ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΞΥΠΝΟ ΣΥΣΤΗΜΑ ΠΕΛΜΑΤΟΓΡΑΦΟΥ ΑΠΟΚΡΙΣΗΣ ΠΡΑΓΜΑΤΙΚΟΥ ΧΡΟΝΟΥ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σταματόπουλος Αλέξανδρος

Επιβλέπων: Ευάγγελος Χριστοφόρου

Καθηγητής Ε.Μ.Π.

Αθήνα, 2018

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΞΥΠΝΟ ΣΥΣΤΗΜΑ ΠΕΛΜΑΤΟΓΡΑΦΟΥ ΑΠΟΚΡΙΣΗΣ ΠΡΑΓΜΑΤΙΚΟΥ ΧΡΟΝΟΥ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σταματόπουλος Αλέξανδρος

Επιβλέπων: Ευάγγελος Χριστοφόρου

Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την Δευτέρα 4 Δεκεμβρίου 2018.

…….	…….	…….
Ευάγγελος Χριστοφόρου	Γεώργιος Ματσόπουλος	Παύλος-Πέτρος Σωτηριάδης
Καθηγητής Ε.Μ.Π	Αν. Καθηγητής Ε.Μ.Π	Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, 2018

.....

Σταματόπουλος Αλέξανδρος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών

Περίληψη

Ζούμε σε μια κοινωνία η οποία καθημερινά γίνεται ολοένα και περισσότερο απαιτητική, για ταχύτερα και καλύτερα ποιοτικά, αποτελέσματα. Ειδικότερα στους κλάδους της άθλησης και της ιατρικής, η ανάγκη συνδυασμού πληροφοριών για βελτίωση των αποτελεσμάτων και των επιδόσεών μας, αυξάνεται με ραγδαίους ρυθμούς. Ως αποτέλεσμα, κρίνεται απαραίτητη η είσοδος έξυπνων τεχνολογιών που θα εκμηδενίζουν τους χρόνους απόκρισης αλλά και την ποιότητα των αποτελεσμάτων της εκάστοτε εφαρμογής. Μια τέτοια εφαρμογή, η οποία εμφανίζεται καθημερινά είτε στον επαγγελματικό κλάδο άθλησης, είτε σε ερασιτέχνες αθλητές αλλά και σε πληθώρα ιατρικών προβλημάτων, είναι το πελματογράφημα. Η διαδικασία, δηλαδή, ανάλυσης της πίεσης που ασκείται σε συγκεκριμένα σημεία του πέλματος κατά τη διάρκεια βάδισης και ηρεμίας. Ένα ζήτημα που παραμένει ελάχιστα εξελιγμένο, παρόλο των πολλών και εξαιρετικά σημαντικών συμπερασμάτων που προσφέρει. Αυτή ήταν και η έμπνευση για την παρούσα διπλωματική. Η σχεδίαση, δηλαδή, μιας νέας πρωτοποριακής λύσης για το πελματογράφημα, χρησιμοποιώντας έξυπνα και αποτελεσματικά μια νέα τεχνολογία για τη σημαντική βελτίωση της εν λόγω διαδικασίας.

Επομένως στη διπλωματική εργασία που ακολουθεί, έγινε αρχικά μια συνολική μελέτη της ελληνικής και διεθνούς βιβλιογραφίας που ασχολείται με το πελματογράφημα σε ιατρικό και αθλητικό επίπεδο με εμβάθυνση στους πληρέστερους τρόπους ανάλυσης της κατανομής της πελματιαίας πίεσης αλλά και στα ορθότερα πιεζοηλεκτρικά υλικά για την ανάλυση αυτή. Αντίστοιχη έρευνα έγινε και για την επιλογή του κατάλληλου μικροελεγκτή, που θα συνδύαζε μικρό μέγεθος, ταχύτητα και απλότητα στη λειτουργία του. Έπειτα, με την τελική επιλογή του πιεζοηλεκτρικού αισθητήρα και του μικροελεγκτή, σχεδιάστηκε ένα απλό κύκλωμα, για την συλλογή και καταγραφή των μετρήσεων πίεσης, σε πραγματικό χρόνο, με τρόπο κατάλληλο για δυνατότητα επεξεργασίας τους, από τον προαναφερθέντα μικροελεγκτή. Παρακάτω, η διπλωματική αυτή εργασία ασχολείται και με τον κώδικα ο οποίος γράφτηκε για την ανάλυση και επεξεργασία των δεδομένων πίεσης.

Ως επίλογο, ο αναγνώστης μπορεί να βρει τα συμπεράσματα, τις προτεινόμενες εφαρμογές του πελματογράφου, καθώς και τις προτάσεις για μελλοντική έρευνα.

Λέξεις κλειδιά: Πιεζοηλεκτρισμός, Πελματογράφος, Πίεση, Arduino Uno, Processing

Abstract

We live in a society that is becoming more and more demanding for faster and better results. Particularly in the fields of sports and medicine, the need to combine data and use them for specific applications is increasing rapidly. As a result, intelligent technology is needed to eliminate response times and improve the quality of the results of each application. Of the thousands of such applications, one that is needed daily either in the professional sports industry or in amateur athletes but also in a variety of medical problems, is pelmatography, which is the process of analyzing the pressure in certain points caused by the tread, during various situations. An issue that remains very sophisticated despite its importance.

This was the inspiration of the present work. A new groundbreaking solution to pelmatography, by using intelligently and effectively new technologies to significantly improve the process in time and quality.

Therefore, in this thesis, a comprehensive study of the Greek and international literature dealing with the medical and athletic pelmatography has been carried out, by studying the best methods of analyzing the distribution of body pressure in the tread in different situations, during running, walking, resting. Moreover, the best piezoelectric materials for the analysis, were studied. Corresponding research was also held, to select the appropriate open source device (microcontroller) that would combine small size, speed and simplicity in its operation. Having chosen the piezoelectric sensor and the microcontroller, a simple circuit was designed to acquire and record pressure measurements in real time and in terms that can be properly interpreted by the microcontroller. Afterwards, the thesis deals with the code written for analyzing the pressure data from the selected microcontroller. Additionally, the algorithm and the corresponding real-time data visualization code which were designed to evaluate them, are explained.

Finally, the reader can find the conclusions about the applications of this work in the fields of medicine and sports, as well as the suggestions for future research.

Keywords: Piezoelectrism, Pelmatogram, Pressure, Arduino Uno, Processing

Πρόλογος

Η παρούσα διπλωματική με θέμα "ΕΞΥΠΝΟ ΣΥΣΤΗΜΑ ΠΕΛΜΑΤΟΓΡΑΦΟΥ ΑΠΟΚΡΙΣΗΣ ΠΡΑΓΜΑΤΙΚΟΥ ΧΡΟΝΟΥ", δημιουργήθηκε με την καθοριστική συμβολή του υπευθύνου Καθηγητή Χριστοφόρου Ευάγγελου, τον οποίο και οφείλω να ευχαριστήσω θερμά για την εμπιστοσύνη που έδειξε καθ' όλη τη διάρκεια της εκπόνησης της εν λόγω εργασίας, καθώς και για την καθοδήγηση και τον χρόνο που μου παρείχε από την αρχή έως το τέλος. Η εκπόνηση της εν λόγω εργασίας θα ήταν ακόμα αδύνατον να προχωρήσει δίχως την συμβολή του υποψήφιου διδάκτορα Αγγελόπουλου Σπύρου, τον οποίο και θέλω να ευχαριστήσω επίσης για τον χρόνο και κόπο τον οποίο αφιέρωσε και αυτός, στην διεκπεραίωση της εν λόγω πτυχιακής. Ο σκοπός της διπλωματικής μου, ήταν να εκσυγχρονίσει και απλοποιήσει την πελματογράφηση με ταυτόχρονη βελτίωση ποιότητας και ταχύτητας απόκρισης των αποτελεσμάτων. Εύχομαι η δουλειά μου και τα αποτελέσματά της να συμβάλουν στον σκοπό αυτόν.

Πίνακας Περιεχομένων

Κεφάλαιο 1. Εισαγωγή

1. Εισαγωγή

Με τον όρο πελματογράφημα αναφερόμαστε σε μια ιατρική εξέταση για την ανατομική μορφολογία του πέλματος καθώς και για το μέγεθος και την κατανομή των δυνάμεων πίεσης σε συγκεκριμένα σημεία του πέλματος κατά την διάρκεια βάδισης και στήριξης. Ο πελματογράφος, το μηχάνημα που πραγματοποιεί τη μέτρηση αυτή, αποτελείται από δύο μέρη: Από μια επιφάνεια εξοπλισμένη με μεγάλο αριθμό αισθητήρων πίεσης και ένα εξειδικευμένο πρόγραμμα εφαρμογής, που μέσω ηλεκτρονικού υπολογιστή συλλέγει, επεξεργάζεται και αναλύει τις δυνάμεις πίεσης που ασκούνται από το πέλμα πάνω στην επιφάνεια σε κατάσταση στήριξης (στατική μέτρηση) ή βάδισης (δυναμική μέτρηση).

Γιατί όμως είναι σημαντικό το πελματογράφημα;

Το πελματογράφημα «αποκαλύπτει» κατά βάση μυοσκελετικά προβλήματα αλλά και προβλήματα στον τρόπο βάδισης και κατανομής της πίεσης που ασκείται στο πέλμα κατά τις δραστηριότητες αυτές και συνεπώς βρίσκει τεράστιες εφαρμογές στους κλάδους της ιατρικής και της άθλησης.

Οι δύο τρόποι μέτρησης του πελματογραφήματος προσφέρονται τόσο ως διαγνωστικό εργαλείο για προβλήματα στην ανατομία των αστραγάλων, οσφυαλγία, ισχιαλγία, σκολίωση κ.ο.κ., αλλά ακόμα σαν εργαλείο πρόληψης διάφορων ιατρικών προβλημάτων που εμφανίζουν συμπτώματα στο πέλμα (π.χ. πρώιμη ανίχνευση του διαβητικού ποδιού). Πρόκειται λοιπόν για μια ανώδυνη, ολιγόλεπτη εξέταση που προσφέρεται για πληθώρα ιατρικών εξετάσεων.

Ο δεύτερος σημαντικός παράγοντας που καθιστά εξαιρετικά σημαντικό το πελματογράφημα, είναι το ότι η ανάλυση των πιέσεων που δέχεται το πέλμα δυναμικά, σε κατάσταση βάδισης ή τρεξίματος, αποτελεί ένα τεράστιο εργαλείο στον τομέα της άθλησης. Παραπάνω από 90% των αθλημάτων περιλαμβάνουν συνεχή κίνηση στα πόδια και άρα εκτεταμένη πίεση στα πέλματα. Το πώς πρέπει να κατανέμεται η πίεση αυτή, αποτελεί συνεπώς ένα βασικό στοιχείο για τους επαγγελματίες αθλητές τόσο για αποφυγή τραυματισμών όσο και για την σημαντική βελτίωση της απόδοσής τους.

