ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Πλοήγηση στο Διαδίκτυο με τη Διαδραστική Κάμερα Kinect

Θεόδωρος Π. Κοτσώνης

Επιβλέπων: **Παναγιώτης Σταματόπουλος,** Επίκουρος Καθηγητής

ΑΘΗΝΑ

ΑΥΓΟΥΣΤΟΣ 2014

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Πλοήγηση στο Διαδίκτυο με τη Διαδραστική Κάμερα Kinect

Θεόδωρος Π. Κοτσώνης
Α.Μ.: 1115200700234

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Παναγιώτης Σταματόπουλος,** Επίκουρος Καθηγητής

ΠΕΡΙΛΗΨΗ

Σκοπός αυτής της πτυχιακής είναι η πλοήγηση του ανθρώπου στο διαδίκτυο με σύγχρονες μεθόδους τεχνολογίας, μέσω του υπολογιστή του.

Κατάφερα να δημιουργήσω μια εφαρμογή η οποία χρησιμοποιεί τη κάμερα Kinect, ως μέσο πλοήγησης για να πετύχουμε τον στόχο μας.

Στο κείμενο που σας παραθέτω, θα μιλήσουμε για το Kinect και θα εξηγήσουμε πλήρως όλα τα μέρη αυτής της εφαρμογής.

Πιο συγκεκριμένα, στη πρώτη ενότητα της πτυχιακής, θα δούμε μια μικρή εισαγωγή για το Kinect. Θα συνεχίσουμε με μια περιγραφή για όλα τα τεχνικά χαρακτηριστικά που διαθέτει (δεύτερη ενότητα), για να ακολουθήσει η επεξήγηση της εφαρμογής όπου εκεί θα βάλουμε τον χρήστη σε ρόλο χειριστή για να εμπεδώσει όλες τις πτυχές τους προγράμματος (τρίτη ενότητα). Στην τέταρτη ενότητα, γίνεται μια πλήρη ανάλυση του κώδικα της εφαρμογής. Εκεί, θα εξηγήσουμε κομμάτι – κομμάτι τον κώδικα και θα τονίσουμε τα σημαντικότερα μέρη του, δίνοντας τις απαραίτητες διευκρινίσεις. Ακόμη θα μιλήσουμε και για όλες τις πιθανές επεκτάσεις που θα μπορούσε να κάνει κάποιος στο μέλλον (πέμπτη ενότητα). Κλείνοντας, μένουν δύο τελευταίες ενότητες. Στη έκτη, όλες οι πηγές πληροφορίας που αναζητήσαμε για την επίτευξη της πτυχιακής είναι εδώ, με στόχο να βοηθήσουμε όποιον θα ήθελε να ασχοληθεί με αυτή και τέλος, στην έβδομη παραθέτουμε τα συμπεράσματα που βγάλαμε από την εργασία και την ενασχόληση μας με το Kinect.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Διαδίκτυο

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Kinect, Αναγνώριση Φωνής, Διαχείριση από Απόσταση

ABSTRACT

The purpose of this qualifying diploma thesis is the navigation on the internet using technologies through our computer.

I manage to create an application which use Kinect camera as a navigation tool to achieve our goal. In this documentation, we discuss about Kinect and I will explain all the parts of this program.

Specifically, in the first part of this thesis, there are some introductory informations about Kinect. Continuing, there is a description of its technical features (part 2) followed by a full explanation of the application where the user can fully understand all its aspects (part 3). In part number 4, there is an explanation of the code used in the app. The code is fully analyzed, the most important parts are highlighted and the necessary explanations are given. Finally in the last two parts (6 and 7), all the sources of information that were used in the thesis are provided to anyone who wish to learn more about the subject and all the conclusions are available.

SUBJECT AREA: Internet navigation

KEYWORDS: Kinect, Voice Recognition, Manage from Distance

Η εργασία είναι αφιερωμένη

σε όλους εκείνους που προσπαθώντας για το αδύνατο,

κατορθώνουν το καλύτερο δυνατό

ΕΥΧΑΡΙΣΤΙΕΣ

Πέρα από την αναζήτηση του θέματος, ήθελα να συνεργαστώ και με έναν καθηγητή, ο οποίος μέσα από αυτή τη διαδρομή, θα μου μάθαινε για νέες τεχνολογίες, έχοντας μια άψογη συνεργασία. Όλα αυτά δεν θα είχαν γίνει χωρίς τη βοήθεια του επιβλέποντα, επίκουρο καθηγητή Παναγιώτη Σταματόπουλο ο οποίος με την πολύτιμη συμβολή του, συνέβαλε στην ολοκλήρωση της εργασίας μου.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΠΗΓΑΙΟΥ ΚΩΔΙΚΑ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΠΡΟΛΟΓΟΣ

Ένας κύκλος σπουδών έφτασε στο τέλος του. Ολοκληρώνοντας την πτυχιακή μου εργασία, κατάφερα να φτιάξω μια εφαρμογή που ο καθένας θα μπορεί να εξερευνήσει το μεγάλο κόσμο του διαδικτύου, με τη βοήθεια νέων εξελισσόμενων τεχνολογιών όπως είναι η κάμερα του Kinect. Μέσα από αυτή την συνεχή αλληλεπίδραση του χρήστη με τη κάμερα, θα μπορέσει να πλοηγηθεί στις αγαπημένες του διευθύνσεις, να ψυχαγωγηθεί, να μάθει και να καταλάβει στο μέγιστο τις δυνατότητες του διαδικτύου.

Είχα την ευκαιρία να μάθω όλα τα χαρακτηριστικά αυτής της κάμερας, το API που μας δίνει η Microsoft αλλά και τεχνικές προγραμματισμού για την σωστή διαχείριση των πολυμέσων που διαθέτει, σε συνδυασμό με μια ασφαλή πλοήγηση στο διαδίκτυο.

Θα ήταν παράβλεψη από μέρος μου, αν δεν ευχαριστούσα τη Microsoft Hellas για την δωρεάν διάθεση της κάμερας Kinect για εκπαιδευτικούς και ερευνητικούς σκοπούς.

Σε αυτή την εργασία, θα συζητήσουμε για όλα τα μέρη της κάμερας Kinect και πως αυτά παίζουν σημαντικό ρόλο στην εφαρμογή. Θα εξηγήσουμε πως λειτουργεί η εφαρμογή, θα δούμε όλες τις συναρτήσεις του προγράμματος αλλά και θα αναλύσουμε μέρη του κώδικα που κρίνουμε σημαντικά. Τέλος θα αναφέρουμε κάποιες δυνατές επεκτάσεις της εφαρμογής, τις πηγές που βασιστήκαμε αλλά και τα τελικά συμπεράσματα που προκύπτουν από τη δημιουργία της.

1. ΕΙΣΑΓΩΓΗ

Το έτος 2010, η Microsoft παρουσίασε τη συσκευή Kinect για το Xbox 360, που επιτρέπει στους χρήστες να παίζουν παιχνίδια, κουνώντας μέρη του σώμα τους. Το Kinect κάνει τη χρήση των κλασικών χειριστηρίων μη απαραίτητη, δίνοντας στους χειριστές του την δυνατότητα να προσαρμόζονται το περιβάλλον του παιχνιδιού. Το Kinect χρησιμοποιεί ένα sensor για να ανιχνεύει την κίνηση του χρήστη και τη μετατρέπει σε εντολές στο παιχνίδι καθώς και μικρόφωνα, για να δίνουμε τις φωνητικές εντολές που επιθυμούμε.

Φυσικά η Microsoft δεν έμεινε μόνο στις παιχνιδομηχανές, και έκανε συμβατή τη συσκευή και στους υπολογιστές. Για αυτό το λόγο, η εταιρία προχώρησε στην δημιουργία ενός Application Programming Interface (API) και έδωσε την ευχέρεια στους χρήστες να χρησιμοποιούν το Kinect. Τόσο η κάμερα όσο και τα μικρόφωνα δέχονται κινήσεις και φωνητικές εντολές του χρήστη αντίστοιχα, δίνοντας ένα φανταστικό αποτέλεσμα.

Στόχος της πτυχιακής είναι αρχικά να πετύχουμε την επικοινωνία του ανθρώπου με τον υπολογιστή και μέσα από αυτή την αλληλεπίδραση να του δώσουμε την δυνατότητα πλοηγηθεί στο διαδίκτυο, επιλέγοντας τις διευθύνσεις που επιθυμεί, κάνοντας την πλοήγηση του πιο εύκολη και πιο διασκεδαστική.

Στην επόμενη ενότητα θα δούμε με λεπτομέρειες τα τεχνικά χαρακτηριστικά της κάμερας Kinect και θα αναλύσουμε όλες τις χρήσεις της.

Εικόνα 1.1: Η κάμερα Kinect

2. ΤΕΧΝΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ

2.1 Γενικά

Το Kinect είναι μια συσκευή εισόδου η οποία ανιχνεύει την κίνηση και δημιουργήθηκε από τη εταιρία Microsoft για τις γνωστές κονσόλες βιντεοπαιχνιδιών Xbox 360 και το Xbox One. Η πρωτοποριακή τεχνολογία πίσω από το Kinect είναι ένας συνδυασμός υλικού και λογισμικού που περιέχονται στον αισθητήρα του Kinect. Η κάμερα που υπάρχει στη συσκευή, επιτρέπει στους χρήστες να ελέγχουν και να αλληλεπιδρούν με το Kinect, χωρίς την ανάγκη χρήσης κάποιου χειριστηρίου.

Μπορεί να αναγνωρίσει μέχρι έξι παίκτες αλλά στις περισσότερες εφαρμογές το Kinect αναγνωρίζει μέχρι δυο. Υπάρχει θύρα η οποία προσφέρει παροχή ρεύματος και συνδέεται με επιτυχία σε υπολογιστές. Διάφορες εφαρμογές έχουν αναπτυχθεί για να επιτρέψουν στο Kinect να ελέγχει ορισμένες πτυχές ενός ηλεκτρονικού υπολογιστή. Η δημιουργία 3D εικόνων, η καταγραφή βίντεο και η χρησιμότητα σε συστήματα ασφαλείας είναι μόνο λίγες από τις εφαρμογές που έχει φανεί χρήσιμη η κάμερα.

Σίγουρα κάποιος μπορεί να ενθουσιαστεί βλέποντας τη συσκευή εξωτερικά αλλά ήρθε η ώρα να δούμε πιο αναλυτικά τα τεχνικά της χαρακτηριστικά.

2.2 Sensor Kinect

Το Kinect αποτελείται αναλυτικά από τα εξής:

- Μια color VGA video camera
- Δυο 3D Depth sensors
- Multi-array microphone
- Μηχανισμός εναλλαγής γωνίας Θέασης

Εικόνα 2.1: Τα μέρη του Kinect

2.2.1 Color VGA Video Camera

Πρόκειται για μια τύπου RGB κάμερα. Ονομάζεται RGB γιατί αναγνωρίζει τρία χρώματα: το κόκκινο, το πράσινο και το μπλε. Αντιπροσωπεύει την κλασική κάμερα που όλοι μας ξέρουμε.

Χρησιμοποιείται για να απεικονίσει το περιβάλλοντα χώρο στον οποίο βρίσκεται ο χρήστης και μας δίνει τη δυνατότητα να την προσαρμόσουμε στις εφαρμογές που επιθυμούμε. Για παράδειγμα, να καταγράψουμε γεγονότα, να επικοινωνήσουμε με άλλους χρήστες, να βγάλουμε φωτογραφίες και άλλα πολλά.

2.2.2 3D Depth Sensors

Είναι 2 αισθητήρες που αναγνωρίζουν τους χρήστες (σκελετούς) με σκοπό να μαζέψουμε δεδομένα για αυτούς. Πρόκειται για τον IR Emitter και τον IR Depth. Για να μαζέψουμε αυτά τα δεδομένα, μπαίνουν σε λειτουργία και οι 2 αισθητήρες.

Ο IR Emitter δημιουργεί ένα πρότυπο από τελείες. Στην συνεχεία ο IR Depth υπολογίζει το μέγεθος και την απόσταση μεταξύ των τελείων με αποτέλεσμα να δημιουργείται ένας χάρτης. Από κει, παίρνουμε τα δεδομένα και το βάθος από τον περιβάλλοντα χώρο.

Πάντως οι περισσότερες συσκευές αναγνώρισης χρησιμοποιούν τη μέθοδο «time of flight». Μιλάμε για «δέσμες φωτός» που εκπέμπει ο IR Emitter, με τον IR Depth να τις διαβάζει καθώς ανακλούνται πίσω στον αισθητήρα της συσκευής. Αυτές οι ανακλώμενες δέσμες μετατρέπονται σε δεδομένα, δίνοντας την απόσταση μεταξύ του χρήστη και της συσκευής. Σαν μέθοδο δεν προτιμήθηκε επειδή θεωρείται ακριβή.

2.2.3 Multi-array Microphone

Είναι το σύνολο των μικροφώνων τα οποία βρίσκονται περιμετρικά της συσκευής. Είναι τέσσερα και είναι τοποθετημένα με τέτοιο τρόπο ώστε να έχουν ίδιες αποστάσεις μεταξύ τους.

Αυτό γίνεται για να αναγνωρίζουν τις φωνές των χρηστών χωρίς να υπάρχει σύγχυση από τους ήχους που υπάρχουν στο χώρο. Τέλος, μπορούν να αντιληφτούν από ποια γωνία ήρθε ο ήχος καθώς και από ποιο χρήστη δημιουργήθηκε. Έτσι ξεχωρίζουν τα ηχητικά σήματα μεταξύ δυο χρηστών.

2.2.4 Μηχανισμός Εναλλαγής Γωνίας Θέασης

Βλέποντας κάποιος τη συσκευή εξωτερικά, θα παρατηρήσει μια βάση που κρατεί το Kinect σταθερό. Αυτός είναι ο μηχανισμός εναλλαγής γωνίας θέασης. Μέσω αυτού του μηχανισμού, μπορούμε να προσαρμόσουμε την κάθετη γωνία της κάμερας. Αυτό μπορεί να επιτευχτεί και προγραμματιστικά, πχ μέσω κάποια φωνητικής εντολής.

