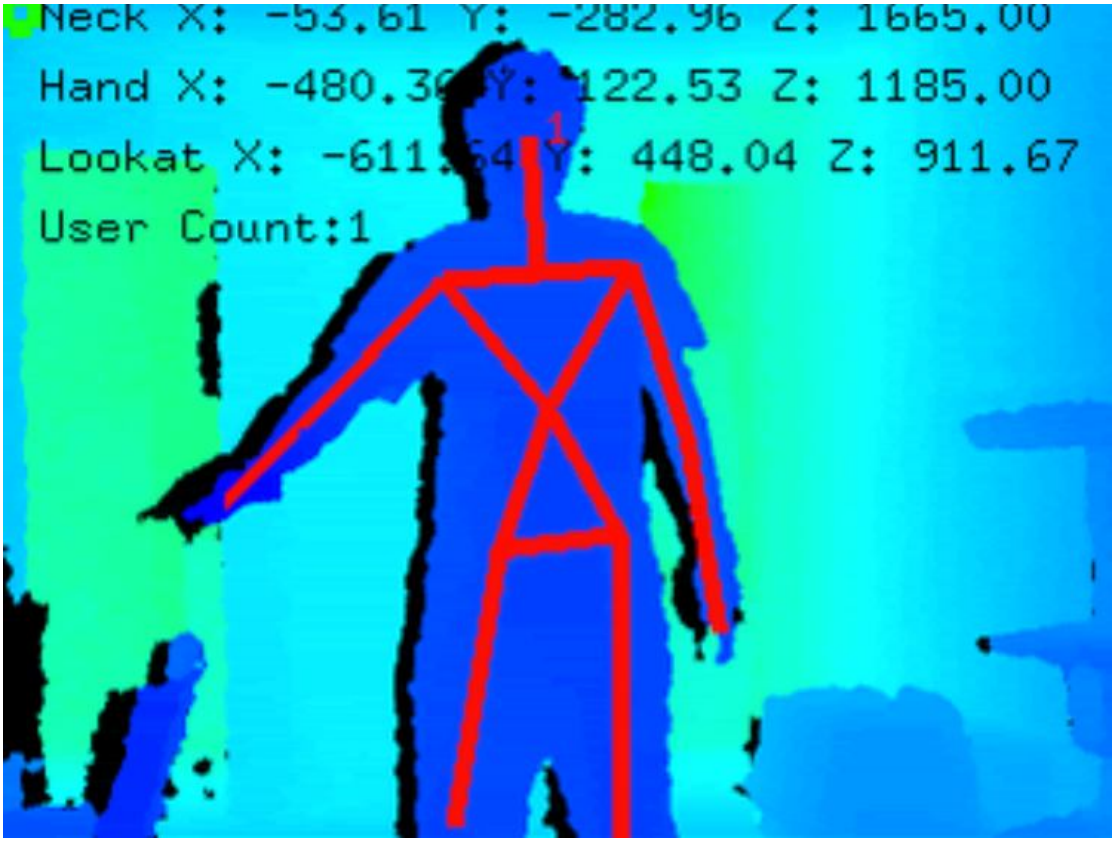
Το OpenNI αναγνωρίζει οτι διαφορετικοί πάροχοι πιθανόν να έχουν διαφορετικές επιλογές διαμόρφωσης για τους κόμβους παραγωγής τους. Έτσι, ενα σύνολο κοινών διαμορφώσεων επιλέχτηκε να είναι υποχρεωτικό απο όλους τους παρόχους. Επιπλέον, μερικές όχι υποχρεωτικές επιλογές διαμόρφωσης ορίστηκαν, και κάθε παροχέας μπορεί να αποφασίσει εαν το επιθυμεί να τους ενσωματώσει ή όχι. Αυτά ονομάζονται δυνατότητες.

Κάθε δυνατότητα απο ένα σύνολο συναρτήσεων τις οποίες εκθέτει το OpenNI. Ένας κόμβος παραγωγής μπορεί να ερωτηθεί εαν υποστηρίζει μια συγκεκριμένη δυνατότητα. Εάν ναι, αυτές οι συναρτήσεις μπορούν να κληθούν για τον συγκεκριμένο κόμβο.

Κάποιος μπορεί να σκεφτεί τις δυνατότητες ως επεκτάσεις στην κοινή διεπαφή.

Προς το παρόν, όλες οι δυνατότητες πρέπει να ορίζονται απο το OpenNI. Στο μέλλον, οι έμποροι μπορούν να προσθέτουν καινούριες δυνατότητες, και να τις προμηθεύουν στις εφαρμογές.

Εικόνα 29



Ενότητες (Modules)

Μια ενότητα OpenNI είναι μια διαμοιρασμένη βιβλιοθήκη (.DLL αρχείο στα Windows, .SO αρχείο στα Linux) η οποία περιέχει μια ενσωμάτωση ενός ή περισσότερων κόμβων. Μια ενότητα είναι πραγματικά ένα πρόσθετο στο OpenNI framework. Όταν μια ενότητα εγγραφεί στο OpenNI, είναι μέρος της διαδικασίας καταμέτρησης, και, βάση στην εφαρμογή, μπορεί να χρησημοποιηθεί.

Συμβατότητα προς τα πίσω

Το OpenNI διακυρήττει πλήρη συμβατότητα προς τα πίσω. Αυτο σημαίνει οτι κάθε εφαρμογή που αναπτύχθηκε στην έκδοση X του OpenNI, θα δουλέψει επίσης (χωρίς την ανάγκη για επαναμεταγλώτισση) σε κάθε μελλοντική έκδοση. Κάθε μηχανή πρέπει να έχει την τελευταία έκδοση του OpenNI (τουλάχιστο η τελευταία έκδοση απο κάθε OpenNI χρειάζεται απο κάθε εφαρμογή που είναι εγκατεστημένη σε αυτή την μηχανή). Για να επιτευχθεί αυτο, κάθε εγκατάσταση εφαρμογής πρέπει επίσης να εγκαθιστα επίσης το OpenNI.

Επιλέγοντας την C απο C++

Η διεπαφή του πυρήνα του OpenNI's ορίζεται η C. Υποθετικά, μια αμιγώς C++ διαπαφη θα ήταν ευκολότερη και ομορφότερη, αλλά προβλήματα όπως το name mangling

και το class representation στην μνήμη δεν είναι τυποποιημένα, έτσι μια C++ διεπαφή θα προκαλούσε inter-compiler προβλήματα συμβατότητας. Αυτό θα μπορούσε να οδηγήσει σε μία περίπτωση στην οποία το OpenNI θα μπορούσε να μεταγλωττιστεί σε ένα μεταγλωττιστή και μια εφαρμογή σε ένα άλλο, και όταν η εφαρμογή που ονομάζεται OpenNI's API θα μπορούσε να προκαλέσει απρόσμενα αποτελέσματα. Τα πλεονεκτήματα μιας διεπαφής C είναι ότι όλοι οι μεταγλωττιστές C χρησημοποιουν το ίδιο naming scheme, και έτσι δεν υπάρχουν προβλήματα memory representation (μια δομή ειναι απλά μια δομή για όλους τους μεταγλωττιστές, αντίθετα με μια C++ κλάση). Μια εναλλακτική σε αυτή την προσέγγιση θα ήταν η χορήγηση ξεχωριστών προμεταγλωττισμένων εκδόσεων της βιβλιοθήκης για κάθε μεταγλωττιστή, αλλά αυτο θα μπορούσε να προκαλέσει μια κατάσταση στην οποία μια επέκταση η οποία φτιάχτηκε στο OpenNI για τον μεταγλωττιστή A θα απαιτούσε ένα OpenNI που μεταγλωττίστηκε για τον μεταγλωττιστή A, και η κατάσταση αυτής της βιβλιοθήκης θα επηρεαζόταν, αλλά μια εφαρμογή που μεταγλωττίστηκε με τον μεταγλωττιστή B δεν θα επηρεαζόταν απο αυτή την αλλαγή της κατάστασης, επειδή βλέπει μια έκδοση της OpenNI βιβλιοθήκης που μεταγλωττίστηκε για τον μεταγλωττιστή B. Επιπλέον, η συντήρηση αυτών των ξεχωριστών εκδόσεων θα μπορούσε να προσθέσει πολύ overhead, και θα μπλόκαρε την χρήση απο μη υποστηριζόμενους μεταγλωττιστές.

C++ Wrappers

Επίσης η διεπαφή του OpenNI είναι πλήρως ορισμένη σε C, το OpenNI επίσης προμηθέυει με ένα σετ απο C++ wrappers, για προγραμματιστές που προτιμούν C++, προμηθεύοντας καλύτερη ασφάλεια-τύπων και καλύτερη ανάγνωση. Αυτοί C++ wrappers ορίζονται αποκλειστικά σε αρχεία επικεφαλίδων, και κάνουν χρήση όλων των OpenNI C συναρτήσεων, και τα χωρίζουν σε κλάσεις. Ο λόγος που αυτά τα wrappers είναι όλα ορισμένα σε αρχεία κεφαλείδων είναι ότι κάθε C++ μεταγλωττιστής μπορεί να διαλέξει να αντιπροσώπει κλάσεις στην μνήμη με διαφορετικό τρόπο, έτσι ώστε να αποφεύγουμε το πρόβλημα, με το να αφήνουμε κάθε project το οποίο χρησιμοποιεί OpenNI να μεταγλωττίζει τα wrappers. Κάθε μέθοδος σε αυτές τις κλάσεις ορίζεται ενσωματομένη, έτσι ώστε να μην επηρεάζεται η απόδοση. Παρακαλούμε να σημειωθέι οτι τα περισσότερα απο αυτά τα δείγματα που φτάνουν με το OpenNI είναι γραμμένα με την χρήση C++ wrappers.