Φυσικά, άθληση δεν σημαίνει απαραίτητα μόνο επαγγελματικός αθλητισμός αλλά και ερασιτεχνικός, και αυτός είναι ίσως και ο κυριότερος παράγοντας που καθιστά εξαιρετικά σημαντικό το πελματογράφημα. Ως εξέταση, αφορά τεράστιο αριθμό ανθρώπων που γυμνάζονται καθημερινά και θέλουν να έχουν την δυνατότητα να παρακολουθούν την εξέλιξή τους αθλητικά, φροντίζοντας παράλληλα να μην τραυματιστούν από την καθημερινή άσκηση.

Εδώ λοιπόν εμφανίζονται και τα προβλήματα του πελματογραφήματος, όπως αυτό πραγματοποιείται μέχρι και σήμερα. Ο προαναφερθείς εξοπλισμός είναι εξαιρετικά ακριβής και προσφέρει πληθώρα αποτελεσμάτων, πρόκειται όμως για ιδιαίτερα ακριβό εξοπλισμό και λόγω αυτού υπάρχει σε ορισμένα μόνο ιδιωτικά ιατρεία και εργαστήρια φυσικοθεραπείας. Ένα ακόμη σημαντικό πρόβλημα, είναι πως καθώς πρόκειται για μια ακριβή ιατρική εξέταση, υπάρχει ελάχιστη ιατρική ασφαλιστική κάλυψη για αυτή και άρα καθίσταται αρκετά μη προσβάσιμη στη πλειοψηφία του κόσμου ειδικά στη σημερινή ελληνική κοινωνία. Ειδικά για επαγγελματίες αθλητές αλλά και ερασιτέχνες, που θέλουν καθημερινά να παρακολουθούν το πελματογράφημά τους, κάτι τέτοιο καθίσταται εξαιρετικά δύσκολο καθώς είναι υπερβολικά χρονοβόρο, με συνεχείς ιατρικές επισκέψεις, και εξίσου πολύ κοστοβόρο. Ένα ακόμα πρόβλημα που προκύπτει, είναι πως ο εξοπλισμός, αλλά και ο χώρος στον οποίο πραγματοποιείται η εξέταση είναι περιορισμένος, με αποτέλεσμα να περιορίζεται και ο τρόπος με τον οποίο μπορεί να γίνει η εξέταση. Είναι αδύνατο παραδείγματος χάριν να εξετάσει κανείς την κατανομή και την συνισταμένη των δυνάμεων πίεσης στο πέλμα σε κατάσταση τρεξίματος, πόσω μάλλον πιο περίπλοκων κινήσεων, με τον υπάρχοντα εξοπλισμό.

1.1 Αντικείμενο της διπλωματικής

Στόχος της παρούσας διπλωματικής είναι η μελέτη, ανάλυση και σχεδίαση ενός νέου πειραματικού πρωτότυπου πελματογράφου, ο οποίος μπορεί να αντικαταστήσει τον έως τώρα χρησιμοποιούμενο εξοπλισμό. Θα παρέχει εξίσου ακριβή αποτελέσματα αλλά ταχύτερα, ευκολότερα και κυρίως εύκολα προσβάσιμα ανά πάσα στιγμή από τον εξεταζόμενο. Μια διάταξη δηλαδή που θα λύνει όλα τα προβλήματα που αντιμετωπίζει ο υπάρχων εξοπλισμός.

Αρχικά γίνεται μια λεπτομερής ανάλυση και έρευνα όλων των στοιχείων που χρειάστηκαν για τη σχεδίαση της διάταξης αυτής, τόσο σε επίπεδο υλικού όσο και σε επίπεδο λογισμικού. Έπειτα, γίνεται και η απαραίτητη μελέτη όλων των ιατρικών

δεδομένων που πρέπει να ληφθούν υπόψιν για την σωστή σχεδίαση της διάταξης. Έρευνες που πραγματοποιήθηκαν, με χρήση της ελληνικής και διεθνούς βιβλιογραφίας πάνω στα προαναφερθέντα ζητήματα και φυσικά του διαδικτύου.

Παράλληλα, θα πραγματοποιηθεί μια σύγκριση και αιτιολόγηση της επιλογής των στοιχείων της διάταξης σε σχέση με άλλες δυνατές επιλογές.

Παρακάτω, ασχολούμαστε με μια εκτενή ανάλυση και επεξήγηση της κυκλωματικής διάταξης που σχεδιάστηκε εξηγώντας τα πλεονεκτήματά της έναντι του υπάρχοντος εξοπλισμού. Έπειτα, αντίστοιχη ανάλυση γίνεται και για τα προγράμματα τα οποία χρησιμοποιήθηκαν για την οπτικοποίηση και επεξεργασία των δεδομένων του πελματογραφήματος και φυσικά τον αντίστοιχο κώδικα που γράφτηκε για την λειτουργία των εν λόγω προγραμμάτων. Θα δοθούν συγκεκριμένα στοιχεία του κώδικα που θα υποδεικνύουν τις κύριες λειτουργίες της διάταξης και πώς αυτές λειτουργούν.

Με βάση αυτήν την ανάλυση και μελέτη που έγινε, πραγματοποιήθηκε και το πειραματικό-κατασκευαστικό κομμάτι της πτυχιακής. Η σχεδίαση δηλαδή της κυκλωματικής διάταξης και η συγγραφή του αντίστοιχου κώδικα. Με τη διάταξη αυτή πραγματοποιήθηκε μια σειρά από μετρήσεις προσομοιώνοντας τις διαφορετικές καταστάσεις στις οποίες θέλουμε να μπορεί να μετράει ο πελματογράφος.

Αργότερα, η συγκεκριμένη εργασία πραγματεύεται και αναλύει τις βασικότερες προτάσεις, για εφαρμογή και εκμετάλλευση της πειραματικής διάταξης βασικά σε δύο κλάδους: της ιατρικής και της άθλησης.

Μια σημαντική σημείωση είναι πως στη συγκεκριμένη πτυχιακή, προσφέρεται ένα πρωτότυπο μοντέλο της διάταξης. Συνιστάται λοιπόν και είναι απαραίτητη περαιτέρω έρευνα στην ανάπτυξη και βελτίωση του μοντέλου, αντικείμενο που αναλύεται εκτενέστερα στο τελικό κομμάτι της πτυχιακής αυτής εργασίας.

1.2 Στόχοι

Η διπλωματική αυτή δημιουργήθηκε ώστε να δοθεί μια συνολική εισαγωγή, ανάλυση και μελέτη του πελματογραφήματος. Θα εξεταστούν το σύνολο των μειονεκτημάτων που αντιμετωπίζει το πελματογράφημα όπως πραγματοποιείται σήμερα. Τελικός στόχος φυσικά, είναι η αντιμετώπιση αυτών των θεμάτων του υπάρχοντος εξοπλισμού της εξέτασης του πελματογραφήματος, με τη σχεδίαση νέου εξοπλισμού.

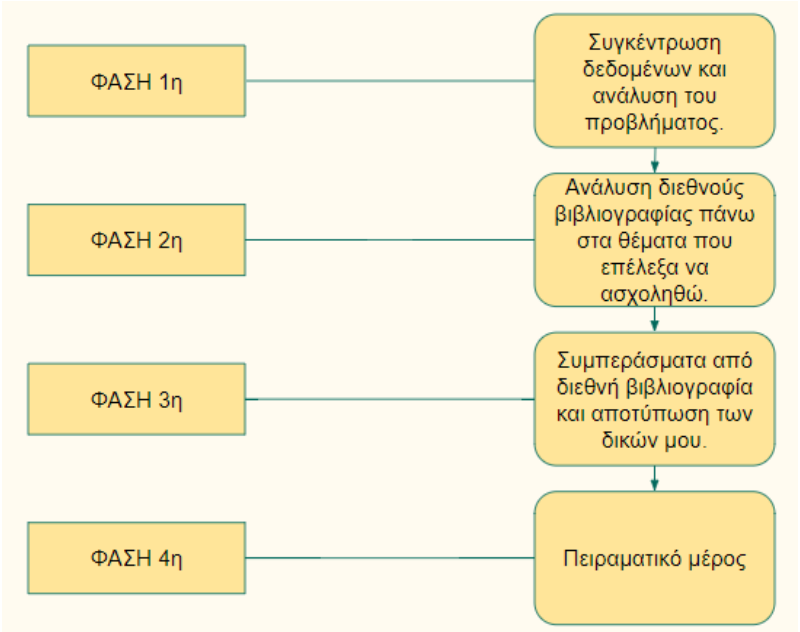
Συνεπώς, ο πρώτος και κύριος στόχος είναι η σχεδίαση μιας πρωτότυπης πειραματικής διάταξης που θα αντιμετωπίζει και θα λύνει τα ζητήματα αυτά (κόστος προσβασιμότητα, περιορισμένες δυνατότητες καταστάσεων εξέτασης κ.ο.κ.) και θα αντικαταστήσει μελλοντικά τον προαναφερθέντα εξοπλισμό.

Ο δεύτερος στόχος αφορά στην παρουσίαση των σημαντικότερων στοιχείων σε υλικό, λογισμικό αλλά και λογική σχεδίασης, που είναι απαραίτητα για την δημιουργία μιας τέτοιας διάταξης και σύγκρισης αυτών των διαφορετικών στοιχείων με άλλα πιθανά στοιχεία, ώστε η διπλωματική αυτή να μπορεί να αποτελεί οδηγό για τη δημιουργία παρόμοιων διατάξεων.

Ο τρίτος στόχος έγκειται στην δημιουργία μιας στέρεας τεχνολογικής βάσης, πάνω στην οποία το πρωτότυπο μοντέλο το οποίο αναπτύσσει η συγκεκριμένη πτυχιακή θα εξελιχθεί από μελλοντικούς ερευνητές σε μια καλύτερη, βελτιωμένη διάταξη.

1.3 Φάσεις υλοποίησης

Η εκπόνηση της διπλωματικής εργασίας πραγματοποιήθηκε μεταξύ Φεβρουαρίου και Οκτωβρίου 2018 και η πορεία αυτής ακολούθησε τις εξής φάσεις, που παρουσιάζονται παρακάτω στο Σχήμα 1.1.



Σχήμα 1.1 Φάσεις υλοποίησης διπλωματικής εργασίας

Η μεθοδολογία που ακολουθήθηκε σε όλα τα στάδια υλοποίησης είναι η εξής:

- Συγκέντρωση πληροφοριών από την ελληνική και διεθνή βιβλιογραφία και από το διαδίκτυο.

- Ανάλυση και σύγκριση όλων των πληροφοριών που επελέγησαν ως χρήσιμες για την διπλωματική.

- Αποτύπωση των αποτελεσμάτων και σύγκριση με τη βιβλιογραφία.

Αυτή η μεθοδολογία αποτυπώνεται στην κατάλληλη επιλογή όλων των υλικών της διάταξης και των προγραμμάτων οπτικοποίησης και αξιολόγησης των δεδομένων, αλλά και στην ανάλυση της μορφολογίας του ανθρώπινου πέλματος για την σωστή επιλογή σημείων μέτρησης της πελματιαίας πίεσης.

Για κάθε έναν από τους 4 στόχους, εφαρμόστηκε η παραπάνω μεθοδολογία με τη συγκεκριμένη σειρά, μετά τη γενική ανάλυση του προβλήματος και τη συγκέντρωση της πληροφορίας.

1.4 Οργάνωση εργασίας

Σχετικά με την οργάνωση της παρούσας διπλωματικής, παρακάτω βρίσκεται η οργάνωση της εργασίας ανά κεφάλαιο.