2.3 Πεδίο Θέασης

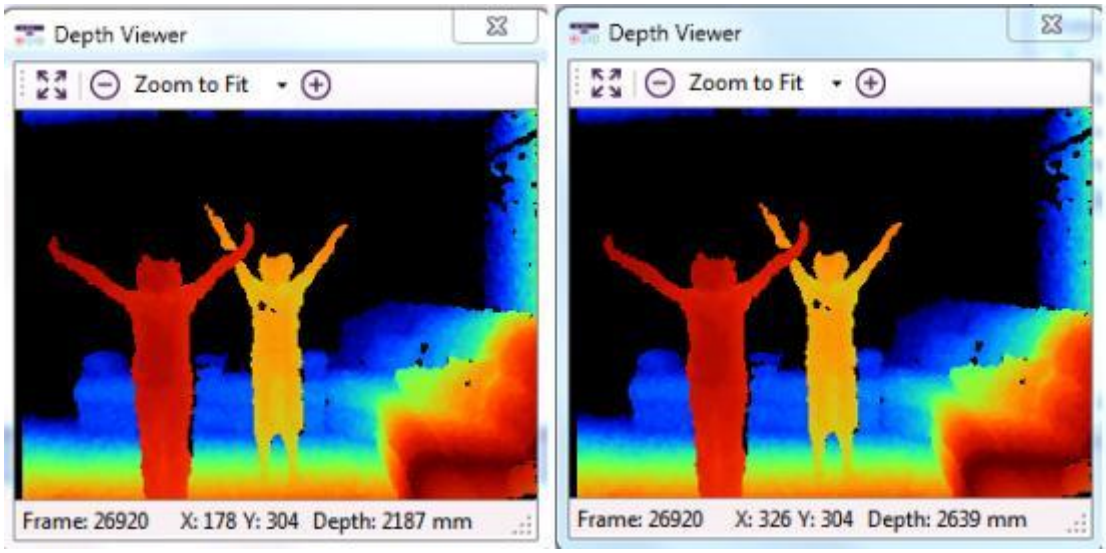
Αναλυτικά το πεδίο θέασης που αναγνωρίζει η κάμερα Kinect είναι το εξής:

- *Οριζόντιο εύρος θέασης*: 57 μοίρες
- *Κάθετο εύρος θέασης*: 43 μοίρες
- *Μηχανισμός εναλλαγής γωνίας θέασης*: ± 27 μοίρες
- *IR Depth sensor*: Απόσταση 1,2 μέχρι 3,5 μέτρα από τη συσκευή

2.4 Data Streams

Τα μέρη της κάμερας Kinect παρουσιάζουν τα εξής χαρακτηριστικά:

- *Color RGB Video camera*: Ανάλυση 640x480, 32-bit color, στα 30 frames per second
- *3D Depth sensors*: Αντιλαμβάνονται τα δεδομένα με ανάλυση 320x240, 16-bit depth στα 30 frames per second
- *Μικρόφωνα*: Αναγνωρίζει τον ήχο με 16-bit audio στα 16 KHz



Εικόνα 2.2: Kinect Data Streams

2.5 Συμβατότητα

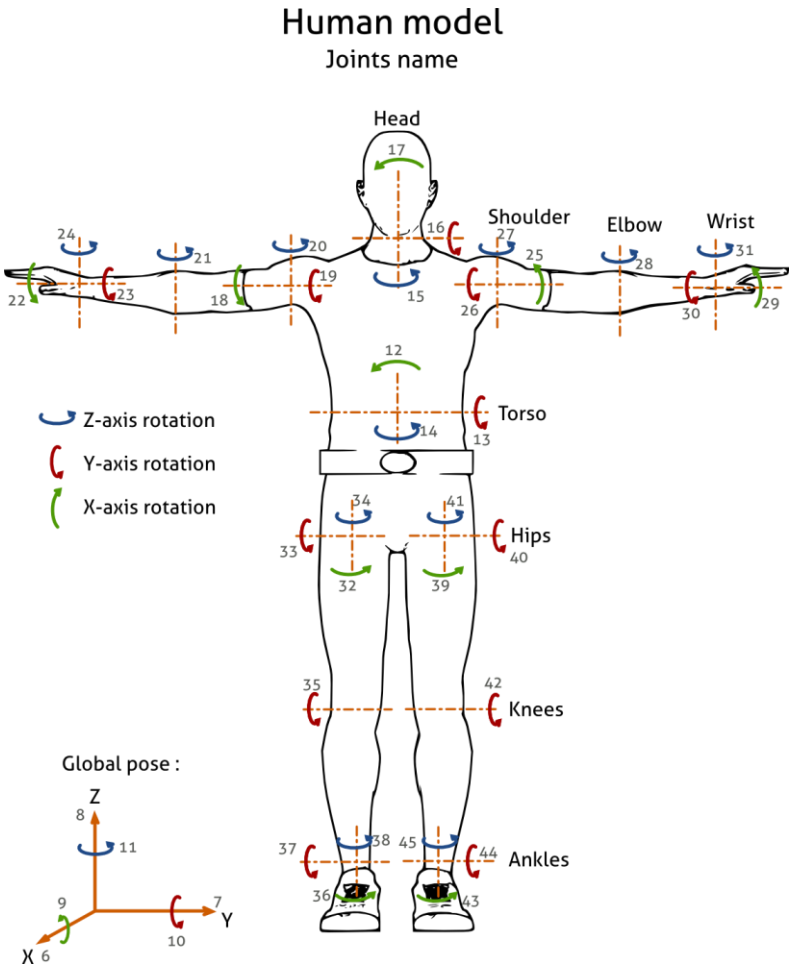
Το Kinect δημιουργήθηκε με σκοπό να είναι συμβατό στη παιχνιδομηχανή Xbox και σε προσωπικό υπολογιστή. Για τη σύνδεση στο Xbox χρησιμοποιείται το καλώδιο που περιέχει η συσκευή ενώ για τη σύνδεση της στον υπολογιστή κάνουμε χρήση αντάπτορα για μετατροπή σε USB καλώδιο.

2.6 Σύστημα Αναγνώρισης Σκελετού

Το Kinect μπορεί να αναγνωρίσει συνολικά έξι άτομα, αλλά οι περισσότερες εφαρμογές μπορούν να διαχειριστούν μέχρι δύο.

Αυτό που αναγνωρίζει η κάμερα από το χρήστη είναι σημεία (Joints) τα οποία όλα μαζί δημιουργούν ένα σκελετό. Αυτά τα σημεία είναι 20 στο σύνολο και είναι τα εξής:

- Center Shoulder
- Center Hip
- Left Shoulder
- Left Elbow
- Left Wrist
- Left Hand
- Left Hip
- Left Knee
- Left Ankle
- Left foot
- Head
- Spine
- Right Shoulder
- Right Elbow
- Right Wrist
- Right Hand
- Right Hip
- Right Knee
- Right Ankle
- Right foot



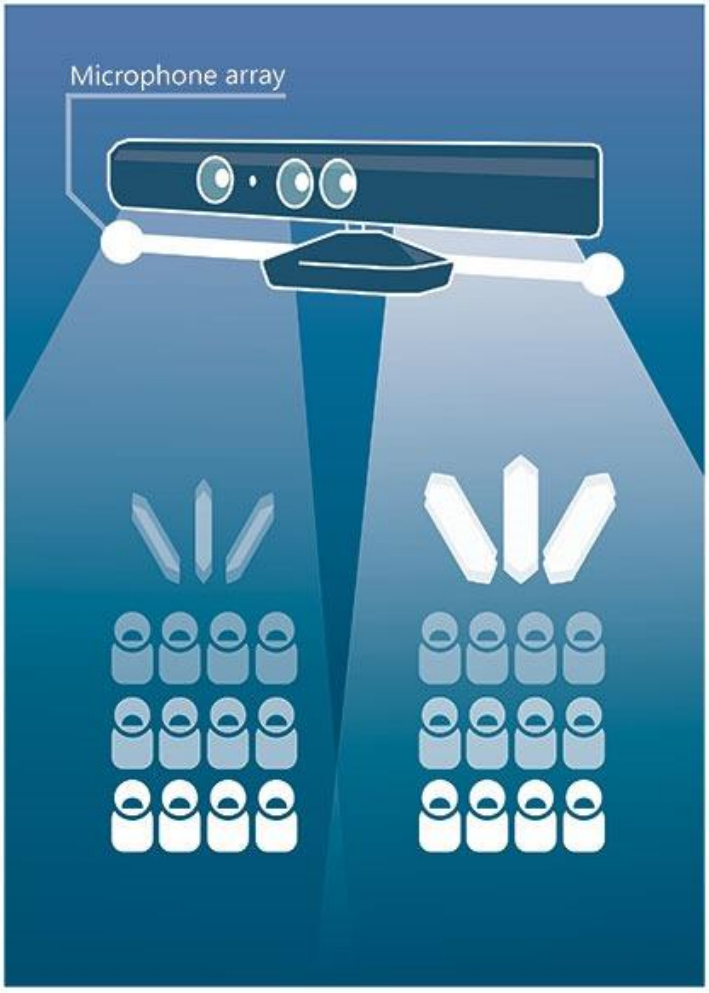
Εικόνα 2.3: Τα Joint που αναγνωρίζει το Kinect

Αυτά τα σημεία μπορεί να τα χρησιμοποιήσει ο κάθε προγραμματιστής, για να διαχειριστεί τις κινήσεις του χρήστη και να καθορίσει τις εντολές που θα εκτελούν.

2.7 Σύστημα Ήχου

Τα μικρόφωνα με τη σειρά τους χρησιμοποιούνται για να καταγράψουν τις εντολές του χρήστη. Αυτή η αλληλεπίδραση που προσφέρει το Kinect, μόνο με τη χρήση της φωνής μπορεί να φανεί πολύ χρήσιμη. Η ποικιλία εφαρμογών που μπορούν να προσφέρουν μέσω των φωνητικών εντολών στο χρήστη είναι μεγάλες.

Ο τρόπος που λειτουργεί το σύστημα είναι πολύ απλό. Όταν ο χρήστης εκφωνήσει τη λέξη, αυτή περνά από κάποια στάδια. Η λέξη χαρακτηρίζεται με μια από τις τρεις καταστάσεις: recognized, hypothesized και rejected. Ανάλογα με τη κατάσταση που θα επιλέξει για τη λέξη η συσκευή, τότε εκτελείται και το αντίστοιχο κομμάτι κώδικα.



Εικόνα 2.4: System Audio Kinect

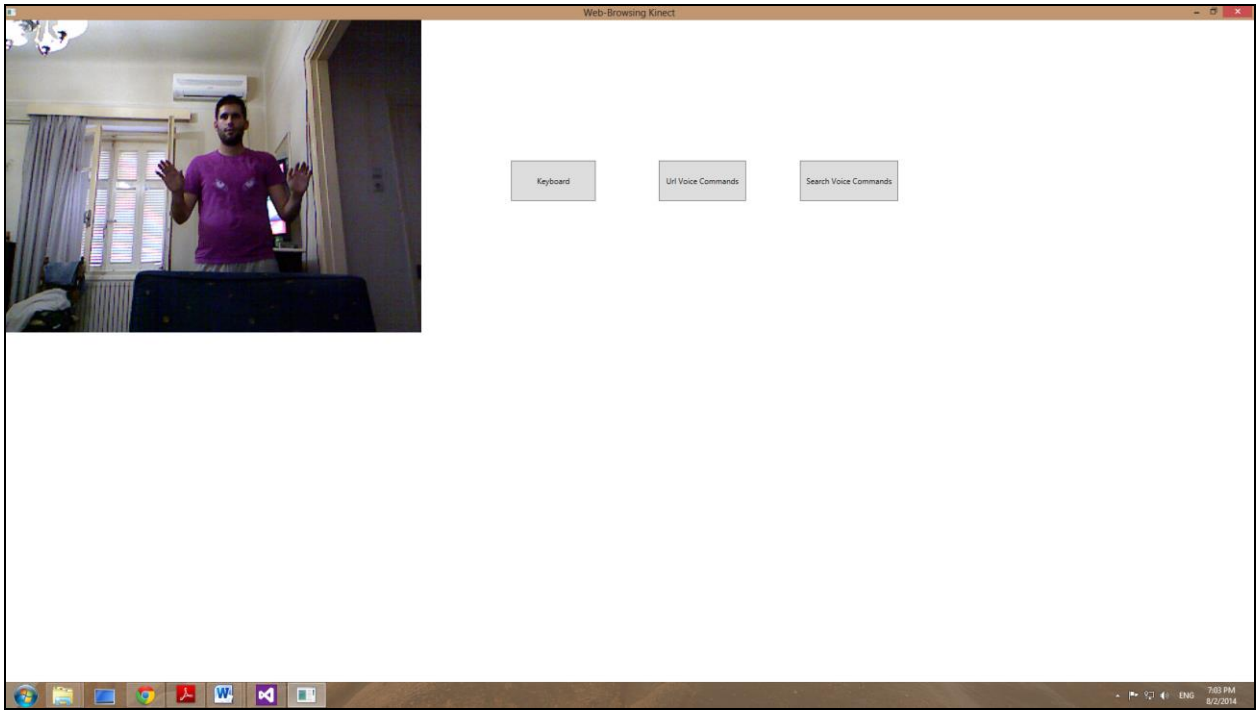
3. ΠΑΡΟΥΣΙΑΣΗ ΠΡΟΓΡΑΜΜΑΤΟΣ

Όταν ο χρήστης τρέξει την εφαρμογή στον υπολογιστή του, θα ανοίξει ένα παράθυρο στην επιφάνεια εργασίας του. Αυτό το παράθυρο περιέχει ένα πλαίσιο με αυτά που προβάλει η κάμερα του Kinect καθώς και 3 κουμπιά, το εικονικό πληκτρολόγιο , μια φόρμα με λέξεις προς αναζήτηση στο διαδίκτυο και μια φόρμα με τις αγαπημένες μας διευθύνσεις. Κατά την έναρξη του προγράμματος υπάρχει η πιθανότητα να εμφανιστεί διαγνωστικό μήνυμα του Kinect. Αυτό σημαίνει ότι η εφαρμογή δεν βρίσκει τη κάμερα του Kinect. Καλό θα ήταν, πριν την βάλετε σε λειτουργία να δείτε ότι το Kinect είναι συνδεδεμένο με τον υπολογιστή σας και οι οδηγοί του έχουν εγκατασταθεί με επιτυχία.

3.1 Θέση Εκκίνησης

Αφού το Kinect εγκατασταθεί και η εφαρμογή λειτούργει, μπορούμε περιηγηθούμε στο περιβάλλοντα χώρο. Πρώτα όμως, το πρόγραμμα πρέπει να αναγνωρίσει το σκελετό του χρήστη και στη συνέχεια τα χέρια του. Είναι σε θέση να διαβάσει και να αναγνωρίσει διάφορα μέρη του σώματος μας κατά την διάρκεια της αρχικοποίηση του, αφού προσαρμόζεται σε κάθε σωματότυπο.

Έχοντας μια απόσταση από τη κάμερα γύρω στα 1,5 μετρο , προεκτείνουμε ελαφρώς τα χέρια μας μπροστά από το κεφάλι και χρησιμοποιώντας το αριστερό χέρι καταφέρνουμε και μετακινούμε το mouse cursor. Το έχουμε δημιουργήσει με τέτοιο τρόπο ώστε να μπορεί να αναγνωρίσει τις διαστάσεις της οθόνης και να κάνει αναπροσαρμογή της θέσης του mouse cursor σε σχέση με το άνοιγμα του χεριού του χρήστη. Άρα δεν χρειάζεται να μετακινούμαστε για να φτάνουμε στα άκρα της οθόνης.

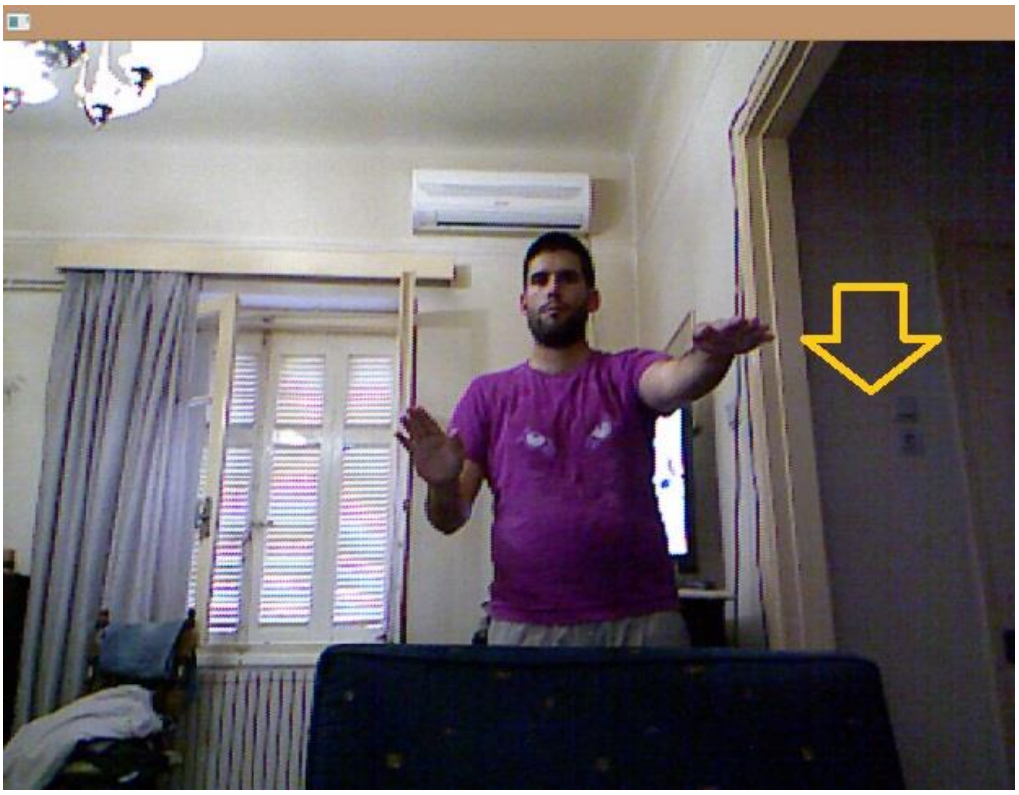


Εικόνα 3.1: Διεπαφή της εφαρμογής

3.2 Κινήσεις του Χρήστη

Το δεξί χέρι το χρησιμοποιούμε για να πετύχουμε κινήσεις (εκτός από μια περίπτωση που θα δούμε στη συνέχεια). Αυτές έχουν να κάνουν τόσο με κινήσεις του mouse cursor όσο και με συντομεύσεις που χρησιμοποιούμε όταν πλοηγούμαστε στο διαδίκτυο. Αφού έχουμε τα χέρια στη σωστή θέση, ανοίγουμε το πλοηγό διαδικτύου της αρεσκείας μας και είμαστε έτοιμοι να εκτελέσουμε τις κινήσεις που επιθυμούμε. Οι κινήσεις αυτές είναι η εξής:

- *Αριστερό κλικ*: Ο χρήστης μετακινεί το δεξί χέρι του προς τη κάμερα σε μια απόσταση 40 εκατοστών από το κεφάλι. Το δεξί χέρι πρέπει να είναι στο ύψος του δεξιού ώμου και η παλάμη να μην βρίσκεται πάνω από το κεφάλι. Σα να ρίχνει «σφαλιάρα» προς τα μπροστά. Έτσι μπορούμε να επιλέγουμε τα link της επιλογή μας. Το χέρι πρέπει να επιστρέψει στην αρχική του θέση για να την ξαναεκτελέσουμε.