7. ZDK(Zigfu Development Kit)

Εισαγωγή

Το Zigfu Development Kit είναι ο ευκολότερος τρόπος να κατασκευαστούν εφαρμογές πολλαπλής-πλατφόρμας , ελεγχόμενες με κίνηση με το Kinect/Xtion Pro σε HTML5/Javascript, Unity3D και Flash. Εφαρμογές που φτιάχνονται με το ZDK μπορούν να μεταφερθούν σε όλα τα λειτουργικά συστήματα, περιηγητές ιστού, ενδιάμεσο λογισμικό όρασης υπολογιστή, και αισθητήρες 3D.

Το ZigFu αφήνει τους προγραμματιστές να τρέξουν "απο το μηδέν σε πλήρως λετούργικές εφαρμογές κίνησης σε μερικά λεπτά," είπε ο συνιδρυτής της εταιρίας Amir Hirsch στο VentureBeat. Αυτή η εκκίνηση είναι απο τη ποιό πρόσφατη παρτίδα εταιριών απο το Y Combinator, και αυτό είναι όλο για την παραγωγή διεπαφών χειρονομιών ταχύτερα και ευκολότερα στην δημιουργία τους. Για τους προγραμματιστές, η εταιρία παρέχει bindings στην μηχανή παιχνιδιών Unity3D, και επίσης δίνει στους προγραμματιστές μια ποικιλία απο δείγματα εφαρμογών για να αρχίσουν την δική τους.

"Δεν έχουμε κυκλοφορίσει ακόμα τα στοιχεία του UI (έχουμε ακόμα να αφαιρέσουμε τις βωμολοχίες που υπάρχουν στα σχόλια του κώδικα), αλλά παίρνει περίπου ένα λεπτό να φτιαχτεί ένας πλοηγός χειρονομιών για μια λίστα απο στοιχεία," είπε ο Hirsch. "Για παράδειγμα, μπορούμε να μετατρέψουμε ένα περιηγητή καταλόγου ή μία παρουσίαση slide show σε μία εφαρμογή ελεγχόμενη απο κίνηση σε μερικά λεπτά."

Ρωτήσαμε την ομάδα του ZigFu πώς ακριβώς δουλεύει η προσφορά τους.Ο Hirsch μας είπε οτι είναι "ένα μέρος laser, δύο μέρη κάμερα και τρία μέρη μαγεία λογισμικού." Συνέχισε, "Χρησημοποιούμε το OpenNI framework για φυσική αλληλεπίδραση, η οποία παρέχει αφηρημένο πλαίσιο για την χρήση σε παρακολούθηση σκελετού ή παρακολούθηση χεριού."

Ο OpenNI είναι ένας οργανισμός ο οποίος προσπαθεί να βεβαιωθεί ότι τα αφοσιωμένα σε χειρονομίες υλικό, ενδιάμεσο λογισμικό και λογισμικό θα λειτουργούν όλα στα ίδια στάνταρντ και θα δουλεύουν καλά μαζί. Του ανοιχτού κώδικα πλαίσιο παρέχει ενα API για φυσική αλληλεπίδραση με υλικό και λογισμικό. "Ενσωματώσαμε το OpenNI στην μηχανή παιχνιδιών Unity για να παρέχουμε υψηλής ποιότητας cross-platform ανάπτυξη," είπε ο Hirsch. "Στην Unity, εκθέτουμε την παρακολούθηση σκελετού και την παρακολούθηση χεριού, και πάνω απο την παρακολούθηση χεριού, παρέχουμε στοιχεία διεπαφών χρήστη όπως μενού, λίστες και ροές."

Επιπλέον απο το OpenNI και το Unity, το ZigFu επίσης χρησημοποιεί το PrimeSense NITE, ένα ενδιάμεσο λογισμικό το οποίο επιτρέπει στους υπολογιστές να αντιλαμβάνονται ένα τρισδιάστατο κόσμο και μία τεχνολογία η οποία έχει χρησημοποιήθεί εκτεταμένα στο Kinect hacking. Φυσιολογικά, το ZigFu υποστηρίζει το πακέτο ανάπτυξης λογισμικού του Kinect, και ο Hirsch είπε ότι η αρχή είναι "δουλεύουμε με άλλους πωλητές ενδιάμεσου λογισμικού οράσεως υπολογιστή για να ενσωματώσουμε περισσοτερη λειτουργικότητα όπως αναγνώριση χειρονομιών κα προσώπου."

Χρησημοποιώντας όλα αυτά τα εργαλεία, είπε oHirsch, "Παίρνει περίπου δύο λεπτά απο δουλειά click-and-drag για να έχεις το δικό σου ελεγχόμενο απο κίνηση avatar να

λειτουργεί. Η εργασία περιλαμβάνει τη σύνδεση των ώμων, αγκώνων, γονάτων και γοφών στον 3-D χαρακτήρα σου στην έξοδο δεδομένων απο την μονάδα αναγνώρισης σκελετού. Όταν έχεις μια μηχανή παιχνιδιών όπως την Unity συνδιασμένη με τον έλεγχο avatar, οι πιθανότητες επεκτείνονται απο εκεί και πέρα."

Ο Hirsch είπε οτι πρίν αρχίσει να δουλεύει στο ZigFu, δούλευε πάνω σε εφαρμογές κίνησης για την διδαχή κινήσεων χορού και ακόμα και στρατιωτικές ασκήσεις. Σαν αποτέλεσμα, είδε δυνατότητες για ελεγχόμενες απο κίνηση, φυσικές διεπαφές στον αληθηνό κόσμο, αληθινά περιβάλλοντα εργασίας— σκέψου εκπαίδευση ελέγχου εναέριας κυκλοφορίας ή φυσική θεραπεία — Όπως στην δημιουργική απασχόληση και τα παχνίδια.

"Νομίζουμε ότι αλλάξαμε την εργασία της εργασίας της παραγωγής μιας εφαρμογής ελεγχόμενης απο κίνηση σε μία σχεδιαστική δουλειά παρά μια προγραμματιστική. Υπάρχει πολύ ενδιαφέρον σε αυτήν την τεχολογία για ψηφιακή σήμανση και διαδραστικό μαρκετινγκ όπως επίσης εξωμοίωση και προπόνηση."

Αυτή την στιγμή,το Microsoft Kinect είναι ο ποιό αποτελεσματικός τρόπος για τους προγραμματιστές να αρχίσουν με τα εργαλεία του ZigFu. Η εκκίνηση επίσης υποστηρίζει τον αισθητήρα βάθους ASUS Xtion. Όμως, είπε ο Hirsch, "Η τηλεόραση είναι το επόμενο σύνορο για την τεχνολογία αναγνώρισης κίνησης. Τίναγμα για την αλλαγή καναλιών και την περιήγηση του DVR σου είναι μόνο μια περίπτωση χρήσης. Φαντάσου να κουνάς το χέρι σου μέσα απο μια λίστα απο φίλους για να πραγματοποιήσεις μια βιντεο-κλήση. Ή φαντάσου να ψωνίζεις στην τηλεόραση σου με χειρονομίες και να χρησημοποιείς ένα εικονικό δοκιμαστήριο για να δοκιμάσεις προϊόντα. Πιστεύουμε ότι ένα οικοσύστημα απο εφαρμογές και παιχνίδια θα είναι ο καθοριστικός παράγοντας σον αγώνα των smart TV."

ZDK με Javascript

Στην εφαρμογή μας θα χρησημοποιήσουμε την javascript πλατφόρμα του ZDK. Μπορουμε να κάνουμε το website μας ελεγχόμενο απο κίνηση με λίγες γραμμές απο JavaScript και HTML. Τα ZDK JavaScript συνδέσεις δίνουν πρόσβαση στα δεδομένα βάθους, εικόνας, και σκελετού απο το Kinect/Xtion Pro όπως επίσης και έλεγχο πάνω απο τον τύπο της εμπλοκής του χρήστη και μερικές υψηλού επιπέδου αφαιρέσεις, για στοιχεία διεπαφής χρήστη. Γραμμένο σε ιδιοματική JavaScript, αλληλεπιδρά απρόσκοπτα με τα περισσότερα δημοφιλή JavaScript frameworks, συμπεριλαμβανομένων των JQuery, prototype.js, και MooTools.