Το 1ο κεφάλαιο αποτελεί μια εισαγωγή της διπλωματικής εργασίας. Ο αναγνώστης μπορεί να σχηματίσει μια γενική εικόνα της εργασίας αλλά και των λόγων για τους οποίους εκπονήθηκε.

Το 2ο κεφάλαιο περιέχει την απαραίτητη γενική θεωρία για την πλήρη κατανόηση ολόκληρης της εργασίας. Στο κεφάλαιο αυτό θα δικαιολογηθεί και η επιλογή όλων των στοιχείων που χρησιμοποιήθηκαν στη σχεδίαση της διάταξης.

Το 3ο κεφάλαιο περιέχει την αναπτυχθείσα διάταξη της συγκεκριμένης εργασίας, εξηγώντας σε βάθος τόσο το υλικό της διάταξης και τη λειτουργία του, όσο και το λογισμικό το οποίο δημιουργήθηκε και χρησιμοποιήθηκε για τις ανάγκες της εργασίας.

Το 4ο κεφάλαιο ασχολείται με προτεινόμενες εφαρμογές της πρωτότυπης διάταξης αυτής, σε δύο βασικούς κλάδους, αυτούς της άθλησης και της ιατρικής.

Τέλος, στο 5^{ο} κεφάλαιο παρουσιάζονται συνοπτικά τα ευρήματα και τα συμπεράσματα από την υλοποίηση της εν λόγω εργασίας.

Κεφάλαιο 2. Γενική Θεωρία

2. Γενική Θεωρία

2.1 Συστήματα πελματογράφων

Παρακάτω θα παρουσιαστούν διαφορετικά συστήματα πελματογράφων. Τόσο ο εξοπλισμός που χρησιμοποιείται ευρέως, όσο και πειραματικές διατάξεις, συμπεριλαμβανομένης της διάταξης που πραγματεύεται η συγκεκριμένη πτυχιακή εργασία. Θα αναλυθούν οι ιδιότητες αυτών και τέλος μια σύγκριση των διαφορετικών αυτών συστημάτων, παρουσιάζοντας τα πλεονεκτήματα και τα μειονεκτήματά τους.

2.1.1 Είδη συστημάτων πελματογράφου

2.1.1.1 Βασικό σύστημα πελματογράφου

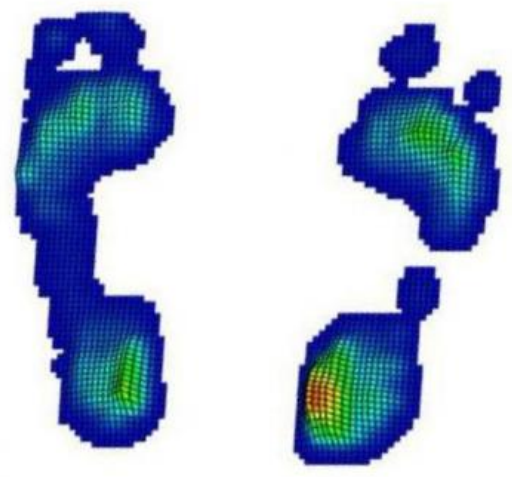
Ο πελματογράφος είναι ένα σύστημα καταγραφής των πιέσεων που ασκούνται στο πέλμα κατά τη στήριξη ή τη βάδιση. Ανάλογα με το σκοπό και το είδος της κίνησης που εξετάζεται, το πελματογράφημα που προκύπτει από το βασικό σύστημα του πελματογράφου μπορεί να ονομάζεται στατικό ή δυναμικό. Το στατικό πελματογράφημα αναφέρεται, κυρίως, στην ανάλυση των πελματιαίων πιέσεων κατά την όρθια στάση. Το δυναμικό πελματογράφημα αναφέρεται στην ανάλυση πελματιαίων πιέσεων κατά τη βάδιση. Δίνεται ακόμα η δυνατότητα ο πελματογράφος να χρησιμοποιηθεί και σαν πλατφόρμα αξιολόγησης της ισορροπίας σε κάθε πέλμα ξεχωριστά, σε σχέση με τις πιέσεις που δέχονται επιμέρους τμήματα του πέλματος κατά τη διάρκεια της εξέτασης.

Το βασικό σύστημα πελματογράφου αποτελείται από μια πλατφόρμα πάνω στην οποία ο εξεταζόμενος καλείται να σταθεί σε όρθια στάση. Η πλατφόρμα αυτή αποτελείται εσωτερικά από 3000-4000 αισθητήρες πίεσης, διαστάσεων 5mm x 7mm ο καθένας, ανάλογα με το μοντέλο του πελματογράφου. Καθένας από τους αισθητήρες αυτούς αντιστοιχεί σε μια έξοδο με δύο άκρα, ένα εκ των οποίων γειώνεται και το άλλο οδηγείται μέσω καλωδίου σε έναν υπολογιστή. Η πίεση που ασκείται σε κάθε αισθητήρα αντιστοιχεί σε ένα επίπεδο τάσης, το οποίο τελικά, επεξεργάζεται μέσω

ειδικού λογισμικού και προσφέρει μια λεπτομερή ανάλυση των πελματικών πιέσεων. Οι βασικές μεταβλητές που αξιολογούνται από το λογισμικό είναι η μέγιστη και μέση πίεση στο πέλμα, ο χρόνος επαφής, η μέγιστη κατακόρυφη δύναμη αντίδρασης, το μέγεθος πίεσης ανά περιοχή πέλματος και η μέγιστη και μέση ώθηση. Με την ανάλυση των μεταβλητών αυτών, σε στατική και δυναμική πίεση, το ειδικό λογισμικό του υπολογιστή θα καταγράψει τελικά με ακρίβεια την κατανομή δυνάμεων που ασκούνται σημειακά στα πέλματα [1], [2].

Το λογισμικό το οποίο χρησιμοποιείται, κατά βάση, είναι το Milletrix. Το λογισμικό αυτό χρησιμοποιεί πολύπλοκες μεθόδους επεξεργασίας δεδομένων για την τελική ψηφιακή απεικόνιση των πελματικών πιέσεων. Εν προκειμένω, χρησιμοποιεί μια χρωμομετρική κλίμακα με δέκα επίπεδα χρώματος, τα οποία αντιστοιχούν κάθε ένα σε διαφορετικό επίπεδο τάσης η κάθε μια και κατά συνέπεια σε διαφορετικές τιμές πιέσεων, που προέρχονται από τους αισθητήρες της πλατφόρμας. Έπειτα, ανάλογα με το είδος της μέτρησης, στατικό ή δυναμικό, το λογισμικό παρουσιάζει μια πλήρη ανάλυση των πελματικών πιέσεων και εμμέσως μέσω των μετρήσεων αυτών, τις κατακόρυφες δυνάμεις αντίδρασης και το κέντρο πίεσης.

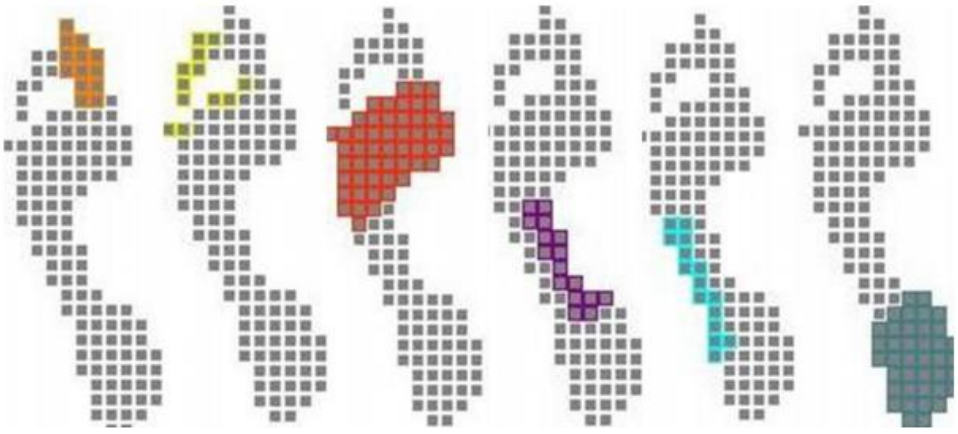
Όπως προαναφέρθηκε, το βασικό σύστημα πελματογράφου έχει δύο, κατά βάση, διαφορετικά είδη μέτρησης, το στατικό και το δυναμικό.



Σχήμα 2.1. Τρισδιάστατη απεικόνιση των πιέσεων που ασκούνται στο πέλμα στην όρθια στάση

Στη στατική μέτρηση, γίνεται ανάλυση των πελματικών πιέσεων κατά την εκτέλεση της όρθιας στάσης. Ένα τυπικό πρωτόκολλο περιλαμβάνει την καταγραφή των πιέσεων, με άνοιγμα των ποδιών στο ύψος των ώμων, και διάρκεια εξέτασης που κυμαίνεται από 10 έως 30 δευτερόλεπτα. Ο πελματογράφος στη στατική μέτρηση, καταγράφει τις πιέσεις που δέχονται οι αισθητήρες του από το πέλμα, συνήθως ανά τετραγωνικό εκατοστό (Σχήμα 2.1). Στη συνέχεια η επιφάνεια του πέλματος διαχωρίζεται σε διάφορες περιοχές και η πίεση αναλύεται ανά περιοχή πέλματος. Υπάρχουν διάφορα μοντέλα διαχωρισμού της πελματιαίας επιφάνειας, αλλά το βασικότερο μοντέλο είναι αυτό των παρακάτω έξι περιοχών (Σχήμα 2.2):

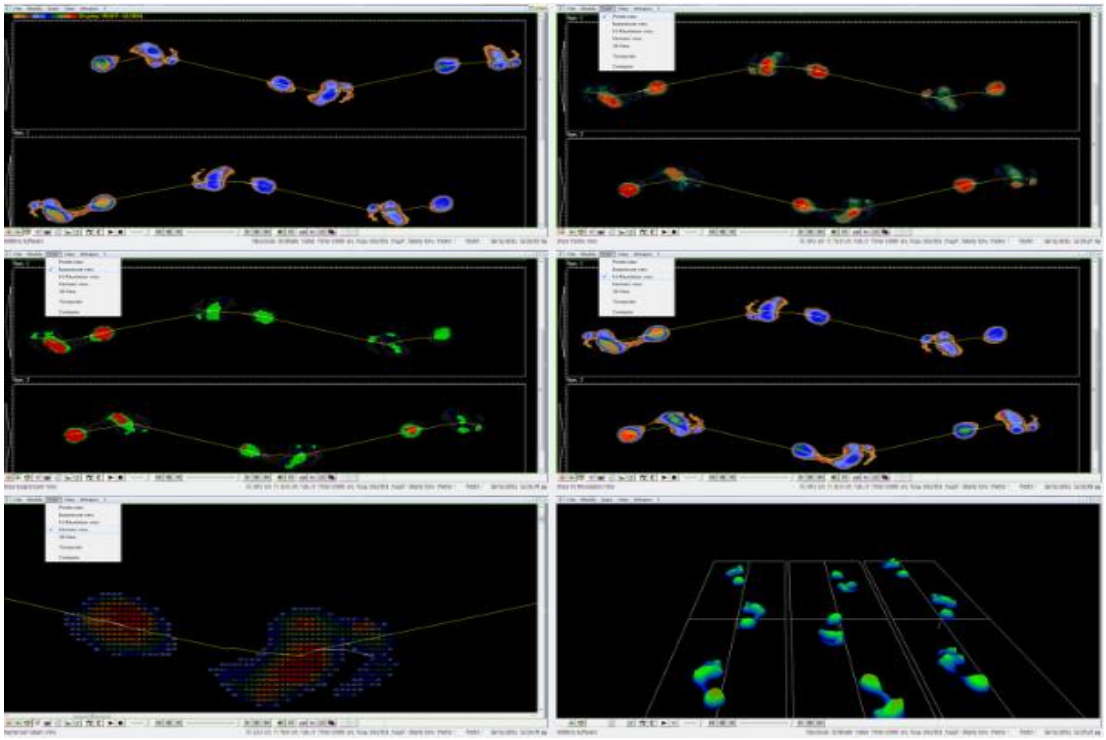
- Μεταταρσιο-φαλαγγική άρθρωση του μεγάλου δακτυλίου
- Δεύτερη έως τέταρτη μεταρσιο-φαλαγγική άρθρωση
- Μετατάρσια
- Έξω καμάρα
- Έσω καμάρα
- Πτέρνα