Εικόνα 3.2: Αριστερό κλικ

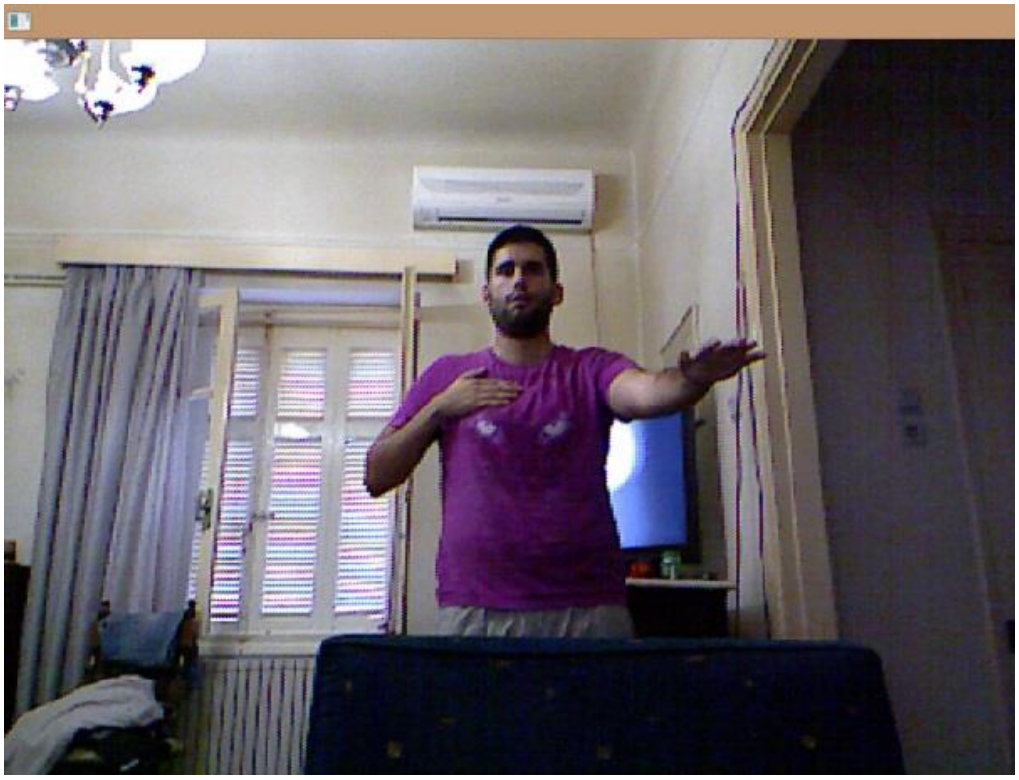
- *Διπλό αριστερό κλικ*: Λειτουργεί με την ίδια λογική όπως το αριστερό κλικ. Αυτό που έπρεπε να προσέξουμε, είναι να πετύχουμε δυο κλικ μέσα σε συγκεκριμένο χρονικό διάστημα (λιγότερο από 1 δευτερόλεπτο) . Για αυτό το σκοπό, ανατρέξαμε στη συνάρτηση stopwatch. Η stopwatch λειτουργεί ως χρονόμετρο. Όταν ο χρήστης κάνει ένα αριστερό κλικ, πρέπει να μέσα σε ένα δευτερόλεπτο να ξανακάνει άλλο ένα για πετύχει τη συγκεκριμένη κίνηση.
- *Δεξί κλικ*: Για να καταφέρουμε αυτή την ενέργεια πρέπει να σηκώσουμε το δεξί χέρι πάνω από το κεφάλι και να κινείται μεταξύ κεφαλιού και δεξιού ώμου. Και εδώ υπάρχει μια Boolean που επιτρέπει μια κίνηση τη φορά.

- *Drag & Drop*: Πρέπει το δεξί χέρι να πάει πίσω από το δεξί ώμο. Όσο παραμένει το χέρι εκεί, το αριστερό κουμπί του mouse μένει πατημένο και μόνο όταν φέρουμε το χέρι πάλι στη θέση του, απελευθερώνεται. Με αυτό το τρόπο μπορούμε να μεταφέρουμε τις καρτέλες των πλοηγών διαδικτύου.
- *Scroll*: Και τα δυο χέρια πρέπει να είναι σε προέκταση 40 εκατοστά μπροστά από το κεφάλι στο ύψος του αριστερού ώμου. Όπως είναι τα χέρια μπροστά, αν αρχίσουμε να τα μετακινούμε προς τα κάτω τότε εκτελείται το scroll down με σταθερό ρυθμό στο ενεργό παράθυρο. Αντίστοιχα αν αρχίσουμε να μετακινούμε τα χέρια προς τα πάνω τότε επιτυγχάνεται το scroll up. Αν μια σελίδα έχει μεγάλο περιεχόμενο έχουμε τη δυνατότητα να μεταβούμε στο σημείο που επιθυμούμε.
- *Zoom*: Όπως και με το scroll, έτσι και εδώ, τα δυο χέρια πρέπει να είναι σε προέκταση 40 εκατοστά μπροστά από το κεφάλι στο ύψος του αριστερού ώμου και η απόσταση των χεριών μεταξύ τους, 5 και 20 εκατοστά. Όταν η απόσταση των δυο χεριών μεγαλώνει (πάνω από 25 εκατοστά) τότε πετυχαίνουμε zoom in. Αν η απόσταση των χεριών είναι μικρότερη από 5 εκατοστά τότε έχουμε zoom out. Είναι μια χρήσιμη ενέργεια για τον χρήστη σε περίπτωση που δεν διακρίνει τα γράμματα στην ιστοσελίδα του και θέλει να αλλάξει το μέγεθος τους.
- *New tab*: Η μόνη κίνηση που γίνεται με τη χρήση του αριστερού χεριού. Ο χρήστης όταν μετακινήσει το αριστερό του χέρι πίσω από το αριστερό του ώμο τότε μια νέα καρτέλα εμφανίζεται στην ενεργή ιστοσελίδα μας.

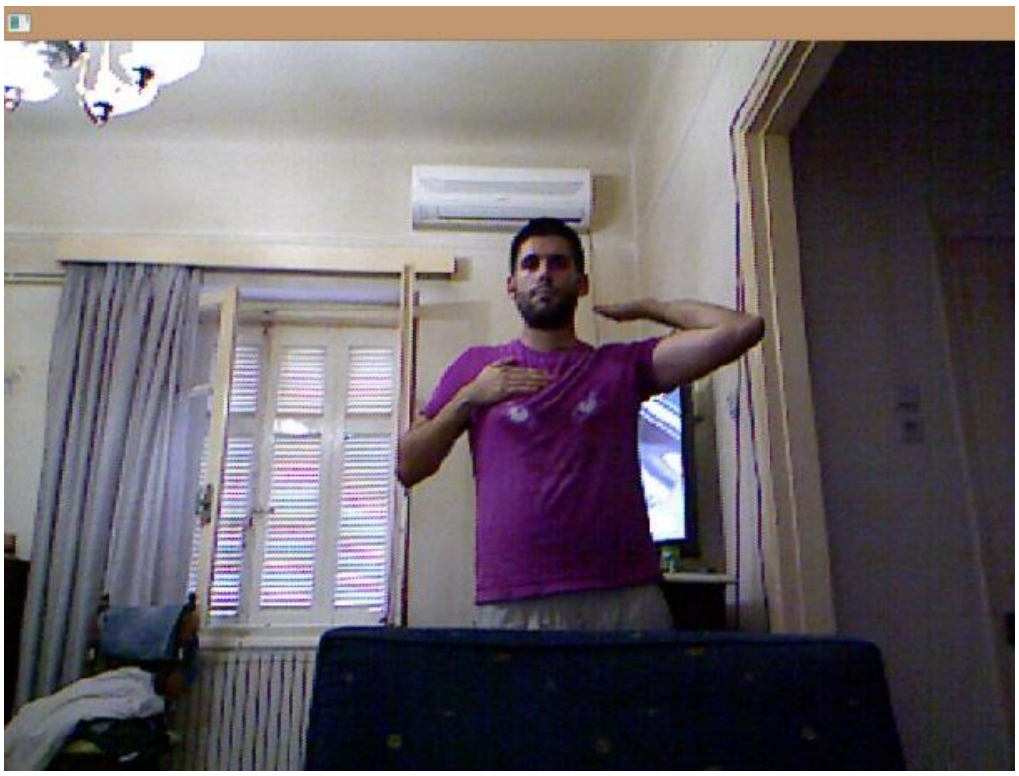
Για τις υπόλοιπες κινήσεις, το αριστερό χέρι πρέπει να βρίσκεται σε μια θέση μεγαλύτερη από τον αριστερό ώμο και σε μια απόσταση μικρότερη από τα 30 εκατοστά ως προς τον άξονα z.

Μια καλή πρόταση είναι να κολλήσετε το χέρι σας στο στήθος σας κοντά στο ύψος της καρδιάς. Και αφήνουμε τις κινήσεις για το δεξί χέρι.

- *Next tab*: Έχει ακριβώς την ίδια κίνηση με το αριστερό κλικ. Έτσι κάνουμε ενεργή την επομένη καρτέλα της ενεργής ιστοσελίδας μας.
- *Previous tab*: Λειτουργούμε όμως το drag & drop. Όταν το δεξί χέρι πάει πίσω από το δεξί ώμο, κάνουμε ενεργή την προηγούμενη καρτέλα της ενεργής ιστοσελίδας μας.
- *Forward page*: Αν προεκτείνουμε το δεξί χέρι κατά 40 εκατοστά προς τα μπροστά και κινούμαστε μεταξύ αριστερού ώμου και της αριστερής πλευράς του κεφαλιού, ο πλοηγός, μας πηγαίνει στην επομένη σελίδα.
- *Back page*: Αν προεκτείνουμε το δεξί χέρι αριστερά του αριστερού ώμου, τότε ο πλοηγός, μας πηγαίνει στη προηγούμενη σελίδα.



Εικόνα 3.3: Next Tab

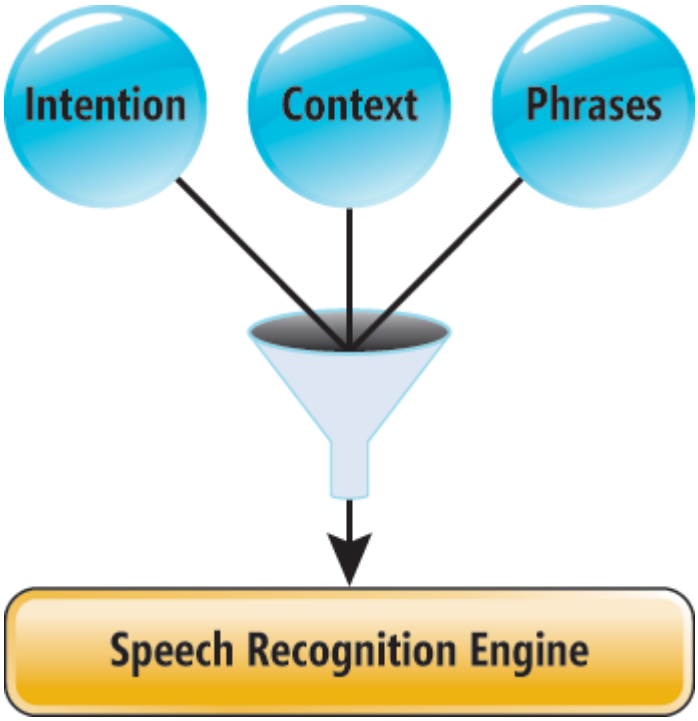


Εικόνα 3.4: Previous Tab

3.3 Φωνητική Αναγνώριση

Τα μικρόφωνα του Kinect παίζουν και αυτά με τη σειρά τους το δικό τους ρόλο στην εφαρμογή. Πάνω σε αυτά βασίστηκε ένα μεγάλο μέρος της άσκησης και πιο συγκεκριμένα η ιδιότητα που έχουν να καταγράφουν και να αναγνωρίζουν τις φωνές των χρηστών του. Όλα αυτά γίνονται μέσα από την φωνητική αναγνώριση. Η λειτουργία αυτή δίνει τη δυνατότητα στο Kinect να δέχεται ήχους από το περιβάλλον που βρίσκεται, και να τους επεξεργάζεται στο βαθμό που επιθυμεί ο χρήστης.

Με το που θέτουμε σε έναρξη το πρόγραμμα μας, μπορεί άμεσα κάποιος να αντιληφθεί ότι έχουν ήδη οριστεί ορισμένες λέξεις στην γραμματική του Kinect. Αυτές οι λέξεις αποσκοπούν στο να εκτελούν κάποιες εντολές στην εφαρμογή, στη προσπάθεια για μια ευκολότερη πλοήγηση στο διαδίκτυο. Χαρακτηριστικό παράδειγμα είναι η κλήση του Google Chrome λέγοντας στο μικρόφωνο τη λέξη «Internet». Τέλος, έχουμε δώσει την ευχέρεια στο χρήστη να προσθαφαιρέσει τις λέξεις και τις διευθύνσεις που τον ενδιαφέρουν.



Εικόνα 3.5: Speech Recognition

Οι λέξεις που είναι αποθηκευμένες στο λεξιλόγιο του Kinect είναι οι εξής:

- Up: Αλλάζει η γωνία της κάμερας του Kinect και ανεβαίνει προς τα πάνω κατά 5 μοίρες.
- Down: Η γωνία της κάμερας του Kinect κατεβαίνει κατά 5 μοίρες.
- Close: Όταν ο χρήστης προφέρει τη λέξη close, το ενεργό παράθυρο που βρίσκεται στην επιφάνεια της οθόνης εκείνη τη στιγμή, κλείνει.
- Maximize: Το ενεργό παράθυρο του υπολογιστή μας, καλύπτει όλη την οθόνη.
- Restore: Επαναφέρει στο αρχικό μέγεθος το ενεργό παράθυρο του υπολογιστή μας.