Εδώ παραθετουμε ένα παράδειγμα δημιουργίας User Radar σε html αρχείο με χρήση javascript

Κώδικας 1

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <title>User Radar</title>
5. <script src='http://cdn.zigfu.com/zigjs/zig.min.js'></script>
6.
7. <script>
8.
```

```
9.      function loaded() {
10.         var radardiv = document.getElementById('radar');
11.
12.         // The radar object will respond to events from the
13.         // zig object and move the dots around accordingly.
14.         // It is also responsible for creating and destroying
15.         // the dots when users are added and removed.
16.         // Functions onuserfound(), onuserlost(), and ondataupdate()
17.         // are called by the zig object when those things happen.
18.         var radar = {
19.             onuserfound: function (user) {
20.                 var userdiv = document.createElement('div');
21.                 userdiv.className = 'user';
22.                 user.radarelement = userdiv; // add the radarelement
    property to the user object
23.                 radardiv.appendChild(user.radarelement);
24.             },
25.             onuserlost: function (user) {
26.                 radardiv.removeChild(user.radarelement);
27.             },
28.             ondataupdate: function (zigdata) {
29.                 for (var userid in zigdata.users) {
30.                     var user = zigdata.users[userid];
31.                     var pos = user.position;
32.                     var el = user.radarelement;
33.                     var parentElement = el.parentNode;
34.                     var zrange = 4000;
35.                     var xrange = 4000;
36.                     var pixelwidth = parentElement.offsetWidth;
37.                     var pixelheight = parentElement.offsetHeight;
38.                     var heightscale = pixelheight / zrange;
39.                     var widthscale = pixelwidth / xrange;
40.                     el.style.left = (((pos[0] / xrange) + 0.5) *
    pixelwidth - (el.offsetWidth / 2)) + "px";
41.                     el.style.top = ((pos[2] / zrange) * pixelheight -
    (el.offsetHeight / 2)) + "px";
42.                 }
43.             }
44.         };
45.         // By adding the radar object as a listener to the zig object,
    the zig object will now start calling
46.         // the callback functions defined in the radar object.
47.         zig.addListener(radar);
48.
49.     }
50.
51.     document.addEventListener('DOMContentLoaded', loaded, false);
52.
53. </script>
54.
55. <style>
```

```
56.     div#radar
57.     {
58.         width: 400px;
59.         height: 400px;
60.         border: 1px solid black;
61.         position: relative;
62.         overflow: hidden;
63.     }
64.     div.user
65.     {
66.         position: relative;
67.         width: 10px;
68.         height: 10px;
69.         background-color: blue;
70.     }
71. </style>
72.
73. </head>
74. <body>
75.
76. <div id='radar'></div>
77.
78. </body>
79. </html>
```

Επίσης παρακάτω παραθέτουμε ένα απόσπασμα πώς να ελέγξεις ένα στοιχείο δρομέα βασισμένο στο ZDK Cursor control:

1. Εμφάνιση και απόκρυψη του στοιχείου δρομέα όταν ενα hand session αρχίζει και τελειώνει.
2. Κίνηση του στοιχείου του δρομέα όταν πρέπει
3. Προσθέτοντας μια κλάση 'pushed' στον δρομέα όταν χρειάζεται
4. Προσομοίωση ενός γεγονότος 'click'

Κώδικας 2

```
1. // Create cursor and cursor dom element
2. var c = zig.controls.Cursor();
3. var ce = document.createElement('div');
4. ce.id = 'mycursor';
5. document.body.appendChild(ce);
6.
7. // 1. show/hide cursor on session start/end
8. zig.singleUserSession.addEventListener('sessionstart',
   function(focusPosition) {
9.     ce.style.display = 'block';
10. });
11. zig.singleUserSession.addEventListener('sessionend', function() {
```

```
12.     ce.style.display = 'none';
13. });
14. 
15. // 2. move the cursor element on cursor move
16. c.addEventListener('move', function(cursor) {
17.     ce.style.left = (c.x * window.innerWidth - (ce.offsetWidth / 2)) +
    "px";
18.     ce.style.top = (c.y * window.innerHeight - (ce.offsetHeight / 2)) +
    "px";
19. });
20. 
21. // 3. Add/remove 'pushed' class on cursor push/release
22. c.addEventListener('push', function(c) {
23.     ce.classList.add('pushed');
24. });
25. c.addEventListener('release', function(c) {
26.     ce.classList.remove('pushed');
27. });
28. 
29. // 4. Simulate mouse click on our virtual cursor
30. c.addEventListener('click', function(c) {
31.     var xpos = c.x * window.innerWidth;
32.     var ypos = c.y * window.innerHeight;
33.     var evt = document.createEvent("MouseEvents");
34.     evt.initMouseEvent("click", true, true, window, 1, xpos, ypos, xpos,
    ypos, false, false, false, false, 0, null);
35.     window.dispatchEvent(evt);
36. });
37. 
38. // Add cursor to our single user UI session
39. zig.singleUserSession.addListener(c);
```

Και φυσικά μερικό βασικό styling για τον δρομέα μας. Ας χρησημοποιήσουμε ένα μπλε τετραγωνο για τώρα, κόκκινο οταν πιέζεται:

```
1.  #mycursor {
2.     position: fixed;
3.     display : none;
4.     width : 40px;
5.     height : 40px;
6.     background-color : blue;
7.  }
8.  
9.  #mycursor.pushed {
10.    background-color: red;
11. }
```

Παρακάτω βλέπουμε ορισμένες συνταγές για javascript οι περισσότερες απο τις οποίες μπορούν να επικολληθούν κατευθείαν στην κονσόλα javascript.

Ειδοποιήσεις User found & lost

Το κύριο zig αντικείμενο εκθέτει 2 γεγονότα τα οποία μας επιτρέπουν να παρακολουθούμε όλους τους εντοπισμένους χρήστες στο πεδίο όρασης του αισθητήρα. Παρακάτω βλέπουμε 2 τρόπους να πάρουμε ειδοποιήσεις όταν τα γεγονότα userfound και userlost συμβαίνουν.

Κώδικας 3

```
1. // Method 1: specific event listeners
2. zig.addEventListener('userfound', function(user) {
3.     console.log('Found user. ID: ' + user.id);
4. });
5. zig.addEventListener('userlost', function(user) {
6.     console.log('Lost user. ID: ' + user.id);
7. });
8.
9. // Method 2: listener object
10. zig.addListener({
11.     onuserfound: function(user){
12.             console.log('Found user. ID: ' + user.id);
13.     },
14.     onuserlost: function(user){
15.             console.log('Lost user. ID: ' + user.id);
16.     }
17. });
```

Αναμονή για τον πρώτο χρήστη με πλήρη δεδομένα σκελετού σώματος

Το EngageUsersWithSkeleton μας βοηθά να διακρίνουμε μεταξύ των χρηστών που παρακολουθούνται, και ότι οι συγκεκριμένοι χρήστες έχουν τον έλεγχο. Ένας χρήστης που ελέγχει ένα παιχνίδι ή ένα UI είναι "κατειλημμένο". Υπάρχουν πολλοί τρόποι να να διαλέξουμε τον κατειλημμένο χρήστη (ή χρήστες) – παρακάτω είναι μερικές απλές περιπτώσεις χρήσης.

Κώδικας 4

```
1. var engager = zig.EngageUsersWithSkeleton(1);
2. engager.addEventListener('userengaged', function(user) {
3.     console.log('User engaged: ' + user.id);
4. });
5. engager.addEventListener('userdisengaged', function(user) {
6.     console.log('User disengaged: ' + user.id);
7. });
```

```
8. zig.addListener(engager);
```

Αναμονή για πολλαπλούς χρήστες με πλήρη δεδομένα σκελετού σώματος

Πολλαπλοί χρήστες (2 χρήστες σε αυτή την περίπτωση). Ιδανικό για παιχνίδια με χωρισμένη την οθόνη στην μέση.

Κώδικας 5

```
1. var multiEngager = zig.EngageUsersWithSkeleton(2);
2. multiEngager.addEventListener('userengaged', function(user) {
3.     console.log('User engaged: ' + user.id);
4. });
5. multiEngager.addEventListener('userdisengaged', function(user) {
6.     console.log('User disengaged: ' + user.id);
7. });
8. multiEngager.addEventListener('allusersengaged', function(users) {
9.     console.log('Engaged all users, starting game');
10.    // start game
11.});
12.zig.addListener(multiEngager);
```

Εκτύπωσε την θέση του κεφαλιού για κάθε frame για τον πρώτο κατειλημμένο χρήστη

Τα δεδομένα σκελετού μπορούν να προσεγγιστούν με το user.skeleton. Σημειώνουμε ότι τα δεδομένα του σκελετού θα είναι έγκυρα μόνο αν το user.skeletonTracked είναι true (αυτό θα είναι πάντα η περίπτωση για κατειλημμένο χρήστη απο το EngageUsersWithSkeleton). Το zig.Joints Περιέχει μια πλήρη λίστα απο πιθανές αρθρώσεις, αλλά δεν θά είναι όλες απο αυτές διαθέσιμες. Αυτό εξαρτάται απο το ενδιάμεσο λογισμικό που χρησημοποιείται απο το ZDK.

Κώδικας 6

```
1. var engager = zig.EngageUsersWithSkeleton(1);
2. engager.addEventListener('userengaged', function(user) {
3.     console.log('User engaged: ' + user.id);
4.
5.     user.addEventListener('userupdate', function(user) {
6.           console.log('Head position: ' +
   user.skeleton[zig.Joint.Head].position);
7.     });
8. });
9. engager.addEventListener('userdisengaged', function(user) {
10.    console.log('User disengaged: ' + user.id);
```