Σχήμα 2.2. Διάκριση του πέλματος σε έξι ανατομικές περιοχές

Στη δυναμική μέτρηση, γίνεται ανάλυση των πιέσεων που ασκούνται στην πελματιαία επιφάνεια, κατά την διάρκεια της βάδισης. Στο είδος μέτρησης αυτό το πρωτόκολλο που ακολουθείται για το βασικό σύστημα του πελματογράφου απαιτεί από τέσσερις έως έξι μετρήσεις (Σχήμα 2.3), κάθε μια από τις οποίες πρέπει να περιλαμβάνει και μέτρηση μισού βήματος. Αυτό οφείλεται στο γεγονός ότι με το μισό βήμα το σύστημα

πελματογράφου είναι ικανό να αναλύσει πελματικές πιέσεις σε ανατομικά σημεία και οστά, στα οποία είναι αδύνατον να αναλυθούν οι πιέσεις αυτές με ολόκληρο βήμα. Ύστερα, τα δεδομένα των βαδίσεων από το σύνολο των διαφορετικών αυτών μετρήσεων συγκρίνονται από το λογισμικό, με την τελική ψηφιακή απεικόνιση να αποτελείται από τον μέσο όρο των μετρήσεων. Η διαδικασία ανάλυσης περιλαμβάνει τον υπολογισμό μεταβλητών τόσο σε επίπεδο πέλματος, όσο και σε επίπεδο περιοχής πέλματος.



Σχήμα 2.3. Σύνολο καταγραφής μετρήσεων κατά τη βάδιση

2.1.1.2 Πειραματικές διατάξεις

2.1.1.2.1 Δυναμοδάπεδο

Το δυναμοδάπεδο έχει σχήμα ορθογώνιου παραλληλόγραμμου (Σχήμα 2.4). Εξωτερικά αποτελείται από δύο μεταλλικές πλάκες, ενώ εσωτερικά αποτελείται από ένα κύκλωμα αισθητήρων δύναμης. Οι αισθητήρες αυτοί μπορεί να είναι αισθητήρες πιεζοαντίστασης ή πιεζοηλεκτρικοί κρύσταλλοι. Με την εφαρμογή μιας δύναμης πάνω στην πλατφόρμα προκαλείται μεταβολή του σχήματος (στην περίπτωση του αισθητήρα πιεζοαντίστασης) ή του ηλεκτρικού φορτίου (στην περίπτωση των πιεζοηλεκτρικών κρυστάλλων), η οποία καταγράφεται ως αλλαγή της τάσης του

ρεύματος και η οποία είναι ανάλογη του μεγέθους της εξωτερικά εφαρμοζόμενης δύναμης [3], [4].

Τα δυναμοδάπεδα με πιεζοηλεκτρικές αντιστάσεις χρησιμεύουν για την καταγραφή δυνάμεων κατά την εκτέλεση αργών κινήσεων, αλλά είναι λιγότερο ευαίσθητα σε γρήγορες κινήσεις. Αντίθετα, τα δυναμοδάπεδα με πιεζοηλεκτρικούς κρυστάλλους είναι πιο ευαίσθητα και, επομένως, καταλληλότερα για την καταγραφή δυνάμεων κατά την εκτέλεση γρήγορων κινήσεων. Ωστόσο υστερούν στις καταγραφές δυνάμεων για παρατεταμένο χρονικό διάστημα. Γενικά και τα δύο είδη δυναμοδαπέδων υπόκεινται σε βαθμονόμηση πριν από την εγκατάστασή τους.

Σχήμα 2.4. Το δυναμοδάπεδο

Η πιο απλή μορφή ενός δυναμοδαπέδου είναι το μονοαξονικό δυναμοδάπεδο. Πρόκειται για μια πλατφόρμα που διαθέτει έναν αισθητήρα, ο οποίος τοποθετείται στο κέντρο της και μπορεί να καταγράψει μόνο την κατακόρυφη δύναμη αντίδρασης του εδάφους. Για το λόγο αυτό η χρήση της συγκεκριμένης πλατφόρμας περιορίζεται, κυρίως, στην αξιολόγηση της κατακόρυφης αλτικής ικανότητας.

Τα σύγχρονα δυναμοδάπεδα διαθέτουν τέσσερις αισθητήρες, οι οποίοι βρίσκονται στις τέσσερις γωνίες της πλατφόρμας. Όταν ένα σώμα έρθει σε επαφή με την πλατφόρμα, τότε ο κάθε αισθητήρας καταγράφει την παραμόρφωση μέσω αλλαγής της τάσης του. Ο συνδυασμός των τάσεων από όλους τους αισθητήρες επιτρέπει την καταγραφή οκτώ μεταβλητών. Εν προκειμένω, την εφαρμοζόμενη δύναμη

αναλυόμενη στους τρείς καρτεσιανούς άξονες, ταυτοχρόνως υπολογίζοντας και τις αντίστοιχες ροπές των δυνάμεων αυτών και τις συντεταγμένες του κέντρου πίεσης, μέσω ενσωματωμένου συστήματος συντεταγμένων για τον ακριβή υπολογισμό της θέσης εφαρμογής, οποιασδήποτε δύναμης εφαρμόζεται πάνω στη πλατφόρμα. Η πλατφόρμα διαθέτει ένα σύστημα συντεταγμένων το οποίο χρησιμοποιείται για να περιγράψει την ακριβή θέση εφαρμογής οποιασδήποτε δύναμης που εφαρμόζεται επάνω της. Με βάση αυτά τα στοιχεία είναι εφικτός ο υπολογισμός της επιτάχυνσης τον κέντρου μάζας, του έργου και της ισχύος κατά την εκτέλεση διάφορων αθλητικών κινήσεων. Όταν συνδυαστούν τα δεδομένα της πλατφόρμας με τα κινηματικά χαρακτηριστικά της κίνησης και τα ανθρωπομετρικά χαρακτηριστικά του εξεταζόμενου, τότε είναι εφικτός ο υπολογισμός των ροπών που ασκούνται γύρω από τις αρθρώσεις με τη χρήση της τεχνικής της αντίστροφης δυναμικής.

Το δυναμοδάπεδο προσφέρει περισσότερες επιλογές είδους μετρήσεων από τον πελματογράφο. Συγκεκριμένα, προσφέρει δυνατότητα μέτρησης όρθιας στάσης, επιτόπιου άλματος και βάδισης. Η ανάλυση της βάδισης συνίσταται στην καταγραφή των δυνάμεων αντίδρασης καθώς το άτομο εκτελεί ένα διασκελισμό και έρχεται σε επαφή με την πλατφόρμα. Εάν χρησιμοποιείται ένα δυναμοδάπεδο, τότε καταγράφεται η φάση στήριξης της βάδισης μόνο στο ένα πόδι. Η καταγραφή και ανάλυση των μετρήσεων ακολουθεί ένα πρωτόκολλο. Οριοθετείται ένας διάδρομος 10 έως 12 μέτρων κατά μήκος του οποίου πραγματοποιείται η βάδιση, εξασφαλίζοντας έτσι ότι η βάδιση του ατόμου πραγματοποιείται με φυσιολογικό τρόπο. Πριν από την έναρξη της μέτρησης γίνεται η καταγραφή του σωματικού βάρους ζητώντας από το άτομο να λάβει όρθια χαλαρή στάση πάνω στην πλατφόρμα. Η διαδικασία αυτή επιτρέπει την αυτόματη συσχέτιση των καταγεγραμμένων δυνάμεων ως προς το σωματικό βάρος. Πραγματοποιείται το κυρίως πρωτόκολλο με προσπάθειες βάδισης με το ένα πέλμα. Αφού ελεγχθεί η καταγραφή των δυνάμεων από το σύστημα, επαναλαμβάνεται η βάδιση με πάτημα στο άλλο πέλμα. Συνιστάται ένας ελάχιστος αριθμός 7 έως 10 προσπαθειών, ώστε να μειωθεί η μεταβλητότητα των μετρήσεων. Σημαντικό κομμάτι της ανάλυσης είναι ο εξεταζόμενος να βαδίζει με τον δικό του ρυθμό, καθώς η καταγραφή των δυνάμεων πίεσης στην πελματιαία επιφάνεια αλλάζει σημαντικά αναλόγως με την ταχύτητα της βάδισης [5].

Στην μέτρηση επιτόπιου άλματος, αρχικά πριν από την έναρξη της καταγραφής γίνεται η μέτρηση του σωματικού βάρους του εξεταζόμενου, ζητώντας από το άτομο

να λάβει όρθια χαλαρή στάση πάνω στην πλατφόρμα. Ομοίως, η διαδικασία αυτή επιτρέπει την αυτόματη συσχέτιση των καταγεγραμμένων δυνάμεων ως προς το σωματικό βάρος. Εάν η πλατφόρμα μηδενιστεί μετά την καταγραφή του σωματικού βάρους, τότε είναι εφικτός και ο αυτόματος υπολογισμός του ύψους άλματος και της ταχύτητας ανύψωσης. Η προσγείωση κατά την εκτέλεση του στατικού άλματος πραγματοποιείται με τα πόδια τεντωμένα και χωρίς να υπάρχει οριζόντια μετατόπιση. Εκτελείται ένας αριθμός δοκιμαστικών προσπαθειών, ώστε το άτομο να εξοικειωθεί με την κίνηση. Κατόπιν γίνεται η λήψη της κανονικής προσπάθειας. Συστήνεται ένας ελάχιστος αριθμός 3 έως 5 προσπαθειών, ώστε να μειωθεί η μεταβλητότητα των μετρήσεων [6], [7].