- Keyboard: Καλείται η φόρμα με το εικονικό πληκτρολόγιο που έχουμε δημιουργήσει. Σε οποιοδήποτε σημείο και αν είμαστε στον υπολογιστή μπορούμε να ενεργοποιήσουμε το εικονικό πληκτρολόγιο. Σε αυτή τη φόρμα έχουμε προσομοιώσει όλα τα γράμματα του Ελληνικού και αγγλικού αλφαβήτου μαζί και με κάποιες συντομεύσεις, απαραίτητες για το διαδίκτυο (www. , .gr , .com). Υπάρχουν επίσης τα κουμπιά Backspace, Space και το Enter. Ο χρήστης μπορεί να γράψει την ακολουθία γραμμάτων που επιθυμεί και για να την προβάλει σε οποιοδήποτε πεδίο κειμένου των ιστοσελίδων που κινούμαστε, πατάει το κουμπί Enter. Μπορούμε μεν να γράψουμε στα πεδία κειμένου και στα ελληνικά αλλά επειδή το Kinect αναγνωρίζει μόνο την αγγλική γλώσσα, τις φόρμες (θα αναφερθούμε αργότερα σε αυτές) καλό είναι να τις συμπληρώνουμε με αγγλικές λέξεις.
- Click: Λέγοντας τη λέξη ο χρήστης, πραγματοποιείται ένα αριστερό click στη θέση που βρίσκεται ο mouse cursor του υπολογιστή μας.
- Double click: Το πρόγραμμα εκτελεί ένα διπλό κλικ στη θέση που βρίσκεται ο mouse cursor του υπολογιστή μας.
- Right click: Σε όποια θέση και αν βρίσκεται ο mouse cursor, όταν προφέρουμε τη συγκεκριμένη λέξη, πιέζεται το δεξί κουμπί του mouse.
- Drag / Drop: Είναι δυο διαφορετικές λέξεις αλλά έχουν ως στόχο την ίδια ενέργεια, το drag & drop. Όταν ο χρήστης πει την λέξη Drag, ο mouse cursor που βρίσκεται στο αντικείμενο, το πιάνει με το αριστερό κλικ και έχοντας το πατημένο, μπορεί με τα χέρια του να το μεταφέρει στο μέρος που θέλει. Αν πει τη λέξη Drop τότε το αντικείμενο μένει στη τελευταία θέση που βρέθηκε ο mouse cursor. Όταν είμαστε στη περίπτωση Drag-Drop δεν εκτελείται κάποια άλλη κίνηση παρά μόνο να μετακινούμε τον mouse cursor.
- Internet: Καλείται μια νέα διεργασία του Google Chrome σε διαφορετικό παράθυρο.
- URL: Ο mouse cursor του υπολογιστή επιλέγει από μόνο του τη μπάρα διευθύνσεων της ενεργής ιστοσελίδας.
- Forward: Πηγαίνει στην επόμενη σελίδα της ενεργής ιστοσελίδας.
- Back: Επιστρέφει στη προηγούμενη σελίδα.
- New tab: Δημιουργεί μια νέα καρτέλα στην ενεργή ιστοσελίδα.
- Next tab: Πηγαίνει στην επόμενη καρτέλα της ενεργής ιστοσελίδας.
- Previous tab: Επιστρέφει στη προηγούμενη καρτέλα της ενεργής ιστοσελίδας.
- Reload: Ανανεώνει το περιεχόμενο της ενεργής ιστοσελίδας.
- Homepage: Επιστρέφει στην αρχική σελίδα με την οποία ανοίγει ο πλοηγός διαδικτύου μας.
- Bookmark: Μαρκάρει την σελίδα που θέλουμε, με σκοπό να την επισκεφτούμε κάποια στιγμή στο μέλλον.
- History: Έχουμε πρόσβαση σε όλο το ιστορικό ιστοσελίδων που έχουμε επισκεφτεί.
- Download history: Όταν προφέρουμε την συγκεκριμένη έκφραση, μπορούμε να δούμε όλο το ιστορικό των αρχείων που έχουμε κατεβάσει από το διαδίκτυο.

- Print page: Στέλνει προς εκτύπωση την ενεργή ιστοσελίδα που διαχειριζόμαστε.
- Save page: Στέλνει προς αποθήκευση (σε μορφή HTML) την ενεργή ιστοσελίδα του υπολογιστή.
- Source code: Ανοίγει μια νέα καρτέλα με τον πηγαίο κώδικα της ενεργής ιστοσελίδας.
- Developer Tools: Ενεργοποιεί τον πίνακα εργαλείων του Google Chrome.

3.4 Δημιουργία Νέων Εντολών

Όπως αναφέραμε και στην φωνητική αναγνώριση, το πρόγραμμα έχει σχεδιαστεί για προσθέτει και να αφαιρει λέξεις και διευθύνσεις που επιθυμεί ο χρήστης.

Πηγαίνοντας στην αρχική διεπαφή της εφαρμογής, εύκολα μπορεί να δει κάποιος αλλά 2 κουμπιά εκτός από το εικονικό πληκτρολόγιο. Το ένα είναι το Url Voice Commands και το άλλο είναι το Search Voice Commands. Αυτά τα 2 κουμπιά, είναι 2 φόρμες των Windows. Η μια περιέχει λέξεις-συντομεύσεις και τις διευθύνσεις που αντιστοιχούν σε αυτές (μαζεύουμε τις διευθύνσεις των ιστοσελίδων της επιλογής μας) και η άλλη φόρμα έχει λέξεις προς αναζήτηση στο διαδίκτυο. Οι λέξεις στις φόρμες είναι στα Αγγλικά και φυσικά σε περίπτωση κενών γραμμών στους πίνακες ή διπλών λέξεων, ο χρήστης ενημερώνεται με αντίστοιχο μήνυμα.

Επειδή και οι 2 φόρμες δεν μπορούν να τρέχουν ταυτόχρονα, έχουμε 2 λέξεις οι οποίες μας μεταφέρουν από την μια κατάσταση στην άλλη. Όταν το πρόγραμμα ξεκινάει, μόνο η φόρμα με τις διευθύνσεις των ιστοσελίδων είναι διαθέσιμη. Αν ο χρήστης προφέρει τη λέξη Search On, τότε ενεργοποιείται η φόρμα Search Voices Commands και μπορεί να προφέρει τις λέξεις προς αναζήτηση. Για να επανέρθει στην προηγούμενη κατάσταση, αρκεί να φωνάξει Search off και τότε οι διευθύνσεις των ιστοσελίδων είναι πάλι διαθέσιμες.

Τέλος να αναφέρουμε ότι προτίμησα να αποθηκεύω τις διευθύνσεις και τις λέξεις από τις φόρμες σε 2 διαφορετικά αρχεία, το searchCommands.txt και το urlCommands.txt αντίστοιχα. Θα μπορούσαμε να κάνουμε μια βάση με τα δεδομένα μας. Αυτό έγινε για να μην υποχρεώνεται ο χρήστης να εγκαταστήσει τίποτα άλλο στον υπολογιστή του παρά μόνο την εργασία.

4. ΠΑΡΟΥΣΙΑΣΗ ΤΟΥ ΚΩΔΙΚΑ

Στο τέταρτο κεφάλαιο θα συζητήσουμε αναλυτικά πως προγραμματίστηκε η εφαρμογή. Θα δείξουμε τη ροή που ακολουθεί το πρόγραμμα και θα αναλύσουμε κομμάτια κώδικα που κρίνουμε ότι παίζουν σημαντικό ρόλο στην άσκηση.

4.1 Κλάσεις της Εφαρμογής

Η εργασία έγινε στη πλατφόρμα της Microsoft, Visual studio 2012 χρησιμοποιώντας C#. Χωρίζεται σε 2 project:

- KinectDevSetup
- Microsoft.Samples.Kinect.WpfViewers

4.1.1 Microsoft.Samples.Kinect.WpfViewers

Είναι ένα έτοιμο project του Kinect Explorer το οποίο είναι διαθέσιμο από τη πλατφόρμα «Microsoft Kinect for Windows SDK Sample Browser» . Περιέχει έτοιμες κλάσεις που είναι απαραίτητες για να δομήσουμε μια εργασία στο Kinect και θα κάνουν το έργο μας πιο εύκολο αφού έχει χρησιμοποιηθεί σε αρκετές εφαρμογές.

Επίσης υπάρχουν χρήσιμα .dll τα όποια μπαίνουν ως αναφορές στην εργασία μας (Coding4Fun.dll - Microsoft.Samples.Kinect.WpfViewers.dll) τα οποία θα μας δώσουν τη δυνατότητα να διαχειριστούμε καλύτερα τη διεπαφή μας και να μας τροφοδοτήσουν με διαγνωστικά μηνύματα σε περίπτωση λάθους κατά την εκτέλεση.

4.1.2 KinectDevSetup

Εδώ έχουμε το σημαντικότερο κομμάτι κώδικα για την υλοποίηση της πτυχιακής το οποίο έχει αναπτυχτεί από εμένα.

Αποτελείται από δυο αρχεία WPF:

- App.xaml
- MainWindow.xaml

Τρία αρχεία Windows Forms:

- Keyboard.cs
- SearchVoiceCommands.cs
- UrlVoiceCommands.cs

Δυο αρχεία text:

- searchCommands.txt
- urlCommands.txt

App.xaml

Με το που τρέξουμε την εφαρμογή μας είναι το πρώτο αρχείο που συναντάμε. Αυτό καλεί με τη σειρά του το MainWindow.xaml.

MainWindow.xaml

Σε αυτό το αρχείο περιέχονται οι πιο σημαντικές πληροφορίες για το Kinect. Εδώ αρχικοποιείται η κάμερα και το μικρόφωνο, αναγνωρίζει το χώρο και τα άτομα που το περιβάλουν, διαβάζει τις κινήσεις που κάνουν τα άτομα, ορίζουμε την γραμματική και άλλα πολλά. Είναι το αρχείο από το οποίο όλα ξεκινούν και όλα τελειώνουν. Για λεπτομέρειες ως προς τις κλάσεις και τι αντιπροσωπεύει ο κώδικας τους θα δούμε στη συνέχεια.

Keyboard.cs

Είναι η φόρμα που εμφανίζει το εικονικό πληκτρολόγιο. Έχει όλα τα γράμματα του Ελληνικού και του Αγγλικού αλφαβήτου, το κουμπί space, το backspace, χρήσιμες συντομεύσεις κουμπιών για το διαδίκτυο (www. , .gr , .com) καθώς και το κουμπί Enter το οποίο στέλνει όλο το περιεχόμενο του κειμένου που γράψαμε ως input σε πεδία κειμένου του διαδικτύου που επιθυμούμε. Τέλος, υπάρχει και ένα πεδίο στο πληκτρολόγιο μας το οποίο μας δίνει τη δυνατότητα να δούμε τι γραφούμε.

SearchVoiceCommands.cs

Το SearchVoiceCommands είναι η φόρμα στην οποία εμφανίζονται οι λέξεις που μπορεί να χρησιμοποιήσει ο χρήστης για την αναζήτηση στο διαδίκτυο. Λειτουργούν ως φωνητικές εντολές αφού ο χρήστης «φωνάζει» τη λέξη που υπάρχει στη λίστα και την εμφανίζει στο πεδίο κειμένου που επιθυμεί. Έχει την δυνατότητα να επεξεργαστεί, να προσθέσει ακόμα και να διαγράψει όσες λέξεις θέλει. Επίσης κατά την διάρκεια της αποθήκευσης, ελέγχεται αν υπάρχουν τυχόν κενά κελία η αν εμφανίζεται μια λέξη παραπάνω από μια φορά.

UrlVoiceCommands.cs

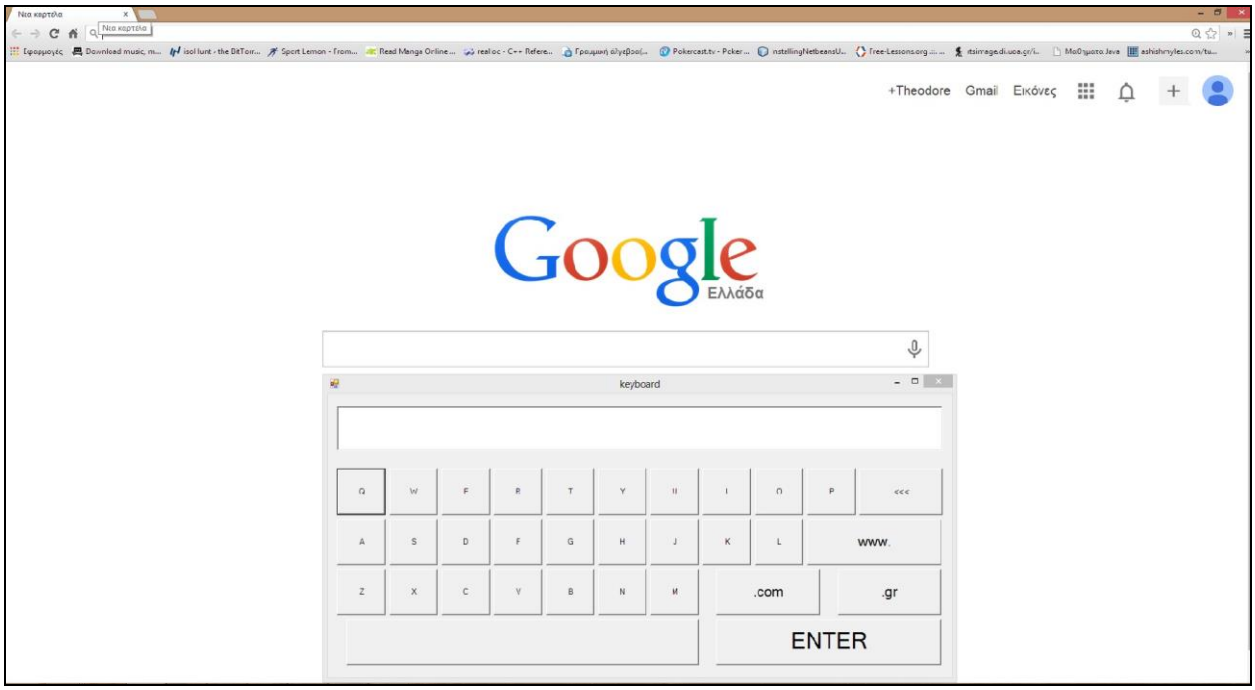
Στο UrlVoiceCommands έχουμε μαζεμένες όλες τις ιστοσελίδες που επιθυμεί ο χρήστης να χρησιμοποιήσει κατά την πλοήγηση του στο διαδίκτυο. Ο χρήστης «φωνάζει» την λέξη-συντόμευση από τη φόρμα και εμφανίζει την αντίστοιχη διεύθυνση στο πεδίο κειμένου στο διαδίκτυο. Είναι μια φόρμα παρόμοιας λογικής με την SearchVoiceCommands, δηλαδή μπορούμε να επεξεργαστούμε, να προσθέσουμε και να διαγράψουμε τις διευθύνσεις που επιθυμούμε. Κατά τη διάρκεια της αποθήκευσης, ελέγχει για κενά κελιά και για διπλές καταχωρήσεις.

searchCommands.txt

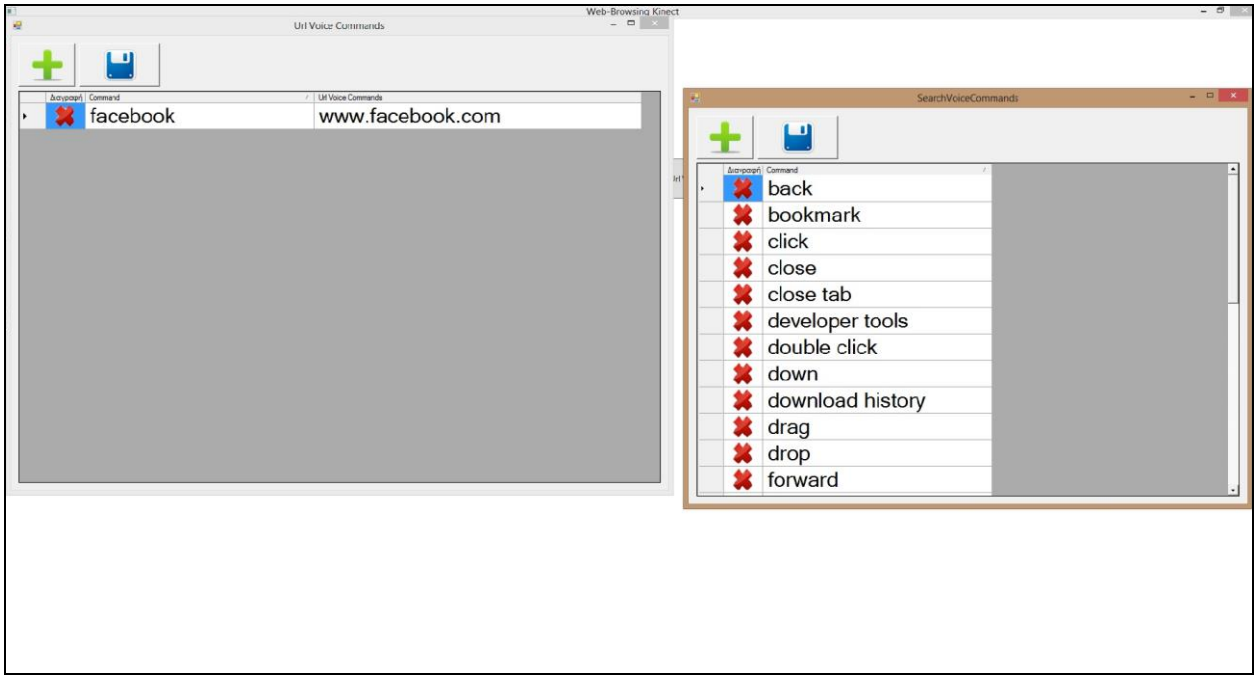
Είναι το έγγραφο κειμένου στο οποίο βρίσκονται αποθηκευμένες όλες οι λέξεις προς αναζήτηση. Η φόρμα SearchVoiceCommands παίρνει τις λέξεις από αυτό το αρχείο και τις προβάλει.

urlCommands.txt

Εδώ βρίσκονται αποθηκευμένες όλες οι λέξεις-συντομεύσεις με τις διευθύνσεις που θέλουμε να καλέσουμε κατά τη πλοήγηση μας στο διαδίκτυο. Η φόρμα UrlVoiceCommands παίρνει τις λέξεις από αυτό το αρχείο και τις προβάλει στη φόρμα της.