```
11. });
12. zig.addListener(engager);
```

Single user UI session

Μία κοινή περίπτωση χρήσης του ZDK είναι ένα UI το οποίο μπορέι να ελεγχθεί απο ένα χρήστη την φορά. Το αντικείμενο singleUserSession (αυτό είνα πραγματικά ένα παράδειγμα του HandSessionDetector) μας ειδοποιεί όταν ένας εντοπισμένος χρήστης προσπαθεί να ελέγξει το UI, και εξασφαλίζει ότι μόνο ένας χρήστης μπορεί να είναι στο control την φορά. Σημειώνουμε ότι αυτό σπουδαίο για εφαρμογές που είναι"Pure UI". Εάν το UI που ελέγχεται είναι ένα σύστημα μενού παιχνιδιού, μπορέι να είναι καλύτερα να διαλέξεις τον κατειλημμένο χρήστη με ένα διαφορετικό τρόπο (EngageUsersWithSkeleton, για παράδειγμα) και τότε πρόσθεσε HandSessionDetector στον κατειλημμένο χρήστη. Αυτό θα εξασφαλίσει ότι μόνο ο χρήστης ο οποίος παίζει πραγματικά το παιχνίδι μπορεί να ελεγξει τα μενού του παιχνιδιού. (Δείτε παρακάτω συνταγή)

Κώδικας 7

```
1. zig.singleUserSession.addEventListener('userengaged', function(user) {
2.     console.log('User started UI session: ' + user.id);
3. });
4. zig.singleUserSession.addEventListener('userdisengaged', function(user) {
5.     console.log('User ended UI session: ' + user.id);
6. });
7. zig.singleUserSession.addEventListener('sessionstart',
   function(initialPosition) {
8.     console.log('Session started at ' + initialPosition);
9. });
10. zig.singleUserSession.addEventListener('sessionend', function() {
11.     console.log('Session ended')
12. });
```

Μενού παιχνιδιού, ελεγχόμενο μόνο απο τον κατειλημμένο χρήστη

Κώδικας 8

```
1. // game menu. just a simple cursor for now
2. var cursor = zig.controls.Cursor();
3. var gameMenu = {
4.     onsessionstart : function(fp) { /* visualize session starting in game
   menu */ },
5.     onsessionupdate : function(p) { /* visualize session ending in game
   menu */},
6.     onattach : function(user) { user.addListener(cursor); }
7. };
8.
9. var engager = zig.EngageUsersWithSkeleton(1);
10. engager.addEventListener('userengaged', function(user) {
```

```
11.     console.log('User engaged: ' + user.id);
12.     // create our hand session detector, add the game menu to it
13.     var hsd = zig.HandSessionDetector();
14.     hsd.addListener(gameMenu)
15.     // add the session detector to the engaged user - only this user can
16.     // control the gameMenu
17.     user.addListener(hsd);
18. });
19. engager.addEventListener('userdisengaged', function(user) {
20.     console.log('User disengaged: ' + user.id);
21. });
22. zig.addListener(engager);
```

Οριζόντιος Fader με έυρος απο 0 – 100

Κώδικας 9

```
1. var f = zig.controls.Fader(zig.Orientation.X);
2. f.addEventListener('valuechange', function (f) {
3.     console.log('Fader value: ' + (f.value * 100));
4. })
5. zig.singleUserSession.addListener(f);
```

Καθετος Fader με 4 αντικείμενα

Κώδικας 10

```
1. var f = zig.controls.Fader(zig.Orientation.Y);
2. f.itemsCount = 4;
3. f.addEventListener('hoverstart', function (f) {
4.     console.log('Fader item hover: ' + f.hoverItem);
5. });
6. f.addEventListener('hoverstop', function(f) {
7.     console.log('Fader item not hovering any more: ' + f.hoverItem);
8. })
9. zig.singleUserSession.addListener(f);
```

Ανιχνευτής γεγονότων Push

Κώδικας 11

```
1. // PushDetector
2. var pushDetector = zig.controls.PushDetector();
3. pushDetector.addEventListener('push', function(pd) {
4.     console.log('PushDetector: Push');
5. });
6. pushDetector.addEventListener('release', function(pd) {
7.     console.log('PushDetector: Release');
8. });
```

```
9.  pushDetector.addEventListener('click', function(pd) {
10.     console.log('PushDetector: Click');
11. });
12. zig.singleUserSession.addListener(pushDetector);
```

Ανιχνευτής γεγονότων Swipe

Κώδικας 12

```
1.  // SwipeDetector
2.  var swipeDetector = zig.controls.SwipeDetector();
3.  swipeDetector.addEventListener('swipeup', function(pd) {
4.      console.log('SwipeDetector: Swipe Up');
5.  });
6.  swipeDetector.addEventListener('swipedown', function(pd) {
7.      console.log('SwipeDetector: Swipe Down');
8.  });
9.  swipeDetector.addEventListener('swipeleft', function(pd) {
10.     console.log('SwipeDetector: Swipe Left');
11. });
12. swipeDetector.addEventListener('swiperight', function(pd) {
13.     console.log('SwipeDetector: Swipe Right');
14. });
15. swipeDetector.addEventListener('swipe', function(dir) {
16.     console.log('SwipeDetector: Swipe direction: ' + dir);
17. });
18. zig.singleUserSession.addListener(swipeDetector);
```

Ανιχνευτής γενονότων Wave

Κώδικας 13

```
1.  // WaveDetector
2.  var waveDetector = zig.controls.WaveDetector();
3.  waveDetector.addEventListener('wave', function(pd) {
4.      console.log('WaveDetector: Wave');
5.  });
6.  zig.singleUserSession.addListener(waveDetector);
```

Ανιχνευτής γεγονότων Steady

Κώδικας 14

```
1.  // SteadyDetector
2.  var steadyDetector = zig.controls.SteadyDetector();
3.  steadyDetector.addEventListener('steady', function(sd) {
4.      console.log('SteadyDetector: Steady');
```

```
5. });
6. steadyDetector.addEventListener('unsteady', function(sd) {
7.     console.log('SteadyDetector: Unsteady');
8. });
9. zig.singleUserSession.addListener(steadyDetector);
```

Top-down "ανιχνευτής χρηστών"

Κώδικας 15

```
1. function usersRadar(parentElement) {
2.     // physical dimensions of radar in room. Lets use 4m x 4m
3.     this.radarWidth = 4000;
4.     this.radarHeight = 4000;
5.
6.     this.onuserfound = function(user) {
7.             // create a new element for this user, add to dom
8.             var el = document.createElement('div');
9.             el.classList.add('user');
10.            parentElement.appendChild(el);
11.            // we can simply add the newly created element to the tracked
    user object
12.            // for later
13.            user.radarElement = el;
14.
15.            // move the element every frame
16.            var that = this;
17.            user.addEventListener('userupdate', function(user) {
18.                    // we need to convert [user.x, user.z] to [screen.x,
    screen.y]
19.                    // first get normalized user position
20.                    var xpos = (user.position[0] / that.radarWidth) + 0.5;
    // 0 for x is actually the center of the depthmap
21.                    var ypos = (user.position[2] / that.radarHeight);
22.                    // convert normalized position to fit into our radar
    element
23.                    var el = user.radarElement;
24.                    el.style.left = xpos * parentElement.offsetWidth -
    (el.offsetWidth / 2) + "px";
25.                    el.style.top = ypos * parentElement.offsetHeight -
    (el.offsetHeight / 2) + "px";
26.            });
27.    }
28.
29.    this.onuserlost = function(user) {
30.            // remove the element we created from the dom and ZDK user
    object
31.            parentElement.removeChild(user.radarElement);
32.            delete user.radarElement;
33.    }
```

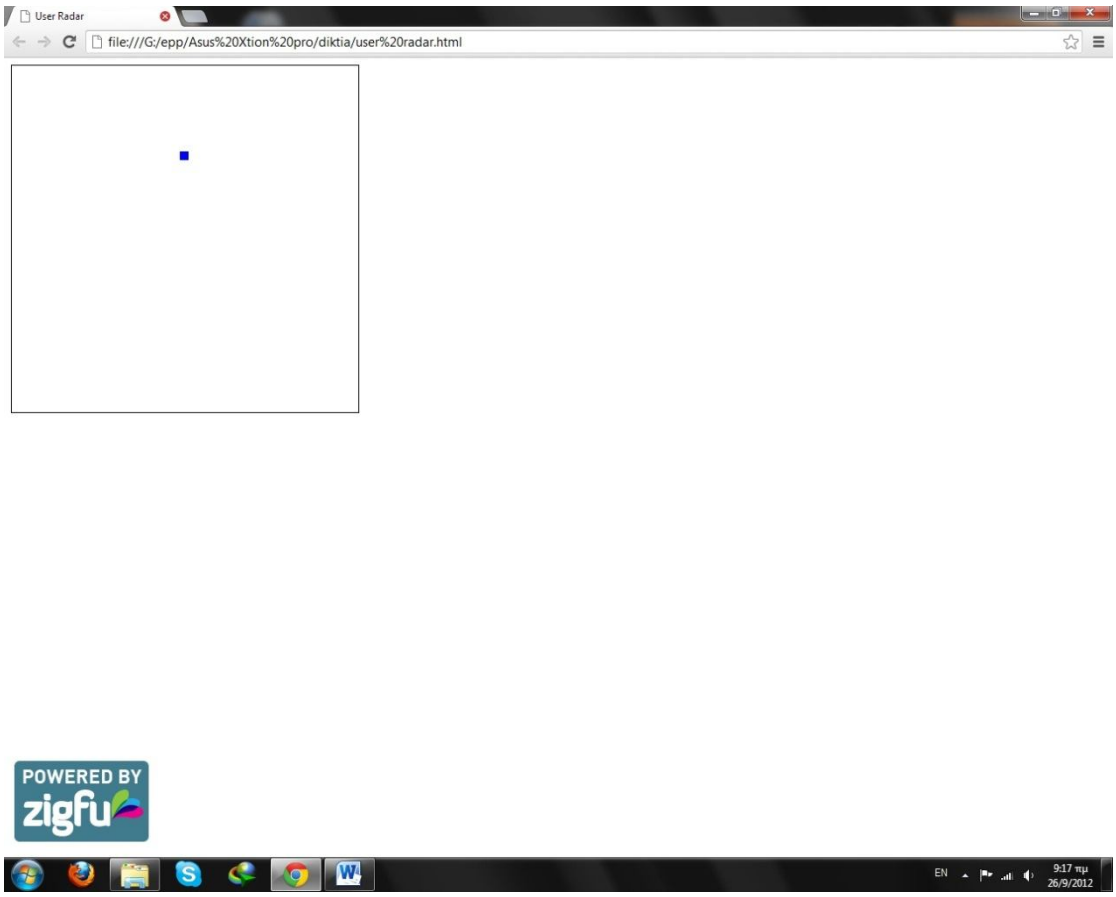
```
34. }
35.
36. // make sure you add <div id='radar'></div> to your html body
37. var radar = new usersRadar(document.getElementById('radar'));
38. zig.addListener(radar);
```

Και το style sheet για το ρανταρ

