ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΤΟΜΙΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δεκέμβριος 2013

Ατομική Διπλωματική Εργασία

Emotion recognition from body signals and EEGs in a game scenario

Ιωάννης Δημητρίου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Δεκέμβριος 2013

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Emotion recognition from body signals and EEGs in a game scenario

Ιωάννης Δημητρίου

Επιβλέπων Καθηγητής

Γιώργος Χρυσάνθου

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Δεκέμβριος 2013

Ευχαριστίες

Ευχαριστώ τον πατέρα μου που με στήριξε και με βοήθησε σε όλη μου τη ζωή μέχρι την μέρα του θανάτου του. Η Διπλωματική αυτή είναι αφιερωμένη σε αυτόν.

Περίληψη

Ο όρος Affective Computing χρησιμοποιείται για τις τεχνολογίες που χρησιμοποιούνται για την αναγνώριση συναισθήματος του ανθρώπου από έναν υπολογιστή. Κάτι το οποίο μπορεί να γίνει με την χρήση διαφόρων τεχνολογιών που διαβάζουν είτε την στάση σώματος είτε εκφράσεις του προσώπου. Εφαρμογές υπάρχουν πάρα πολλές και κυρίως σε αυτιστικά άτομα ή άτομα με πλήρη σωματική παράλυση, που έχουν δυσκολίες στην έκφραση των συναισθημάτων τους καθώς και σε άτομα που πάσχουν από αλεξιθυμία Μπορούν να χρησιμοποιηθούν από τους γονείς, τους θεραπευτές ή τους νοσοκόμους των συγκεκριμένων ατόμων.

Συγκεκριμένα σε αυτή τη διπλωματική σκοπός ήταν η αναγνώριση συναισθήματος σε ένα σενάριο παιγνίου με τη χρήση των σωματικών κινήσεων και EEG και κατά πόσο μπορεί το παιχνίδι να μεταβληθεί για να κρατήσει περισσότερο το ενδιαφέρον του παίκτη και να τον κάνει να το απολαμβάνει περισσότερο.

Την αναγνώριση σωματικών κινήσεων δεν την χρησιμοποιήσαμε στο τέλος λόγω έλλειψης χρόνου αλλά καταγράψαμε δεδομένα που μπορεί να χρησιμοποιηθούν ως database για μελλοντική χρήση.

Table of contents

Table of Figures

Κεφάλαιο 1: Εισαγωγικά

1.0 Εισαγωγή

Το πόσο συνεχίζει να λαμβάνει μέρος σε ένα παιχνίδι ένας παίκτης επηρεάζεται πάρα πολύ από τη συναισθηματική του κατάσταση. Αν για παράδειγμα βαριέται, δυσκολεύεται υπερβολικά και νιώθει διάφορα άλλα αρνητικά συναισθήματα είναι πολύ πιθανό να σταματήσει το παιχνίδι ή να παίξει κάποιο άλλο παιχνίδι. Αντιθέτως όταν το παιχνίδι του προσφέρει ευχαρίστηση και ευφορία είναι πολύ πιθανό πώς θα συνεχίζει να ασχολείται με το ίδιο παιχνίδι. Αν λοιπόν μεταβάλλουμε το παιχνίδι ανάλογα με το συναίσθημα του παίκτη τότε θα μπορούμε να κρατήσουμε τον παίκτη στο παιχνίδι για περισσότερο χρονικό διάστημα. Ο τομέας μελέτης τα Ατομικής Διπλωματικής μου εργασίας εμπίπτει στον τομέα του Affective Computing.

1.1 Τι είναι το Affective Computing;

Η λέξη Affection σημαίνει συναίσθημα. Μόνο με τον όρο αυτό είναι εύκολο να κατανοηθεί τι σημαίνει ο όρος «Affective Computing». Ο όρος αυτός χρησιμοποιείται για οτιδήποτε έχει σχέση με υπολογιστές και περιλαμβάνει συναίσθημα. Συγκεκριμένα αναφέρετε στην μελέτη και τη δημιουργία πληροφορικών συστημάτων και συσκευών οι οποίες έχουν την δυνατότητα να αναγνωρίζουν, να μεταφράζουν να επεξεργάζονται και να μιμούνται ανθρώπινα συναισθήματα. Τα συστήματα αυτά ανάλογα με το συναίσθημα που αναγνωρίζουν μπορούν είτε να δίνουν αναφορά στο χρήστη τους ή να μεταβάλλουν τις πράξεις τους ανάλογα με το τι ανιχνεύουν. [9]

1.2 Σκοπός και συνολική διαδικασία της Ατομικής Διπλωματικής Εργασίας

Σκοπός της Διπλωματικής αυτής εργασίας είναι η δημιουργία ενός συστήματος και μίας βάσης δεδομένων με καταγραμμένες κινήσεις που αντιπροσωπεύουν συναισθήματα. Θα δημιουργηθεί μία εφαρμογή η οποία χρησιμοποιώντας τα δεδομένα των πειραμάτων μας θα εκπαιδευτεί και θα μπορεί πλέον και με την χρήση του Microsoft Kinect και το Emotiv EPOC να αναγνωρίζει τα συναισθήματα ενός ανθρώπου από τα Body Signals, καθώς και από τα Brain Signals. Έτσι λοιπόν, καθώς αναπτύσσονται τα παιχνίδια στα οποία ο χρήστης χρησιμοποιεί ολόκληρο το σώμα του για να λάβει μέρος στα παιχνίδια, προσπαθούμε να δημιουργήσουμε μία εφαρμογή που να μπορεί να αναγνωρίζει τα συναισθήματα του παίκτη και να μπορεί να μεταβάλλει το παιχνίδι ανάλογα με το συναίσθημα του. Επίσης αυτό μπορεί να εφαρμοστεί και σε καταστάσεις έξω από παιχνίδια. Για παράδειγμα σε άτομα με αυτισμό που δυσκολεύονται να εκφράσουν την άποψη τους και το τι νιώθουν και οι πράξεις τους. Ίσως σε μελλοντικό στάδιο να μπορέσει να βοηθήσει άτομα με αλεξιθυμία να εκφράσουν το συναίσθημα τους στους άλλους.

Στη διπλωματική αυτή καταγράψαμε διάφορες μορφές Body Signals για συναισθήματα αλλά προσπαθήσαμε να βρούμε τη συσχέτιση των EEGs με το συναίσθημα. Τα Body Signals δεν τα χρησιμοποιήσαμε για αναγνώριση συναισθήματος παρά μόνο δημιουργήσαμε μία βάση δεδομένων για μελλοντική μελέτη.

1.2.1 Τι είναι η αλεξιθυμία;

Αλεξιθυμία είναι η κατάσταση που χαρακτηρίζει τους ανθρώπους που δυσκολεύονται ή γενικά είναι ανήμποροι να εκφράσουν ή να κατανοήσουν το συναίσθημα τους. [1] Οι άνθρωποι που βρίσκονται στην κατάσταση που περιγράφεται ως αλεξιθυμία έχουν τεράστια δυσκολία στο να αναγνωρίσουν συναισθηματικές καταστάσεις καθώς εκδηλώνονται. Μπορεί να έχουν μια αίσθηση, όταν βρίσκονται σε μια πολύ δυνατή συναισθηματική κατάσταση, όπως θλίψη με δάκρυα ή ασυγκράτητος θυμός, αλλά όταν προσπαθούν να καταλάβουν τι προκάλεσε αυτή την κατάσταση, δεν μπορούν να καθορίσουν τι ήταν αυτό που τους ερέθισε. Οι περισσότεροι μπορεί να αισθάνονται άβολα, σαν κάτι να αλλάζει στο σώμα τους, όπως αυξημένη καρδιακή πίεση ή πόνοι στο στομάχι, και όταν πιέζονται να εξηγήσουν τα συναισθήματα τους δεν έχουν λέξεις για να προσφέρουν. Μπορεί να προσπαθήσουν με αδέξιο τρόπο να δώσουν μια αριστοτεχνική απάντηση, ίσως να απαντήσουν αυτό που θέλουν οι άλλοι να ακούσουν, ή απλά να αλλάξουν θέμα στην κουβέντα.

Επίσης, το άτομο με αλεξιθυμία μπορεί να παρερμηνεύσει τη σωματική έκφραση του συναισθήματος σαν σωματική έκφραση ασθενειών. Για παράδειγμα, τα δάκρυα στα μάγουλα δεν είναι έκφραση στενοχώριας, αλλά ανωμαλία στους δακρυϊκούς αδένες ή μια καρδιά που πάλλεται δυνατά και γρήγορα λόγω πάθους ερμηνεύεται ως προβληματική βαλβίδα στην καρδιά κλπ. Ανάλογα μπορούν να θεωρηθούν οι συναισθηματικές εκδηλώσεις ως αποτέλεσμα των περιβαλλοντολογικών συνθηκών (μολυσμένος αέρας, αλλαγή στην ατμοσφαιρική πίεση κλπ).

Έτσι με τη βοήθεια του συστήματος τα άτομα αυτά θα μπορούν να αναγνωρίζουν και οι ίδιοι ακριβώς τι νιώθουν και τη στιγμή που ξεκίνησε το συναίσθημα τους αυτό, βοηθώντας τους έτσι να βρουν και τι προκάλεσε το συναίσθημα τους αυτό.

Ουσιαστικά ένα τέτοιο σύστημα μπορεί να χρησιμεύσει στην θεραπεία καθώς και στην ευκολότερη έκφραση τέτοιων ατόμων.

1.3 Τι είναι το συναίσθημα;

Όλοι γνωρίζουμε ότι το συναίσθημα είναι μία κατάσταση που δεν έχει σχέση με την λογική και το «ζύγισμα» των πιθανών επιλογών αλλά είναι περισσότερο συνδεδεμένο με τον παρορμητισμό. Γενικά όμως υπάρχει μία τεράστια ασάφεια γύρω από το συναίσθημα και τι είναι. Πολλοί θα πουν ότι είναι καθαρά θέμα της ψυχής. Κάποιοι ακόμα πιστεύουν πως το συναίσθημα εδρεύει στην καρδιά. Τι είναι λοιπόν το συναίσθημα;

Το συναίσθημα είναι μια σύνθετη υποκειμενική συνειδητή εμπειρία : συνδυασμός νοητικών καταστάσεων, ψυχοσωματικών εκφράσεων και βιολογικών αντιδράσεων του σώματος. Είναι αυτό που ένας άνθρωπος «αισθάνεται», όχι ως απλή αίσθηση αλλά ως κάτι βαθύ, εσωτερικό, που επιδρά στο σώμα (πχ καρδιακός ρυθμός) και την «ψυχή» του και σχεδόν πάντα εκφράζεται (στο πρόσωπο, στη φωνή, στη στάση του σώματος) και είναι παρατηρήσιμο από τους άλλους.

1.4 Προηγούμενη Εργασία στον Τομέα

Ο τομέας μελέτης της στάσης του σώματος και διαφόρων κινήσεων και η σχέση που έχουν με το συναίσθημα είναι πολύ μεγάλος.

1.4.1 Brain Reaction when viewing emotions

Είναι αποδεδειγμένο ότι η Αμυγδάλα του εγκεφάλου (το σημείο που είναι υπεύθυνο για πάμπολλες λειτουργίες του εγκεφάλου, μία από αυτές είναι το συναίσθημα) αντιδρά με διάφορους τρόπους ανάλογα με το συναίσθημα που βλέπει. Σε πολλές όμως από τις μελέτες αυτές ασχοληθήκαν μόνο με τις εκφράσεις προσώπου και την συσχέτιση τους με το συναίσθημα. [2]

1.4.2 Brain's Reaction When viewing emotion expressed solely by Body Posture

Σε πρόσφατο Paper των Νικόλαο Σάββα, Alfonsina Scarinzi και Nadia Bianchi-Berthouze προέβαλαν ολόκληρο σώμα ανθρώπου με το πρόσωπο θολωμένο έτσι ώστε να μετρηθεί κατά πόσο η στάση του σώματος και μόνο μπορεί να επηρεάσει τη αντίδραση της Αμυγδάλας. [3] Αποδείχτηκε λοιπόν με την προβολή αυτών των εικόνων σε «πειραματόζωα» και τη μέτρηση των εγκεφαλικών σημάτων που εκπέμψαν ανάλογα με την εικόνα που έβλεπαν βρέθηκε ότι το ανθρώπινος εγκέφαλος μπορεί να εκλάβει το συναίσθημα του άλλου ατόμου όχι μόνο από την έκφραση του προσώπου αλλά και από τη στάση του σώματος. Έτσι λοιπόν αφού ο ανθρώπινος εγκέφαλος έχει την δυνατότητα να αναγνωρίζει το συναίσθημα μόνο από τη στάση του σώματος τότε θα μπορούσε να δημιουργηθεί ένα υπολογιστικό σύστημα το οποίο θα χρησιμοποιήσει τη γνώση, που αποκτούμε με την μελέτη του ανθρωπίνου εγκεφάλου, για την αναγνώριση ανθρωπίνου συναισθήματος μόνο με τη χρήση του Body Posture.

1.4.3 Creating a System that recognizes Emotion using Body Posture

Σε άλλο paper των Χάρη Ζαχαράτο, Χρίστο Γκατζούλη και Γιώργο Χρυσάνθου, μελέτησαν κατά πόσο ένα σύστημα μπορεί να εκπαιδευτεί για να αναγνωρίζει συναίσθημα με μόνη είσοδο τη στάση του σώματος. [4] Έκαναν διάφορα πειράματα σε ελεγχόμενο περιβάλλον δύο παικτών και πήραν μετρήσεις με τη χρήση ενός Kinect. Οι παίκτες έπαιζαν παιχνίδια στο Xbox360 με τη χρήση Kinect για να μπορούν να κινούνται ελεύθερα και να είναι αρκετά εκφραστικοί. Χρησιμοποιώντας αλγόριθμο Back-Propagation εκπαίδευσαν ένα σύστημα το οποίο αναγνώριζε συναισθήματα. Το σύστημα αυτό είχε μεγάλο ποσοστό επιτυχίας στην αναγνώριση του συναισθήματος αλλά δεν ήταν 100% ορθό.

1.4.4 Real-time EEG-based Emotion Recognition and its Applications

Στο paper αυτό ασχοληθήκαν με την δημιουργία ενός real time συστήματος που να αναγνωρίζει συναισθήματα χρησιμοποιώντας Electroengephalophy [12]. Δημιούργησαν δύο σετ πειραμάτων με auditory stimuli. Το ένα αποτελείτο από τραγούδια ενός λεπτού τα οποία κατηγοριοποίησαν σε sad, fear, pleasant, happy και angry. Το δεύτερο πείραμα αποτελείτο από ήχους που πήραν από το IADS (Inernation Affective Digitized Sounds). Το IADS παρέχει ένα σετ από standarized ήχους που αποσκοπούν στην προξενήσουν συγκεκριμένο συναίσθημα ανάλογα με τον ήχο. Για την καταγραφή των EEG χρησιμοποίησαν το Emotiv που χρησιμοποίησαμε και εμείς.

Για την αναγνώριση συναισθήματος χρησιμοποίησαν μόνο τα ηλεκτρόδια στις θέσεις AF3, F4, FC6 και στο band 2-42 Hz.