Εικόνα 4.1: Το εικονικό πληκτρολόγιο



Εικόνα 4.2: Οι φόρμες με τις φωνητικές εντολές

4.2 Πίνακας Μεθόδων Εφαρμογής

Σε αυτή την ενότητα, θα περιγράψουμε μέσω ενός πινάκα, κάθε μέθοδο και κάθε κλάση που παρουσιάζονται στο KinectSetupDev. Στη συνέχεια θα εξηγήσουμε με περισσότερες λεπτομέρειες κάποια κομμάτια κώδικα που αξίζουν να αναφερθούν.

4.2.1 MainWindow.xaml

Όνομα συνάρτησης	Περιγραφή
`public MainWindow()`	Ο constructor της MainWindow. Σε αυτή τη μέθοδο, αρχικοποιείται το παράθυρο της εφαρμογής μας.
`private void Window_Loaded(object sender, RoutedEventArgs e)`	Με το που φορτώσει το παράθυρο της εφαρμογής μας, καλείται η διπλανή μέθοδος. Εδώ καθορίζεται ο handler, για τη διαχείριση του event, όταν αλλάζει ο sensor.
`void kinectSensorChooser1_KinectSensorChanged(object sender, DependencyPropertyChangedEventArgs e)`	Σε αυτή τη μέθοδο, καθορίζουμε τις παραμέτρους του sensor (ενεργοποίηση κάμερας , μικροφώνων) .
`void newSensor_AllFramesReady(object sender, AllFramesReadyEventArgs e)`	Αυτή η μέθοδος αναφέρεται στη κάμερα και καλείται αρκετές φορές καθώς τρέχει η εφαρμογή. Μέσω αυτής, μπορούμε και καλούμε τις συναρτήσεις της κίνησης.
`private void ProcessGesture(Joint head, Joint handleft, Joint handright, Joint shoulderight, Joint shoulderleft)`	Μέσω αυτής της συνάρτησης, η εφαρμογή μας αναγνωρίζει τη κίνηση του χρήστη. Μέσω αυτών των κινήσεων μπορούμε να ορίσουμε το πρόγραμμα να κάνει όποια ενέργεια επιθυμούμε.
`Skeleton GetFirstSkeleton(AllFramesReadyEventArgs e)`	Εδώ, η εφαρμογή διαβάζει το πρώτο σκελετό που έχει αναγνωρίσει και μπορούμε να πάρουμε όλα τα διαθέσιμα Joints που χρειαζόμαστε για τη κίνηση του χρήστη.
`private void StopKinect(KinectSensor sensor)`	Τερματίζει τη λειτουργία του sensor
`private void Window_Closing(object sender, System.ComponentModel.CancelEventArgs e)`	Η μέθοδος αυτή ενεργοποιείται όταν κλείνει η εφαρμογή μας. Σε αυτή, καλείται η μέθοδος StopKinect για να κλείσει και ο sensor.
`private SpeechRecognitionEngine CreateSpRecognizer()`	Σε αυτή τη συνάρτηση γίνεται η αρχικοποίηση της φωνητικής αναγνώρισης και της γραμματικής που θα χρησιμοποιήσουμε. Αφού πάρουμε το αντικείμενο από το recognizer, στη συνέχεια, φτιάχνουμε την γραμματική για να τις συγκρίνει με τις λέξεις που προφέρει ο χρήστης.
`private static RecognizerInfo GetKinectRecognizer()`	Ο χρήστης σε αυτή τη μέθοδο παίρνει τη γλώσσα αναγνώρισης για τις φωνητικές εντολές.

`private void RejectSpeech(RecognitionResult result, float conf)`	Η μέθοδος καλείται όταν το Kinect ακούσει μια λέξη η οποία δεν υπάρχει στη γραμματική της.
`private void RecSpeechRecognitionRejected(object sender, SpeechRecognitionRejectedEventArgs e)`	Στη μέθοδο αναγνώρισης της φωνής, αν το Kinect αποφασίσει πως μια λέξη δεν είναι αποδεκτή, μπαίνει στη διπλανή μέθοδο και στη συνέχεια καλεί τη RejectSpeech
`private void RecSpeechHypothesized(object sender, SpeechHypothesizedEventArgs e)`	Αυτή η μέθοδος καλείται από τη μέθοδο αναγνώρισης της φωνής όταν μια λέξη μοιάζει πολύ με κάποια της γραμματικής.
`private void RecSpeechRecognized(object sender, SpeechRecognizedEventArgs e)`	Η συνάρτηση καλείται όταν η μέθοδος αναγνώρισης της φωνής αποφασίσει ότι η λέξη που προφέραμε υπάρχει στη γραμματική μας. Ανάλογα με τις λέξεις που έχουμε βάλει στη γραμματική, κάνουμε και τις ανάλογες ενέργειες.
`public void AllVoiceCommands()`	Διαβάζει τα αρχεία searchCommands.txt και urlCommands.txt και τις λέξεις που περιέχουν και τις βάζουμε σε ένα Dictionary και μια List αντίστοιχα. Τόσο το Dictionary όσο και τη List θα τα χρησιμοποιήσουμε στη συνέχεια για να τα βάλουμε στη γραμματική και να φορτώσουμε τις φόρμες με τις φωνητικές εντολές.
`private void KeyboardButton_Click(object sender, RoutedEventArgs e)`	Δημιουργεί τη φόρμα με το εικονικό πληκτρολόγιο και το εμφανίζει. Λόγω της Boolean που υπάρχει, δεν μας αφήνει να έχουμε 2 ίδιες φόρμες ανοικτές.
`private void UrlVoiceCmd_Click(object sender, RoutedEventArgs e)`	Αυτή η μέθοδος δημιουργεί τη φόρμα με τις διευθύνσεις και τις συντομεύσεις τους και δίνουμε σαν μεταβλητή το Dictionary. Τέλος, εμφανίζει τη φόρμα με τα στοιχεία της. Λόγω της Boolean που υπάρχει δεν μας αφήνει να έχουμε 2 ίδιες φόρμες ανοικτές.
`private void SearchVoiceCmd_Click(object sender, RoutedEventArgs e)`	Αυτή η μέθοδος δημιουργεί τη φόρμα με τις λέξεις για αναζήτηση στο διαδίκτυο και δίνουμε σαν μεταβλητή τη List. Τέλος, εμφανίζει τη φόρμα με τα στοιχεία της. Λόγω της Boolean που υπάρχει, δεν μας αφήνει να έχουμε 2 ίδιες φόρμες ανοικτές.

Πίνακας 4.1: Συναρτήσεις της κλάσης MainWindow.xaml

4.2.2 Keyboard.cs

Όνομα συνάρτησης	Περιγραφή
`protected override CreateParams CreateParams`	Κάνει την φόρμα να φαίνεται απενεργοποιημένη ώστε να μπορούμε να βλέπουμε το πεδίο κειμένου που γράφουμε στο

	διαδίκτυο.
`public Keyboard()`	Είναι ο constructor της του εικονικού πληκτρολογίου. Αρχικοποιούνται όλα τα κουμπιά για να εμφανιστούν στην οθόνη
`private void writeKey(string key)`	Η συνάρτηση παίρνει τη συμβολοσειρά και την γράφει στο πεδίο κειμένου του εικονικού πληκτρολογίου για να βλέπει ο χρήστης τι γράφει.
`private void buttoni_Click(object sender, EventArgs e)`	Για i από 1 μέχρι 31 υπάρχει και από μια συνάρτηση. Εδώ ο χρήστης καλεί τη μέθοδο writeKey με όρισμα το γράμμα που αντιστοιχεί σε κάθε κουμπί. Η μέθοδος i=31 διαγράφει το τελευταίο χαρακτήρα από το πεδίο κειμένου του εικονικού πληκτρολογίου (Το κουμπί backspace).
`private void button32_Click(object sender, EventArgs e)`	Όταν ο χρήστης πατά το κουμπί ENTER, τότε γράφει έναν-έναν τους χαρακτήρες από το πεδίο κειμένου του πληκτρολογίου στο πεδίο κειμένου που διαλέξαμε στο διαδίκτυο. Τέλος κλείνει τη φόρμα.
`private void Keyboard_FormClosing(object sender, FormClosingEventArgs e)`	Καλείται όταν κλείνουμε τη φόρμα. Αλλάζει τιμή στη Boolean του πληκτρολογίου για να μπορέσουμε να το επιλέξουμε.
`private void setLanguage(string lan)`	Όταν αλλάξει η γλώσσα στα Windows, μετατρέπει τα γράμματα από ελληνικά σε αγγλικά και ανάποδα.

Πίνακας 4.2: Συναρτήσεις της κλάσης Keyboard.cs

4.2.3 SearchVoiceCommands.cs

Όνομα συνάρτησης	Περιγραφή
`public SearchVoiceCommands()`	Είναι ο constructor της φόρμας με τις λέξεις προς αναζήτηση στο διαδίκτυο. Εδώ αρχικοποιεί όλα τα στοιχεία της.
`private void SearchVoiceCommands_Load(object sender, EventArgs e)`	Καλείται αυτή η μέθοδος όταν φορτώσει η φόρμα. Εδώ δημιουργεί το DataGridView και φορτώνει τα δεδομένα από τη List.
`private void AddButton_Click(object sender, EventArgs e)`	Σε αυτή τη μέθοδο εισάγουμε μια νέα γραμμή στο πλέγμα και ο χρήστης έχει τη δυνατότητα να βάλει τα δεδομένα όπου θέλει.
`private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)`	Μόλις κάνουμε κλικ σε κελιά του DataGridView, ενεργοποιείται η συνάρτηση. Εδώ ελέγχουμε αν πατήθηκε το κουμπί της διαγραφής.
`private void CreateDataGrid()`	Αυτή είναι η μέθοδος δημιουργείται και γεμίζει το DataGridView. Τα στοιχεία τα παίρνει από τη

	List και τα φορτώνει στις γραμμές
`private void SaveButton_Click(object sender, EventArgs e)`	Όταν το κουμπί της αποθήκευσης πατηθεί, τότε ενεργοποιείται η μέθοδος. Στην αποθήκευση διαγράφονται τα δεδομένα όλου του αρχείου και εγγράφονται εκ νέου όλος ο πινάκας με τα στοιχεία. Σε περίπτωση κενής γραμμής η διπλών λέξεων εμφανίζεται μήνυμα.
`private void SearchVoiceCommands_FormClosing(object sender, FormClosingEventArgs e)`	Η συνάρτηση καλείται όταν κλείνει η φόρμα με τις λέξεις προς αναζήτηση στο διαδίκτυο. Εμφανίζεται και μήνυμα για επανεκκίνηση της εφαρμογής σε περίπτωση που έχουμε προσθέσει νέες λέξεις.

Πίνακας 4.3: Συναρτήσεις της κλάσης SearchVoiceCommands.cs

4.2.4 UrlVoiceCommands.cs

Όνομα συνάρτησης	**Περιγραφή**
`public UrlVoiceCommands()`	Είναι ο constructor της φόρμας με τις διευθύνσεις. Εδώ αρχικοποιεί όλα τα στοιχεία της.
`private void UrlVoiceCommands_Load(object sender, EventArgs e)`	Αυτή η μέθοδος καλείται όταν φορτώσει η φόρμα. Εδώ δημιουργεί το DataGridView και φορτώνει τα δεδομένα από το Dictionary.
`private void AddButton_Click(object sender, EventArgs e)`	Σε αυτή τη μέθοδο εισάγουμε μια νέα γραμμή στο πλέγμα και ο χρήστης έχει τη δυνατότητα να βάλει τα δεδομένα όπου θέλει.
`private void dataGridView1_CellContentClick ( object sender, DataGridViewCellEventArgs e )`	Μόλις κάνουμε κλικ σε κελιά του DataGridView, ενεργοποιείται η συνάρτηση. Εδώ ελέγχουμε αν πατήθηκε το κουμπί της διαγραφής.
`private void CreateDataGrid()`	Αυτή είναι η μέθοδος που δημιουργείται και γεμίζει το DataGridView. Τα στοιχεία τα παίρνει από το Dictionary και τα φορτώνει στις γραμμές
`private void SaveButton_Click(object sender, EventArgs e)`	Όταν το κουμπί της αποθήκευσης πατηθεί, τότε ενεργοποιείται η μέθοδος. Στην αποθήκευση διαγράφονται τα δεδομένα όλου του αρχείου και εγγράφονται εκ νέου όλος ο πίνακας με τα στοιχεία. Σε περίπτωση κενής γραμμής η διπλών λέξεων-συντομεύσεων εμφανίζεται μήνυμα.
`private void UrlVoiceCommands_FormClosing(object sender, FormClosingEventArgs e)`	Η συνάρτηση καλείται όταν κλείνει η φόρμα με τις λέξεις προς αναζήτηση στο διαδίκτυο. Εμφανίζεται και μήνυμα για επανεκκίνηση της εφαρμογής σε περίπτωση που έχουμε προσθέσει νέες διευθύνσεις.

Πίνακας 4.4: Συναρτήσεις της κλάσης UrlVoiceCommands.cs

4.3 Σημαντικά σημεία στο Κώδικα

Σε αυτή την ενότητα θα παρουσιάσουμε κάποια κομμάτια κώδικα που παίζουν σημαντικό ρολό στη εργασία και χρειάζονται καλύτερη επεξήγηση.

Από τις πρώτες γραμμές κώδικα του project, μπορεί κάποιος να παρατηρήσει τις δηλώσεις των συναρτήσεων που χρησιμοποιήθηκαν ώστε ο χρήστης να μπορεί να διαχειριστεί τα Windows. Η κίνηση του δρομέα, η ανάκτηση παραθύρων στα Windows και η εναλλαγή αυτών, διαγνωστικά μηνύματα σε περίπτωση λάθους είναι μερικά από τα αποτελέσματα αυτών των συναρτήσεων. (Κώδικας 4.1).

```
[DllImport("user32.dll")]
public static extern int SetCursorPos(int x, int y);

[DllImport("user32.dll",
CharSet = CharSet.Auto, CallingConvention = CallingConvention.StdCall)]
public static extern void mouse_event(int dwFlags, int dx, int dy, int cButtons, int
dwExtraInfo);

[DllImport("user32.dll")]
private static extern IntPtr GetForegroundWindow();

[DllImport("user32.dll")]
static extern bool SetForegroundWindow(IntPtr hWnd);

[DllImport("user32.dll")]
private static extern int GetWindowText(IntPtr hWnd, StringBuilder text, int count);

[DllImport("user32.dll", CharSet = CharSet.Auto)]
static extern IntPtr SendMessage(IntPtr hWnd, UInt32 Msg, IntPtr wParam, IntPtr lParam);

[DllImport("user32.dll")]
static extern bool ShowWindow(IntPtr hWnd, int nCmdShow);
```

Κώδικας 4.1: Συναρτήσεις Συστήματος

Στη συνέχεια βλέπουμε πως το Kinect βάζει σε λειτουργία τον sensor και αρχικοποιεί σημαντικές παραμέτρους της κάμερας (Κώδικας 4.2). Μαζί με αυτά ξεκινάμε και τα μικρόφωνα και αρχικοποιείται η φωνητική αναγνώριση δίνοντας τη δυνατότητα στο χρήστη κάνει ενέργειες στην εφαρμογή μέσω της ομιλίας του.