```
1.  #radar {
2.      width:        300px;
3.      height: 300px;
4.      border-radius: 15px;
5.      bottom: 20px;
6.      right: 20px;
7.      background:url('http://cdn.zigfu.com/zigjs/toys/radar.png');
8.      overflow:hidden;
9.      position: fixed;
10. }
11.
12. #radar .user {
13.     width: 35px;
14.     height: 35px;
15.     background-image: url('http://cdn.zigfu.com/zigjs/toys/user.png');
16.     position:relative;
17. }
```

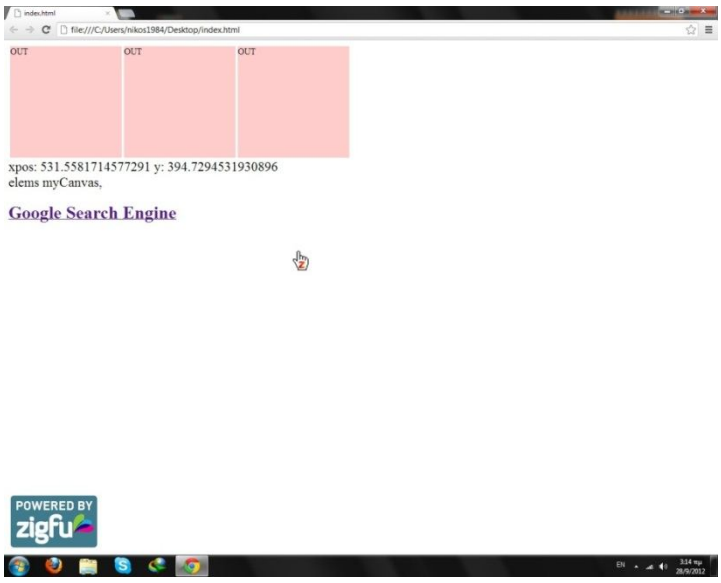
Εδω βλέπουμε το αποτέλεσμα του user radar στον περιηγητήμας:

Εικόνα 30



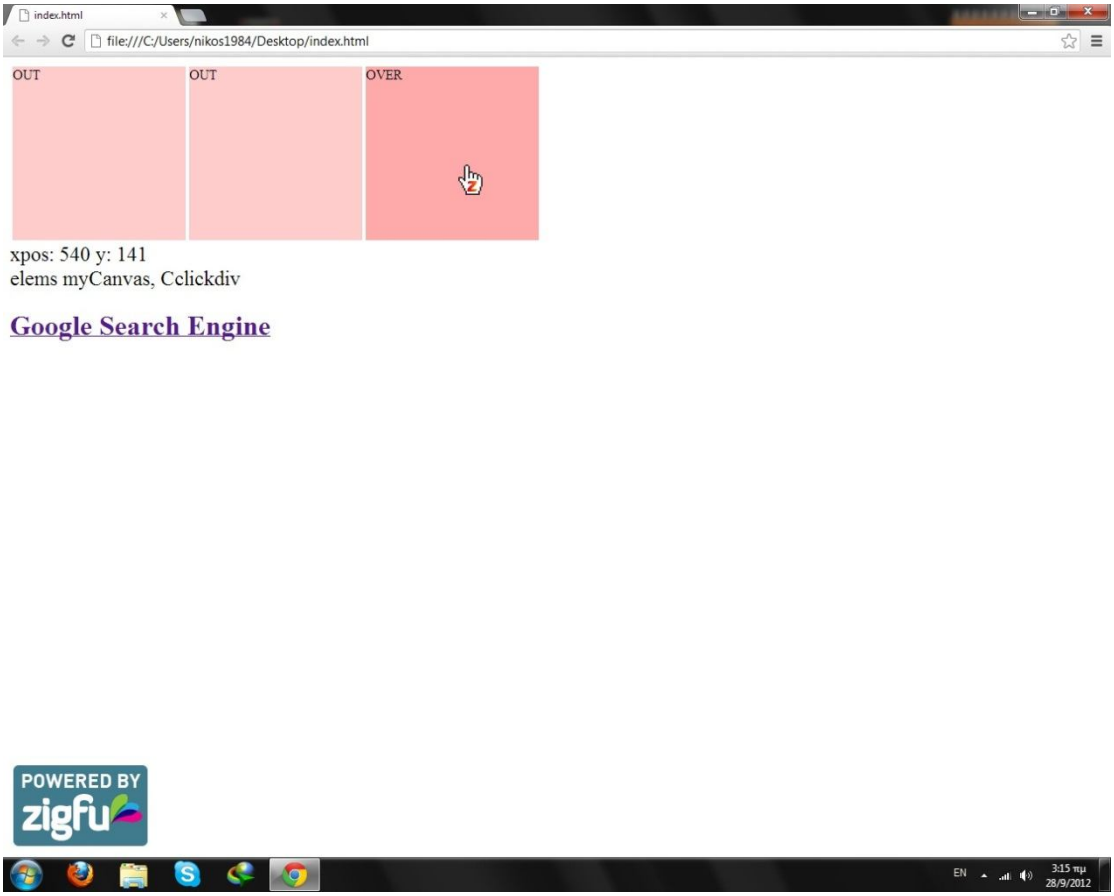
Μετά αφου εφαρμόσουμε το javascript στην ιστοσελίδα μας πέρνουμε το παρακάτω αποτέλεσμα:

Εικόνα 31

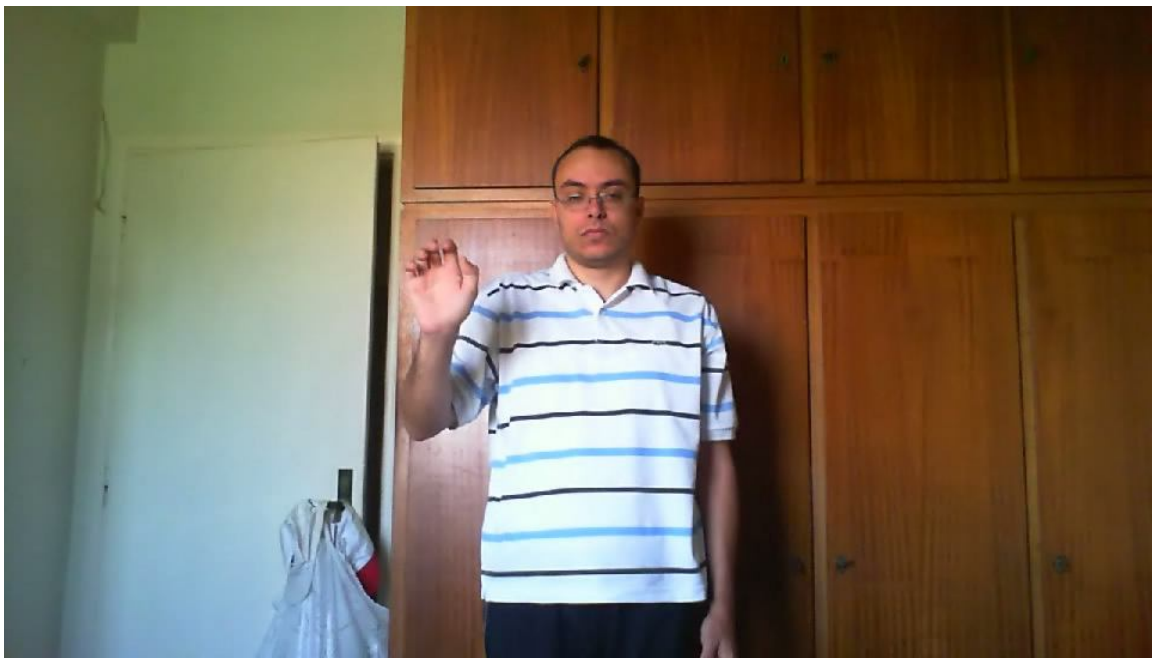


Εδώ βλεπουμε τον δείκτη και την κίνηση του σε συνάρτηση με τον χρήστη.

Εικόνα 32

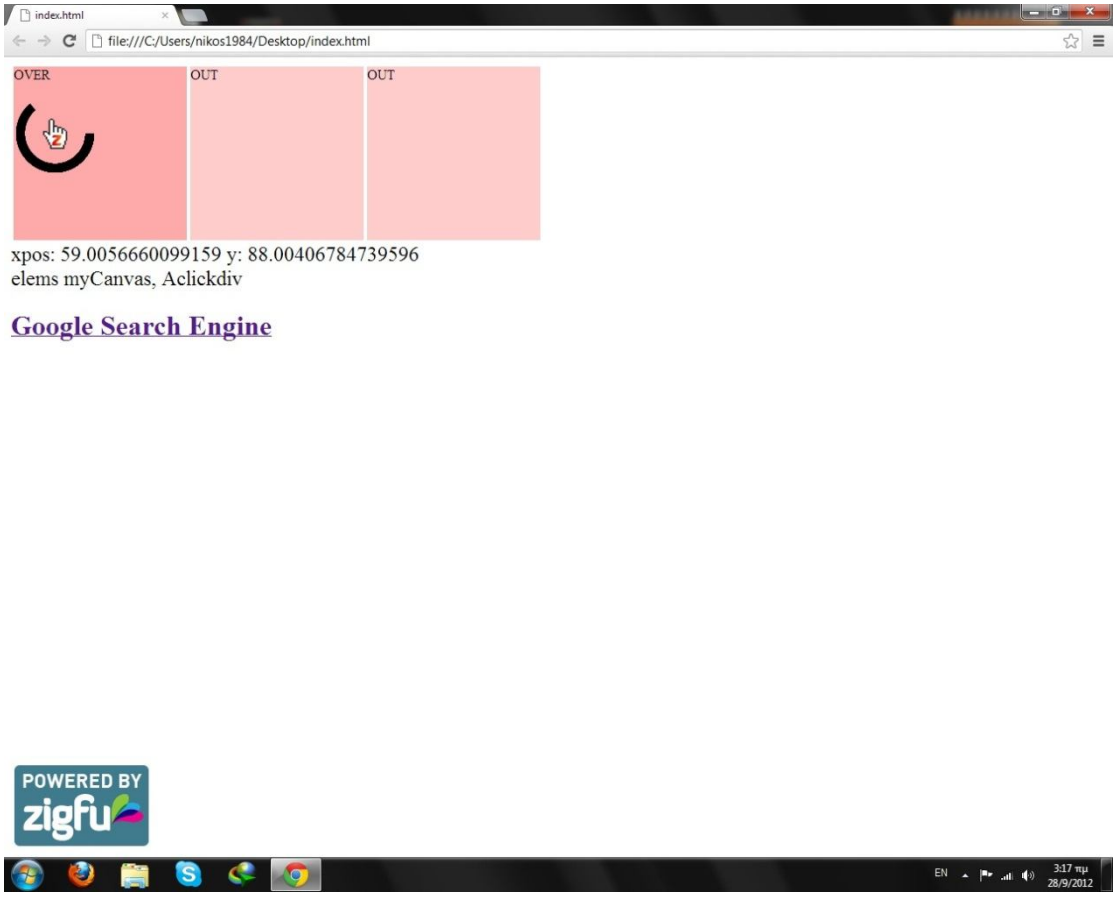


Εικόνα 33



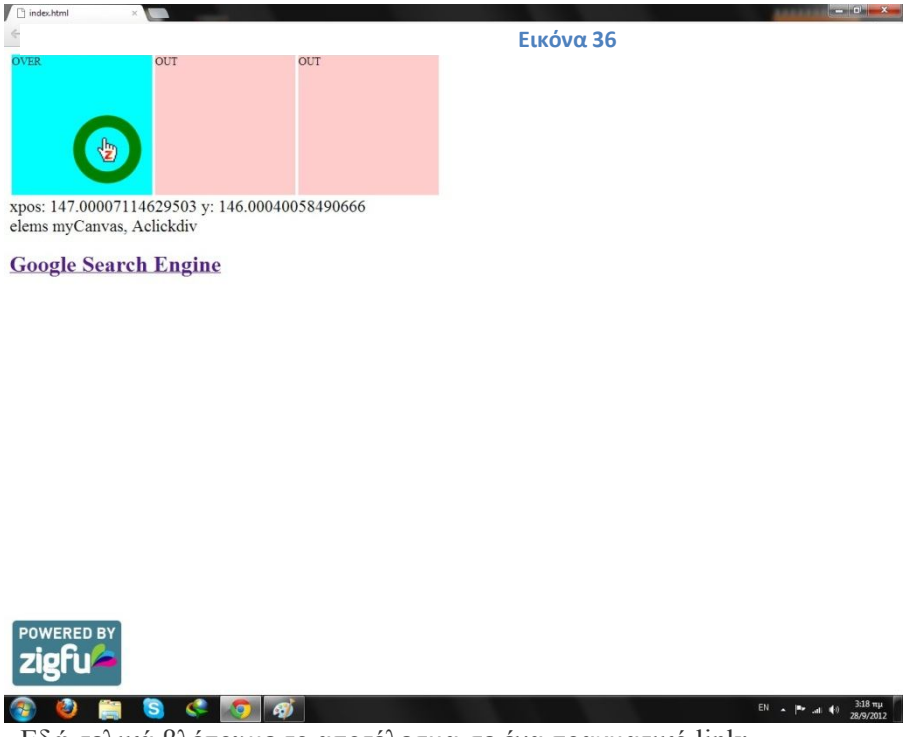
Εδώ βλέπουμε το γέμισμα του hover timer που βάλαμε για να γίνει το κλικ:

Εικόνα 34



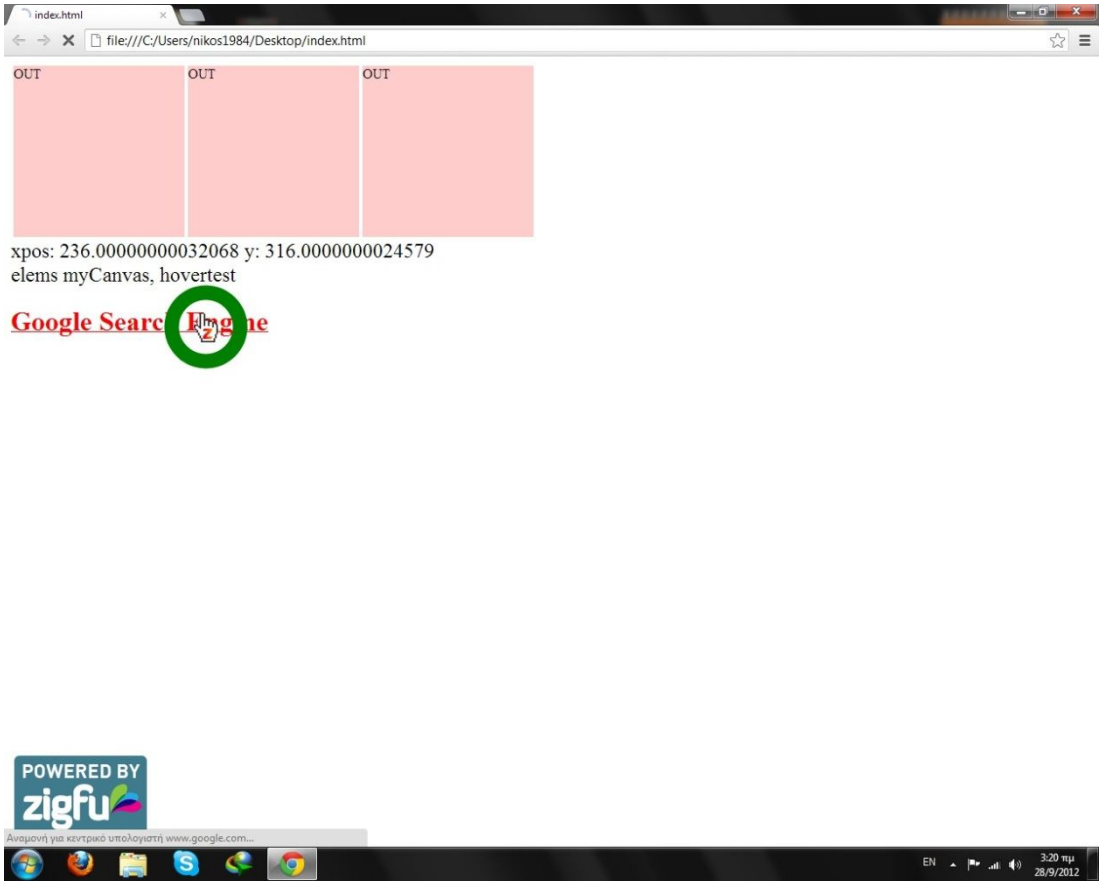
Και εδώ το αποτέλεσμα όταν γεμίσει ο timer

Εικόνα 35

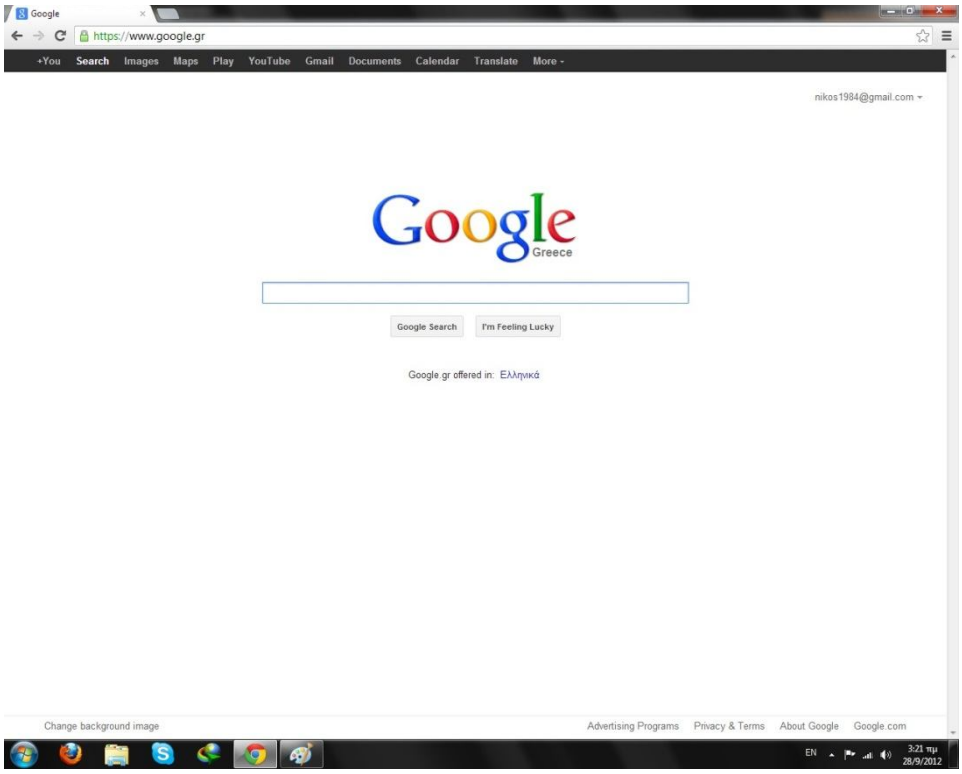


Εικόνα 36

Εδώ τελικά βλέπουμε το αποτέλεσμα σε ένα πραγματικό link:



Εικόνα 37



Παρακάτω ακολουθεί ο κώδικας που χρησμοποιήσαμε στην ιστοσελίδα μας:

```
<html>

<head>

<script src='zig.js'></script>

<style>

#cursor {

position: fixed;

display: block;

width : 100px;

height : 100px;

background-image: url("zigcursor.png");

background-position: center;

background-repeat: no-repeat;

}

#output

{

font-size: 24px;

}

.clickdiv {

width: 200px;

height: 200px;

background-color: #FFCCCC;

}

a:hover
```

```
{

color: #FF0000;

}

</style>

</head>

<body>

<div id="cursor"><canvas id="myCanvas" width="100" height="100"
style="width:100px;height:100px"></canvas></div>

<table><tr>

<td><div id="Aclickdiv" class="hoverable clickdiv" onclick="divclick(this);"
onmousedown="divdown(this);" onmouseup="divup(this);" onmouseover="divover(this);"
onmouseout="divout(this);">A</div></td>

<td><div id="Bclickdiv" class="clickdiv" onclick="divclick(this);"
onmousedown="divdown(this);" onmouseup="divup(this);" onmouseover="divover(this);"
onmouseout="divout(this);">B</div></td>

<td><div id="Cclickdiv" class="clickdiv" onclick="divclick(this);"
onmousedown="divdown(this);" onmouseup="divup(this);" onmouseover="divover(this);"
onmouseout="divout(this);">C</div></td>

</table>

<div id="output">OUT</div>

<div><h1><a id="hovertest" class="hoverable" href="http://google.com">Google Search
Engine</a></h1></div>

<script type="text/javascript">

function hasClass(ele,cls) {

return ele.className.match(new RegExp('(\\s|^)'+cls+'(\\s|$)'));

}
```

```
function addClass(ele,cls) {
if (!this.hasClass(ele,cls)) ele.className += " "+cls;
}
function removeClass(ele,cls) {
if (hasClass(ele,cls)) {
var reg = new RegExp('(\\s|^)'+cls+'(\\s|$)');
ele.className=ele.className.replace(reg,'');
}
}
function divdown(t)
{
console.log("down: " + t.id);
t.innerHTML = "DOWN";
t.style.backgroundColor = '#AAFFAA';
}
function divup(t)
{
console.log("up: " + t.id);
t.innerHTML = "UP";
t.style.backgroundColor = '#FFAAAA';
}
function divover(t)
{
console.log("divover: " + t.id);
```

```
t.innerHTML = "OVER";
t.style.backgroundColor = '#FFAAAA';
}
function divout(t)
{
console.log("divout: " + t.id);
t.innerHTML = "OUT";
t.style.backgroundColor = '#FFCCCC';
}
function divclick(t)
{
console.log("click: " + t.id);
t.style.backgroundColor = '#00FFFF';
setTimeout(function(){t.style.backgroundColor='#FFCCCC';}, 1000);
}
var context, canvas, centerX, centerY, radius, counterclockwise;
window.onload = function(){
canvas = document.getElementById("myCanvas");
context = canvas.getContext("2d");
centerX = canvas.width / 2;
centerY = canvas.height / 2;
radius = 40;
counterclockwise = false;
context.strokeStyle = "black"; // line color
```

```
context.lineWidth = 10;

window.onmousemove = function(e){

target[0] = e.clientX / window.innerWidth;

target[1] = e.clientY / window.innerHeight;

};

window.setInterval(moveCursor,15);

}

function fillArc(start,total)

{

//console.log("start: " + start + " total: " + total);

context.arc(centerX, centerY, radius, start*(3.1415926/180),
(start+total)*(3.1415926/180), counterclockwise);

context.stroke();

}

var pos = 0;

var del = 20;

var amt = 10;

var endHover = false;

function clear()

{

endHover = true;

canvas.width = canvas.width;

}

function startHoverClick()
```

```
{
console.log("START HOVER CLICK " + mouseovertarget.id);
pos = 0;
endHover = false;
fillCircle();
}
function fillCircle()
{
if (!endHover)
{
context.lineWidth = 10;
fillArc(pos,amt);
pos += amt;
if (pos < 360)
{
window.setTimeout(fillCircle,del);
}
else
{
pos = 0;
window.setTimeout(end,del);
}
}
}
```

```
function end()

{

console.log("HOVER CLICK " + mouseovertarget.id);

context.lineWidth = 17;

context.strokeStyle = "green"; // line color

fillArc(0,360);

xpos = c.x * window.innerWidth;

ypos = c.y * window.innerHeight;

var evt = document.createEvent("MouseEvents");

evt.initMouseEvent("click", true, true, window, 1, xpos, ypos, xpos, ypos, false, false, false,
false, 0, null);

mouseovertarget.dispatchEvent(evt);

}


function vlerp(p1,p2,r)

{

out = [];

for (i=0;i<p1.length;i++)

{

out.push(p2[i]*r+p1[i]*(1-r));

}

return out;

}

// Create cursor and cursor dom element

var c = zig.controls.Cursor();
```

```
var ce = document.getElementById("cursor")

// 1. show/hide cursor on session start/end

zig.singleUserSession.addEventListener('sessionstart', function(focusPosition) {

ce.style.display = 'block';

});

zig.singleUserSession.addEventListener('sessionend', function() {

ce.style.display = 'block';

});

// 2. move the cursor element on cursor move

c.addEventListener('move', function(cursor) {

target[0] = c.x;

target[1] = c.y;

});

var last = [0,0];

var target = [0,0];

function moveCursor()

{

//console.log(target);

last = vlerp(last,target,.3);

setCursor(last[0],last[1]);

}

var mouseovertarget = null;

function setCursor(x,y)

{
```

```
xpos = x * window.innerWidth;

ypos = y * window.innerHeight;

ce.style.left = xpos - (ce.offsetWidth / 2); "px";

ce.style.top = ypos - (ce.offsetHeight / 2); "px";

elem = document.elementFromPoint(xpos,ypos);

output = document.getElementById("output");

if (elem)

{

var r = elem;

var n=elem, el=n, elems=[];

k=0;

do {

elems.push(el);

if (el.id == "myCanvas")

{

//console.log("decrease canvas z");

//console.log(el.parentNode.id)

el.parentNode.style.zIndex = parseInt(el.parentNode.style.zIndex||0)-(1000+k);

}

else

{

el.style.zIndex = parseInt(el.style.zIndex||0)-(1000+k);

//var evt = document.createEvent("MouseEvents");