1.4.5 EEG-based Emotion Recognition, The Influence of Visual and Auditory Stimuli

Στο paper αυτό χρησιμοποιήσαν Ηλεκτροεγκεφαλογράφημα για την καταγραφή εγκεφαλικής δραστηριότητας και όχι μία consumer based συσκευή [13]. Χρησιμοποίησαν Linked Ears μεθοδολογία (**5.3** Different EEG methodologies) για την καταγραφή των EEGs και μονάχα τρία ηλεκτρόδια. Συγκεκριμένα τοποθέτησαν ηλεκτρόδια στις θέσεις Fpz, F3 και F4 με βάση το International 10-20 System (**5.3** Different EEG methodologies). Δημιούργησαν ένα σύστημα που να αναγνωρίζει συναισθηματική διέγερση και ηρεμία. Δεν ασχοληθήκαν με συγκεκριμένα συναισθήματα.

Κεφάλαιο 2: Συλλογή Δεδομένων – Πειράματα – Τεχνολογίες που χρησιμοποιηθήκαν

2.1 Μικρή περιγραφή Microsoft Kinect

Το Kinect είναι μία συσκευή που κατασκευάστηκε από την Microsoft για το Xbox360. Σκοπός κατασκευής της ήταν, καταρχάς η απάντηση με κάτι καλύτερο στο Wii της Nintendo, αλλά κυρίως η δημιουργία ενός νέου τύπου παιχνιδιών που δεν χρειάζεται η χρήση κάποιου χειριστηρίου παρά μόνο με κινήσεις χεριών ποδιών και γενικά του σώματος. Λειτουργεί με την χρήση υπέρυθρων ακτινών που φεύγουν από το Kinect κτυπούν στο σώμα του παίκτη και αντανακλώνται πίσω στην συσκευή. Η συσκευή διαβάζει τις ακτίνες και ανάλογα με την γωνία με την οποία επέστρεψε και τον χρόνο που τους πήρε για να επιστρέψουν δημιουργεί το σκελετό του ανθρωπίνου σώματος. Η Microsoft το Φεβρουάριο 2013 κυκλοφόρησε επίσημο API για δημιουργία εφαρμογών με τη χρήση του Kinect για το PC. Είχε

κυκλοφορήσει προηγουμένως API σε open beta distribution αλλά το Φεβρουάριο του 2012 κυκλοφόρησε επίσημη

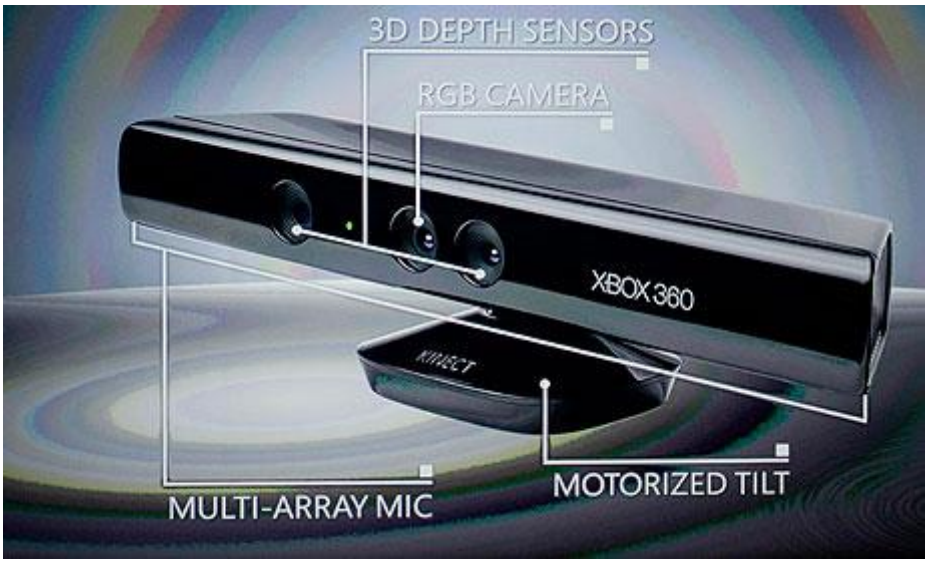

Figure 1: Microsoft Kinect

ολοκληρωμένη έκδοση διορθώνοντας όλα τα σφάλματα και απλοποιώντας πολλές λειτουργίες. Η γλώσσα στην οποία μπορεί το API να χρησιμοποιηθεί ευκολότερα και με περισσότερες δυνατότητες είναι η C# αλλά μπορεί επίσης να χρησιμοποιηθεί με Visual Basic. Εμείς προτιμήσαμε C# για να έχουμε περισσότερη ελευθερία στο τι ζητούμε από το σύστημα μας καθώς και επειδή είναι μία γλώσσα που χρησιμοποιείτε περισσότερο στο εμπόριο παρά η Visual Basic.

2.2 Τρόπος συλλογής δεδομένων – Πειράματα

Η συλλογή δεδομένων έγινε σε δύο στάδια. Και για τις δύο περιπτώσεις ζητήσαμε από φοιτητές και μη, άνδρες και γυναίκες, ηλικίας 20-25 ετών να μας βοηθήσουν στα πειράματα μας. Τα «πειραματόζωα» μας καλούνταν να συμμετάσχουν σε ένα παιχνίδι δύο παικτών. Σκοπός μας ήταν να καταγράψουμε κινήσεις συναισθημάτων με βάση αλληλεπίδραση δύο ατόμων. Τους βάζαμε σε ελεγχόμενο περιβάλλον να παίξουν ένα παιχνίδι δύο παικτών και

καταγράφαμε τις κινήσεις τους. Συγκεκριμένα έπαιζαν παιχνίδι σε Xbox360 με την χρήση του Kinect έτσι ώστε να μπορούν να κινούνται ελεύθερα. Με την χρήση του σώματος τους ως χειριστήριο βλέπουμε περισσότερες κινήσεις συναισθήματος καθώς και συνεχή καταγραφή όλων των κινήσεων. Για παράδειγμα όταν έπαιζαν ποδόσφαιρο διαφορετικά κλωτσούσαν την μπάλα όταν έχαναν και διαφορετικά όταν κέρδιζαν. Τους δώσαμε τέτοιου είδους παιχνίδια να παίξουν, Kinect και two-player και μάλιστα προσπαθούσαμε να είναι δύο άτομα γνωστά μεταξύ τους, έτσι ώστε να είναι πιο εκφραστικοί μεταξύ τους χωρίς να νιώσουν ντροπή. Ήταν όλα groups των δύο ατόμων (στα οποία καταγράφαμε το ένα άτομο) διάρκειας 5-10 λεπτών. Και μετά εναλλαγή μεταξύ των δύο ή εισαγωγή δεύτερου group. Στα μισά πειράματα καταγράψαμε την κίνηση τους με τη χρήση του Kinect και τα Brain Signals με την χρήση του Emotiv και στα άλλα μισά καταγράψαμε την κίνηση τους με τη χρήση της Motion Capture στολής που παρέχει η Phasepace και δεν μπορούσαμε να καταγράψουμε τα brain signals διότι για την καταγραφή της θέσης της κεφαλής οι χρήστες χρειαζόταν να φοράνε ένα καπελάκι ως μέρος της στολής και δεν υπήρχε δυνατότητα να έχουν ταυτόχρονα στο κεφάλι τους το Emotiv

2.3 Υλοποίηση κώδικα Kinect

Χρησιμοποιήσαμε το Kinect και το API της Microsoft για να δημιουργήσαμε μία εφαρμογή η οποία καταγράφει τις κινήσεις του συμμετέχοντα στο πείραμα μας. Ο κώδικας είναι γραμμένος σε γλώσσα C# με Object Oriented και Event Driven λογική. Είναι μία WPF εφαρμογή. Την καταγραφή δεδομένων του σκελετού αρχικός σκοπός ήταν να την χρησιμοποιήσουμε για το Emotion Recognition αλλά τελικά δημιουργήσαμε μία βάση δεδομένων για μελλοντική ανάλυση καθώς και με αυτά δημιουργήσαμε ένα ερωτηματολόγιο για Data Validation.

2.3.1 Skeleton Data Capture

Η Εφαρμογή αυτή καταγράφει τις συντεταγμένες, στον τρισδιάστατο χώρο, της κάθε κλείδωσης, τις γωνίες σε σχέση με τον πατέρα της κάθε κινηματικής αλυσίδας καθώς επίσης και την χρονική στιγμή της καταγραφής του σκελετού. Συγκεκριμένα παίρνουμε 60 frames το δευτερόλεπτο και καταγράφουμε την χρονική στιγμή σε ακρίβεια μέχρι χιλιοστό δευτερολέπτου. Αποθηκεύονται όλα τα δεδομένα σε ένα αρχείο μορφής .txt για να επεξεργαστούν αργότερα.

Ο κώδικας είναι αναρτημένος στο Παράρτημα **A.1**: Κώδικας Kinect για Data Capturing

2.3.2 Converting Clock ticks to timestamp

Επειδή στην καταγραφή των δεδομένων μας χρειαζόμασταν ταχύτητα και πολλά frames/second δεν υπολογίζαμε το timestamp την στιγμή της καταγραφής των δεδομένων μας για να μειώσουμε την πολυπλοκότητα του κώδικα. Για να μην χαθεί όμως ο χρόνος (ένα δεδομένο πολύ σημαντικό στα πειράματα μας) αποθηκεύαμε τα clock ticks. Μετά γράψαμε μία διαφορετική εφαρμογή (Console Application) σε C# με την οποία διαβάζαμε το κάθε αρχείο των Skeleton Data και παίρναμε τα clock ticks και τα μετατρέπαμε σε timestamp.

Ο κώδικας είναι αναρτημένος στο Παράρτημα **A.2**: Κώδικας για μετατροπή των clock ticks σε timestamp

2.3.3 Real-Time On screen Skeleton Rendering

Η εφαρμογή εκτός από την καταγραφή των δεδομένων έχει ένα interface το οποίο δείχνει σε πραγματικό χρόνο τη θέση του σκελετού και τις κλειδώσεις του. Το interface αυτό γράφτηκε σε γλώσσα XAML. Τα δεδομένα την στιγμή που υπολογίζονται, ταυτόχρονα με την αποθήκευση τους, χρησιμοποιούνται για να δημιουργήσουν τον σκελετό του «πειραματόζωου» καθώς και τον σκελετό του συμπαίκτη του. Το On Screen Rendering του

σκελετού το χρησιμοποιήσαμε για τη δημιουργία ενός ερωτηματολογίου που θα μας βοηθούσε στο καλύτερο annotation των συναισθημάτων που βρήκαμε.

Ο κώδικας είναι αναρτημένος στο Παράρτημα **A.3**: Κώδικας για On Screen Rendering του σκελετού

2.3.4 Kinect video Stream

Το interface της εφαρμογής μας εκτός από τους σκελετούς παρέχει και συνεχές video stream των παικτών. Το video stream που μας παρέχει η camera του Kinect μας βοήθησε να βρούμε διάφορα συναισθήματα από τα πειράματα μας.

Figure 2: Screenshot of Kinect App for Skeleton Data Capturing

2.3.5 Screen Recording

Κάθε πείραμα που τρέχαμε καταγράφαμε συνεχώς και την οθόνη του υπολογιστή. Το video αυτό το χρησιμοποιήσαμε για να βρούμε ακριβώς τις στιγμές στις οποίες υπάρχει έκφραση

συναισθήματος (video stream και timestamp) και κάνοντας crop το video και παίρνοντας μόνο τον σκελετό δημιουργήσαμε ερωτηματολόγιο με τις συγκεκριμένες στιγμές αυτές για να μας βοηθήσουν στο καλύτερο annotation των συναισθημάτων που βρήκαμε.

2.4 Kinect Data Smoothing

Στην πορεία συλλογής των δεδομένων προσέξαμε ότι είχαμε λίγο θόρυβο. Για να φιλτράρουμε λοιπόν τα δεδομένα μας αποφασίσαμε να χρησιμοποιήσουμε τις built-in μεθόδους και κλάσεις που μας προσφέρει το Kinect API. Όσο περισσότερο φιλτράρισμα όμως γίνεται τόσο πιο αργό γίνεται το σύστημα. Αυτό συμβαίνει εκ φύσεως διότι πλέον επεξεργάζονται τα δεδομένα πριν καταγραφούν άρα χρειάζεται μία άλφα καθυστέρηση πριν την καταγραφή. Προσπαθήσαμε λοιπόν να βρούμε την χρυσή τομή μεταξύ καθυστέρησης και φιλτραρίσματος με διάφορους πειραματισμούς. Για να εφαρμοστεί το φιλτράρισμα χρειάζεται η δημιουργία ενός αντικειμένου που παίρνει σαν παραμέτρους κανονικοποιημένες τιμές (0-1) για

1. **Smoothing:** Τιμή για το πόσο θα φιλτράρουμε τα δεδομένα μας.
2. **Correction:** Τιμή για το πόσο θα διορθώσουμε τα δεδομένα μας.
3. **Prediction:** Τιμή για το πόσα frames θα προβλέψουμε στο μέλλον το τι θα συμβεί
4. **JitterRadius:** Τιμή που υποδηλώνει την ακτίνα σε μέτρα όπου θα μειώσουμε την παραμόρφωση
5. **MaxDeviationRadius:** Η μέγιστη δυνατή απόκλιση των φιλτραρισμένων δεδομένων από τα raw data

Για το smoothing και το correction αν δώσουμε την τιμή 0 τότε μας επιστρέφονται τα raw data.

Τρέξαμε διάφορα πειράματα για να βρούμε τις ιδανικές τιμές για smoothing και ξεκινήσαμε από την αρχή την καταγραφή νέων δεδομένων χωρίς θόρυβο.

2.5 Emotiv

Για την καταγραφή των Brain Signals χρησιμοποιήσαμε την συσκευή Emotiv EEG neuroheadset. Το Emotiv έχει 14 ηλεκτρόδια που λειτουργούν ως αισθητήρες για την καταγραφή των Brain Signals και γυροσκόπιο σε 2 άξονες για την καταγραφή των κινήσεων της κεφαλής. Περισσότερες πληροφορίες στο **Κεφάλαιο 5: EEG Analysis**

2.6 Mocap – Phasespace

Για motion capture data χρησιμοποιήσαμε δύο διαφορετικούς τρόπους. Ο ένας ήταν με την χρήση του Microsoft Kinect και ο άλλος με τη χρήση στολής Motion Capture από τη Phasespace. Σε κάποια από τα πειράματα λοιπόν ζητήσαμε από τον ένα από τους δύο συμμετέχοντες (αυτόν που θα καταγράφαμε με το Kinect) να φορέσει στολή Motion Capture της Phasespace για να πάρουμε δεδομένα σκελετού και κλειδώσεων με περισσότερη ακρίβεια για να κτίσουμε ένα ρεαλιστικό χαρακτήρα που να αναπαράγει τις κινήσεις αυτές έτσι ώστε να είναι εύκολα αναγνωρίσιμες από τους responders του ερωτηματολογίου.

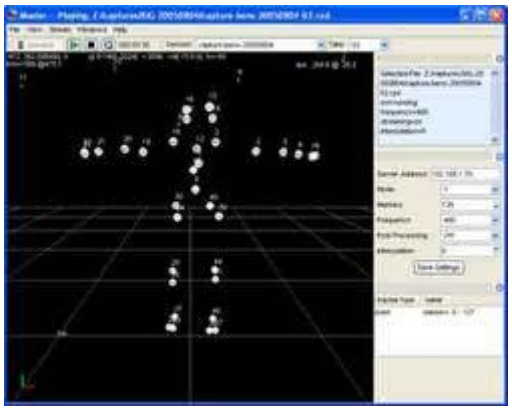

Figure 3: Mocap Data Visualization

Η στολή της Phasespace φέρει 42 markers. Το όλο σύστημα αποτελείται από την στολή, 4 κάμερες με 2 «μάτια» η κάθε μια, τον server, ένα LED Controller καθώς και το tracking software.