Όπως φαίνεται και στο κώδικα, αρχικά βάζουμε τιμές στις παραμέτρους του TransformSmoothParameters. Ανάλογα τις τιμές που θα επιλέξουμε, παρατηρούμε αλλαγή στη ροή της κίνησης. Ανοίγουμε το sensor αφού πρώτα καλέσουμε τις DepthStream, ColorStream και προσθέτουμε έναν handler στο sensor που θα χειριστεί το event AllFramesReady. Ολοκληρώνοντας αυτή τη διαδικασία είμαστε έτοιμη να πάρουμε την εικόνα από την κάμερα.

Τέλος αρχικοποιούμε την φωνητική αναγνώριση καλώντας τη συνάρτηση CreateSpeechRecognizer ώστε να καθορίσουμε γλώσσα αρεσκείας και τη γραμματική-λέξεις που θα αναγνωρίζει το Kinect. Όπως θα διαπιστώσετε, ανοίγουν τα μικρόφωνα και το input, από τη φωνητική αναγνώριση, το δεχόμαστε από το AudioStream της Kinect. Η φωνητική αναγνώριση γίνεται ασύγχρονα με την υπόλοιπη εφαρμογή για να

μπορεί ο χρήστης να δίνει εντολές με τη φωνή και να κάνει οποία άλλη ενέργεια επιθυμεί.

```
var parameters = new TransformSmoothParameters
{
     Smoothing = 0.1f,
     Correction = 0.0f,
     Prediction = 0.0f,
     JitterRadius = 0.05f,
     MaxDeviationRadius = 0.5f
};

newSensor.SkeletonStream.Enable(parameters);

AllVoiceCommands();

newSensor.AllFramesReady += new
EventHandler<AllFramesReadyEventArgs>(newSensor_AllFramesReady);
newSensor.DepthStream.Enable(DepthImageFormat.Resolution640x480Fps30);
newSensor.ColorStream.Enable(ColorImageFormat.RgbResolution640x480Fps30);

spRecognizer = CreateSpRecognizer();

try
{
     newSensor.Start();
     newSensor.ElevationAngle = 14;
}
catch (System.IO.IOException)
{
     kinectSensorChooser1.AppConflictOccurred();
}
audsensor.AudioSource.BeamAngleMode = BeamAngleMode.Adaptive;
spRecognizer.SetInputToAudioStream( audsensor.AudioSource.Start(), new
SpeechAudioFormatInfo(EncodingFormat.Pcm, 16000, 16, 1, 32000, 2, null));

spRecognizer.RecognizeAsync(RecognizeMode.Multiple);
audsensor.AudioSource.EchoCancellationMode = EchoCancellationMode.None;
audsensor.AudioSource.AutomaticGainControlEnabled = false;
```

Κώδικας 4.2: Αρχικοποίηση κάμερας και ήχου

Στο επόμενο κομμάτι κώδικα, ελέγχουμε αν η εφαρμογή πάει να κλείσει. Αλλιώς εντοπίζει στη κάμερα τον χρήστη και μπορούμε να πάρουμε όλα τα διαθέσιμα Joints που χρειαζόμαστε για τη κίνηση του. Μετά γίνεται η κλήση της συνάρτησης ProcessGesture. Είναι η μέθοδος που διαχειρίζεται τις κινήσεις του χρήστη από τη κάμερα. Καλείται πολλές φορές μέσα σε ένα δευτερόλεπτο (32 φορές) για να βλέπει όλες τις κινήσεις του χρήστη. Παίρνει σαν ορίσματα όλα τα απαραίτητα Joint για να αναγνωρίζονται όλες οι κινήσεις που θέλουμε να πετύχουμε (Κώδικας 4.3).

```
if (closing)
{
     return;
}

Skeleton first = GetFirstSkeleton(e);

if (first == null)
{
     return;
}

ProcessGesture(first.Joints[JointType.Head],first.Joints[JointType.HandLeft],
first.Joints[JointType.HandRight],first.Joints[JointType.ShoulderRight],
first.Joints[JointType.ShoulderLeft]);
```

Κώδικας 4.3: Αναγνώριση του χρήστη και κλήση της ProcessGesture

Ας μιλήσουμε τώρα για την μέθοδο ProcessGesture και να δούμε αναλυτικά πως πλοηγούμαστε στο διαδίκτυο με τη κίνηση των χεριών μας (Κώδικας 4.4).

Στην αρχή του κώδικα προσαρμόζουμε τις συντεταγμένες των χεριών του χρήστη σε κλίμακα (scale) τέτοια ώστε να μπορεί να φτάνει σε όλο το εύρος της οθόνης χωρίς να μετακινεί όλο το κορμό του. Η κάμερα αναγνωρίζει 3 διαστάσεις. Μήκος ύψος και βάθος (x,y,z). Η λογική πίσω από τις κινήσεις των χεριών του χρήστη είναι πολύ απλή. Βασιζόμαστε σε 5 Joints του χρήστη. Το κεφάλι, το δεξί ώμο, αριστερό ώμο, δεξί χέρι και αριστερό χέρι. Το αριστερό χέρι χρησιμοποιείται για να κουνάμε το mouse cursor και το δεξί για να εκτελεί τις ενέργειες. Το αριστερό χέρι πρέπει να έχει μια απόσταση μεγαλύτερη από 30 εκατοστά από το κεφάλι ως προς τον άξονα z (βάθος) για να κουνηθεί ο mouse cursor. Για ευκολία στην εξήγηση του κώδικα θα ονομάσουμε αυτή τη θέση λειτουργίας. Το δεξί χέρι είναι εκείνο που θα εκτελεί τις ενέργειες.

Οι ενέργειες αυτές (φαίνονται και στο κώδικα παρακάτω) είναι το κλικ, το διπλό κλικ, το δεξί κλικ και το drag & drop. Σε όλες αυτές τις ενέργειες το αριστερό χέρι είναι σε θέση λειτουργίας και το δεξί χέρι σε ύψος μεγαλύτερο του δεξί ώμου.

Η κίνηση για το αριστερό κλικ είναι απλή. Αν προεκτείνουμε το δεξί χέρι (παλάμη) 40 εκατοστά μπροστά από το κεφάλι ως προς τον άξονα z, μεταξύ δεξί ώμου και κεφαλιού (ως προς τον άξονα x) και ύψος μικρότερο του κεφαλιού (άξονα y) τότε εκτελείται το αριστερό κλικ.

Στην ιδία λογική είναι και το διπλό κλικ μόνο που υπάρχει μια διαφορά. Όταν εκτελείται το απλό κλικ τότε ξεκινά ένα χρονόμετρο (stopwatch). Μόνο αν ο χρήστης κάνει 2 κλικ μέσα σε 1 δευτερόλεπτο, εκτελείται το διπλό κλικ. Θυμίζει πολύ στη λογική του ποντικιού και είναι εύκολο να το εμπεδώσουν όλοι.

Το drag & drop γίνεται όταν το δεξί χέρι κάνει κίνηση πίσω από το δεξί ώμο. Όσο παραμένει εκεί το αριστερό κλικ παραμένει πατημένο και μόνο όταν έρθει πάλι μπροστά το χέρι, απελευθερώνεται το κουμπί.

Τέλος, το δεξί κλικ επιτυγχάνεται όταν το δεξί χέρι είναι 25 εκατοστά πάνω από το κεφάλι, μπροστά από το δεξί ώμο και να μην ξεπερνά τα 30 εκατοστά (ως προς το βάθος) το κεφάλι.

Άρα το κεφάλι και ο δεξί ωμός παίζουν βασικό ρόλο ώστε να πραγματοποιηθούν αυτές οι εντολές. Όλες οι κινήσεις με το που εκτελεστούν έχουν μια Boolean που τις αποτρέπει να γίνονται συνέχεια γιατί όπως αναφέραμε και πιο πριν μέσα σε ένα δευτερόλεπτο, ο κώδικας τρέχει 32 φορές.

Για να επανέλθουν αυτές οι Boolean στην αρχική τους κατάσταση και να μπορέσουμε να ξαναπετύχουμε τις παραπάνω λειτουργίες, πρέπει και το δεξί χέρι να βρίσκεται σε κατάσταση λειτουργίας (να μην ξεπερνά τα 30 εκατοστά μπροστά από το κεφάλι).

```
Joint scJointLeft = handleft.ScaleTo(screenWidth, screenHeight, .3f, .3f);
Joint scJointRight = handright.ScaleTo(screenWidth, screenHeight, .3f, .3f);

if (handleft.Position.Z < head.Position.Z - 0.3f)
{
     screenXpos = Convert.ToInt32(scJointLeft.Position.X);
     screenYpos = Convert.ToInt32(scJointLeft.Position.Y);

     if (!GrabItem)
     {
          if (handright.Position.Y > shoulderight.Position.Y)
          {
               if ((handright.Position.Z < shoulderight.Position.Z - 0.4f) &&
(handright.Position.X > head.Position.X) && (handright.Position.Y < head.Position.Y +
0.1f) && cbutton)
               {
                    cbutton = false;
                    stopwatch.Stop();
                    if ((!counterclicks) && (stopwatch.Elapsed.TotalSeconds < 1) &&
(stopwatch.Elapsed.TotalSeconds > 0))
                    {
                         counterclicks = true;
                         mouse_event(MOUSEEVENTF_LEFTDOWN, screenXpos, screenYpos, 0, 0);
                         Thread.Sleep(10);
                         mouse_event(MOUSEEVENTF_LEFTUP, screenXpos, screenYpos, 0, 0);
                         Thread.Sleep(100);
                         mouse_event(MOUSEEVENTF_LEFTDOWN, screenXpos, screenYpos, 0, 0);
                         Thread.Sleep(10);
                         mouse_event(MOUSEEVENTF_LEFTUP, screenXpos, screenYpos, 0, 0);
                         Thread.Sleep(10);
                    }
                    else
                    {
                         counterclicks = false;
                         mouse_event(MOUSEEVENTF_LEFTDOWN, screenXpos, screenYpos, 0, 0);
                         Thread.Sleep(10);
                         mouse_event(MOUSEEVENTF_LEFTUP, screenXpos, screenYpos, 0, 0);
                         Thread.Sleep(10);
                    }
                    stopwatch.Restart();
               }
               if ((handright.Position.X > head.Position.X) && (handright.Position.Z >
shoulderight.Position.Z) && (handright.Position.Y < head.Position.Y) &&
(handright.Position.Y > shoulderight.Position.Y) && (drdrop == 1))
               {
                    drdrop = 2;
                    mouse_event(MOUSEEVENTF_LEFTDOWN, screenXpos, screenYpos, 0, 0);
                    Thread.Sleep(10);
               }
               if ((handright.Position.X > head.Position.X) && (handright.Position.Z >
shoulderleft.Position.Z) && (handright.Position.Y > head.Position.Y + 0.25f) && rbutton)
```

```
                {
                     rbutton = false;
                     mouse_event(MOUSEVENTF_RIGHTDOWN, screenXpos, screenYpos, 0, 0);
                     Thread.Sleep(10);
                     mouse_event(MOUSEVENTF_RIGHTUP, screenXpos, screenYpos, 0, 0);
                     Thread.Sleep(10);
                }
                if ((handright.Position.Z > shoulderight.Position.Z - 0.3f) &&
(handright.Position.Z < shoulderight.Position.Z) && (handright.Position.X >
head.Position.X) && (handright.Position.Y < head.Position.Y + 0.15f))
                {
                     cbutton = true;
                     rbutton = true;
                     newtabutton = true;
                     if (drdrop == 2)
                     {
                          drdrop = 1;
                          mouse_event(MOUSEEVENTF_LEFTUP, screenXpos, screenYpos, 0, 0);
                          Thread.Sleep(10);
                     }
                     else drdrop = 1;
                }
           }
           else
           {
                stopwatch.Stop();
                stopwatch.Reset();
           }
```

Κώδικας 4.4: Αριστερό κλικ. Διπλό αριστερό κλικ, δεξί κλικ, drag & drop

Εκτός από βασικές κινήσεις που κάνουμε με το ποντίκι, έχουμε ενσωματώσει στην εφαρμογή μας το scroll και το zoom (Κώδικας 4.5). Αυτή τη φορά θα δουλέψουμε από την αριστερή πλευρά του κεφαλιού. Αν υποθέσουμε ότι ένας άξονας ως προς το x περνάει από το κεφάλι μας, το οποίο θεωρείται ως το μέσο, οι παραπάνω λειτουργίες γίνονταν στο δεξί εύρος και το scroll και το zoom στο αριστερό.

Παρατηρούμε ότι για να πετύχουμε αυτές τις 2 λειτουργίες πρέπει και τα δυο χέρια να είναι σε προέκταση 50 εκατοστά από το κεφάλι. Πρέπει να βρίσκονται μεταξύ του αριστερού ώμου και του κεφαλιού και να μην ξεπερνάμε το κεφάλι ως προς το ύψος.

Πρώτα θα μιλήσουμε για το zoom. Αν η απόσταση των 2 χεριών είναι μεγαλύτερη από 25 εκατοστά τότε έχουμε zoom in. Αν η απόσταση είναι μικρότερη από 5 εκατοστά τότε έχουμε τη λειτουργία του zoom out. Και σε αυτή τη περίπτωση οι Boolean παίζουν το ρόλο τους και αποτρέπουν να επαναλαμβάνονται συνεχώς οι λειτουργίες. Για αυτό και όταν τα χεριά βρίσκονται σε απόσταση μεταξύ 5 και 20 εκατοστών μεταξύ τους, έχουμε μια κατάσταση ηρεμίας και οι Boolean παίρνουν τις αρχικές τους τιμές.

Στο scroll τώρα, αν ο χρήστης έχοντας και τα 2 χέρια παρατεταμένα, τα κατεβάσουμε προς τα κάτω ενεργοποιείται η λειτουργία του scroll down και η ροδέλα του mouse cursor κατεβαίνει προς τα κάτω. Αντίθετα αν τα παρατεταμένα χέρια ανέβουν προς τα πάνω γίνεται το scroll up. Όπως και με το zoom και εδώ, οι Boolean κάνουν τη δουλειά τους και όταν είμαστε στη ενδιάμεση κατάσταση, οι Boolean παίρνουν πάλι τις πρώτες τιμές τους.

```
              if ((handright.Position.Z < head.Position.Z - 0.5f) &&
(handleft.Position.Z < head.Position.Z - 0.5f) && (handright.Position.X <
head.Position.X) && (handleft.Position.X < head.Position.X))
              {
                   double Xs = Math.Pow(handright.Position.X - handleft.Position.X, 2);
                   double Ys = Math.Pow(handright.Position.Y - handleft.Position.Y, 2);
                   double currentHandDistance = Math.Pow(Xs + Ys, 0.5);

                   if ((currentHandDistance < 0.2f) && (currentHandDistance > 0.1f))
                   {
                        zoominbutton = true;
                        zoomoutbutton = true;
                   }
                   else if (currentHandDistance > 0.25f && zoominbutton)
                   {
                        zoominbutton = false;
                        SendKeys.SendWait("^({ADD})");
                        Thread.Sleep(1000);
                   }
                   else if (currentHandDistance < 0.05f && zoomoutbutton)
                   {
                        zoomoutbutton = false;
                        SendKeys.SendWait("^({SUBTRACT})");
                        Thread.Sleep(1000);
                   }

                   if (handright.Position.Y < 0.10 && handright.Position.Y > -0.10 &&
handleft.Position.Y < 0.10 && handleft.Position.Y > -0.10)
                   {
                          //Do nothing
                   }
                   if ((handright.Position.Y > 0) && (handleft.Position.Y > 0))
                   {
                        mouse_event(MOUSEVENTF_WHEEL, screenXpos, screenYpos, 30, 0);
                        Thread.Sleep(10);
                   }
                   else if (handright.Position.Y < 0 && handleft.Position.Y < 0)
                   {
                        mouse_event(MOUSEVENTF_WHEEL, screenXpos, screenYpos, -30, 0);
                        Thread.Sleep(10);
                   }
              }
         }
         SetCursorPos(screenXpos, screenYpos);
    }
```

Κώδικας 4.5: Zoom και Scroll

Από τις κινήσεις μας, δεν θα μπορούσαν να λείπουν λειτουργίες που αφορούν την πλοήγηση στο διαδίκτυο. Όλοι μας έχουμε χρησιμοποιήσει το back, το forward, το new tab, το next tab και το previous tab. Παρακάτω περιγράφουμε, πως τις προσαρμόσαμε στις κινήσεις του χρήστη (Κώδικας 4.6).