Το λογισμικό που χρησιμοποιεί το δυναμοδάπεδο, βάσει των παρακάτω σχέσεων κάνει την ανάλυση στην μέτρηση επιτόπιου άλματος. Επειδή η κίνηση του σώματος κατά την εκτέλεση του κατακόρυφου άλματος πραγματοποιείται στον κατακόρυφο άξονα, οι δυνάμεις στους άλλους άξονες δεν εξετάζονται. Επομένως, η ανάλυση του άλματος βασίζεται στην καμπύλη της κατακόρυφης δύναμης-αντίδρασης και του χρόνου. Συγκεκριμένα, η κατακόρυφη δύναμη δίνεται από τη σχέση:

$$\boldsymbol{GRF_z = m * \alpha_{KM}} \qquad \textbf{Εξ.1}$$

όπου **m** = μάζα, $\boldsymbol{\alpha_{KM}}$ = κατακόρυφη επιτάχυνση του κέντρου μάζας.

Επειδή κατά την εκτέλεση του άλματος η τριβή θεωρείται αμελητέα, ενώ το σωματικό βάρος έχει ήδη καταγραφεί και αφαιρεθεί από τη μέτρηση, η εξίσωση είναι απλοποιημένη. Εφόσον το άτομο στην αφετηρία έχει μηδενική ταχύτητα, το ολοκλήρωμα της εξίσωσης ως προς το χρόνο επιτρέπει τον υπολογισμό της ταχύτητας ανύψωσης του κέντρου μάζας (**V_{KM}**) του σώματος ως εξής:

$$\int_0^t \boldsymbol{GRF_z dt = m * V_{KM}} \qquad \textbf{Εξ.2}$$

Στη συνέχεια υπολογίζονται το ύψος (H), η μηχανική ισχύς (P) και το μηχανικό έργο (W) ως εξής:

$$H = V_{KM}^2/2g \qquad \text{Εξ.3}$$

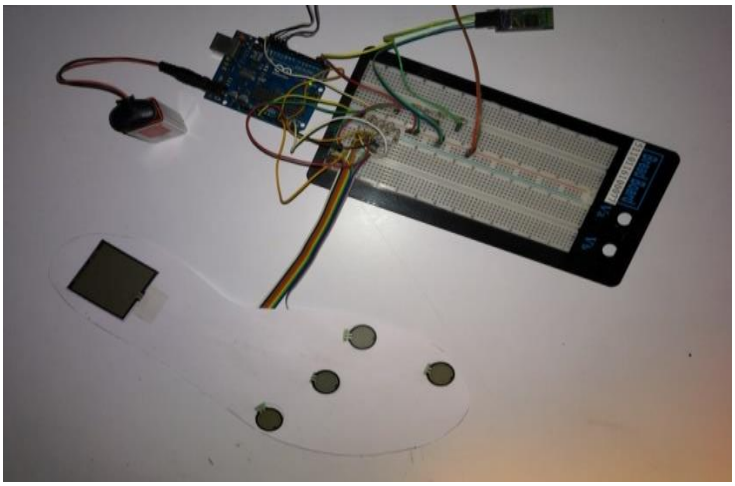
$$P = GRF_Z * V_{KM} \qquad \text{Εξ.4}$$

$$W = \int_0^T P \qquad \text{Εξ.5}$$

Τέλος, η ανάλυση ισορροπίας, μέτρηση δηλαδή κατά την όρθια στάση, κυρίως, συνίσταται στην καταγραφή της θέσης του κέντρου πίεσης στο σύστημα αναφοράς της πλατφόρμας. Οι βασικές μεταβλητές που αξιολογούνται είναι: Η μέγιστη μετατόπιση του κέντρου πίεσης στον οριζόντιο και τον κατακόρυφο άξονα, η οποία εκφράζει το μέγεθος της μετατόπισης του κέντρου πίεσης κατά τη διάρκεια μιας προσπάθειας (δηλαδή, το μέγεθος της ταλάντωσης τον σώματος), η τυπική απόκλιση της μετατόπισης του κέντρου πίεσης σε κάθε άξονα, η οποία εκφράζει τη μεταβλητότητα του κέντρου πίεσης κατά τη διάρκεια της προσπάθειας (δηλαδή, το πόσες ταλαντώσεις εκτελεί το σώμα), η ταχύτητα μετατόπισης του κέντρου πίεσης και τελικά, η επιφάνεια που προσδιορίζεται από τη συνολική γραμμή μετατόπισης του κέντρου πίεσης, η οποία εκφράζει (έμμεσα) τη συνολική συμπεριφορά του κέντρου πίεσης κατά τη διάρκεια της προσπάθειας [8], [9], [10].

2.1.1.2.2 Η προτεινόμενη διάταξη

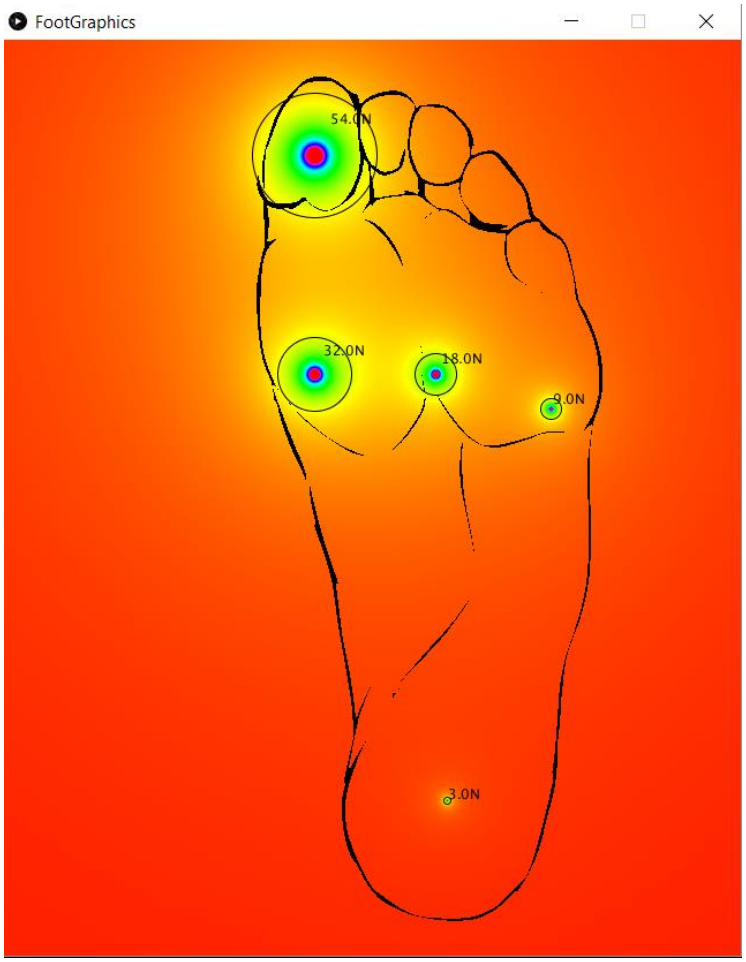
Στο σημείο αυτό είναι σημαντικό να τονιστεί πως το υλικό και το λογισμικό που σχεδιάστηκε για την εν λόγω πειραματική διάταξη θα αναλυθεί πλήρως στο Κεφάλαιο 3.



Σχήμα 2.5. Η κυκλωματική διάταξη της πτυχιακής εργασίας

Συνοπτικά λοιπόν, η πειραματική κυκλωματική διάταξη η οποία σχεδιάστηκε και υλοποιήθηκε (Σχήμα 2.5), αποτελείται από πέντε αισθητήρες δύναμης πίεσης (FSR),

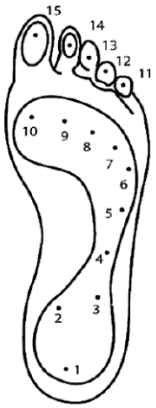
οι οποίοι οδηγούνται ο κάθε ένας, σε πέντε διαφορετικές αναλογικές εισόδους ενός μικροελεγκτή Atmega328p (Arduino Uno). Επίσης, χρησιμοποιήθηκε πρόσθετο επικοινωνίας βασισμένο στην τεχνολογία Bluetooth, το οποίο επιτρέπει την αποστολή της πληροφορίας σε υπολογιστή ή κινητό τηλέφωνο, με σκοπό την οπτικοποίηση των δεδομένων. Τέλος, χρησιμοποιήθηκε μια κάρτα μνήμης για αποθήκευση του συνόλου των δεδομένων. Ο μικροελεγκτής προγραμματίστηκε έτσι ώστε να αναγνωρίζει την πίεση που εφαρμόζεται σε κάθε έναν αισθητήρα, να την αντιστοιχεί σε μια κλίμακα 0 έως 100 Newton και να την οπτικοποιεί, σε μορφή χάρτη θερμότητας, πάνω στην εικόνα ενός πέλματος (Σχήμα 2.6). Η απόκριση της μέτρησης και οπτικοποίησης των δεδομένων είναι σε πραγματικό χρόνο.

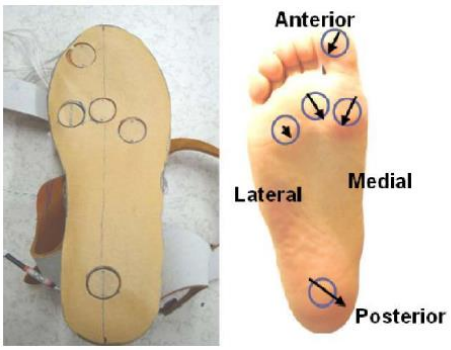


Σχήμα 2.6. Η οπτικοποίηση των πιέσεων στην πελματιαία επιφάνεια από τον κώδικα που σχεδιάστηκε για την διάταξη της πτυχιακής

Σχεδιάστηκε, ακόμα, λογισμικό το οποίο συγκεντρώνει το σύνολο των μετρήσεων από μια άσκηση και δίνει την δυνατότητα στον χρήστη να αξιολογήσει το σύνολο του πελματογραφήματος ή κομμάτια αυτού σε οποιαδήποτε χρονική στιγμή.

Σημαντικό κομμάτι της θεωρίας αποτελεί η μελέτη που έγινε για την επιλογή των συγκεκριμένων πέντε σημείων στο πέλμα για την τοποθέτηση των αισθητήρων. Αναλυτική μελέτη της ανατομικής μορφολογίας του πέλματος δείχνει ότι το σωματικό βάρος κάθε ανθρώπου μοιράζεται μεταξύ 15 περιοχών στο πέλμα, όπως φαίνεται και στο Σχήμα 2.7. Ιδανικά λοιπόν θα χρειαζόμασταν δεκαπέντε διαφορετικούς αισθητήρες σε κάθε μια από τις περιοχές αυτές. Βασικά όμως, πέντε είναι οι περιοχές οι οποίες στηρίζουν ολόκληρο το ανθρώπινο βάρος (Σχήμα 2.8) και χρησιμοποιήθηκαν στην ανάλυση της συγκεκριμένης διάταξης. Στις πέντε περιοχές αυτές, βρίσκονται τα οστά τα οποία χρησιμοποιούνται κατά βάση στο σύνολο των κινήσεων ενός αθλητή στο μεγαλύτερο ποσοστό αθλημάτων και χρησιμοποιούνται για την πλειοψηφία των ιατρικών διαγνώσεων σχετικών με τη πελματιαία πίεση [13], [14], [15], [16].