Οι 42 markers λειτουργούν με active LED τεχνολογία. Η τεχνολογία αυτή μας βοηθά πολύ εύκολα να αναγνωρίσουμε τον κάθε marker καθώς εκπέμπουν όλοι κόκκινο φως διαφορετικής συχνότητας ο κάθε ένας. Έτσι λοιπόν η κάθε κάμερα μπορεί εύκολα να αναγνωρίσει ποιος marker είναι ποιος βάση της συχνότητας του φωτός που εκπέμπει.

Η κάθε κάμερα είναι εφοδιασμένη με δύο μάτια. Αυτό είναι υπερβολικά χρήσιμο καθώς μας βοηθάει να λύσουμε το πρόβλημα του βάθους. Αυτό γίνεται με ακριβώς τον ίδιο τρόπο που το κάνει το ανθρώπινο μάτι.

Figure 4: Led Camera

Ο LED Controller της στολής είναι μία μικρή συσκευή η οποία ρυθμίζει τον κάθε marker. Επικοινωνεί με τον server και ρυθμίζει την συχνότητα του κάθε marker για να είναι διακριτά αναγνωρίσιμη από τις υπόλοιπες για επιτάχυνση της διαδικασίας του tracking. Αυτό σαν αποτέλεσμα έχει την καταγραφή περισσότερων frames ανά δευτερόλεπτο. Συγκεκριμένα παίρνουμε 120 frames το δευτερόλεπτο.

Figure 5: Led Controller

Δυστυχώς οι συνθήκες του εργαστηρίου δεν είναι optimal και οι κάμερες δεν είναι απολύτως σταθερές. Ακόμα και αν κλείσει η πόρτα στο διπλανό ή στο παραδίπλα εργαστήριο η δόνηση μεταφέρετε δια μέσω των τοίχων και οι κάμερες μετακινούνται έστω και λίγο. Έτσι κάθε νέα πειραματική μέρα πρέπει το όλο σύστημα να γίνει ξανά calibrate. Μία κάπως επαναλαμβανόμενη διαδικασία αλλά απαραίτητη για να είναι σωστές οι μετρήσεις μας.

Κεφάλαιο 3: Δημιουργία Χαρακτήρων και Ερωτηματολογίου

3.1 Autodesk Motionbuilder

Χρησιμοποιώντας τα skeleton data του Motion Capture και με τη βοήθεια του Autodesk Motionbuilder κτίσαμε ένα digital χαρακτήρα για κάθε πείραμα χωρίς όμως να προσθέσουμε facial expressions. Αφού λοιπόν στη συνέχεια τους χαρακτήρες αυτούς θα τους χρησιμοποιούσαμε σε ερωτηματολόγια για να γίνουν annotate το κάθε συναίσθημα ήταν υποχρεωτικό να μην προσθέσουμε facial expressions έτσι ώστε να μην επηρεαστεί η κρίση του ερωτηθέντα από τις εκφράσεις του προσώπου του παίκτη και να μπορεί να κρίνει μόνο από τις κινήσεις του σώματος.

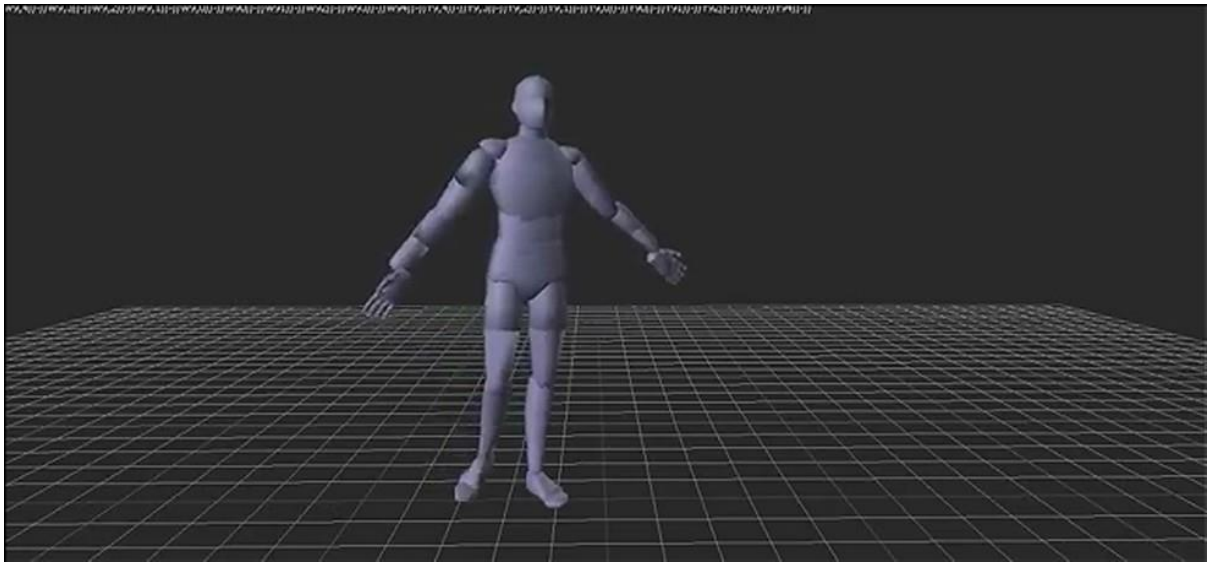

Figure 6: Autodesk Motionbuilder Character Creation

Έτσι λοιπόν πήραμε τα δεδομένα από το Motion Capture τα εισάγαμε στο Autodesk Motionbuilder και δημιουργήσαμε το χαρακτήρα για τον κάθε συμμετέχοντα. Με τα δεδομένα αυτά πήραμε το κάθε πείραμα και βρήκαμε τις στιγμές όπου υπάρχει ένδειξη συναισθήματος, τις απομονώσαμε και τις κάναμε export σε μικρά video (2-3 δευτερόλεπτα). Πήραμε συνολικά 309 video συναισθηματικών εκφράσεων.

3.2 Πρώτο ερωτηματολόγιο

Χρησιμοποιώντας τα clips που πήραμε από το Motion Capture και το Autodesk Motionbuilder δημιουργήσαμε τρία ερωτηματολόγια. Το κάθε ένα από αυτά περιλάμβανε 100 video με εξαίρεση το τρίτο που είχε 109. Τα διασπάσαμε σε τρία ερωτηματολόγια και δεν τα βάλαμε όλα μαζί για ευκολία απαντήσεων από τους χρήστες διότι θα ήταν πολύ κουραστικό να απαντήσουν σε 309 ερωτήσεις απευθείας.

Τα άτομα που συμμετείχαν στο ερωτηματολόγιο για κάθε βίντεο είχαν να διαλέξουν μία από τις εξής πέντε επιλογές:

1. **Frustration** = similar with – Angry, upset, confused
2. **Excitement** = similar with - joy , happiness
3. **Meditation** = similar with - stretching, wondering, relaxed. (a situation when the user focuses on his own body.)
4. **Concentration** = similar with - focus, stand by
5. **None of the above**

Για την επιλογή των συγκεκριμένων συναισθημάτων βασιστήκαμε στην μεταπτυχιακή διπλωματική του Θεοχάρη Ζαχαράτου[5] καθώς και σε αντίστοιχο paper που δημοσίευσε μαζί με τους Γιώργος Χρυσάνθου και Χρίστο Γκατζούλη[4].

Αυτό ήταν το πρώτο ερωτηματολόγιο το οποίο λόγω αλλαγής στη μορφή καταγραφής των δεδομένων (δεν ασχοληθήκαμε με Motion Capture καθώς μας εμπόδιζε στην καταγραφή των EEG) δεν το χρησιμοποιήσαμε.

Μας βοήθησε όμως να αναγνωρίσουμε ότι δεν υπάρχει μεγάλη διαφορά μεταξύ Meditation και Concentration. Υπήρξε αρκετό μπέρδεμα από τους απαντητές του ερωτηματολογίου στο πιο συναίσθημα είναι πιο και τα συμπτύξαμε σε ένα συναίσθημα (Engagement) για το μεταγενέστερο μέρος της διπλωματικής.

3.3 Mocap Data analysis. Inverse Dynamics

Χρησιμοποιώντας τα δεδομένα που πήραμε από το Mocap μπορούμε να υπολογίσουμε την αντίστροφη δυναμική για κάθε κλείδωση. Έτσι θα μπορέσουμε στη πορεία να δημιουργήσουμε ένα σύστημα που, με τον υπολογισμό των δυνάμεων που ασκούνται σε κάθε κλείδωση , να μπορέσει να συμπεράνει το συναίσθημα του χρήστη. Για τους υπολογισμούς αυτούς υπάρχουν αρκετοί εξομοιωτές που με την χρήση των Mocap data μπορούν να το κάνουν.

Τα Mocap data χρησιμοποιηθήκαν για την δημιουργία ενός data bank για μελλοντική χρήση με το OpenSim για να βρούμε τα Inverse Dynamics.

3.4 OpenSim

Το OpenSim είναι ένα σύστημα ανοικτού κώδικα για την εμβιομηχανική μοντελοποίηση, προσομοίωση και ανάλυση . Σκοπός του είναι να παρέχει δωρεάν και ευρέως προσβάσιμα εργαλεία για τη διεξαγωγή της έρευνας και της εμβιομηχανικής. Το OpenSim χρησιμεύει σε ένα ευρύ φάσμα μελετών , συμπεριλαμβανομένης της ανάλυσης της δυναμικής κίνησης , τις μελέτες των αθλητικών επιδόσεων , προσομοιώσεις των χειρουργικών διαδικασιών , ανάλυση των δυνάμεων που ασκούνται στις κλειδώσεις , το σχεδιασμό των ιατρικών συσκευών και δημιουργία animation ανθρώπων και ζώων .

Το λογισμικό εκτελεί αντίστροφη ανάλυση της δυναμικής και προς τα εμπρός προσομοιώσεις δυναμικής . Το OpenSim χρησιμοποιείται σε εκατοντάδες εμβιομηχανικά

εργαστήρια σε όλο τον κόσμο για τη μελέτη κίνησης και έχει μια τεράστια crowdsourcing βάση από προγραμματιστές που συνεχώς συμβάλλουν στη δημιουργία νέων χαρακτηριστικών. .

Το OpenSim είναι μία από τις flagship εφαρμογές της από Simbios , το Κέντρο Βιοϊατρικής Πληροφορικής στο Πανεπιστήμιο του Stanford . Η Simbios, ιδρύθηκε το 2004, και αποσκοπεί στην παροχή κορυφαίου λογισμικού και υπολογιστικών εργαλείων για τη physics-based μοντελοποίηση και προσομοίωση των βιολογικών δομών . Το OpenSim σχεδιάστηκε για να ωθήσει την έρευνα της εμβιομηχανικής, παρέχοντας ένα κοινό πλαίσιο για την έρευνα και ένα μέσο για την ανταλλαγή πολύπλοκων μυοσκελετικών μοντέλων .

Η διαδικασία για να πάρουμε τα Inverse Dynamics περνά από το Scaling, το Inverse Kinematics και τέλος φυσικά το Inverse Dynamics.

Και στις τρεις περιπτώσεις σαν input στο tool ο χρήστης δίνει μόνο ένα Settings file(.xml) το οποίο περιλαμβάνει τα ονόματα των υπολοίπων inputs καθώς και διάφορες ρυθμίσεις για το πώς να τρέξει το κάθε Simulation.

Τα inputs και outputs του OpenSim περιγράφονται αναλυτικά στο **Error! Reference source not found.**

.

Κεφάλαιο 4: Δημιουργία Ερωτηματολογίου, Clip Annotation και Στατιστική ανάλυση των αποτελεσμάτων

4.1 Δημιουργία ερωτηματολογίου με τα Kinect Data

Μετά το πρώτο ερωτηματολόγιο πήραμε νέα δεδομένα και τα διαχωρίσαμε πλέον σε 3 συναισθήματα αντί τέσσερα.

Με την χρήση των **2.3.3** Real-Time On screen Skeleton Rendering και **2.3.5** Screen Recording δημιουργήσαμε ένα ερωτηματολόγιο με τα clips που βρήκαμε.

Από τα καταγραμμένα video κόψαμε στις χρονικές στιγμές που βρήκαμε ότι υπάρχει έκφραση συναισθήματος. Από τα κομμένα video απομονώσαμε μονάχα τον σκελετό έτσι ώστε να μην υπάρχουν facial expressions στα video μας καθώς τα facial expressions ευκολύνουν την διαδικασία απάντησης ενώ εμείς θέλαμε οι απαντητές να κρίνουν με βάση μόνο τον σκελετό.

Δημιούργησα ένα Google Form και ανέβασα τα 39 video των στιγμών που βρήκαμε και σε κάθε video οι χρήστες είχαν την δυνατότητα να επιλέξουν μία από τις τρεις απαντήσεις για κάθε video, στην ερώτηση «Ποιο είναι το συναίσθημα στο πιο κάτω video;»

Είχαν τις εξής τρεις απαντήσεις:

1. **Engagement:** Similar with Concentration, Focus
2. **Excitement:** Similar with Joy, Happiness
3. **Frustration:** Similar with Angry, Upset, Confused

Τα δεδομένα της κάθε απάντησης του ερωτηματολογίου αποθηκεύονταν σαν ξεχωριστή γραμμή σε ένα Excel αρχείο για να τα επεξεργαστούμε αργότερα.

4.2 Clip Annotation και Στατιστική ανάλυση των αποτελεσμάτων

Χρειαζόμασταν μία εξακρίβωση για το πόσο ορθές είναι οι στιγμές που βρήκαμε σε σχέση με την έκφραση συναισθήματος και ποιο συναίσθημα είναι αυτό. Ακολουθήσαμε μία επιστημονική μέθοδο που χρησιμοποιήθηκε και στην Μεταπτυχιακή Διπλωματική του Χάρη Ζαχαράτου χρησιμοποιώντας το Cohen's Kappa, που ασχολήθηκε με παρόμοιο θέμα. [5]

4.2.1 Τι είναι το Cohen's Kappa

Το στατιστικό kappa, που για πρώτη φορά χρησιμοποιήθηκε από τον Galton (1892), είναι μία στατιστική μέτρηση της αξιοπιστίας αποτελεσμάτων μίας έρευνας. Γενικά πιστεύεται ότι είναι ένα πιο ισχυρό μέτρο από τον απλό υπολογισμό της συμφωνία με βάση τοις εκατό καθώς το Kappa λαμβάνει υπόψη τις απαντήσεις που συμφωνούν κατά τύχη. [6]

Το Kappa παρέχει μία μέτρηση η οποία δείχνει πόσο δύο κριτές , A και B , συμφωνούν στις αντίστοιχες ταξινομήσεις N στοιχείων σε k κατηγορίες που αλληλοαποκλείονται . Ένας «κριτής» , στο πλαίσιο αυτό μπορεί να είναι ένας μεμονωμένος ανθρώπινος, ένα σύνολο ατόμων που ταξινόμησε τα αντικείμενα N συλλογικά , ή ένα πρόγραμμα υπολογιστή ή διαγνωστικό τεστ , που εκτελεί μια ταξινόμηση με βάση καθορισμένων κριτηρίων. Δεν παίρνουμε το καθαρό ποσοστό της συμφωνίας διότι τα ποσοστά μπορεί να είναι

παραπλανητικά και δεν αντικατοπτρίζουν την πραγματική εικόνα. Ο λόγος είναι ότι ο υπολογισμός των ποσοστών δεν λαμβάνει υπόψη τις συμφωνίες που έχουν μεταξύ τους οι κριτές και οφείλονται στην τύχη. Χρησιμοποιώντας τα ποσοστά μπορεί να οδηγήσει σε δύο κριτές να φαίνονται ότι συμφωνούν πλήρως, ή πάρα πολύ ακόμη και αν έχουν συμπληρώσει το ερωτηματολόγιο τους στην τύχη.