Επειδή είναι κινήσεις που δεν χρειάζονται το mouse cursor, αποφάσισα να εκτελούνται όταν το αριστερό χέρι δεν είναι σε κατάσταση λειτουργίας. Δηλαδή σε μια θέση μεγαλύτερη από τον αριστερό ώμο και μια μικρότερη απόσταση από τα 30 εκατοστά ως προς τον άξονα z. Μια ιδανική τέτοια θέση θα ήταν να κολλήσετε το χέρι σας στο

στήθος σας, κοντά στο ύψος της καρδίας. Έτσι το δεξί χέρι είναι έτοιμο να εκτελέσει τις λειτουργίες που θα του ζητήσουμε.

Αρχικά το next tab έχει ακριβώς ίδια κίνηση με το κλικ. Προέκταση του δεξιού χεριού προς τα μπροστά και πηγαίνουμε στην επομένη καρτέλα της ενεργής ιστοσελίδα μας.

Το previous tab τώρα είναι όπως το drag & drop. Το δεξί χέρι πίσω από τον δεξί ώμο και αυτή τη φορά μεταβαίνουμε στην προηγούμενη καρτέλα.

Όλα αυτά από το δεξί μέρος της κεφαλής. Στο αριστερό τώρα έχουμε το back και το forward με τα οποία πηγαίνουμε στη προηγούμενη και στην επόμενη ιστοσελίδα που είχαμε επιλέξει.

Προέκταση του δεξιού χεριού από την αριστερή πλευρά (μεταξύ αριστερού ώμου και κεφάλι) προς τα μπροστά σε μια απόσταση μεγαλύτερη από 40 εκατοστά από το κεφάλι ως προς τον άξονα z, πτωχαίνουμε το forward. Το back έχει μια ιδιαιτερότητα. Επιτυγχάνεται όταν το δεξί χέρι περάσει τον αριστερό ώμο ως προς τον νοητό άξονα x.

Η μονή κίνηση που γίνεται με το αριστερό χέρι είναι το new tab. Όταν το αριστερό χέρι πάει πίσω από τον αριστερό ώμο τότε μια νέα καρτέλα εμφανίζεται στο πλοηγό διαδικτύου.

Όπως και με το zoom και με το scroll, έχουμε πάλι Boolean που εμποδίζουν τη συνεχή επανάληψη των λειτουργιών.

```
     else
     {
          stopwatch.Stop();
          stopwatch.Reset();
          if ((handleft.Position.Z > shoulderleft.Position.Z) && (handleft.Position.X >
shoulderleft.Position.X) && (handleft.Position.X < head.Position.X) &&
(handleft.Position.Y > shoulderleft.Position.Y) && (handleft.Position.Y <
head.Position.Y) && newtabutton)
          {
               newtabutton = false;
               SendKeys.SendWait("^({t})");
               Thread.Sleep(10);
          }
          if ((handright.Position.Z < shoulderleft.Position.Z - 0.4f) &&
(handright.Position.X < head.Position.X - 0.04f) && (handright.Position.Y <
head.Position.Y) && (handright.Position.Y > shoulderleft.Position.Y) && forwbutton)
          {
               forwbutton = false;
               SendKeys.SendWait("%({RIGHT})");
               Thread.Sleep(10);
          }
          if ((handright.Position.Z < shoulderleft.Position.Z) && (handright.Position.X <
shoulderleft.Position.X) && (handright.Position.Y < head.Position.Y) &&
(handright.Position.Y > shoulderleft.Position.Y) && backbutton)
          {
               backbutton = false;
               SendKeys.SendWait("%({LEFT})");
               Thread.Sleep(10);
          }
          if ((handright.Position.Z < shoulderight.Position.Z - 0.5f) &&
(handright.Position.X > head.Position.X + 0.04f) && (handright.Position.Y <
head.Position.Y) && (handright.Position.Y > shoulderight.Position.Y) && nexttabutton)
          {
               nexttabutton = false;
```

```
                SendKeys.SendWait("^({TAB})");
                Thread.Sleep(10);
            }
            if ((handright.Position.X > head.Position.X + 0.04f) && (handright.Position.Z >
shoulderight.Position.Z) && (handright.Position.Y < head.Position.Y) &&
(handright.Position.Y > shoulderight.Position.Y) && prevtabutton)
            {
                prevtabutton = false;
                SendKeys.SendWait("^+({TAB})");
                Thread.Sleep(10);
            }
            if ((handright.Position.Z > shoulderight.Position.Z - 0.3f) &&
(handright.Position.Z < shoulderight.Position.Z))
            {
                backbutton = true;
                nexttabutton = true;
                prevtabutton = true;
                forwbutton = true;
            }
        }
```

Κώδικας 4.6: New tab, previous tab, next tab, forward page και back page

Αφού τελειώσαμε την περιγραφή για τις κινήσεις, ήρθε η ώρα να μιλήσουμε για τη φωνητική αναγνώριση. Θα εξηγήσουμε πως γίνεται φωνητική αναγνώριση των εντολών του χρήστη και μπορούμε να προβάλουμε λέξεις ή τις αγαπημένες ιστοσελίδες στο πλοηγό διαδικτύου (Κώδικας 4.7).

Ας ξεκινήσουμε με τον καθορισμό του λεξιλογίου μας. Πρώτα παίρνουμε από τη συνάρτηση GetKinectRecognizer τη γλώσσα της αρεσκείας μας (EN-US), ορίζουμε την φωνητική αναγνώριση της Kinect (SpeechRecognitionEngine) και προσθέτουμε τις βασικές λέξεις της εφαρμογής ώστε το αριστερό κλικ, διπλό αριστερό και οι άλλες ενέργειες που αναφέραμε πριν να εκτελούνται με τη φωνή του χρήστη. Προς το τέλος έχουμε 2 επαναλήψεις foreach να προσθέτουν λέξεις σ τη γραμματική μας. Είναι οι διευθύνσεις και οι λέξεις προς αναζήτηση για την πλοήγηση μας στο διαδίκτυο. Αφού ανακτήσουμε τις λέξεις από τα 2 αρχεία κειμένου (searchCommands και urlCommands) τις βάζουμε και αυτές μέσα για να ολοκληρώσουμε την διαδικασία. Επειδή δεν θέλουμε να περιορίσουμε τις λέξεις προς αναζήτηση, χρησιμοποιούμε όλες τις λέξεις που βρίσκονται στη γραμματική.

```
RecognizerInfo ri = GetKinectRecognizer();

SpeechRecognitionEngine speechrec;
speechrec = new SpeechRecognitionEngine(ri.Id);

Choices grammar = new Choices();

grammar.Add("search on");
grammar.Add("search off");
VSCmd.Add("up");
VSCmd.Add("down");
VSCmd.Add("close");
VSCmd.Add("maximize");
VSCmd.Add("minimize");
VSCmd.Add("restore");
```

```
VSCmd.Add("keyboard");
VSCmd.Add("click");
VSCmd.Add("double click");
VSCmd.Add("drag");
VSCmd.Add("drop");
VSCmd.Add("right click");
VSCmd.Add("internet");
VSCmd.Add("url");
VSCmd.Add("forward");
VSCmd.Add("back");
VSCmd.Add("new tab");
VSCmd.Add("close tab");
VSCmd.Add("next tab");
VSCmd.Add("previous tab");
VSCmd.Add("reload");
VSCmd.Add("homepage");
VSCmd.Add("bookmark");
VSCmd.Add("history");
VSCmd.Add("download history");
VSCmd.Add("print page");
VSCmd.Add("save page");
VSCmd.Add("source code");
VSCmd.Add("developer tools");

foreach (KeyValuePair<string, string> pair in VoiceCommands)
{
     try
     {
          grammar.Add(pair.Key);
     }
     catch (ArgumentException)
     {
          //Do nothing
     }
}
foreach (string dkey in VSCmd)
{
     try
     {
          grammar.Add(dkey);
     }
     catch (ArgumentException)
     {
           //Do nothing
     }
}
GrammarBuilder gb = new GrammarBuilder { Culture = ri.Culture };
gb.Append(grammar);
Grammar g = new Grammar(gb);
speechrec.LoadGrammar(g);
```

Κώδικας 4.7: Δημιουργία Λεξιλογίου

Στα τελευταία κομμάτια κώδικα, θα εξηγήσουμε πως χειριζόμαστε τις αναγνωρισμένες φωνητικές εντολές-λέξεις και τι υλοποιούν (Κώδικας 4.8 – Κώδικας 4.9). Όταν μια λέξη αναγνωρίζεται από το Kinect τότε περνάει από μια μεταβλητή, η οποία ελέγχει κατά ποσό είναι σίγουρο το Kinect για αυτή. Αυτή η μεταβλητή ονομάζεται confidence και έχει εύρος τιμών από το 0 μέχρι το 1. Όσο πιο μικρή είναι η τιμή της confidence τόσο το Kinect δεν είναι σίγουρο για την εντολή αυτή. Αν μια λέξη γίνει δεκτή, αφήνει εμάς να αποφασίσουμε πως θα την διαχειριστούμε. Στη προκειμένη περίπτωση, αν η λέξη που αναγνωρίστηκε, έχει confidence μικρότερο από το 0,75 , απορρίπτουμε τη λέξη.

Αφού η λέξη γίνει δεκτή, τότε επιλεγούμε τις ενέργειες που θα κάνει η κάθε μια. Αυτό φαίνεται μέσα στην συνθήκη switch-case. Για παράδειγμα αν ο χρήστης προφέρει τη λέξη «UP» τότε η κάμερα του Kinect αλλάζει τη γωνία του προς τα πάνω κατά 5 μοίρες. Αν ο χρήστης πει τη λέξη «MINIMIZE» τότε το παράθυρο που έχουμε ενεργό εκείνη τη στιγμή, ελαχιστοποιείται.

```
if (e.Result.Confidence < .75f)
{
     RejectSpeech(e.Result, e.Result.Confidence);
     return;
}
IntPtr state = GetForegroundWindow();

VoiceCommands.Clear();
AllVoiceCommands();

String rec;
Console.WriteLine("Accept: {0} with confidence : {1}", e.Result.Text.ToUpperInvariant(),
e.Result.Confidence);

if (e.Result.Text.ToUpperInvariant() == "SEARCH ON")
{
     System.Windows.Forms.MessageBox.Show(new Form() { TopMost = true }, "Αναζήτηση mode:
ON");
     Onsearch = true;
}

if (e.Result.Text.ToUpperInvariant() == "SEARCH OFF")
{
     System.Windows.Forms.MessageBox.Show(new Form() { TopMost = true }, "Αναζήτηση mode:
OFF");
     Onsearch = false;
}

if (!Onsearch)
{
     switch (rec = e.Result.Text.ToUpperInvariant())
     {
          case "UP":
               audsensor.ElevationAngle += 5;
               break;
          case "DOWN":
               audsensor.ElevationAngle -= 5;
               break;
          case "CLOSE":
               SendMessage(state, WM_CLOSE, IntPtr.Zero, IntPtr.Zero);
               break;
```

Κώδικας 4.8: Αναγνώριση λέξης

Επειδή θέλουμε να ξεχωρίζουμε πότε θα αναζητούμε λέξεις και πότε να ζητάμε τις διευθύνσεις, χρησιμοποιούμε μια Boolean για αυτή τη δουλειά. Άρα όταν η Onsearch είναι true τότε χρησιμοποιούμε τη φωνητική αναγνώριση για να αναζητούμε λέξεις στο διαδίκτυο. Αν είναι false τότε προφέρουμε τις διευθύνσεις μας. Τόσο στο Dictionary (για τις διευθύνσεις) όσο και για τη List (αναζήτηση λέξεων) ψάχνουμε μέσω τις foreach για λέξεις-εντολές που έχουμε κατοχυρώσει και μέσω της SendKeys, τις εμφανίζουμε στα πεδία κειμένου που επιθυμούμε.

```
if (!Onsearch)
{
     foreach (KeyValuePair<string, string> pair in VoiceCommands)
     {
          if (e.Result.Text.ToUpperInvariant().Equals(pair.Key.ToUpperInvariant()))
          {
               char[] input = pair.Key.ToCharArray();
               foreach (char ch in input)
               {
                    if (ch != ' ')
                         SendKeys.SendWait("{" + ch + "}");
                    else
                         SendKeys.SendWait(" ");
                    }
                    SendKeys.SendWait("{ENTER}");
               }
          }
}

if (Onsearch)
{
     foreach (string dkey in VSCmd)
     {
          if (e.Result.Text.ToUpperInvariant().Equals(dkey.ToUpperInvariant()))
          {
               char[] inputs = dkey.ToCharArray();
               foreach (char chs in inputs)
               {
                    if (chs != ' ')
                         SendKeys.SendWait("{" + chs + "}");
                    else
                         SendKeys.SendWait(" ");
                    }
                    SendKeys.SendWait(" ");
               }
          }
     }
}
```

Κώδικας 4.9: Αναγνώριση διευθύνσεων και λέξεων προς αναζήτηση

5. ΠΙΘΑΝΕΣ ΕΠΕΚΤΑΣΕΙΣ ΕΦΑΡΜΟΓΗΣ

Σε αυτή την ενότητα θα συζητήσουμε για τυχόν αλλαγές ή προσθήκες που θα μπορούσαμε να βάλουμε στο πρόγραμμα, οι οποίες λόγω όγκου δεν μπορούσαν να υλοποιηθούν στα πλαίσια της πτυχιακής μου. Άρα είναι μια καλή ευκαιρία, για όποιον θα ήθελε να συνεχίσει την εργασία μου, να τα προσθέσει ώστε η εφαρμογή να γίνει πιο λειτουργική και πιο εύχρηστη για αυτόν που τη χρησιμοποιεί.

5.1 Αναγνώριση της Ελληνικής Γλώσσας στο Kinect

Όταν κατανόησα όλες της πτυχές της φωνητικής αναγνώρισης, μπήκα στη διαδικασία και πρόσθετα ελληνικές λέξεις ως εντολές αφού η πτυχιακή μου είναι στα ελληνικά και θα μπορούσε να απευθύνει στο κόσμο που κάνει χρήση της ελληνικής γλώσσα. Φανταστείτε πόσο ωραία θα ήταν να λέμε λέξεις από τη μητρική μας γλώσσα και να διασκεδάζουν με το πρόγραμμα και άτομα τα οποία δεν κατέχουν την αγγλική γλώσσα. Άμα ανατρέξετε στο κώδικα μου, υπάρχει ένα σημείο στο SpeechRecognitionEngine όπου προσπαθώ να ανακτήσω από το Kinect τη γλώσσα που θα χρησιμοποιεί ο χρήστης για να προφέρει τις λέξεις-εντολές του. Κάνοντας μια αναζήτηση στο διαδίκτυο, προσπάθησα να βρω ένα language pack για το Kinect στα ελληνικά στην ιστοσελίδα της Microsoft αλλά δυστυχώς δεν υπήρχε κάτι. Άρα χρησιμοποίησα αναγκαστικά το πακέτο EN-US (English USA) και όλες οι φωνητικές εντολές είναι στα αγγλικά.

Μια καλή άσκηση λοιπόν θα ήταν κάποιος να φτιάξει από την αρχή ένα language pack για το Kinect χρησιμοποιώντας τα ελληνικά. Σίγουρα μια τέτοια διαδικασία δεν θα είναι εύκολη. Η ελληνική γλώσσα θεωρείται μια από τις πιο πολύπλοκες, με τεράστιο όγκο λέξεων αλλά πλέον υπάρχουν τα εργαλεία και οι βάσεις για να το φτιάξει κάποιος από το μηδέν.