//evt.initMouseEvent("mousemove", true, true, window, 1, xpos, ypos, xpos, ypos, false,
```

```
false, false, false, 0, null);

//el.dispatchEvent(evt);

if (el != mouseovertarget)

{

clear();

if (mouseovertarget)

{

console.log("CLASS: " + mouseovertarget.classList);

//mouseovertarget.classList.remove("zigHover");

removeClass(mouseovertarget, "zigHover");

console.log("REMOVE zigHover class from " + mouseovertarget.id );

console.log("CLASS: " + mouseovertarget.classList);

var evt = document.createEvent("MouseEvents");

evt.initMouseEvent("mouseout", true, true, window, 1, xpos, ypos, xpos, ypos, false, false,
false, false, 0, null);

console.log("mouseout: " + mouseovertarget.id);

mouseovertarget.dispatchEvent(evt);

}

mouseovertarget = el;

console.log("mouseover: " + mouseovertarget.id);

var evt = document.createEvent("MouseEvents");

evt.initMouseEvent("mouseover", true, true, window, 1, xpos, ypos, xpos, ypos, false,
false, false, false, 0, null);

mouseovertarget.dispatchEvent(evt);

//mouseovertarget.classList.add("zigHover");
```

```
addClass(mouseovertarget, "zigHover");

console.log("ADD zigHover class to " + mouseovertarget.id );

if (hasClass(mouseovertarget, "hoverable"))

{

startHoverClick();

}

}

}

//console.log("next");

r = el;

el = document.elementFromPoint(xpos, ypos);

//console.log(el);//console.log(el.id);

k++;

} while(el!=n && el!=r && k<2);

//console.log(elems.length);

elems.toString=function(){

var buf=[];

for(var ii=0; ii<this.length; ii++)

buf.push(this[ii].id);

return buf.join(", ");

}

//console.log(elems);

for(var ii=0; ii<elems.length; ii++)

{
```

```
if (elems[ii].id == "myCanvas")

{

//console.log("increase canvas z: " + elems[ii].parentNode.style.zIndex);

elems[ii].parentNode.style.zIndex = parseInt(elems[ii].parentNode.style.zIndex) +
(1000+ii);

}

else

{

elems[ii].style.zIndex = parseInt(elems[ii].style.zIndex) + (1000+ii);

}

}

output.innerHTML = "xpos: " + xpos + " y: " + ypos + "<br>elems " + elems;

}

}

// 3. Add/remove 'pushed' class on cursor push/release

c.addEventListener('push', function(c) {

//console.log("mousedown: " + mouseovertarget.id);

addClass(ce,'pushed');

xpos = c.x * window.innerWidth;

ypos = c.y * window.innerHeight;

var evt = document.createEvent("MouseEvents");

evt.initMouseEvent("mousedown", true, true, window, 1, xpos, ypos, xpos, ypos, false,
false, false, false, 0, null);

mouseovertarget.dispatchEvent(evt);
```

```
});

c.addEventListener('release', function(c) {

removeClass(ce,'pushed');

//console.log("mouseup: " + mouseovertarget.id);

xpos = c.x * window.innerWidth;

ypos = c.y * window.innerHeight;

var evt = document.createEvent("MouseEvents");

evt.initMouseEvent("mouseup", true, true, window, 1, xpos, ypos, xpos, ypos, false, false,
false, false, 0, null);

mouseovertarget.dispatchEvent(evt);

});

// 4. Simulate mouse click on our virtual cursor

c.addEventListener('click', function(c) {

var xpos = c.x * window.innerWidth;

var ypos = c.y * window.innerHeight;

var evt = document.createEvent("MouseEvents");

elem = document.elementFromPoint(xpos,ypos);

//console.log("xpos: " + xpos + " y: " + ypos);

if (elem)

{

var r = elem;

var n=elem, el=n, elems=[];

k=0;

do {
```

```
elems.push(el);

if (el.id == "myCanvas")

{

//console.log("decrease canvas z");

//console.log(el.parentNode.id)

el.parentNode.style.zIndex = parseInt(el.parentNode.style.zIndex||0)-(1000+k);

}

else

{

//console.log("click");

evt.initMouseEvent("click", true, true, el, 1, xpos, ypos, xpos, ypos, false, false, false, false,
0, null);

el.dispatchEvent(evt);

el.style.zIndex = parseInt(el.style.zIndex||0)-(1000+k);

}

//console.log("next");

r = el;

el = document.elementFromPoint(xpos, ypos);

//console.log(el);//console.log(el.id);

k++;

} while(el!=n && el!=r && k<3);

//console.log(elems.length);

elems.toString=function(){

var buf=[];
```

```
for(var ii=0; ii<this.length; ii++)

buf.push(this[ii].id);

return buf.join(", ");

}

//console.log(elems);

for(var ii=0; ii<elems.length; ii++)

{

if (elems[ii].id == "myCanvas")

{

//console.log("increase canvas z: " + elems[ii].parentNode.style.zIndex);

elems[ii].parentNode.style.zIndex = parseInt(elems[ii].parentNode.style.zIndex) +
(1000+ii);

}

else

{

elems[ii].style.zIndex = parseInt(elems[ii].style.zIndex) + (1000+ii);

}

}

}


});

// Add cursor to our single user UI session

zig.singleUserSession.addListener(c);



</script>
```

```
<script type="text/javascript">
for (var i = 0; i < document.styleSheets.length; i++)
{
console.log("stylesheet " + i);
ss = document.styleSheets[i];
for (var r = 0; r < ss.rules.length; r++)
{
console.log(ss.rules[r].selectorText);
if (ss.rules[r].selectorText.search(":hover") > 0)
{
console.log("add hover");
txt = ss.rules[r].cssText;
newstyle = txt.replace(":hover", ".zigHover");
console.log(newstyle);
ss.insertRule(newstyle);
r++; //make sure we don't do it for the same rule twice after inserting
}
}
}
</script>
</body>
</html>
```

8 Συμπεράσματα

Σε αυτή την εργασία παρουσιάσαμε μία εφαρμογή για τον έλεγχο της περιήγησης ιστοσελίδων μέσω της πολυ-τροπικής συσκευής εισόδου Asus Xtion Pro.

Περιγράψαμε και αναλύσαμε σύντομα τα Natural User Interfaces. Και μέτα παρουσιάσαμε σύντομες περιγραφές των συσκευών πολυ-τροπικής εισόδου που υπάρχουν σήμερα. Ακόμα γνωρίσαμε τη ιστορία της κατασκεύαστριας εταιρίας της συσκευής μας και είδαμε εφαρμογές πέρα πο την κύρια χρήση των συσκευών. Επίσης είδαμε την περιγραφή του API OpenNi και του ZDK που χρησημοποιήσαμε για να ενσωματόσουμε τον πολυτροπικό έλεγχο στο site μας.

Μελλοντικά αυτο θα μπορούσε να επεκταθέι σε ακριβέστερο έλεγχο τω ιστοσελίδων με την εισαγωγή ποιό περίπλοκων κινήσεων για την άυξηση της διαδραστικότητας με την ιστοσελίδα.

Βιβλιογραφία

1. http://www.wikipedia.com
2. http://www.asus.com/Multimedia/Motion_Sensor/Xtion_PRO/
3. http://www.openni.org/
4. https://leapmotion.com/
5. http://zigfu.com/
6. http://www.microsoft.com/en-us/kinectforwindows/
7. http://venturebeat.com/2011/08/24/zigfu/

Παράρτημα

1 Εγκατάσταση συσκευής/OpenNi

Εγκάσταση

Συνδέοντας την συσκευή

1. Συνδέσε τον αισθητήρα Xtion PRO(Xtion PRO Live) στον υπολογιστή σας μέσω USB καλωδίου.
2. Βάλτε το DVD υποστήριξης στο οπτικό μέσο. Η διεπαφή του οδηγού εμφανίζεται στην οθόνη αυτόματα. Κάντε κλικ στην καρτέλα Installation, τότε το SDK αρχίζει να εγκαθίσταται.
3. Παρακαλώ τρέξτε το αρχείο <Samples\Bin\Release\NiViewer.exe> μετά την εγκατάσταση.
   Η βασικό μονοπάτι είναι <C:\Program Files\OpenNI\Samples\Bin\Release\NiViewer.exe>.Το SDK έχει εγκατασταθεί επιτυχώς όταν ο χάρτης βάθους φαινεται στην αριστερή γωνία της οθόνη. (και η εικόνα χρώματος φαίνεται στην δεξια πλευρά της οθόνης(για το Xtion Pro Live)).
4. Το δείγμα κώδικα περιλαμβάνεται στο SDK για τιν αναφορά σας. Μπόρείς να βρείς δείγματα κώδικα στα παρακάτω μονοπάτια:
   a) OpenNI : The default path is: <C:\Program Files\OpenNI\Samples>.
   b) NITE: The default path is: <C:\Program Files\Prime Sense\NITE\Samples>.

2 Εγκατάσταση Zigfu Browser plugin

Για την εγκατάστασταση του browser plugin για να λειτουργήσει η εφαρμογή στον υπολογιστή μας πρέπει να περιηγιθούμε στην παρακάτω διέυθυνση:
http://zigfu.com/en/downloads/browserplugin/