4.2.2 Πώς υπολογίζουμε το Kappa

Η συνάρτηση για τον υπολογισμό του Kappa είναι:

$$k = \frac{\Pr(a) - \Pr(e)}{1 - Pr(e)}$$

Figure 7: Cohen's Kappa

Όπου Pr(a) είναι η σχετική συμφωνία μεταξύ των κριτών και Pr(e) είναι η πιθανότητα συμφωνίας κατά τύχη, χρησιμοποιώντας τα δεδομένα των κριτών για να υπολογιστούν οι πιθανότητες οι κριτές να επιλέξουν τυχαία τις ίδιες κατηγορίες. Αν οι κριτές συμφωνούν πλήρως τότε το kappa ισοδυναμεί με 1. Αν δεν υπάρχει καθόλου συμφωνία, εκτός αυτή που αναμένεται αν επιλέξουν τυχαία αποτέλεσμα, τότε το kappa ισοδυναμεί με 0.

4.2.3 Στατιστική ανάλυση αποτελεσμάτων

Χρησιμοποιώντας λοιπόν το Cohen's Kappa αναλύσαμε το ερωτηματολόγιο για να βρούμε το σωστό Emotion Annotation των τμημάτων που βρήκαμε.

Είχαμε 4 annotators για το ερωτηματολόγιο. Η συνάρτηση του Cohen's Kappa (**Figure 7**) υπολογίζει την συμφωνία που έχουν στις απαντήσεις τους δύο άτομα. Έτσι εμείς πήραμε τις 6 δυάδες απαντήσεων και υπολογίσαμε την συμφωνία τους με βάση το Cohen's Kappa.

Αναλύοντας λοιπόν τα αποτελέσματα, για το κάθε ζευγάρι observers, οι τιμές που πήρα στο Kappa για το κάθε ζευγάρι ήταν:

OBSERVER	**1**	**2**	**3**	**4**
1	X	0.7	0.63	0.63
2	0.7	X	0.81	0.67
3	0.63	0.81	X	0.74
4	0.63	0.67	0.74	X

Με βάση τη θεωρία του Cohen’s Kappa σε ανθρώπινους κριτές για να θεωρηθούν καλά τα αποτελέσματα και να μειωθεί πολύ η «κατά τύχην συμφωνία» πρέπει να είναι πάνω από 0,7. Για να θεωρηθούν «ορθά» πρέπει να είναι πάνω από 0,8. Το Kappa μας σε όλα τους συνδυασμούς είναι σχετικά καλό αλλά για τελικό annotation επιλέξαμε το ζευγάρι με το καλύτερο Kappa που είναι το ζευγάρι 2,3 με Kappa 0,81

Κεφάλαιο 5: EEG Analysis

5.1 Το συναίσθημα και ο εγκέφαλος

Όπως προαναφέραμε στο **1.3** Τι είναι το συναίσθημα; το συναίσθημα είναι συνδυασμός νοητικών καταστάσεων, ψυχοσωματικών εκφράσεων και βιολογικών αντιδράσεων του σώματος. Πόσοι λοιπόν είναι η συσχέτιση του με τον εγκέφαλο;

Περισσότερο από δυόμισι χιλιάδες χρόνια πριν, ο Έλληνας φιλόσοφος Επίκτητος διατύπωσε ότι "ο άνθρωπος δεν συγκινείται από τα γεγονότα, αλλά απλώς από την άποψή του για αυτά" Με αυτό, ο Επίκτητος άφησε να εννοηθεί ότι η αντίληψη οδηγεί στο συναίσθημα, το οποίο με τη σειρά του οδηγεί σε κίνηση που απαντά σε αυτό το συναίσθημα.

Σύγχρονοι νευροεπιστήμονες και ψυχολόγοι έχουν ερευνήσει και έχουν βρει ότι η αμυγδάλα του εγκεφάλου καθώς και ο προμετωπιαίος φλοιός συνδέονται άρρητα με το συναίσθημα.

Έτσι λοιπόν με την χρήση ενός ηλεκτροεγκεφαλογραφήματος μπορούμε να μετρήσουμε με κάποιο τρόπο την λειτουργία του εγκεφάλου και από αυτή να εξάγουμε το συναίσθημα του ατόμου.

5.2 Τι είναι το EEG

Ο ανθρώπινος εγκέφαλος έχει δισεκατομμύρια νευρώνες. Η λειτουργία του εγκεφάλου είναι εφικτή χάριν στους ηλεκτρικούς παλμούς που αποστέλλονται από τον ένα νευρώνα στον άλλο. Οι ηλεκτρικοί παλμοί αντιστοιχούν σε «μηνύματα» με τα οποία επικοινωνούν οι νευρώνες μεταξύ τους. Έτσι λοιπόν μπορούμε εμείς να μετρήσουμε τους παλμούς αυτούς και να χαρτογραφήσουμε τα brain waves του εγκεφάλου. Το πρώτο ηλεκτροεγκεφαλογράφημα (ElectroEncephaloGram – EEG) δημιουργήθηκε το 1924 από τον Γερμανό Hans Berger. Την σήμερον ημέρα οι γιατροί μπορούν να χρησιμοποιήσουν το EEG για να διαγνώσουν την επιληψία καθώς και άλλες πιθανές νευρολογικές παθήσεις.

5.3 Different EEG methodologies

Υπάρχουν πολλές μεθοδολογίες για καταγραφή EEG data. Κάποιες χρησιμοποιούν περισσότερα ηλεκτρόδια κάποιες λιγότερα. Οι ονομασίες και οι τοποθεσίες των ηλεκτροδίων όμως είναι προκαθορισμένες από το Internation 10-20 System [11].

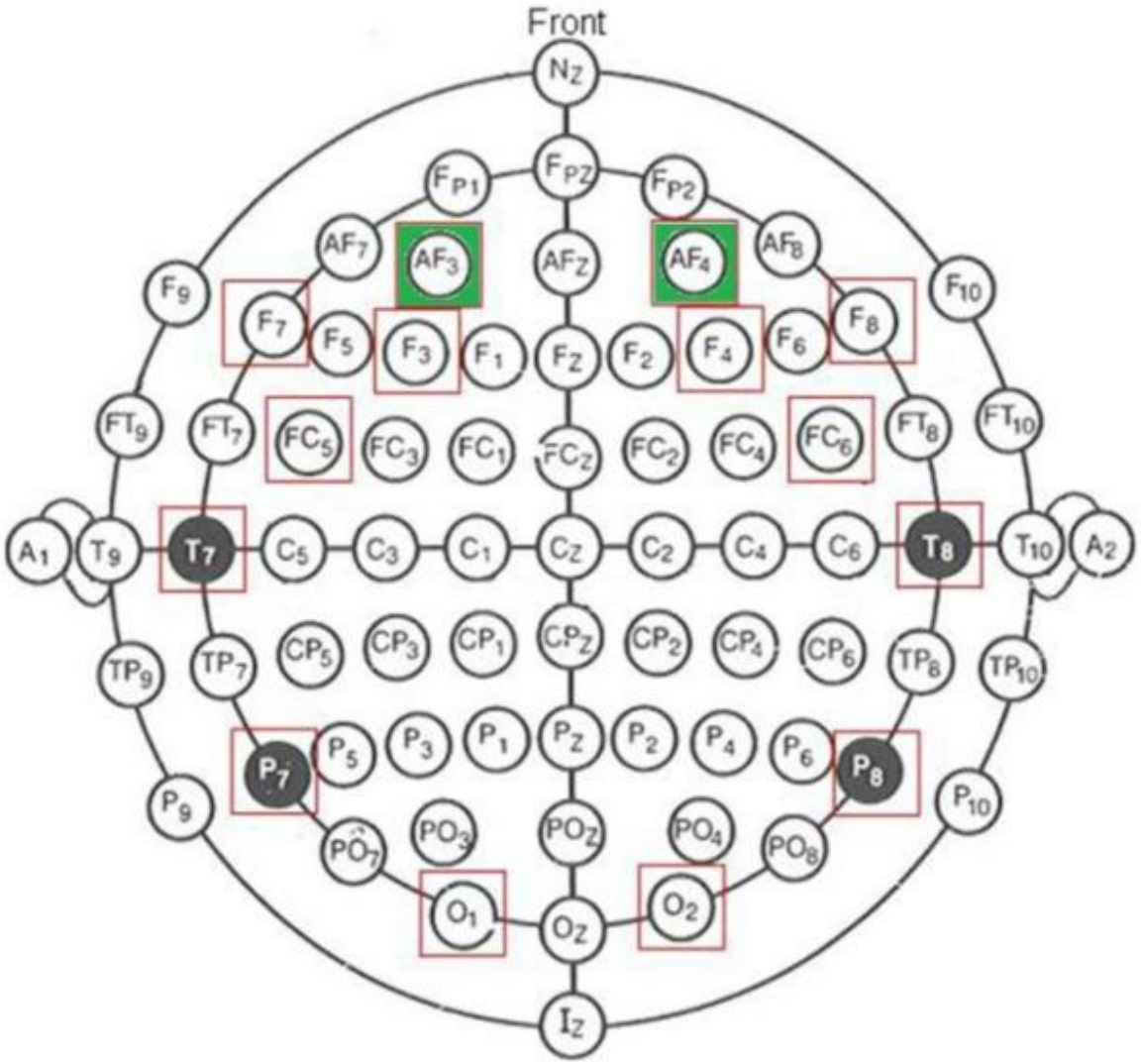

Figure 8: International 10-20 System, Electrode Placing and labeling

Δεδομένου ότι ένα σήμα τάσεως EEG αντιπροσωπεύει μια διαφορά μεταξύ των τάσεων σε δύο ηλεκτρόδια, η απεικόνιση και αναγνώριση ενός EEG μπορεί να συσταθεί με διάφορους τρόπους . Η αναπαράσταση των καναλιών EEG ονομάζεται montage.

5.3.1 Squential montage

Κάθε κανάλι αντιπροσωπεύει τη διαφορά μεταξύ δύο γειτονικών ηλεκτροδίων . Το σύνολο του montage αποτελείται από μια σειρά των καναλιών αυτών.

Για παράδειγμα, το κανάλι «FP1 -F3» αντιπροσωπεύει τη διαφορά στην τάση μεταξύ του ηλεκτροδίου FP1 και του ηλεκτροδίου F3. Αντίστοιχα το επόμενο κανάλι στο montage «F3-C3» αντιπροσωπεύει τη διαφορά τάσεως μεταξύ των F3 και C3 , και ούτω καθεξής δρ ολόκληρη της συστοιχία των ηλεκτροδίων.

5.3.2 Referential montage

Κάθε κανάλι αντιπροσωπεύει τη διαφορά μεταξύ ενός συγκεκριμένου ηλεκτροδίου και ενός ηλεκτροδίου αναφοράς. Δεν υπάρχει τυπική θέση για το ηλεκτρόδιο αναφοράς. Είναι ωστόσο σε διαφορετική θέση από τα ηλεκτρόδια καταγραφής. Συνήθως χρησιμοποιούνται κεντρικές θέσεις για τα reference electrodes επειδή δεν ενισχύουν το σήμα σε ένα ημισφαίριο έναντι του άλλου.

Μια άλλη δημοφιλής Referential montage τεχνική είναι «Linked Ears». Η μεθοδολογία αυτή μας δίνει τον μαθηματικό μέσο όρο από δύο reference electrodes τα οποία είναι τοποθετημένα πίσω από τα δύο αυτιά του ατόμου. Έτσι είναι κάπως πιο ακριβή τα αποτελέσματα καθώς χρησιμοποιεί δύο reference electrodes ταυτόχρονα

5.3.3 Average Reference montage

Οι έξοδοι όλων των reference electrodes αθροίζονται και ο υπολογίζεται μέσος όρος. Αυτός ο μέσος όρος χρησιμοποιείται ως κοινή αναφορά για την καταγραφή των υπόλοιπων electrodes.

5.3.4 Laplacian montage

Κάθε κανάλι αντιπροσωπεύει τη διαφορά μεταξύ ενός ηλεκτροδίου και τον σταθμισμένο μέσο όρο των γύρω ηλεκτροδίων. [10]

5.4 Το Emotiv

Το Emotiv EEG Neuroheadset είναι μία σύγχρονη απλοποιημένη μορφή του πρώτου EEG το οποίο σήμερα χρειάζεται μονάχα 14 αισθητήρες (ηλεκτρόδια) για το «διάβασμα» των εγκεφαλικών κυμάτων. Είναι μίας φθηνή consumer συσκευή για home EEG capture. Συγκεκριμένα χρησιμοποιώντας τις βιβλιοθήκες που παρέχει η ίδια η εταιρεία για την συσκευή γράψαμε μία Windows Console εφαρμογή σε γλώσσα C++ η οποία καταγράφει τα εγκεφαλικά κύματα του «πειραματόζωου» μας χωρίς να τον εμποδίζει καθώς η συσκευή επικοινωνεί με τον υπολογιστή μέσω Bluetooth. Έτσι δεν υπάρχουν καθόλου σύρματα να εμποδίζουν το πειραματόζωο καθώς και τον μεταγενέστερο χρήση και να μην επεμβαίνει στις καθημερινές του λειτουργίες.

Η montage μεθοδολογία που χρησιμοποιά το Emotiv για την καταγραφή των δεδομένων είναι η μεθοδολογία Linked Ears.

Το sampling rate μας είναι 128Hz (captures/second)

5.5 Emotiv Setup

Το Emotiv χρειάζεται κάποια προεργασία για να μπορεί να λειτουργήσει σωστά. Συγκεκριμένα τα 14 ηλεκτρόδια χρειάζεται να είναι αρκετά υγρά ώστε να λαμβάνουν πιο εύκολα τα ηλεκτρικά σήματα του εγκεφάλου. Μετά από πειράματα βρήκαμε ότι το υγρό που λειτουργεί ως ο καλύτερος αγωγός των εγκεφαλικών σημάτων είναι το υγρό καθαριστικών των φακών επαφής. Επίσης όταν τοποθετείται το Emotiv στο κεφάλι του «πειραματόζωου» πρέπει να τοποθετηθεί με τέτοιο τρόπο ώστε όλα τα ηλεκτρόδια να έχουν άμεση επαφή με το κρανίο χωρίς το εμπόδιο των μαλλιών. Κύριος βοηθός στην τοποθέτηση του Emotiv είναι το Emotiv Control Panel το οποίο ξεκαθαρίζει ποια ηλεκτρόδια παίρνουν σήμα και ποια όχι. Επίσης καθορίζει την ποιότητα του σήματος του κάθε ηλεκτροδίου, την χωρητικότητα της μπαταρίας καθώς και το πόσο δυνατό είναι το σήμα που λαμβάνει ο υπολογιστής από το Emotiv.