5.2 Extension για το Google Chrome

Η google δίνει τη δυνατότητα στους χρήστες του Google Chrome να χρησιμοποιούν extensions τα οποία κάνουν την πλοήγηση μας στο διαδίκτυο πιο διασκεδαστική και πιο ευχάριστη. Από την επεξεργασία της διεπαφής του Google Chrome, τη πρόσβαση σε πληροφορίες άλλων ιστοσελίδων, παιχνίδια, χάρτες είναι μόνο λίγα από τις εκατοντάδες δυνατότητες που προσφέρουν τα extensions τα οποία κυκλοφορούν στο Google Store της Google.

Μια καλή πρόταση θα ήταν να δημιουργήσουμε ένα extension του Google Chrome βασισμένο στην πτυχιακή μου. Οποιαδήποτε στιγμή ο χρήστης έμπαινε στο Google Chrome, θα μπορούσε να ενεργοποιήσει το extension και όλες οι λειτουργιές του προγράμματος θα ήταν στην διάθεση του. Έτσι δεν θα χρειάζεται η εργασία να τρέχει από πίσω και να πιάνει τόσο χώρο στην μνήμη του υπολογιστή.

5.3 Καταγραφή Κινήσεων και Μετατροπή τους σε λέξεις-εντολές

Στην κοινωνία μας υπάρχουν πολλά άτομα με ειδικές ανάγκες τα οποία έρχονται καθημερινά σε επαφή με τον υπολογιστή τους και πιο συγκεκριμένα με το διαδίκτυο.

Μια κατηγορία είναι και τα άτομα με πρόβλημα ομιλίας και ακοής. Όπως καταλαβαίνετε, η χρήση του υπολογιστή αλλά και του διαδικτύου είναι περιορισμένη για αυτούς επειδή δεν μπορούν να επικοινωνήσουν με τον συνομιλητή τους παρά μόνο με τη χρήση του γραπτού λόγου. Άρα για τα συγκεκριμένα άτομα, η εφαρμογή μας τους περιορίζει σε μεγάλο βαθμό. Ένα μεγάλο ποσοστό της εργασίας βασίζεται στη φωνητική αναγνώριση και στην καταχώρηση λέξεων-εντολών. Μια λύση σε αυτό θα ήταν να καταγράφαμε κινήσεις και να τις αντιστοιχούσαμε σε ένα λεξιλόγιο ώστε η αλληλεπίδραση με το Kinect να γίνεται εξολοκλήρου με αυτές.

Μια καλή ιδέα θα ήταν να είχαμε μια φόρμα όπου ο χρήστης θα έφτιαχνε ένα σύνολο κινήσεων, χρησιμοποιώντας όλο το σώμα του ώστε να καλύψει όσο τη δυνατόν περισσότερες λέξεις της γραμματικής. Η φόρμα θα είναι της ίδιας λογικής με την φόρμα με τις αγαπημένες μας διευθύνσεις αλλά αντί για αυτές, θα βάζαμε τις κινήσεις της επιλογής μας.

Θέλει όμως προσοχή επειδή τα Joints που παίρνει το Kinect σε συνδυασμό με το ότι ανά 1 δευτερόλεπτο η εφαρμογή περνά από την κάμερα 32 φορές, μας δυσκολεύουν σε πολλές σύνθετες κινήσεις. Για αυτό και υπάρχουν εργαλεία με tutorial στο διαδίκτυο που λύνουν αυτό το πρόβλημα.

6. ΠΗΓΕΣ ΑΝΑΖΗΤΗΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ

Το διαδίκτυο πλέον είναι μια μεγάλη βάση πληροφοριών, έτοιμη να εξυπηρετήσει τους χρήστες του. Όσο και αν με ενθουσιάζει μια πτυχιακή εργασία πάνω στο Kinect έπρεπε κάποιος να έχει και τα κατάλληλα εφόδια για να ασχοληθεί μαζί του. Να είναι έτοιμος να αντιμετωπίσει όλα τα προγραμματιστικά κενά και δυσκολίες που θα έβρισκε και να τα έλυνε.

Η αναζήτηση αυτής της πληροφορίας δεν ήταν εύκολη. Λόγω των συνεχών αναβαθμίσεων που γίνονται στο λογισμικό του αλλά και της κυκλοφορίας της νέας κάμερας, καθιστούσε την εύρεση των νέων αλλαγών και των tutorial δύσκολη.

Μετά από αρκετό ψάξιμο, κατέληξα σε μερικές ιστοσελίδες τις οποίες και θα σας παραθέσω.

6.1 Φροντιστηριακοί Οδηγοί και Κοινότητες για το Kinect

Θα δούμε ιστοσελίδες με βίντεο και παραδείγματα, ώστε να κατανοήσουμε στο μέγιστο πως είναι να προγραμματίζεις με το Kinect.

6.1.1 Microsoft Developer Network

Η βάση δεδομένων της Microsoft παρέχει ένα πλήρες όγκο με πληροφορίες για το Kinect. Αν θέλει κάποιος να προγραμματίσει μια εφαρμογή σε αυτό, πρέπει να το εγκαταστήσουμε στα Windows του υπολογιστή. Πληροφορίες για τον τρόπο εγκατάστασης θα δείτε στη ιστοσελίδα της Microsoft

http://www.microsoft.com/en-us/kinectforwindows/

και

http://msdn.microsoft.com/en-us/library/hh855356.aspx

Για τη φωνητική αναγνώριση θα πρέπει να εγκαταστήσουμε διαφορετικό πακέτο ώστε να έχουμε τον έλεγχο της μηχανής της φωνητικής αναγνώρισης. Για να κατεβάσετε το πακέτο δείτε στον παρακάτω σύνδεσμο

http://www.microsoft.com/en-us/download/details.aspx?id=27226

Τέλος, αν θέλουμε το kinect να αναγνωρίζει κάποια άλλη γλώσσα εκτός της αγγλική, αρκεί να κατεβάσουμε κάποιο άλλο πακέτο

http://www.microsoft.com/en-us/download/details.aspx?id=34809

Αν προτιμάτε να δείτε σε βίντεο κάποια από τα παραπάνω, περισσότερα στην παρακάτω διεύθυνση

http://www.microsoft.com/en-us/kinectforwindows/develop/how-to-videos.aspx

και

http://www.microsoft.com/en-us/kinectforwindowsdev/Community.aspx

6.1.2 Channel 9

Ένα αρκετά χρήσιμο website με πλούσια video και αναλύσεις για το πώς θα προγραμματίσουμε στο Kinect. Ίσως το πιο αναλυτικό κείμενο αφού αναλύει διεξοδικά πως λειτουργούν η κάμερα, η φωνητική αναγνώριση, η γραμματική και πώς να φτιάξουμε δίκες μας κινήσεις. Χαρακτηριστικά υπάρχει ένας σύνδεσμος με 6 βίντεο που θα δώσει τα πρώτα βήματα για την υλοποίηση της άσκησης.

http://channel9.msdn.com/Series/KinectQuickstart

6.1.3 DevelopKinect

Η ιστοσελίδα http://developkinect.com/ , στοχεύει σε όλους εκείνους τους ανθρώπους που ενδιαφέρονται για την ανάπτυξη εφαρμογών με τη χρήση του Kinect. Υπάρχει ένα μεγάλο σύνολο διαθέσιμων πόρων. Αν σας ενδιαφέρει ένα συγκεκριμένο θέμα πάνω στο Kinect, θα βρείτε περισσότερα από όσα θα χρειαστείτε στο φόρουμ του.

6.1.4 KinectHacks

Είναι μια κοινότητα αφιερωμένη στη συνεχή παροχή ειδήσεων, πηγαίου κώδικα και πληροφοριών σχετικά με τις τελευταίες και πιο καινοτόμες χρήσεις του Kinect. Πιστεύουν στις τεράστιες δυνατότητες που διαθέτει η Kinect. Αποτελείται από άτομα-χρήστες με δημιουργικότητα και εφευρετικότητα στην ανάπτυξη κώδικα.

http://www.kinecthacks.com/

6.1.5 OpenKinect

Είναι μια ανοιχτή κοινότητα για ανθρώπους που ενδιαφέρονται για τη χρήση του Kinect τόσο με τους υπολογιστές όσο και με άλλες συσκευές. Υπάρχουν βιβλιοθήκες ανοικτού κώδικα οι οποίες θα επιτρέψουν στο Kinect να χρησιμοποιείται με τα Windows, Linux και Mac. Η κοινότητα OpenKinect αποτελείται από πάνω από 2000 μέλη που συνεισφέρουν ενεργά. Είναι μια πραγματική "Open Source" κοινότητα.

http://openkinect.org/wiki/Main_Page

7. ΣΥΜΠΕΡΑΣΜΑΤΑ

Το διαδίκτυο έχει γίνει ένα απαραίτητο εργαλείο για τον άνθρωπο, αφού συνέβαλε στο να έρθει πιο κοντά με πράγματα που επιθυμεί. Χρησιμοποιείται παντού, παρέχει πληροφόρηση, ψυχαγωγία, διευρύνει την επικοινωνία και εκπαιδεύει τις μάζες. Η πλοήγηση στο διαδίκτυο γινόταν εξολοκλήρου από προσωπικό υπολογιστή. Με τη συνεχόμενη εξέλιξη της τεχνολογίας αυτό άλλαξε. Πλέον ο χρήστης μπορεί να πλοηγεί από laptop, κινητά και tablets. Αλλά δεν μείναμε μόνο εκεί. Αρχίσαμε να χρησιμοποιούμε συσκευές για να βελτιώσουμε την πλοήγηση. Μια από αυτές τις συσκευές είναι και η κάμερα Kinect.

Μέσα από την ενασχόληση μου με αυτή, ανακάλυψα όλες τις δυνατότητες που μπορεί να προσφέρει. Κατάφερα να εξερευνήσω όλες τις πτυχές της κάμερας υλοποιώντας κινήσεις με τα χεριά και μέσω της φωνητικής αναγνώρισης να δημιουργήσω φωνητικές εντολές. Και είναι και αρκετά εύχρηστη. Μέσα σε λίγο διάστημα και με βασικές γνώσεις προγραμματισμού μπόρεσα να βγάλω κάποια αξιόλογα αποτελέσματα.

Άρα πέρα από το διασκεδαστικό μέρος της υλοποίησης, η διαφορετικότητα της πλοήγησης που μας δίνει, τυγχάνει να έχει και πολλά πλεονεκτήματα.

Καταργεί πλήρως το πληκτρολόγιο και το ποντίκι. Χαρακτηριστικό παράδειγμα εφαρμογής είναι στην ώρα της διδασκαλίας. Ο καθηγητής θα μπορούσε μέσα από κινήσεις και φωνητικές εντολές να πλοηγηθεί χωρίς να ασχολείται με τα περιφερειακά μέσα του υπολογιστή και να επικεντρωθεί στο μάθημα του. Μια άλλη αξιόλογη περίπτωση είναι η βοήθεια που θα μπορούσε να προσφέρει σε άτομα με ειδικές ανάγκες. Άτομα με πρόβλημα όρασης, ακοής και μερικής κίνησης, θα κατάφερναν να περιηγηθούν στο κόσμο του διαδικτύου με άνεση, απολαμβάνοντας τις χάρες του.

Μπορεί βεβαία να υπάρχουν και κάποια αρνητικά σε αυτή την εξ αποστάσεων πλοήγηση. Πολλοί πιστεύουν ότι μια τέτοια εφαρμογή θα κούραζε το χρηστή λόγω της συνεχής κίνησης των χεριών, οδηγώντας τον σε λάθη. Μια άλλη περίπτωση είναι να βρίσκεται η κάμερα του Kinect σε ένα χώρο με πολλά άτομα που θα μιλάνε όλα μαζί. Αυτό θα δημιουργούσε προβλήματα στην αναγνώριση του χειριστή. Το Kinect θα μπερδευόταν γιατί το πρόγραμμα είναι σχεδιασμένο ώστε να αναγνωρίζει ένα άτομο τη φορά με αποτέλεσμα να μην υπακούει στις κινήσεις μας. Το ίδιο ισχύει και με την φωνητική αναγνώριση. Μέσα από όλη αυτή την φασαρία, η συσκευή θα ήταν σε σύγχυση και μπορεί να αναγνώριζε λέξεις που δεν άκουσε από εμάς.

Πιστεύω ότι αυτή η εργασία είναι μια καλή αρχή στη προσπάθεια του ανθρώπου για μια καλύτερη και ευκολότερη πλοήγηση στο διαδίκτυο. Κάποια από αυτά είναι σε πρώιμο στάδιο, κάποια θα μπορούσαν να αξιοποιηθούν από τώρα. Ο άνθρωπος δείχνει ότι δεν επαναπαύεται. Θέλει να εξελίσσεται και βρίσκει τρόπους ώστε να κάνει την καθημερινότητα του ακόμα πιο εύκολη. Ίσως τελικά μας λείπει η τεχνολογία για ένα ικανοποιητικό αποτέλεσμα. Ίσως και όχι. Το Kinect όμως δείχνει το σωστό δρόμο και όσο περνάει ο καιρός θα έχει να μας δείξει ακόμα περισσότερα.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
3D Depth Sensor	Τρισδιάστατος αισθητήρας βάθους
Audio	Ήχος
Back	Πίσω
Bookmark	Σελιδοδείκτης
Constructor	Κατασκευαστής
DataGridView	Δεδομένα σε πίνακα
Data streams	Ροή δεδομένων
Dictionary	Λεξικό
Developer Tools	Εργαλεία προγραμματιστή
Event	Γεγονός
Forward	Μπροστά
Frames per second	Καρέ ανά δευτερόλεπτο
Handler	Χειριστής
Homepage	Αρχική σελίδα
Hypothesized	Υποθέτουμε
Internet	Διαδίκτυο
Joints	Αρθρώσεις
Kinect Explorer	Πλοήγηση Kinect
List	Λίστα
Mouse cursor	Δείκτης ποντικιού
Multi-array microphone	Πολλαπλές σειρές μικροφώνων
New	Νέο
Previous	Προηγούμενο
Recognized	Αναγνωρίστηκε
Rejected	Απορρίφθηκε
Reload	Ανανέωση
Scale	Κλίμακα
Scroll up - down	Μετακίνηση προς τα πάνω - κάτω
Sensor	Αισθητήρας
Source code	Πηγαίος κώδικας
Speech Recognition	Αναγνώριση φωνής
Tab	Καρτέλα
Text	Κείμενο
«time of flight»	«Χρόνος πτήσης»
Voice Commands	Ηχητικές εντολές
Website	Ιστοσελίδα
Windows Form	Φόρμα Windows
Zoom in – out	Μεγέθυνση - Σμίκρυνση

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

API	Application Programming Interface
DLL	Dynamic-Link Library
HTML	Hypertext Markup Language
KHz	Kilohertz
RGB	Red Green Blue Camera
USB	Universal Serial Bus
VGA	Video Graphics Array/Adaptor
WPF	Windows Presentation Foundation

ΑΝΑΦΟΡΕΣ

[1] http://electronics.howstuffworks.com/microsoft-kinect2.htm
[2] http://kotaku.com/5576002/here-are-kinects-technical-specs
[3] http://www.triballabs.net/2011/06/kinectapis/
[4] https://wiki.zimt.uni-siegen.de/fertigungsautomatisierung/index.php/Einsatzm%C3%B6glichkeiten_einer_3D-Kamera_in_der_Produktionstechnik_am_Beispiel_der_Kinect-Kamera
[5] http://msdn.microsoft.com/en-us/magazine/jj650892.aspx
[6] http://www.openrobots.org/morse/doc/latest/user/sensors/mocap_posture.html
[7] http://blogs.msdn.com/b/eternalcoding/archive/2013/02/26/kinect-dawanoid-how-to-create-an-ball-game-controlled-only-with-ambient-noise.aspx
[8] http://msdn.microsoft.com/en-us/magazine/hh882450.aspx
[9] http://channel9.msdn.com/Series/KinectQuickstart
[10] http://msdn.microsoft.com/en-us/library/hh855356.aspx
[11] http://msdn.microsoft.com/en-us/library/windows/desktop/ff468919(v=vs.85).aspx
[12] http://msdn.microsoft.com/en-us/library/system.windows.forms.sendkeys.send.aspx
[13] http://www.microsoft.com/en-us/kinectforwindowsdev/Community.aspx
[14] http://developkinect.com/
[15] http://www.kinecthacks.com/
[16] http://openkinect.org/wiki/Main_Page