Σχήμα 2.7 Οι 15 περιοχές κατανομής της πίεσης

Σχήμα 2.8 Οι 5 περιοχές που επιλέχθηκαν

2.1.2 Σύγκριση συστημάτων πελματογραφήματος

Και τα τρία παραπάνω συστήματα έχουν τα δικά τους πλεονεκτήματα και αδυναμίες. Παραθέτουμε λοιπόν, σε αυτό το σημείο μια συνοπτική ανάλυση αυτών. Η ανάλυση θα καταλήξει και στον λόγο, για τον οποίο θεωρούμε, πως η πειραματική διάταξη που σχεδιάστηκε υπερτερεί των άλλων δύο, κάτι που οδηγεί στην αιτία πραγματοποίησής της.

Στο βασικό σύστημα του πελματογράφου λοιπόν, ένα από τα βασικά προτερήματά του είναι η πολύ μεγάλη ακρίβεια των αποτελεσμάτων. Η πλατφόρμα, όπως

προαναφέρθηκε, διαθέτει πολύ μεγάλο αριθμό αισθητήρων και συνεπώς είναι δυνατόν να δώσει αποτελέσματα με ακρίβεια τετραγωνικών χιλιοστών, στην πελματιαία επιφάνεια. Η διαδικασία του πελματογραφήματος, είναι εύκολη, γρήγορη και ανώδυνη γεγονός που προσφέρει τα αποτελέσματα ταχύτατα και με την προαναφερθείσα υψηλή ποιότητα. Επιπλέον, είναι ένα σύστημα το οποίο βρίσκεται πάρα πολλά χρόνια στην αγορά, και συνεπώς, σε επίπεδο υλικού αλλά και σε επίπεδο λογισμικού, αναπτύσσεται διαρκώς [11], [12].

Τα μειονεκτήματά του όμως, παραμένουν αρκετά. Πρόκειται, για πολύ ακριβό εξοπλισμό και άρα είναι διαθέσιμος σε συγκεκριμένα μόνο, ιατρικά και φυσικοθεραπευτικά κέντρα. Σαν συνέπεια αυτού, η ασφαλιστική κάλυψη για την εξέταση του πελματογραφήματος είναι σπάνια και ελάχιστα ασφαλιστικά ταμεία την προσφέρουν. Αυτό, καθιστά το πελματογράφημα προσβάσιμο από λίγους ή πιο σωστά, προσβάσιμο λίγες φορές μόνο τον χρόνο, στην πλειοψηφία του κόσμου.

Έπειτα, οι μετρήσεις που προκύπτουν από τον πελματογράφο είναι πολύ υψηλής ακρίβειας και ποιότητας αλλά περιορίζονται μόνο σε δύο διαφορετικά είδη, τα οποία και προαναφέρθηκαν. Στην πλειοψηφία των αθλημάτων, πραγματοποιούνται πελματιαίες πιέσεις που είναι αδύνατον να καταγραφούν και να αναλυθούν μόνο με την όρθια μέτρηση ή μέτρηση βάδισης. Ακόμα και για διαγνωστικούς σκοπούς, ο πελματογράφος αδυνατεί να καταγράψει πλάγιες και οριζόντιες δυνάμεις αντίδρασης, που είναι απαραίτητες για την διάγνωση συγκεκριμένων παθήσεων, όπως η ισχιαλγία [15].

Τέλος, οι μετρήσεις του συστήματος αυτού, γίνονται σε περιβάλλον εργαστηρίου όπου όλες οι συνθήκες είναι ελεγχόμενες. Αυτό, μπορεί να αποτελεί προτέρημα σε μια διαγνωστική εξέταση, σε περιπτώσεις όμως, που ένας αθλητής θέλει να χρησιμοποιήσει το πελματογράφημα για βελτίωση των αθλητικών του επιδόσεων, οι μετρήσεις είναι απαραίτητο και αναγκαίο να πραγματοποιηθούν υπό κανονικές συνθήκες προπόνησης [17].

Στο σύστημα του δυναμοδαπέδου, παρόμοια και με το σύστημα του πελματογράφου, έχουμε μια εξέταση υψηλής ακρίβειας, γρήγορη, εύκολη και ανώδυνη για τον εξεταζόμενο. Παρέχεται, μια πλήρης ανάλυση των δυνάμεων σε τρείς διαφορετικούς άξονες, ξεπερνώντας ουσιαστικά έτσι το πρόβλημα το οποίο αντιμετωπίζει το βασικό σύστημα του πελματογράφου, με τις πλάγιες και οριζόντιες δυνάμεις.

Δίνεται ακόμα, δυνατότητα για μεγαλύτερο εύρος μετρήσεων για τις ροπές των δυνάμεων που ασκούνται στην πελματιαία επιφάνεια, κάτι που με τη σειρά του σημαίνει ότι υπάρχει δυνατότητα περισσότερων μετρήσεων. Εν προκειμένω, το δυναμοδάπεδο αποτελεί μια πλατφόρμα με δυνατότητα μέτρησης επιτόπιου άλματος, γεγονός εξαιρετικά σημαντικό, καθώς αυξάνει σημαντικά το σύνολο και το είδος των μετρήσεων που μπορεί να προσφέρει.

Παρόλο που το διαφορετικό μέγεθος και η μορφή της πλατφόρμας επιτρέπουν μεγαλύτερη πληθώρα μετρήσεων, είναι το μέγεθος αυτό που καθιστά σχεδόν αδύνατη τη μετακίνηση του συστήματος εκτός εργαστηρίου ιδιαίτερα σε αγωνιστικούς χώρους προπόνησης αθλητών. Οι δυνατότητες μετρήσεων λοιπόν, σαφώς αυξάνονται σε σύγκριση με το βασικό σύστημα του πελματογράφου, και πάλι όμως παραμένουν περιορισμένες σε πολύ συγκεκριμένο αριθμό και είδος. Ακόμα, όπως και στην περίπτωση του πελματογράφου, πρόκειται για ένα δυσεύρετο και ακριβό σύστημα, το οποίο το καθιστά δύσκολα προσβάσιμο σε μεγάλο κομμάτι του κόσμου.

Τέλος, οι μετρήσεις του δυναμοδαπέδου είναι μετρήσεις δυνάμεων και όχι πελματικών πιέσεων, πράγμα που σημαίνει ότι η δυνατότητα της διάταξης αυτής για ιατρική χρήση, μειώνεται αισθητά.

Η πειραματική διάταξη που σχεδιάστηκε στην συγκεκριμένη πτυχιακή εργασία, δίνει λύση στο βασικό μειονέκτημα όπως αυτό αναφέρθηκε στις δύο παραπάνω περιπτώσεις. Είναι μια φορητή διάταξη, που προσαρμόζεται σαν σόλα στο παπούτσι του εξεταζόμενου και άρα δεν φέρει κανέναν περιορισμό, στο είδος των μετρήσεων που θέλει ο εξεταζόμενος να πάρει. Ακόμα, μπορεί να μεταφερθεί σε οποιονδήποτε αγωνιστικό χώρο και συνεπώς οι μετρήσεις που θα δώσει, θα αφορούν πραγματικές συνθήκες προπόνησης. Τα αποτελέσματα των μετρήσεων δίνονται σε πραγματικό χρόνο, συνεπώς είναι δυνατό κατά τη διάρκεια της προπόνησης ο αθλητής να παρακολουθεί το πελματογράφημά του και να προσαρμόζει την προπόνησή του αναλόγως.

Ένα επιπλέον πολύ σοβαρό προτέρημα της διάταξης αυτής, είναι ότι πρόκειται για έναν εξοπλισμό πολύ φθηνό, τον οποίο μπορεί ο καθένας να αγοράσει αλλά το κυριότερο να τον χρησιμοποιεί καθημερινά και όσες φορές επιθυμεί. Τέλος, η πλατφόρμα της διάταξης είναι σχεδιασμένη έτσι, που μπορεί να χρησιμοποιηθεί για συγκομιδή ενέργειας, το οποίο θα αναφερθεί στο τελευταίο κομμάτι της εργασίας,

στις προτάσεις για μελλοντική έρευνα.

Η πειραματική διάταξη βέβαια έχει επίσης μειονεκτήματα. Όντας πρωτότυπο μοντέλο υστερεί στην ποιότητα της οπτικοποίησης των πελματιαίων πιέσεων, αλλά και στην ποιότητα των υλικών που χρησιμοποιήθηκαν για τη σχεδίαση της, σε σύγκριση με το βασικό σύστημα πελματογράφου, ο οποίος φυσικά χρησιμοποιεί πολύ πιο ακριβά υλικά. Ακόμα, ένα βασικό μειονέκτημα αποτελεί το γεγονός, πως όντας σε μορφή σόλας ή ακόμα και ενσωματωμένη σε αθλητικό παπούτσι, είναι πολύ περισσότερο ευαίσθητη σε φθορά.

Το συμπέρασμα των παραπάνω, σαν μια συνολική σύγκριση των διατάξεων, οδήγησε στην έμπνευση και σχεδίαση του νέου αυτού συστήματος πελματογράφου. Προσφέρει σημαντικά περισσότερα πλεονεκτήματα σε σχέση με τις άλλες δύο διατάξεις, καθώς και με άλλες παρόμοιες, ενώ τα μειονεκτήματά της οφείλονται βασικά στο ότι πρόκειται για ένα πρωτότυπο μοντέλο. Είναι προβλήματα που με βελτίωση της συγκεκριμένης εργασίας είναι εύκολο να ξεπεραστούν.

2.2 Αισθητήρες πίεσης

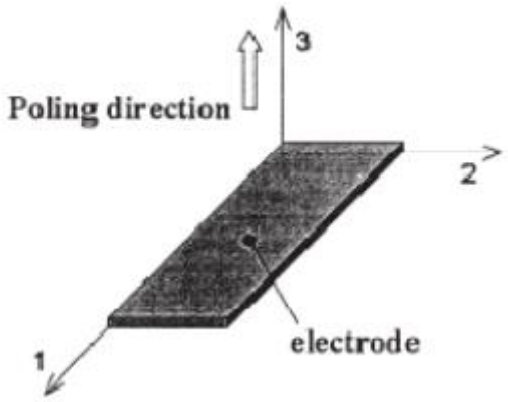
Στο κεφάλαιο αυτό, παρουσιάζονται συνοπτικά τα βασικά αισθητήρια υλικά που χρησιμοποιούνται σε συστήματα πελματογράφων και γίνεται μια σύγκριση αυτών που τελικά θα δικαιολογήσει και τις επιλογές για την προτεινόμενη διάταξη.

2.2.1 Ορισμός - μαθηματικό μοντέλο πιεζοηλεκτρισμού

Ο πιεζοηλεκτρισμός είναι ιδιότητα κεραμικών και κρυσταλλικών υλικών κατά βάση, σύμφωνα με την οποία όταν βρίσκονται υπό μηχανική πίεση ή ταλάντωση, παράγουν ηλεκτρική τάση. Το φαινόμενο μπορεί να εξηγηθεί ποιοτικά με τη μεταφορά ελεύθερων φορτίων στα άκρα του κρυσταλλικού πλέγματος του εκάστοτε πιεζοηλεκτρικού υλικού.