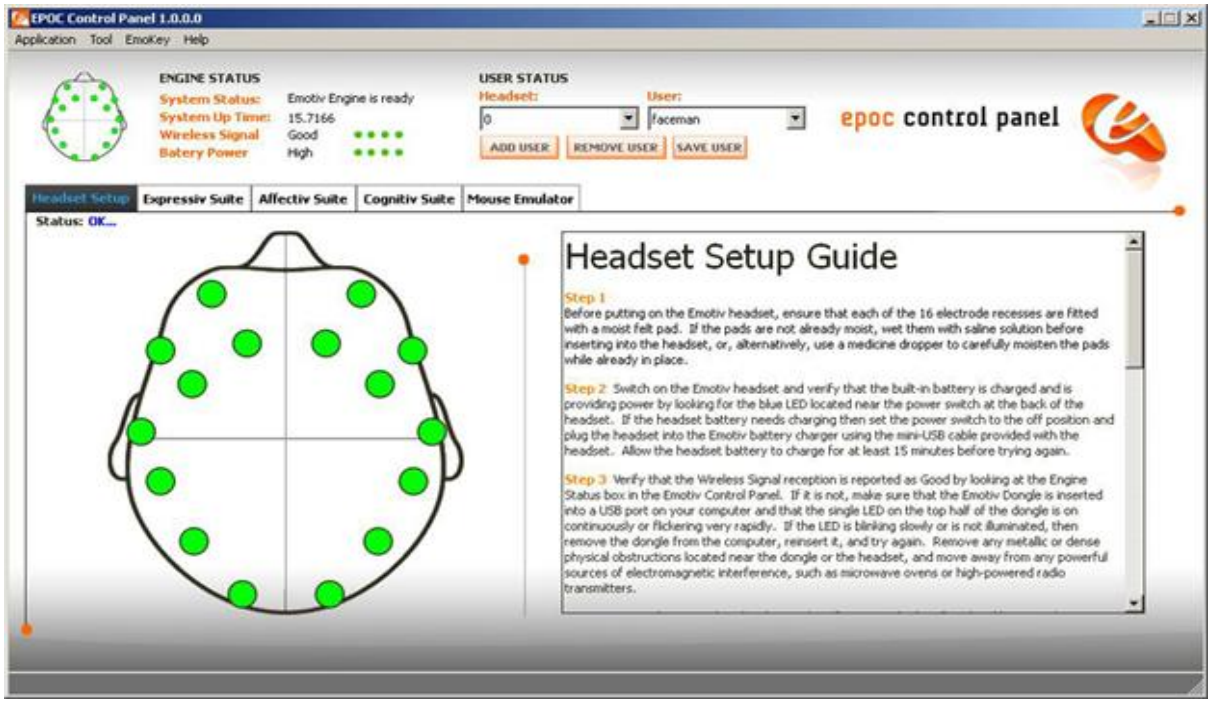

Figure 9: Emotiv Setup Guide

Αν στην εφαρμογή όλα τα ηλεκτρόδια συμβολίζονται με πράσινο κύκλο τότε έχουμε την καλύτερη δυνατή καταγραφή σήματος με καθόλου ή ελάχιστο θόρυβο. Αντιθέτως αν έχουμε κίτρινο τότε δεν είναι τόσο καλό, κόκκινο σημαίνει καθόλου καλό και μαύρο σημαίνει καθόλου σήμα.

5.6 EEG Procedures

5.6.1 Exporting data from csv files

Κατά την μελέτη των πειραμάτων βρήκαμε 39 στιγμές στις οποίες υπάρχει έντονη έκφραση ενός από τα τρία συναισθήματα που μελετούμε. Έτσι λοιπόν δημιουργήσαμε ένα φάκελο για κάθε στιγμή. Στον φάκελο προσθέσαμε ένα αρχείο video το οποίο κάναμε crop έτσι ώστε να φαίνεται μονάχα ο σκελετός του ατόμου (έτσι ώστε να μην έχουμε facial expressions), σε ένα αρχείο .txt τα Kinect Data για την συγκεκριμένη χρονική στιγμή και σε ένα αρχείο .csv

αποθηκεύσαμε τα EEG data για την χρονική στιγμή αυτή. Στο αρχείο .csv αποθηκεύουμε και τις τιμές των δύο γυροσκοπίων (ένα για κάθε άξονα) του Emotiv σαν ξεχωριστές στήλες.

Δείγμα των EEG data μπορεί να βρεθεί στον ακόλουθο σύνδεσμο

https://drive.google.com/file/d/0B1EbI-EMJxmkVHhSelFfNklLVzQ/edit?usp=sharing

5.6.2 Preprocessing

Σαν πρώτο στάδιο, με τη χρήση ενός Matlab script, διαβάζουμε το αρχείο .csv για το κάθε clip στην Matlab και σπάζουμε τα EEG από τα Gyro data σε δύο διαφορετικούς πίνακες και τα επιστρέφουμε. Τον πίνακα με τα EEG προτού τον επιστρέψουμε βρίσκουμε τον ανάστροφό του και επιστρέφουμε αυτόν διότι έτσι μας χρησιμεύει περισσότερο για το επόμενο στάδιο.

Ο κώδικας είναι αναρτημένος στο Παράρτημα **Παράρτημα Γ.**1: Preprocessing

5.6.3 EEG Band pass filtering using EEGLAB

5.6.3.1 Τι είναι το EEGLAB

Το EEGLAB είναι ένα διαδραστικό πακέτο εργαλείων για την επεξεργασία συνεχών και event-related EEG, MEG και άλλα ηλεκτροφυσιολογικά δεδομένα που ενσωματώνουν ανάλυση ανεξάρτητων συνιστωσών (independent component analysis-ICA), χρόνου/συχνότητας ανάλυσης, αφαίρεση θορύβου, event-related στατιστικές, και άλλους χρήσιμους τρόπους απεικόνισης των δεδομένων. Το EEGLAB δημιουργήθηκε,

αναπτύσσεται και συντηρείται από το Swartz Center for Computational Neuroscience του University of California San Diego.

5.6.3.2 Γιατί με το EEGLAB

Το EEGLAB παρέχει μια διαδραστική γραφική διεπαφή χρήστη (GUI), επιτρέποντας στους χρήστες να επεξεργαστούν υψηλής πυκνότητας EEG καθώς και άλλων δυναμικών δεδομένων εγκεφάλου, ευέλικτα και διαδραστικά, με ανάλυση ανεξάρτητων συνιστωσών (independent component analysis-ICA). Το EEGLAB ενσωματώνει επίσης εκτεταμένα tutorials και τα παράθυρα βοήθειας , καθώς και μια λειτουργία ιστορικού εντολών που να διευκολύνει τη μετάβαση των χρηστών από το GUI -based εξερεύνηση των δεδομένων στο να οικοδομήσουν και να λειτουργούν batch scripts που να χρησιμοποιούν τις βιβλιοθήκες του EEGLAB και να κάνουν την επεξεργασία πιο γρήγορη. Το EEGLAB προσφέρει μια πληθώρα μεθόδων για visualization και modeling δεδομένων που σχετίζονται με τον εγκέφαλο. Για έμπειρους χρήστες της Matlab, το EEGLAB προσφέρει ένα δομημένο περιβάλλον προγραμματισμού για την αποθήκευση, την πρόσβαση, τη μέτρηση, το χειρισμό και την οπτικοποίηση δεδομένων EEG.

5.6.3.3 Importing data in EEGLAB

Το EEGLab μας δίνει τη δυνατότητα να δημιουργούμε datasets χρησιμοποιώντας πίνακες της Matlab. Με την διαδικασία που αναφέρουμε στο **5.5.2** και τον κώδικα στο παράρτημα **Γ.1** σπάσαμε το αρχείο του κάθε clip και το μετατρέψαμε στην μορφή που το δέχεται το EEGlab. Μέσω αυτού δημιουργήσαμε ένα dataset για κάθε band pass filtering που θέλαμε να κάνουμε.

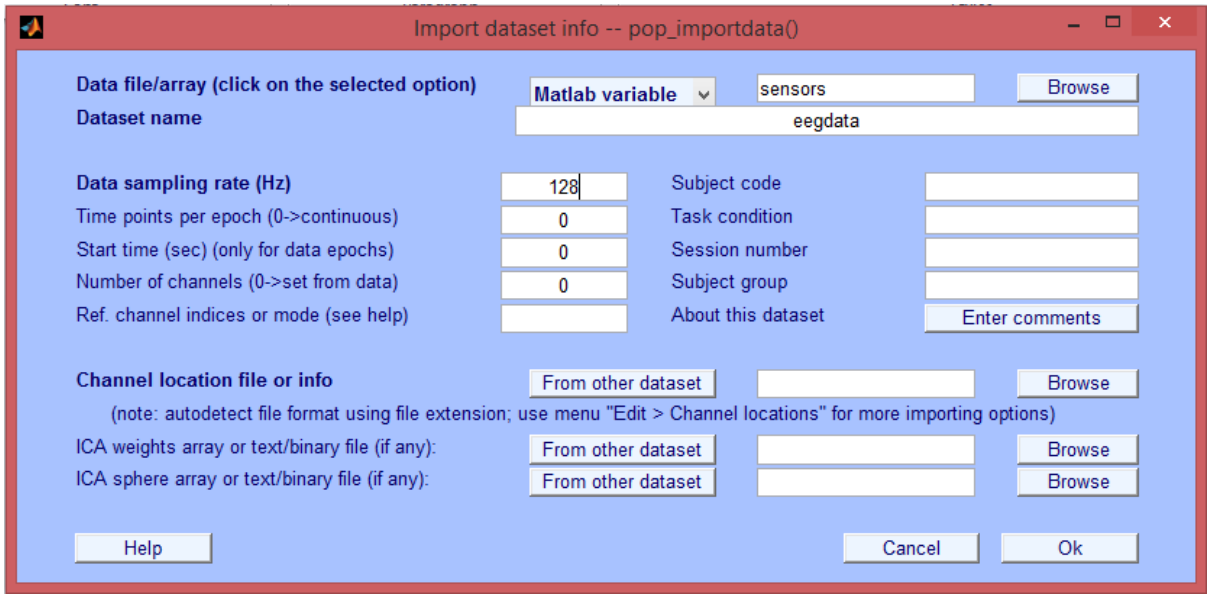

Figure 10: Importing EEGs in EEGLab and dataset Creation

Μέσω του πιο πάνω παραθύρου καθορίζουμε στο EEGLab ποια μεταβλητή να πάρει σαν input, πώς να ονομάσει το dataset μας καθώς και τη συχνότητα καταγραφής των δεδομένων μας

5.6.3.4 Band Pass filtering and exporting

Αφού λοιπόν δημιουργήσουμε το dataset πλέον μπορούμε να το επεξεργαστούμε και να το φιλτράρουμε. Εμείς θέλουμε μόνο συγκεκριμένες συχνότητες να περάσουν. Συχνότητες που ανταποκρίνονται σε συγκεκριμένα bands. Το EEGlab μας δίνει τη δυνατότητα να φιλτράρουμε τα δεδομένα μας με βάση το band που επιθυμούμε να αφήσουμε να περάσει από το φίλτρο

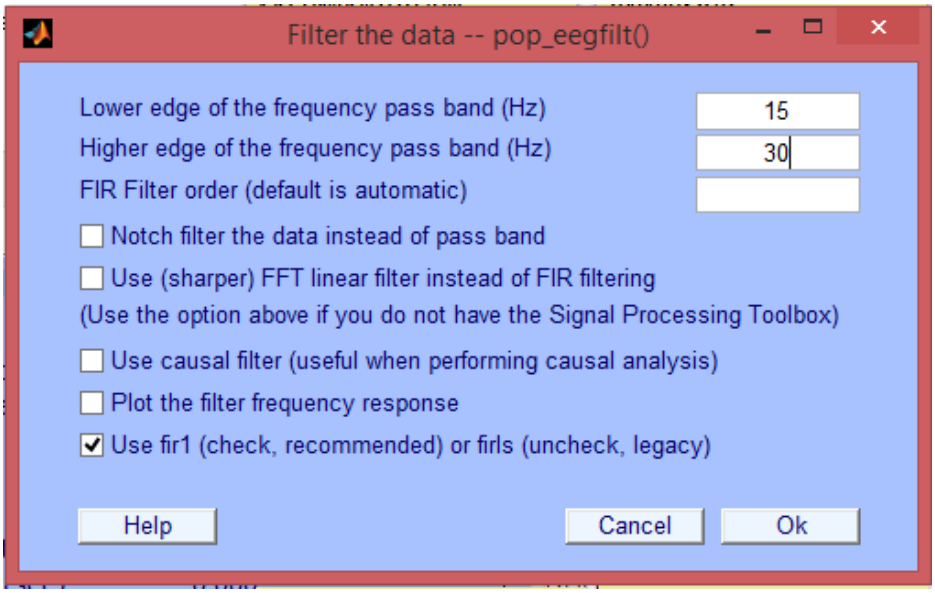

Figure 11: Band Pass Filtering

Στο πιο πάνω παράθυρο δίνουμε τις παραμέτρους με τις οποίες θέλουμε να φιλτράρουμε τα δεδομένα μας. Εμείς θέλουμε για το κάθε clip να πάρουμε συγκεκριμένα bands που αντιπροσωπεύουν συγκεκριμένες λειτουργίες του εγκεφάλου.

Η εγκεφαλική δραστηριότητα του ανθρώπου μοιράζεται σε τέσσερα βασικά bands ανάλογα με την λειτουργία. Με λίγα λόγια τα εγκεφαλικά κύματα έχουν συγκεκριμένη συχνότητα ανάλογα με το τι σκέφτεται το άτομο τη συγκεκριμένη στιγμή. [7]

- **Delta 0-4 Hz:** Τέτοιου είδους εγκεφαλική δραστηριότητα συναντάτε σε άτομα που έχουν μπροστά τους δραστηριότητες που απαιτούν συνεχής προσοχή

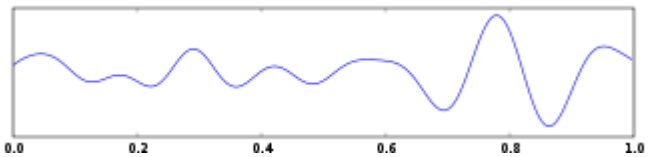

Figure 12: Delta Hz signals Visualization

- **Theta 4-7 Hz:** Έχει βρεθεί σε περιπτώσεις όπου το άτομο προσπαθεί να καταστείλει μία ενέργεια ή αντίδραση.

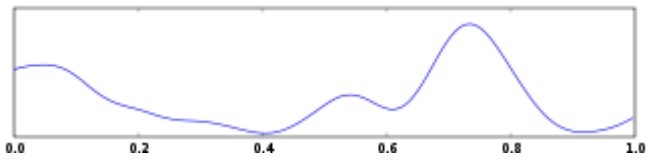

Figure 13: Theta Hz Signals Visualization

- **Alpha 7-14 Hz:** Σχετίζεται με τον έλεγχο της αναστολής των δράσεων ενός ατόμου

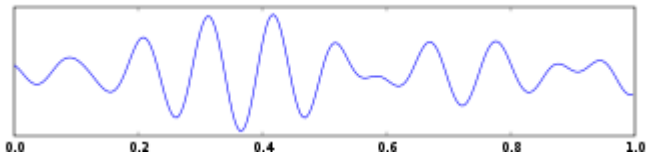

Figure 14: Alpha Hz Signals Visualization

- **Beta 15-30 Hz:** Σχετίζεται με ενεργή, απασχολημένη ή αγχώδης σκέψη

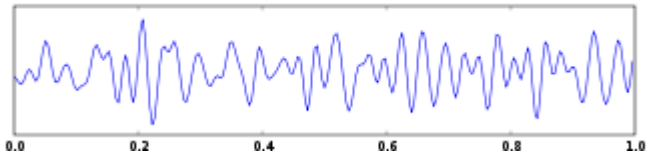

Figure 15: Beta Hz Signals Visualization

Τα φιλτραρισμένα data τα εξάγουμε σε ξεχωριστό .csv αρχείο για κάθε είδους φιλτραρίσματος, για περεταίρω επεξεργασία.

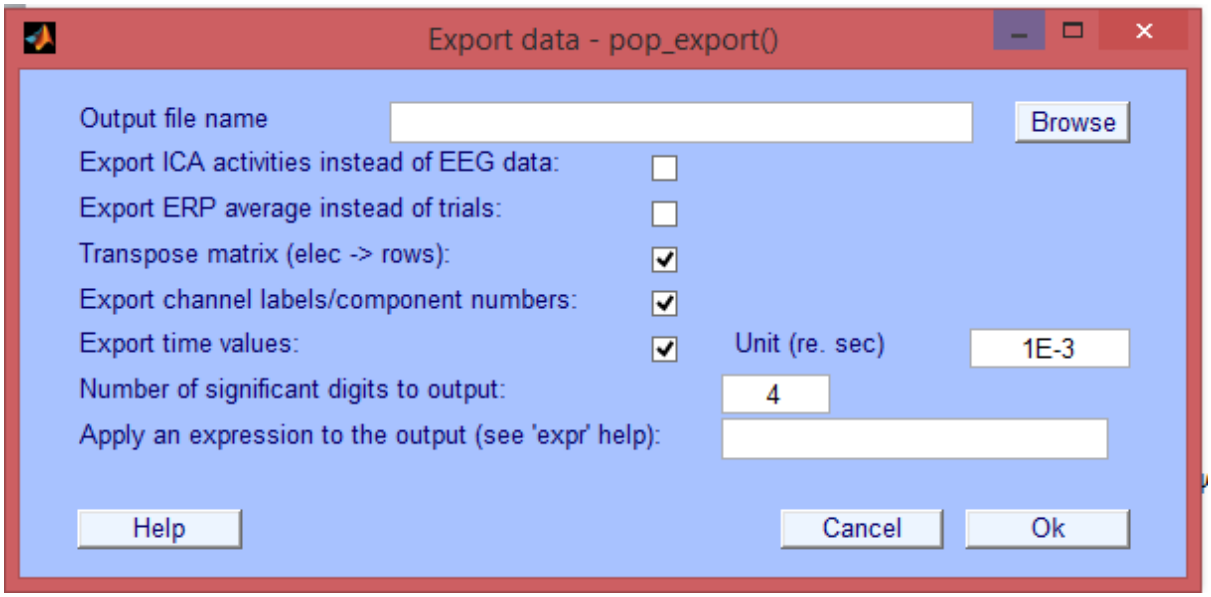

Figure 16: EEGLab data export window

Στην πιο πάνω εικόνα βλέπουμε το παράθυρο εξαγωγής των φιλτραρισμένων EEG δεδομένων. Επιλέγουμε αρχείο από το browse στο οποίο θα φυλάξουμε τα δεδομένα μας (πρέπει να είναι .txt αρχείο) και κάνουμε transpose τον πίνακα των δεδομένων για να μπορούμε να τα επεξεργαστούμε πιο εύκολα προγραμματιστικά.

5.6.4 Gyro Data Processing

Τα δεδομένα του γυροσκοπίου δύο αξόνων του Emotiv αφού τα εξάγουμε από το .csv αρχείο που ήταν μαζί με τα EEG τα επεξεργαζόμαστε για να προστεθούν στο Weka file. Έτσι λοιπόν με ένα Matlab Script διαβάζουμε τον πίνακα που μας επέστρεψε ο κώδικας, και από αυτό βρίσκουμε τον μέσο όρο κίνησης, στον κάθε άξονα, της κεφαλής του ατόμου και τον αποθηκεύουμε σε ξεχωριστό αρχείο για κάθε άξονα.

Ο κώδικας είναι αναρτημένος στο **Παράρτημα Γ.2** Gyro data processing

5.6.5 Filtered Data Processing

Όπως έχω προαναφέρει στο **5.5.3.4** φιλτράραμε τα δεδομένα μας στα διάφορα bands με την βοήθεια του EEGLab και δημιουργήσαμε ένα αρχείο για κάθε band, για το κάθε clip. Δημιούργησα στον υπολογιστή μου ένα φάκελο για το κάθε band και μέσα στον κάθε φάκελο δημιούργησα 14 αρχεία .txt με τις ονομασίες AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4 που αντιστοιχούν στους 14 neuroreceptors του Emotiv. Με ένα απλό Matlab Script, το οποίο σαν είσοδο παίρνει το όνομα του αρχείου που θα επεξεργαστεί και το είδος του band filtering που πέρασαν τα δεδομένα πριν γίνουν export στο αρχείο αυτό, παίρνει το αρχείο και βρίσκει τον μέσο όρο για τον κάθε neuroreceptor. Στη συνέχει χρησιμοποιώντας το input του χρήστη για τι είδους band filtered data έχουμε εντοπίζει τον σωστό φάκελο και κάνει append στο αρχείο του κάθε neuroreceptor τα αντίστοιχα δεδομένα που υπολόγισε για αυτόν.

Ο κώδικας είναι αναρτημένος στο Παράρτημα Παράρτημα **Γ.**3 Filtered Data Processing

5.6.6 Artifact Removal

Τα εγκεφαλικά κύματα είναι ηλεκτρικά σήματα χαμηλής έντασης. Κατά την διάρκεια της καταγραφής τους χρειάζεται να ενισχυθούν (amplify) για να μπορέσουν να καταγραφούν. Αυτό όμως οδηγά στο να ενισχύεται και ο θόρυβος. Ακόμα και το να ανοιγοκλείνει τα μάτια του το «πειραματόζωο» μπορεί να επηρεάσει τα δεδομένα μας και την καταγραφή αυτών.

Τα Artifacts είναι ένα πραγματικό πρόβλημα για τους κλινικούς γιατρούς και ερευνητές. Κάνει την κλινική ερμηνεία των EEG δύσκολη και μπερδεύει πολύ τα αποτελέσματα της έρευνας. Η παραδοσιακή μέθοδος οπτικού ελέγχου για αφαίρεση των Artifacts από τα δεδομένα των πειραμάτων μας είναι αναξιόπιστη, και πολλές αυτοματοποιημένες μέθοδοι εισάγουν καλύτερη διόρθωση. Νέες εξελίξεις στην επεξεργασία των σημάτων μας επιτρέπουν να μειώσουμε τα Artifacts των EEG, διατηρώντας σχέσεις φάσεως. Έτσι αφαιρώντας τα Artifacts θα μειωθεί ο θόρυβος και θα αυξηθεί η αναγνώριση των δεδομένων.

Δεν προλάβαμε να κάνουμε Artifact Removal στα δεδομένα μας αλλά ακόμα και χωρίς αυτό τα αποτελέσματα μας είναι αρκετά ελπιδοφόρα.

Κεφάλαιο 6: Weka Runs

6.1 Τι είναι το Weka;

Το Weka (Waikato Environment for Knowledge Analysis) είναι ένα πολύ χρήσιμο και πολύ γνωστό σύστημα μηχανικής μάθησης γραμμένο στην Java και υλοποιημένο από το πανεπιστήμιο του Waikato στην Νέα Ζηλανδία. [8] Το εργαλείο Weka περιέχει μια συλλογή από εργαλεία απεικόνισης και αλγόριθμους για την ανάλυση δεδομένων και την προγνωστική μοντελοποίηση, μαζί με γραφικά περιβάλλοντα χρήστη για εύκολη πρόσβαση στις λειτουργίες αυτές. Η αρχική έκδοση του Weka είχε αρχικά σχεδιαστεί ως ένα εργαλείο για την ανάλυση των δεδομένων από γεωργικές περιοχές, αλλά η πιο πρόσφατη πλήρως Java-based έκδοση (Weka 3), για την οποία η ανάπτυξη ξεκίνησε το 1997, χρησιμοποιείται σήμερα σε πολλές διαφορετικές περιοχές, κυρίως για εκπαιδευτικούς σκοπούς και έρευνα.

6.2 Γιατί με το Weka;

Το Weka έχει πάρα πολλά πλεονεκτήματα με τα κυριότερα να είναι:

1. Δωρεάν πρόσβαση σε όλους κάτω από το GNU General Public License
2. Πλήρης φορητότητα καθώς είναι γραμμένο στην Java και μπορεί να τρέξει σε οποιανδήποτε σύγχρονη πλατφόρμα.
3. Περιέχει μια ολοκληρωμένη συλλογή προ-επεξεργασίας δεδομένων και τεχνικές μοντελοποίησης.
4. Τεράστια ευκολία χρήσης λόγω των πολύ απλών GUI που παρέχει.

6.3 Αρχείο Weka

Το αρχείο Weka έχει δύο τμήματα. Στο πρώτο τμήμα καθορίζεται το είδος της βάσης. Πρέπει μπροστά να έχει το String @relation. Στην συνέχεια του πρώτου τμήματος καθορίζονται τα attributes της βάσης μας. Κάθε attribute καθορίζεται με ένα συγκεκριμένο όνομα και ένα τύπο. Ο τύπος καθορίζεται το είδος των δεδομένων που θα εισαχθούν στη βάση για το συγκεκριμένο attribute. Οι τύποι δεδομένων που δέχεται το Weka είναι numeric, nominal, string και date. Εξ ορισμού το στοιχείο που ψάχνουμε να βρούμε αν μπορεί να αναγνωριστεί με συστήματα τεχνητής νοημοσύνης είναι το τελευταίο attribute στη λίστα. Αμέσως μετά μπαίνει η λέξη @data και από εκεί ξεκινούν τα δεδομένα μας. Κάθε γραμμή περιλαμβάνει όλα τα attributes που καλύψαμε στο header και χωρίζονται μεταξύ τους με ένα κόμμα.

6.4 Ετοιμασία δεδομένων

Αφού ολοκληρώθηκε η διαδικασία του ερωτηματολογίου και η ανάλυση των EEG πρέπει να ετοιμάσουμε τα filtered data και τα Gyro Data για να εισαχθούν στο Weka για να βρούμε πόσο εύκολα μπορεί να δημιουργηθεί ένα σύστημα Τεχνητής Νοημοσύνης, με τα δεδομένα που έχουμε βρει, το οποίο να αναγνωρίζει το συναίσθημα του χρήστη αυτόματα.

Με μία εφαρμογή στη C# παίρνω όλα τα δεδομένα που εξάγαμε από το EEGLab και δημιουργώ το Weka File. Σαν attributes στο αρχείο έβαλα το κάθε band από όλους τους neuroreceptors του Emotiv καθώς και τους δύο άξονες του γυροσκοπίου.

Κάθε γραμμή λοιπόν στο Weka περιλάμβανε 56 αριθμητικές τιμές για τους 14 neuroreceptors, 2 για τα γυροσκόπια και μία τελική τιμή που καθόριζε το συναίσθημα. Με βάση την τιμή αυτή ζητούσαμε από το Weka να μας βγάλει πόσο μπορεί να αναγνωριστεί το συναίσθημα με βάση τα δεδομένα που του δώσαμε

Ο κώδικας της C# εφαρμογής είναι αναρτημένος στο παράρτημα **Παράρτημα Δ:** Weka File Creation source code

6.5 First Weka Run

Αφού δημιουργήσαμε το αρχείο Weka με τα στοιχεία που προαναφέραμε στο **6.4** Ετοιμασία δεδομένων τα φορτώσαμε στο Weka και τρέξαμε αλγόριθμο Logistic Regression για να δούμε πόσο εύκολο είναι για ένα σύστημα να αναγνωρίσει το συναίσθημα με βάση τα EEG Data που του δώσαμε. Τον αλγόριθμο Logistic Regression τον επέλεξα διότι είναι ο αλγόριθμος που έδωσε τα καλύτερα αποτελέσματα, παρά το γεγονός ότι δεν είναι και τόσο καλά.

6.5.1 Weka Results

Τρέχοντας λοιπόν το Weka με αλγόριθμο Logistic Regression το Weka μας έδωσε τα εξής αποτελέσματα:

Correctly Classified Instances **11** **28.2051 %**

Incorrectly Classified Instances **28** **71.7949 %**

Kappa statistic **-0.0931**

Mean absolute error **0.4691**

Root mean squared error **0.6767**

Relative absolute error **106.8565 %**

Root relative squared error **144.1801 %**

Total Number of Instances **39**

και το εξής confusion matrix

	A	**B**	**C**
A	6	4	5
B	5	3	3
C	6	6	1

Figure 17: Confusion Matrix for First Weka Run

Όπου a αντιστοιχεί στο συναίσθημα 1 που θέσαμε στο data file μας, b στο 2 και c στο 3. Οι αντιστοιχίες αριθμών με συναίσθημα είναι 1=Engagement, 2=Excitement και 3=Frustration.

Στο συγκεκριμένο πίνακα μας δηλώνει σε κάθε κατηγορία πώς αναγνωρίζει το σύστημα το κάθε στοιχείο της. Δηλαδή από τα 16 στοιχεία που ανήκουν στην κατηγορία a=Engagement τα 7 από αυτά αναγνωρίζονται σωστά ενώ τα υπόλοιπα αναγνωρίζονται λανθασμένα. Ένα ποσοστό κάτω από το 50%. Ο κύριος λόγος στον οποίο οφείλεται το χαμηλό αυτό ποσοστό αναγνώρισης είναι το overfeeding των δεδομένων μας καθώς έχουμε σύνολο 58 attributes αλλά μόνο 39 γραμμές δεδομένων.

6.5.2 First Run Conclusions

Έτσι λοιπόν πρέπει να βρούμε ένα τρόπο να μειώσουμε τα attributes, να μείνουν μόνο τα σημαντικά που καθορίζουν τα συναισθήματα που θέλουμε να αναγνωρίσουμε διότι ανάλογα με το ποιο συναίσθημα νιώθουμε ενεργοποιούνται και απενεργοποιούνται διαφορετικά τμήματα του εγκεφάλου και πρέπει εμείς να βρούμε τη σωστή σχέση μεταξύ αυτών για να γίνει σωστή η αναγνώριση από το σύστημα.

6.6 Second Weka Run

6.6.1 Remove Gyros

Δύο attributes που αποφασίσαμε να αφαιρέσουμε χωρίς αναλυτική μελέτη είναι τα δύο γυροσκόπια που σκοπός τους είναι να μας δώσουν την θέση του κεφαλιού του χρήστη σε δύο άξονες. Το κεφάλι του χρήστη είναι ένα από τα joints που παίρνουμε με το Kinect γι’ αυτό αποφασίσαμε να το αφαιρέσουμε από τα EEG δεδομένα μας.

6.6.2 Bayes Network

Αποφασίσαμε να αλλάξουμε τον αλγόριθμο Τεχνητής Νοημοσύνης τον οποίο χρησιμοποιούσαμε για την εκπαίδευση του συστήματος και να χρησιμοποιήσουμε ένα δίκτυο Bayes. Ο σκοπός της αλλαγής ήταν το γεγονός ότι το Bayes Network είναι πολύ καλύτερο στην εκπαίδευση όταν τα data είναι over-fitted κάτι που ισχύει στην περίπτωση μας καθώς έχουμε 56 attributes και μόνο 39 γραμμές δεδομένων. [14]

6.6.3 Second Run Results

Τρέχοντας το Weka με το Δίκτυο Bayes αντί Logistic Regression καθώς και με την αφαίρεση τον Gyros βελτιωθήκαν πάρα πολύ τα αποτελέσματα μας. Συγκεκριμένα πήραμε τα εξής αποτελέσματα

Correctly Classified Instances **21** **53.8462 %**

Incorrectly Classified Instances **18** **46.1538 %**

Kappa statistic **0.3036**

Mean absolute error **0.396**

Root mean squared error **0.473**

Relative absolute error **89.2869 %**

Root relative squared error **100.178 %**

Total Number of Instances **39**

και τον πιο κάτω Confusion Matrix:

	A	**B**	**C**
A	9	1	5
B	2	3	6
C	0	4	9

Figure 18: Confusion Matrix for Second Weka Run

Βλέπουμε ότι πλέον δεν μπερδεύονται τα στιγμιότυπα του Frustration με το Engagement καθώς και το Engagement έχει τον πιο ψηλή αναγνώριση. Επίσης βλέπουμε ότι υπάρχει ακόμα ένα μπέρδεμα μεταξύ Excitement και Frustration. Με το κύριο συναίσθημα που να μην αναγνωρίζετε καλά να είναι το Excitement καθώς μόνο τρία από τα 11 clips αναγνωρίζονται σωστά.