Οι πιεζοηλεκτρικοί αισθητήρες, έχουν κατά βάση μορφή φύλλων, με τις δύο τους όψεις επικαλυμμένες με λεπτά στρώματα ηλεκτροδίου, μεταξύ των οποίων βρίσκεται ένα στρώμα πιεζοηλεκτρικού υλικού. Οι άξονες 1 και 2, όπως αυτοί φαίνονται στο Σχήμα 2.8, του πιεζοηλεκτρικού υλικού είναι στο επίπεδο του φύλλου. Με την

μορφολογία αυτή του αισθητήρα γίνεται μέτρηση του παραγόμενου φορτίου από το πιεζοηλεκτρικό υλικό, με τις σχέσεις που αναλύονται παρακάτω. Δίνεται η δυνατότητα να υπολογίσουμε την πίεση ανά μονάδα επιφάνειας στο υλικό και κατά συνέπεια γνωρίζοντας την παραμόρφωση του υλικού υπολογίζεται τελικά και η τάνυση [18].



Σχήμα 2.8. Μορφολογία πιεζοηλεκτρικού φύλλου

Υπό συνθήκες ασθενούς πεδίου, οι καταστατικές εξισώσεις για ένα πιεζοηλεκτρικό υλικό είναι:

$$\boldsymbol{D_i = e_{ij}^{\sigma} + d_{im}^{d}\,\sigma_m} \qquad \textbf{Εξ.6}$$

$$\boldsymbol{\varepsilon_k = d_{jk}^{c} + s_{km}^{E}\,\sigma_m} \qquad \textbf{Εξ.7}$$

όπου το διάνυσμα **D** είναι η ηλεκτρική μετατόπιση (Coulomb/m^2), **ε** είναι το διάνυσμα τάνυσης, **E** είναι το εφαρμοζόμενο ηλεκτρικό πεδίο και το $\boldsymbol{\sigma_m}$ είναι το διάνυσμα πίεσης ανά μονάδα επιφάνειας. Οι πιεζοηλεκτρικές σταθερές είναι: η διηλεκτρική επιτρεπτότητα $\boldsymbol{e_{ij}^{\sigma}}$ (Farad/m), οι πιεζοηλεκτρικοί συντελεστές $\boldsymbol{d_{im}^{d}}$ και $\boldsymbol{d_{jk}^{c}}$ (Coulomb/Newton), που προσδιορίζουν την ηλεκτρική μετατόπιση ανά μονάδα δύναμης σε σταθερό ηλεκτρικό πεδίο και την τάνυση ανά μονάδα πεδίου σε σταθερή καταπόνηση, και η ελαστική παραμόρφωση $\boldsymbol{s_{km}^{E}}$ (m^2/N). Οι δείκτες **c** και **d** έχουν προστεθεί για να διαφοροποιείται το ευθύ από το αντίστροφο πιεζοηλεκτρικό φαινόμενο, αν και στην πράξη είναι αριθμητικά ίσοι. Οι δείκτες **σ** και **E** υποδεικνύουν ότι το μέγεθος μετράται υπό σταθερή δύναμη και σταθερό ηλεκτρικό πεδίο

αντίστοιχα. Για ένα φύλλο πιεζοηλεκτρικού υλικού, η κατεύθυνση πόλωσης, η οποία είναι συνήθως κατά μήκος του πάχους, υποδηλώνεται καθώς οι άξονες είναι στο επίπεδο του φύλλου [18]. Ο πίνακας $\boldsymbol{d_{jk}^{c}}$ εκφράζεται ως εξής:

$$\mathbf{d} = \begin{bmatrix} 0 & 0 & d_{31} \\ 0 & 0 & d_{32} \\ 0 & 0 & d_{33} \\ 0 & d_{24} & 0 \\ d_{15} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \textbf{Εξ.8}$$

όπου οι συντελεστές $\mathbf{d_{31}}$, $\mathbf{d_{32}}$, και $\mathbf{d_{33}}$ σχετίζουν την κανονική τάνυση στις διευθύνσεις 1, 2, 3 αντίστοιχα με ένα πεδίο κατά μήκος της διεύθυνσης πόλωσης, $\mathbf{E_3}$. Οι συντελεστές $\mathbf{d_{15}}$, και $\mathbf{d_{24}}$ σχετίζουν την εγκάρσια τάνυση στο επίπεδο 1-3 με το επίπεδο E_1 και την εγκάρσια τάνυση στο επίπεδο 2-3 στο επίπεδο $\mathbf{E_2}$ αντιστοίχως [18]. Σημειώνεται ότι δεν είναι δυνατόν να διατηρηθεί εγκάρσια τάνυση στο επίπεδο 1-2 με απλή εφαρμογή ενός ηλεκτρικού πεδίου. Γενικά, ο πίνακας της παραμόρφωσης είναι της μορφής:

$$\mathbf{S^E} = \begin{bmatrix} S_{11} & S_{12} & S_{13} & 0 & 0 & 0 \\ S_{21} & S_{22} & S_{23} & 0 & 0 & 0 \\ S_{31} & S_{32} & S_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & S_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & S_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & S_{66} \end{bmatrix} \qquad \textbf{Εξ.9}$$

Και ο πίνακας επιτρεπτότητας της μορφής:

$$\mathbf{e^{\sigma}} = \begin{bmatrix} \boldsymbol{e_{11}^{\sigma}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{e_{22}^{\sigma}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{e_{33}^{\sigma}} \end{bmatrix} \qquad \textbf{Εξ.10}$$

Το διάνυσμα πίεσης ανά μονάδα επιφάνειας γράφεται ως:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{bmatrix} = \begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{23} \\ \sigma_{31} \\ \sigma_{12} \end{bmatrix} \qquad \textbf{Εξ.11}$$

Η εξίσωση 6 είναι η εξίσωση αισθητήρα και η εξίσωση 7 είναι η εξίσωση του ενεργοποιητή. Εφαρμογές ενεργοποιητή βασίζονται στο αντίστροφο πιεζοηλεκτρικό φαινόμενο. Οι εφαρμογές αισθητήρων βασίζονται στο ευθύ πιεζοηλεκτρικό φαινόμενο. Ο αισθητήρας εκτίθεται σε ένα πεδίο πίεσης ανά μονάδα επιφάνειας και αποκρίνεται παράγοντας ένα φορτίο, το οποίο και μετράται. Στην περίπτωση του αισθητήρα, όπου το εφαρμοζόμενο εξωτερικό πεδίο είναι μηδενικό, οι εξισώσεις 6, 7 γίνονται, σε διανυσματική μορφή:

$$\begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & d_{15} & 0 \\ 0 & 0 & 0 & d_{24} & 0 & 0 \\ d_{31} & d_{32} & d_{33} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{bmatrix} \qquad \textbf{Εξ.12}$$

Αυτή είναι τελικά και η μαθηματική εξίσωση που συνοψίζει την αρχή λειτουργίας των πιεζοηλεκτρικών αισθητήρων, με την ηλεκτρική μετατόπιση **D** να σχετίζεται με το παραγόμενο φορτίο στην πιεζοηλεκτρική επιφάνεια μέσω της σχέσης [18]:

$$\boldsymbol{q} = \iint [\boldsymbol{D_1} \quad \boldsymbol{D_2} \quad \boldsymbol{D_3}] \begin{bmatrix} \boldsymbol{dA_1} \\ \boldsymbol{dA_2} \\ \boldsymbol{dA_3} \end{bmatrix} \qquad \textbf{Εξ.13}$$

όπου, τα $\mathbf{dA_1}$, $\mathbf{dA_2}$, $\mathbf{dA_3}$ είναι τα συστατικά της περιοχής ηλεκτροδίου στα επίπεδα 2-3, 1-3 και 1-2 αντίστοιχα. Μπορεί να παρατηρηθεί ότι το συλλεγόμενο φορτίο, **q**, εξαρτάται μόνο από τα συστατικά της απειροστής επιφάνειας ηλεκτροδίου ανάλογα με την μετατόπιση **D** [18].

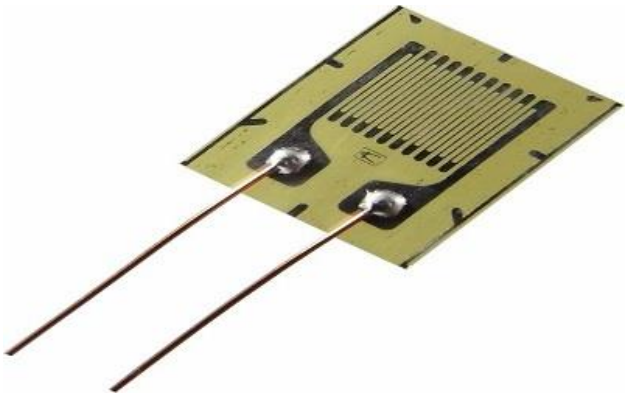
2.2.2 Βασικά είδη πιεζοηλεκτρικών στοιχείων

2.2.2.1 Αισθητήρες πιεζοαντίστασης

Οι αισθητήρες αυτοί στηρίζονται στη μεταβολή της τιμής της αντίστασης ενός μεταλλικού σύρματος όταν αυτό πιεστεί κατά μήκος της κύριας διάστασής του (Σχήμα 2.9). Για τον υπολογισμό της ποσοτικής σχέσης θεωρούμε την εξίσωση που σχετίζει την αντίσταση ενός σύρματος με τα φυσικά του χαρακτηριστικά:

$$\boldsymbol{R = \frac{\rho l}{a}} \qquad \textbf{Εξ.14}$$

όπου $\boldsymbol{\rho}$ είναι η ειδική αντίσταση του υλικού, από το οποίο είναι φτιαγμένο το σύρμα, και $\boldsymbol{l}$ και $\boldsymbol{A}$ είναι το μήκος και το εμβαδό διατομής του σύρματος.



Σχήμα 2.9. Αισθητήρας πιεζοαντίστασης

Εάν το υλικό συμπιεστεί (και άρα το μήκος του μειωθεί και η ενεργός διατομή του αυξηθεί), και εκφράσουμε τη μεταβολή $\boldsymbol{\Delta R}$ της αντίστασης ως προς τις αντίστοιχες μεταβολές των ποσοτήτων $\boldsymbol{\rho}$, $\boldsymbol{l}$ και $\boldsymbol{A}$, αποδεικνύεται ότι είναι:

$$\boldsymbol{\frac{\Delta R}{R_o} = \frac{\Delta l}{l_o} G} \qquad \textbf{Εξ.15}$$

όπου $\boldsymbol{R_0}$ και $\boldsymbol{l_0}$ είναι οι τιμές της αντίστασης και του μήκους όταν η πίεση μετρητή είναι μηδέν, $\boldsymbol{\Delta l}$ είναι η μεταβολή του μήκους και $\boldsymbol{G}$ είναι μία ποσότητα χαρακτηριστική του υλικού, που ονομάζεται gauge factor. Η ποσότητα $\boldsymbol{G}$ σχετίζεται με τα φυσικά χαρακτηριστικά του υλικού και έχει γνωστή τιμή, η οποία εξαρτάται ελαφρά από τη συμπίεση $\boldsymbol{\Delta l}$ που προκαλείται στο υλικό. Το πηλίκο $\boldsymbol{\Delta l/l_0}$ εκφράζει την ποσοστιαία συμπίεση του υλικού και είναι γνωστό με τον αγγλικό όρο strain. Αυτό

υποδεικνύει και ένα από τα βασικά μειονεκτήματα του τύπου αισθητήρων αυτών. Οι διαστάσεις και αντοχές του αισθητήρα πιεζοαντίστασης σε χαμηλές και υψηλές θερμοκρασίες, τον τοποθετούν σε περίοπτη θέση, στις επιλογές αισθητήρων για εφαρμογές πελματογραφημάτων [19], [20].