Κεφάλαιο 7 Future Work

Ο συγκεκριμένος τομέας είναι υπερβολικά τεράστιος και υπάρχουν πολλές μελλοντικές διεργασίες οι οποίες μπορούν να γίνουν.

Για αρχή να εκτελεστούν περισσότερα πειράματα με καινούρια άτομα έτσι ώστε να έχουμε περισσότερα δεδομένα για ανάλυση για να βγουν και καλύτερα αποτελέσματα. Σε ένα αλγόριθμο τεχνητής νοημοσύνης όσο περισσότερα δεδομένα του δώσουμε, τόσο καλύτερη θα είναι η αναγνώριση που θα έχει.

Στη συνέχεια να εκτελεστεί το artifact removal που αναφέρουμε και περιγράφουμε στο **5.6.6** Artifact Removal.

Στη συνέχεια μπορούμε να βρούμε τα Inverse Dynamics των clips χρησιμοποιώντας το opensim και τα Mocap Data που έχουμε ήδη καταγράψει. Έτσι μπορούμε να υπολογίσουμε τις δυνάμεις που ασκούνται στο κάθε κόκαλο ή κάθε κομμάτι του σκελετού κατά τη διάρκεια έκφρασης συγκεκριμένου συναισθήματος και να δημιουργήσουμε μία μαθηματική απεικόνιση της συσχέτισης των δυνάμεων που ασκούνται σε κάθε κόκαλο με το συναίσθημα που εκφράζουν.

Επίσης χρησιμοποιώντας τα δεδομένα που πήραμε με το Microsoft Kinect μπορούμε να βρούμε μία συσχέτιση μεταξύ των Skeleton Data και των EEG για καλύτερη αναγνώριση του συναισθήματος.

Works Cited

1. *The prevalence of 'alexithymic' characteristics in psychosomatic patients. Psychotherapy and psychosomatics.* **Petros, Sifneos.**

2. *Seeing Fearful Body Expressions Activates the Fusiform Cortex and Amygdala.* **Gelder, Nouchine Hadjikhani1 and Beatrice de.**

3. *Continuous recognition of player's affective body expression as dynamic quality of aesthetic experience* . **Nikolaos Savva, Alfonsina Scarinzi, Nadia Bianchi-Berthouze.**

4. *Affect Recognition During Active Game Playing Based on Posture Skeleton Data.* **Haris Zacharatos, Christos Gatzoulis and Yiorgos Chrysanthou.**

5. *Affect recognition for games using body postures* . **Zacharatos, Theocharis.**

6. *Assessing agreement on classification tasks: The kappa statistic. Computational Linguistics.* **Carletta, Jean.**

7. *Data Mining: Practical machine learning tools and techniques, 3rd Edition.* **Witten, Ian H. και Eibe Frank, Mark A. Hall.**

8. *Comparative analysis of event-related potentials during Go/NoGo and CPT: Decomposition of electrophysiological markers of response inhibition and sustained attention.* **Kirmizi-Alsan, Elif, και συν., και συν.**

9. *Affective Computing: A Review.* **Tao, Jianhua, Tieniu Tan.**

10. *The Spline-Laplacian in Clinical Neurophysiology.* **Nunez, Paul L. και Pilgreen, Kenneth L.**

11. *The spatial location of EEG electrodes: Locating the best-fitting sphere relative to cortical anatomy.* **Towle, Vernon L., και συν., και συν.**

12. *Real-time EEG-based Emotion Recognition and its Applications.* **Yisi Liu, Olga Sourina, and Minh Khoa Nguye.**

13. *EEG-based Emotion Recognition, The Influence of Visual and Auditory Stimuli.* **Bos, Danny Oude.**

14. *Advantages and challenges of Bayesian networks in environmental modelling.* **Uusitalo, Laura.**

Παράρτημα Α Κώδικας για τα δεδομένα του σκελετού

Παράρτημα Α.1: Κώδικας Kinect για Data Capturing

```
// this code is added to give the software functionality to export skeletal
information captured through the sensors
// a while loop is runs when the RecordTextFiles function is activated through a
button in the interface
// at every frame of the second ,the loop exports the X,Y,Z values of all identified
joints right after rendering
// the skeletal structure
// a Boolean "RecordTextFiles" is set from the interface class "KinectDiagnosticViewer
int i = 0;
JointType[] allJoints = new JointType[20] {JointType.HipCenter, JointType.Spine,
JointType.ShoulderCenter, JointType.Head, JointType.ShoulderLeft, JointType.ElbowLeft,
JointType.WristLeft, JointType.HandLeft, JointType.ShoulderRight,
JointType.ElbowRight, JointType.WristRight, JointType.HandRight, JointType.HipLeft,
JointType.KneeLeft, JointType.AnkleLeft, JointType.FootLeft, JointType.HipRight,
JointType.KneeRight,JointType.AnkleRight, JointType.FootRight};
//an array is created to contain all joint values (X,Y,Z for each joint)

while (RecordTextFiles == true & i < 20)
{
Joint j = this.currentSkeleton.Joints[allJoints[i]];
string output = (j.Position.X + @"," + j.Position.Y + @"," + j.Position.Z + @",");
//append each joint current values to a string
BoneRotation r =
this.currentSkeleton.BoneOrientations[allJoints[i]].HierarchicalRotation;
output = output + (r.Quaternion.X + @"," + r.Quaternion.Y + @"," + r.Quaternion.Z +
@"," + r.Quaternion.W + @",");
System.IO.File.AppendAllText(@"E:\haris\experiment\data\Skeleton" +
currentSkeleton.TrackingId + ".txt", output);
//Append the current values of the joints for the current frame to the exported
file,the file will be name after the
//skeleton tracking Id so that it'd write a different file for multiple skeletons
if (i == 19)
{

long milliseconds = DateTime.Now.Ticks / TimeSpan.TicksPerMillisecond;
System.IO.File.AppendAllText(@"E:\haris\experiment\data\Skeleton" +
currentSkeleton.TrackingId + ".txt", "" + milliseconds.ToString() +
System.Environment.NewLine);
}
//save the current CPU tick at the end of each line

i++; // go to next joint

}
```

Ο πιο πάνω κώδικας δημιουργεί ένα πίνακα στον οποίο να είναι φυλαγμένα όλα τα Joints του σκελετού που θέλουμε να καταγράψουμε (σύνολο 20) και στη συνέχεια με ένα Loop

διασχίζει τον πίνακα και για κάθε Joint αποθηκεύει σε ένα αρχείο τις x,y,z συντεταγμένες του Joint καθώς, το Τετραδόνιο (Quaternion) x,y,z για την κάθε μια καθώς και το Orientation του κάθε Joint με βάση τον πατέρα του στην κινηματική αλυσίδα την οποία ανήκει

Παράρτημα Α.2: Κώδικας για μετατροπή των clock ticks σε timestamp

```
static void Main(string[] args)
        {
            String path = args[0] + "\\Skeleton675.txt";
            List<String> lines = File.ReadAllLines(path).ToList<String>();
            List<String> newLines=new List<string>();
            foreach (String s in lines)
            {
                int locationAt = s.LastIndexOf(',');
                string tickString = s.Substring(locationAt + 1);
                long ticks = long.Parse(tickString);

                long hours = TimeSpan.FromMilliseconds(ticks).Hours;
                long min = TimeSpan.FromMilliseconds(ticks).Minutes;
                long sec = TimeSpan.FromMilliseconds(ticks).Seconds;
                long milsec = TimeSpan.FromMilliseconds(ticks).Milliseconds;

                String timestamp = hours.ToString() + ":" + min.ToString() + ":" +
sec.ToString() + ":" + milsec.ToString();
                String newLine = s.Substring(0, locationAt) + ","+timestamp + "\n";
                newLines.Add(newLine);
            }

            File.WriteAllLines(path, newLines);


        }
```

Ο πιο πάνω κώδικας διαβάζει το κάθε αρχείο με τα Skeleton Data, πάει στο τελευταίο στοιχείο της κάθε γραμμής διαβάζει τον αριθμό των Clock Ticks την στιγμή του Data Capture και το μετατρέπει σε Timestamp. Το Path δίνεται σαν command line argument και το όνομα του αρχείου είναι hard coded στον κώδικα. Γίνεται ένα string concatenation με τα δύο. Διαβάζονται όλες οι γραμμές σαν ξεχωριστά Strings και προσθέτονται σε μία λίστα. Διασχίζεται η λίστα, επεξεργάζεται το κάθε String και δημιουργείτε ένα νέο με τα διορθωμένα στοιχεία το οποίο φυλάγεται σε μια νέα λίστα. Μόλις ολοκληρωθεί η

επεξεργασία τότε η νέα λίστα αποθηκεύεται στο ίδιο αρχείο κάνοντας Overwrite τα προηγούμενα data.

Παράρτημα Α.3: Κώδικας για On Screen Rendering του σκελετού

```
if (this.ShowBones)
{
// Render Torso
this.DrawBone(drawingContext, JointType.Head, JointType.ShoulderCenter);
this.DrawBone(drawingContext, JointType.ShoulderCenter, JointType.ShoulderLeft);
this.DrawBone(drawingContext, JointType.ShoulderCenter, JointType.ShoulderRight);
this.DrawBone(drawingContext, JointType.ShoulderCenter, JointType.Spine);
this.DrawBone(drawingContext, JointType.Spine, JointType.HipCenter);
this.DrawBone(drawingContext, JointType.HipCenter, JointType.HipLeft);
this.DrawBone(drawingContext, JointType.HipCenter, JointType.HipRight);

// Left Arm
this.DrawBone(drawingContext, JointType.ShoulderLeft, JointType.ElbowLeft);
this.DrawBone(drawingContext, JointType.ElbowLeft, JointType.WristLeft);
this.DrawBone(drawingContext, JointType.WristLeft, JointType.HandLeft);

// Right Arm
this.DrawBone(drawingContext, JointType.ShoulderRight, JointType.ElbowRight);
this.DrawBone(drawingContext, JointType.ElbowRight, JointType.WristRight);
this.DrawBone(drawingContext, JointType.WristRight, JointType.HandRight);

// Left Leg
this.DrawBone(drawingContext, JointType.HipLeft, JointType.KneeLeft);
this.DrawBone(drawingContext, JointType.KneeLeft, JointType.AnkleLeft);
this.DrawBone(drawingContext, JointType.AnkleLeft, JointType.FootLeft);

// Right Leg
this.DrawBone(drawingContext, JointType.HipRight, JointType.KneeRight);
this.DrawBone(drawingContext, JointType.KneeRight, JointType.AnkleRight);
this.DrawBone(drawingContext, JointType.AnkleRight, JointType.FootRight);

if (this.ShowJoints)
   {
   // Render Joints
      foreach (JointMapping joint in this.jointMappings.Values)
      {
        Brush drawBrush = null;
        switch (joint.Joint.TrackingState)
          {
          case JointTrackingState.Tracked:
            drawBrush = this.trackedJointBrush;
            break;
          case JointTrackingState.Inferred:
            drawBrush = this.inferredJointBrush;
            break;
          }

        if (drawBrush != null)
```

```
        {
         drawingContext.DrawEllipse(drawBrush, null, joint.MappedPoint, JointThickness
* this.scale, JointThickness * this.scale);
        }
      }
  }
}
```

Ο παραπάνω κώδικας είναι υπεύθυνος για την να παρουσιάζει στην οθόνη τον κάθε σκελετό που λαμβάνει μέρος στο πείραμα. Αυτό το χρησιμοποιήσαμε για να καταγράφουμε video της οθόνης (Screen Capture) για να δημιουργήσουμε ερωτηματολόγια έτσι ώστε να γίνουν σωστά annotate τα clips που κόψαμε.

Παράρτημα Β: Κώδικας εφαρμογής για καταγραφή των Brain Signals

```
#include <iostream>
#include <fstream>
#include <conio.h>
#include <sstream>
#include <windows.h>
#include <map>
#include <time.h>

#include "EmoStateDLL.h"
#include "edk.h"
#include "edkErrorCode.h"

#pragma comment(lib, "../lib/edk.lib")

EE_DataChannel_t targetChannelList[] = {
		ED_COUNTER,
		ED_AF3, ED_F7, ED_F3, ED_FC5, ED_T7,
		ED_P7, ED_O1, ED_O2, ED_P8, ED_T8,
		ED_FC6, ED_F4, ED_F8, ED_AF4, ED_GYROX, ED_GYROY, ED_TIMESTAMP,
		ED_FUNC_ID, ED_FUNC_VALUE, ED_MARKER, ED_SYNC_SIGNAL
	};

const char header[] = "COUNTER,AF3,F7,F3, FC5, T7, P7, O1, O2,P8"
			", T8, FC6, F4,F8, AF4,GYROX, GYROY, TIMESTAMP, "
			"FUNC_ID, FUNC_VALUE, MARKER, SYNC_SIGNAL,TIME,";
const char affectivSuitesName[] =
"Time,Engagement,Frustration,Meditation,Excitement,";

// output files : Affectiv_Data.csv and EEG_Data.csv

int main(int argc, char** argv) {

	EmoEngineEventHandle eEvent = EE_EmoEngineEventCreate();
	EmoStateHandle eState = EE_EmoStateCreate();
	unsigned int userID =0;
	const unsigned short composerPort= 1726;
	float secs= 1;
	unsigned int datarate= 0;
	bool readytocollect= false;
	int option= 0;
	int state= 0;

	std::string input;

	try {


std::cout << "==================================================================" <<
std::endl;
std::cout << "This application logs EEG Data and Affectiv Data from
EmoEngine/EmoComposer."		<< std::endl;
```

```
std::cout << "================================================================" <<
std::endl;
std::cout << "Press '1' to start and connect to the EmoEngine                 " <<
std::endl;
std::cout << "Press '2' to connect to the EmoComposer                         " <<
std::endl;
std::cout << ">> ";
std::getline(std::cin, input, '\n');
option = atoi(input.c_str());

switch (option) {
	case 1:
	{
		if (EE_EngineConnect() != EDK_OK) {
			throw std::exception("Emotiv Engine start up failed.");
		}
		break;
	}
	case 2:
	{
		std::cout << "Target IP of EmoComposer? [127.0.0.1] ";
		std::getline(std::cin, input, '\n');
		if (input.empty()) {
			input = std::string("127.0.0.1");
		}
		if (EE_EngineRemoteConnect(input.c_str(), composerPort) != EDK_OK) {
			std::string errMsg = "Cannot connect to EmoComposer on ["+ input
+ "]";
			throw std::exception(errMsg.c_str());
		}
		break;
	}
	default:
		throw std::exception("Invalid option...");
		break;
}


std::cout << "Start receiving EEG Data and affectiv data! Press any key to stop
logging...\n" << std::endl;
std::ofstream ofs("../bin/EEG_Data.csv",std::ios::trunc);
ofs << header << std::endl;
std::ofstream ofs2("../bin/Affectiv_Data.csv",std::ios::trunc);
ofs2 << affectivSuitesName << std::endl;

DataHandle hData = EE_DataCreate();
EE_DataSetBufferSizeInSec(secs);

std::cout << "Buffer size in secs:" << secs << std::endl;

while (!_kbhit()) {

	state = EE_EngineGetNextEvent(eEvent);
	EE_Event_t eventType;

	if (state == EDK_OK) {

		eventType = EE_EmoEngineEventGetType(eEvent);
		EE_EmoEngineEventGetUserId(eEvent, &userID);
		EE_EmoEngineEventGetEmoState(eEvent, eState);
```

```
		// Log the EmoState if it has been updated
		if (eventType == EE_UserAdded) {
			std::cout << "User added";
			EE_DataAcquisitionEnable(userID,true);
			readytocollect = true;
		}


		if (readytocollect && (eventType == EE_EmoStateUpdated)) {

			EE_DataUpdateHandle(0, hData);

			unsigned int nSamplesTaken=0;

			EE_DataGetNumberOfSample(hData,&nSamplesTaken);

			std::cout << "Updated " << nSamplesTaken << std::endl;


			time_t currentTime = time(0);

			if (nSamplesTaken != 0  ) {

				double* data = new double[nSamplesTaken];
		for (int sampleIdx=0 ; sampleIdx<(int)nSamplesTaken ; ++ sampleIdx) {
		for (int i=0; i<sizeof(targetChannelList)/sizeof(EE_DataChannel_t) ;
i++) {

			EE_DataGet(hData, targetChannelList[i], data, nSamplesTaken);
			ofs << data[sampleIdx] << ",";
		}
			ofs << ","<< ctime(&currentTime) << std::endl;
                                        }
			delete[] data;
}



float affEngegement = ES_AffectivGetEngagementBoredomScore(eState);
float affFrus = ES_AffectivGetFrustrationScore(eState);
float affMed = ES_AffectivGetMeditationScore(eState);
float affExcitement = ES_AffectivGetExcitementShortTermScore(eState);
printf("Engagement: %f, Frustration: %f, ...\n",affEngegement,affFrus);
ofs2 <<
ctime(&currentTime)<<","<<affEngegement<<","<<affFrus<<","<<affMed<<","<<affExcitement
<<","<<std::endl;

}
}
Sleep(100);
}

ofs.close();
ofs2.close();
EE_DataFree(hData);

}
	catch (const std::exception& e) {
		std::cerr << e.what() << std::endl;
```

```
            std::cout << "Press any key to exit..." << std::endl;
            getchar();
        }

        EE_EngineDisconnect();
        EE_EmoStateFree(eState);
        EE_EmoEngineEventFree(eEvent);

        return 0;
}
```

Ο παραπάνω κώδικας είναι μία εφαρμογή η οποία επικοινωνεί με Emotiv EPOC και καταγράφει τα brain signals του πειραματόζωου. Η εφαρμογή είναι γραμμένη σε C++ διότι οι βιβλιοθήκες της Emotiv παρέχονται για όλα τα λειτουργικά συστήματα κι έτσι μία εύκολη και γρήγορη γλώσσα που τρέχει παντού είναι η C++. Πρέπει να ξαναγίνουν compile σε κάθε λειτουργικό σύστημα που θα χρησιμοποιήσουμε την συσκευή. Χρειαζόμαστε ακρίβεια millisecond και γι' αυτό χρησιμοποιήσαμε C++. Μπορεί η Java να απλοποιεί τις διαδικασίες χρησιμοποίησης της συσκευής σε διάφορα λειτουργικά συστήματα αλλά η μεταφορά από Intermediate Code σε JVM Code και από εκεί σε specific kernel code είναι πολύ χρονοβόρα και μας είναι πλήρως άχρηστο.

Παράρτημα Γ: Matlab Scripts

Παράρτημα Γ.1: Preprocessing

```
%This function's main purpose is to read the EEG file and
split EEG from
%Gyro data so it can be proccessed in eeglab
function [or, gyro,sensors] =split(path)

%Read the csv file provided
or=xlsread(path);

% get the gyro data in a separate matrix so we proccess only
EEG data
gyro=or(:,15:16);

%get the sensor data so we can proccess it
sensors=or(:,1:14);

%transpose the array so it will be in the format that eeglab
accepts
sensors=sensors';
```

Ένα μικρό κομμάτι Matlab κώδικα το οποίο διαβάζει το αρχείο με τα EEG data και σπάζει τα EEG και τα Gyro data σε δύο διαφορετικούς πίνακες και τα επιστρέφει. Επίσης βρίσκει τον ανάστροφο των EEG data για να είναι σε μορφή που τα δέχεται το EEGLab.

Παράρτημα Γ.2 Gyro data processing

```
function[]=gyroproccess(gyro)
format long
[rowsg,colsg]=size(gyro);
sumg=zeros(2);
sumg=sumg(:,1);

%add contents of each column to get average of GyroX and GyroY
for col=1:colsg
    for row=1:rowsg
        sumg(col)=sumg(col)+gyro(row,col);
    end
end

%get average of gyroX and GyroY
 sumg(1)=sumg(1)/rowsg;
 sumg(2)=sumg(2)/rowsg;
d='D:\Dropbox\data\gyroX.txt';
fid=fopen(d,'at');
fprintf(fid,'%f',sumg(1));
fprintf(fid,'\n');
fclose(fid);
d='D:\Dropbox\data\gyroY.txt';
fid=fopen(d,'at');
fprintf(fid,'%f',sumg(2));
fprintf(fid,'\n');
```

Ένα μικρό script το οποίο παίρνει το δεύτερο output του κώδικα στο Παράρτημα Ε.1 το επεξεργάζεται και το αποθηκεύει σε δύο ξεχωριστά αρχεία. Ένα για τον άξονα X και ένα για τον άξονα Y.

Παράρτημα Γ.3 Filtered Data Processing

```
function []=sensorprocessing(path,band)
%read the excel file of the filtered data
filtered=xlsread(path);
%get rows and columns of each matrix
[rowsf,colsf]=size(filtered);
%remove header line
filtered=filtered(:,2:colsf);
%remove counter
filtered=filtered(2:rowsf,:);
%string of the path of files
str01='D:\Dropbox\data\';
%create sum array full of zeros
sumf=zeros(colsf);
sumf=sumf(:,1);
%get rows and columns of the resized matrix
[rowsf,colsf]=size(filtered);
%add contents of each column to get the average of sensors
for col = 1:colsf
    for row = 1:rowsf
        sumf(col)=sumf(col)+filtered(row,col);
    end
end

%get average of each column of sensors
for col=1:colsf
    sumf(col)=sumf(col)/rowsf;
end

%concatenate strings for the text file
str01=strcat(str01,band);
str01=strcat(str01,'\');

%create an array for the text files represnting each sensor
strSensor={'AF3','F7','F3','FC5','T7','P7','O1','O2','P8','T8'
,'FC6','F4','F8','AF4'};
[rowsS,colsS]=size(strSensor);

%for each sensor find the proper file, open it and append data
for x=1:colsS
    str02=strcat(str01,strSensor{x});
    str02=strcat(str02,'.txt');
    fid=fopen(str02,'at');
    fprintf(fid,'%f',sumf(x));
    fprintf(fid,'\n');
    fclose(fid);
end
```

Παράρτημα Δ: Weka File Creation source code

```
static void Main(string[] args)
{
    //Initialize Strings to easier work with paths
    String alphaPath = @"D:\Dropbox\data\alpha\";
    String betaPath = @"D:\Dropbox\data\beta\";
    String deltaPath = @"D:\Dropbox\data\delta\";
    String thetaPath = @"D:\Dropbox\data\theta\";
    String wekaPath = @"D:\Dropbox\data\eeg.arff";
    String gyroPathX = @"D:\Dropbox\data\gyroX.txt";
    String gyroPathY = @"D:\Dropbox\data\gyroY.txt";

    List<String> wekafile = new List<string>();
    wekafile.Add("@relation eeg");
    wekafile.Add("@attribute 'AF3alpha' numeric");
    wekafile.Add("@attribute 'AF3beta' numeric");
    wekafile.Add("@attribute 'AF3delta' numeric");
    wekafile.Add("@attribute 'AF3theta' numeric");
    wekafile.Add("@attribute 'F7alpha' numeric");
    wekafile.Add("@attribute 'F7beta' numeric");
    wekafile.Add("@attribute 'F7delta' numeric");
    wekafile.Add("@attribute 'F7theta' numeric");
    wekafile.Add("@attribute 'F3alpha' numeric");
    wekafile.Add("@attribute 'F3beta' numeric");
    wekafile.Add("@attribute 'F3delta' numeric");
    wekafile.Add("@attribute 'F3theta' numeric");
    wekafile.Add("@attribute 'FC5alpha' numeric");
    wekafile.Add("@attribute 'FC5beta' numeric");
    wekafile.Add("@attribute 'FC5delta' numeric");
    wekafile.Add("@attribute 'FC5theta' numeric");
    wekafile.Add("@attribute 'T7alpha' numeric");
    wekafile.Add("@attribute 'T7beta' numeric");
    wekafile.Add("@attribute 'T7delta' numeric");
    wekafile.Add("@attribute 'T7theta' numeric");
    wekafile.Add("@attribute 'P7alpha' numeric");
    wekafile.Add("@attribute 'P7beta' numeric");
    wekafile.Add("@attribute 'P7delta' numeric");
    wekafile.Add("@attribute 'P7theta' numeric");
    wekafile.Add("@attribute 'O1alpha' numeric");
    wekafile.Add("@attribute 'O1beta' numeric");
    wekafile.Add("@attribute 'O1delta' numeric");
    wekafile.Add("@attribute 'O1theta' numeric");
    wekafile.Add("@attribute 'O2alpha' numeric");
    wekafile.Add("@attribute 'O2beta' numeric");
    wekafile.Add("@attribute 'O2delta' numeric");
    wekafile.Add("@attribute 'O2theta' numeric");
    wekafile.Add("@attribute 'P8alpha' numeric");
    wekafile.Add("@attribute 'P8beta' numeric");
    wekafile.Add("@attribute 'P8delta' numeric");
    wekafile.Add("@attribute 'P8theta' numeric");
    wekafile.Add("@attribute 'T8alpha' numeric");
    wekafile.Add("@attribute 'T8beta' numeric");
    wekafile.Add("@attribute 'T8delta' numeric");
    wekafile.Add("@attribute 'T8theta' numeric");
    wekafile.Add("@attribute 'FC6alpha' numeric");
    wekafile.Add("@attribute 'FC6beta' numeric");
    wekafile.Add("@attribute 'FC6delta' numeric");
    wekafile.Add("@attribute 'FC6theta' numeric");
    wekafile.Add("@attribute 'F4alpha' numeric");
    wekafile.Add("@attribute 'F4beta' numeric");
```

```
            wekafile.Add("@attribute 'F4delta' numeric");
            wekafile.Add("@attribute 'F4theta' numeric");
            wekafile.Add("@attribute 'F8alpha' numeric");
            wekafile.Add("@attribute 'F8beta' numeric");
            wekafile.Add("@attribute 'F8delta' numeric");
            wekafile.Add("@attribute 'F8theta' numeric");
            wekafile.Add("@attribute 'AF4alpha' numeric");
            wekafile.Add("@attribute 'AF4beta' numeric");
            wekafile.Add("@attribute 'AF4delta' numeric");
            wekafile.Add("@attribute 'AF4theta' numeric");
            wekafile.Add("@attribute 'GyroX' numeric");
            wekafile.Add("@attribute 'GyroY' numeric");

            wekafile.Add("@attribute Mood {1,2,3}\r\n");
            wekafile.Add("@data\r\n ");

            //create an Array of Lists for each band pass filtered data. Array has a
size of 14, one for each sensors
            List<String>[] alpha = new List<string>[14];
            List<String>[] beta = new List<string>[14];
            List<String>[] delta = new List<string>[14];
            List<String>[] theta = new List<string>[14];

            //AF3 represents Fp1, AF4 represents Fp2, T7 represents A1 and T8
represents A2 in a regular EEG
            //An array that contains the names of each sensor
            String[] sensors = {
                                    "AF3",
                                    "F7",
                                    "F3",
                                    "FC5",
                                    "T7",
                                    "P7",
                                    "O1",
                                    "O2",
                                    "P8",
                                    "T8",
                                    "FC6",
                                    "F4",
                                    "F8",
                                    "AF4"
                                };


            //get alpha data
            for (int i = 0; i < sensors.Length; i++)
            {
                String temp = alphaPath + sensors[i] + ".txt";
                alpha[i] = File.ReadAllLines(temp).ToList<String>();
            }


            //get beta data
            for (int i = 0; i < sensors.Length; i++)
            {
                String temp = betaPath + sensors[i] + ".txt";
                beta[i] = File.ReadAllLines(temp).ToList<String>();
            }

            //get delta data
```

```
            for (int i = 0; i < sensors.Length; i++)
            {
                String temp = deltaPath + sensors[i] + ".txt";
                delta[i] = File.ReadAllLines(temp).ToList<String>();
            }

            //get theta data
            for (int i = 0; i < sensors.Length; i++)
            {
                String temp = thetaPath + sensors[i] + ".txt";
                theta[i] = File.ReadAllLines(temp).ToList<String>();
            }
            //get Gyro Data
            List<String> gyroX = File.ReadAllLines(gyroPathX).ToList<String>();
            List<String> gyroY = File.ReadAllLines(gyroPathY).ToList<String>();

            //add the first 16 lines which represent Engagement in a list to write
later
            for (int j = 0; j < 16; j++)
            {
                String Line = "";
                for (int i = 0; i < sensors.Length; i++)
                {
                    Line += alpha[i].First() + "," + beta[i].First() + "," +
delta[i].First() + "," + theta[i].First() + ",";

                    alpha[i].RemoveAt(0);
                    beta[i].RemoveAt(0);
                    theta[i].RemoveAt(0);
                    delta[i].RemoveAt(0);

                }

                Line += gyroX.First() + "," + gyroY.First() + ",";
                gyroY.RemoveAt(0);
                gyroX.RemoveAt(0);
                Line += "1";
                wekafile.Add(Line);


            }

            //add 10 lines which represent Excitement in a list to write later
            for (int j = 0; j < 10; j++)
            {
                String Line = "";
                for (int i = 0; i < sensors.Length; i++)
                {
                    Line += alpha[i].First() + "," + beta[i].First() + "," +
delta[i].First() + "," + theta[i].First() + ",";
                    alpha[i].RemoveAt(0);
                    beta[i].RemoveAt(0);
                    theta[i].RemoveAt(0);
                    delta[i].RemoveAt(0);

                }
                Line += gyroX.First() + "," + gyroY.First() + ",";
                gyroY.RemoveAt(0);
                gyroX.RemoveAt(0);
                Line += "2";
```

```
                wekafile.Add(Line);


            }


            //add the Last 13 lines which represent Frustration in a list to write
later
            for (int j = 0; j < 10; j++)
            {
                String Line = "";
                for (int i = 0; i < sensors.Length; i++)
                {
                    Line += alpha[i].First() + "," + beta[i].First() + "," +
delta[i].First() + "," + theta[i].First() + ",";
                    alpha[i].RemoveAt(0);
                    beta[i].RemoveAt(0);
                    theta[i].RemoveAt(0);
                    delta[i].RemoveAt(0);

                }
                Line += gyroX.First() + "," + gyroY.First() + ",";
                gyroY.RemoveAt(0);
                gyroX.RemoveAt(0);
                Line += "3";
                wekafile.Add(Line);


            }


            File.WriteAllLines(wekaPath, wekafile);
        }
```