Για την κατασκευή πιεζοαντιστάσεων μπορούν να χρησιμοποιηθούν μέταλλα ή κράματα μετάλλων αλλά και γνωστοί ημιαγωγοί, όπως πυρίτιο με προσμίξεις (δηλαδή τύπου p ή n). Για την επιλογή του υλικού πρέπει να ληφθούν υπόψη οι θερμοκρασιακές συνθήκες, καθώς οι διακυμάνσεις της θερμοκρασίας προκαλούν μεταβολή της τιμής του παράγοντα G αλλά και της τιμής R_0 της αντίστασης.

Τα μέταλλα έχουν παράγοντες G με μικρές τιμές (της τάξης του 2.0), οπότε η άσκηση πίεσης σε αυτά προκαλεί μικρές μεταβολές στην αντίστασή τους. Από την άλλη πλευρά όμως, οι μεταβολές του παράγοντα G με τη θερμοκρασία είναι μικρές και ο θερμικός συντελεστής της αντίστασης μπορεί να είναι μικρός, της τάξης του 4×10^{-5} $^{o}C^{-1}$. Μπορούν επίσης να χρησιμοποιηθούν κράματα με πολύ μικρό συντελεστή γραμμικής διαστολής, ο οποίος εκφράζει την εξάρτηση της τιμής του αρχικού μήκους l_0 από τη θερμοκρασία.

Οι ημιαγωγοί έχουν μεγάλες τιμές παράγοντα G, που ενδέχεται να είναι και αρνητικές. Για παράδειγμα, το πυρίτιο τύπου p έχει τιμές παράγοντα G μεταξύ +100 και +175, ενώ το πυρίτιο τύπου n έχει τιμές παράγοντα G μεταξύ −100 και −140. Το αρνητικό πρόσημο σημαίνει ότι η συμπίεση προκαλεί αύξηση της αντίστασης του υλικού. Οι μεγάλες τιμές του παράγοντα G μας οδηγούν στη διαπίστωση ότι η αντίσταση των ημιαγωγικών υλικών είναι πολύ πιο ευαίσθητη στη συμπίεση από την αντίσταση των μεταλλικών υλικών. Από την άλλη όμως πλευρά είναι επίσης πιο ευαίσθητη σε μεταβολές της θερμοκρασίας. Αναφέρεται ότι η αύξηση της θερμοκρασίας από 0^o σε 40 ^{o}C προκαλεί μείωση της τιμής του παράγοντα G κατά περίπου 11% (από π.χ. 135 σε 120). Επίσης αναφέρεται ότι η αύξηση της θερμοκρασίας από 20^o σε 60 ^{o}C προκαλεί αύξηση της τιμής της αντίστασης R_0 κατά 4% (από π.χ. 120 Ω σε 125 Ω) [19], [20].

Με τη βοήθεια πιεζοαντιστάσεων κατασκευάζονται μετρητικές διατάξεις που ονομάζονται μετρητές μηχανικής τάσης (strain gauges). Αυτές αποτελούνται συνήθως από τέσσερις πιεζοαντιστάσεις σε διάταξη γέφυρας Wheatstone. Οι πιεζοαντιστάσεις κατασκευάζονται συνήθως από ένα λεπτό φύλλο αλουμινίου, το οποίο υφίσταται μία

διαδικασία απόξεσης (etching) και λαμβάνει τη μορφή ενός σύρματος. Το αλουμίνιο συνδέεται σε μία βάση από ρητίνη και στη συνέχεια αυτή συγκολλάται στην επιφάνεια που μας ενδιαφέρει. Η έξοδος της γέφυρας Wheatstone είναι της τάξης των 100 mV και έτσι είναι αναγκαία η ενίσχυσή της προτού διαβιβαστεί σε ένα μετατροπέα A/D. Εναλλακτικά, η έξοδος μπορεί να μην υποστεί ενίσχυση εάν ο μετατροπέας A/D έχει υψηλή ευαισθησία (π.χ. έχει ανάλυση 12 bits).

2.2.2.2 Πιεζοηλεκτρικοί αισθητήρες πίεσης

Όταν συμπιέζεται ή εφελκύεται ένας κρύσταλλος, μετατοπίζονται οι θέσεις των θετικών και αρνητικών φορτίων του και έτσι εμφανίζεται στα άκρα του μία ποσότητα φορτίου (θετικού στο ένα άκρο και αρνητικού στο άλλο άκρο), δηλαδή διαφορά δυναμικού (ηλεκτρική τάση). Το φαινόμενο αυτό ονομάζεται πιεζοηλεκτρικό φαινόμενο (piezoelectric effect) και εμφανίζεται σε όλα τα κρυσταλλικά υλικά, είναι όμως ιδιαίτερα έντονο σε ορισμένα μόνο υλικά, τα οποία ονομάζονται για το λόγο αυτό πιεζοηλεκτρικά. Τα υλικά αυτά εμφανίζουν και το αντίστροφο πιεζοηλεκτρικό φαινόμενο, κατά το οποίο η εφαρμογή στα άκρα του υλικού μίας τάσης **V** προκαλεί συμπίεση (ή εφελκυσμό, ανάλογα με το πρόσημο της τάσης) κατά διάστημα **x**:

$$x = d\,V \qquad \textbf{Εξ.16}$$

όπου η ποσότητα **d** είναι η πιεζοηλεκτρική σταθερά του υλικού [20].

Η περιγραφή του πιεζοηλεκτρικού φαινομένου γίνεται πολύ απλά θεωρώντας ότι ένας κρύσταλλος λειτουργεί σαν ελατήριο σταθεράς **k** όταν συμπιέζεται κατά διάστημα **x**. Η σχέση μεταξύ δύναμης και διαστήματος είναι η εξής:

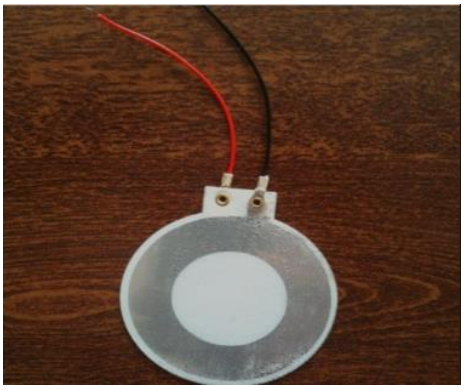
$$F = k\,x \qquad \textbf{Εξ.17}$$

Συνδυάζοντας τις δύο παραπάνω σχέσεις εξάγεται η σχέση μεταξύ της εφαρμοζόμενης δύναμης και της πιεζοηλεκτρικής τάσης αναπτύσσεται στα άκρα του υλικού:

$$\mathbf{F = kdV \;\Leftrightarrow V = \frac{F}{kd}} \qquad \textbf{Εξ.18}$$

Οι πιεζοηλεκτρικοί κρύσταλλοι είναι συνήθως κρύσταλλοι χαλαζία (quartz).

Μπορούν να έχουν πολύ μικρό μέγεθος και είναι ανθεκτικοί σε υψηλές θερμοκρασίες. Η σταθερά **k** των κρυστάλλων είναι πολύ μεγάλη (δηλαδή οι κρύσταλλοι είναι σκληροί), με αποτέλεσμα η τάση V που παράγεται να έχει μικρές τιμές και να απαιτείται η ενίσχυσή της ή η τοποθέτηση πολλών κρυστάλλων σε σειρά. Οι κρύσταλλοι τίθενται σε επαφή με ένα κινητό διάφραγμα, το οποίο δέχεται την πίεση, και συμπιέζονται από αυτό [21], [22].



Σχήμα 2.10. Πιεζοηλεκτρικός αισθητήρας

Οι πιεζοηλεκτρικοί κρύσταλλοι δε χρησιμοποιούνται μόνο για τη μέτρηση πιέσεων και δυνάμεων αλλά και επιταχύνσεων, καθώς γνωρίζουμε ότι η δύναμη και η επιτάχυνση είναι μεγέθη ανάλογα, με βάση τη γνωστή σχέση:

$$F = m\,a \qquad \textbf{Εξ.19}$$

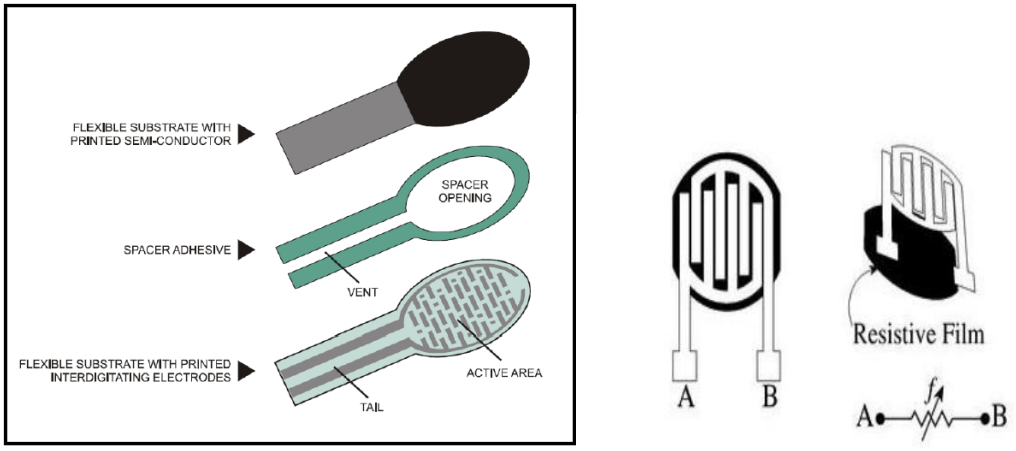
όπου ***a*** είναι η επιτάχυνση και **m** είναι η μάζα που δέχεται την επιτάχυνση [21].

Τα πιεζοηλεκτρικά υλικά διακρίνονται μεταξύ τους ανάλογα με την τιμή του ηλεκτρικού σήματος που παράγουν εξαιτίας μίας δεδομένης πίεσης, την απόκρισή τους σε πιέσεις διαφόρων συχνοτήτων, τη συχνότητα στην οποία εμφανίζουν το φαινόμενο του συντονισμού κ.ά. Λόγω των δυναμικών χαρακτηριστικών της λειτουργίας των πιεζοηλεκτρικών υλικών, οι πιεζοηλεκτρικοί αισθητήρες πίεσης χρησιμοποιούνται περισσότερο για τη μέτρηση δυναμικών φαινομένων πίεσης. Το γεγονός αυτό, σε συνδυασμό με την δομή ενός πιεζοηλεκτρικού αισθητήρα (Σχήμα 2.10), τον καθιστά τη λιγότερο ορθή επιλογή για διατάξεις πελματογραφήματος [21], [22].

2.2.2.3 Αισθητήρες δύναμης-πίεσης (FSR)

Οι αισθητήρες που αναφέρομαι στο κομμάτι αυτό της πτυχιακής είναι και οι αισθητήρες που τελικά χρησιμοποιήθηκαν για τη σχεδίαση της διάταξης. Προσφέρουν πάρα πολλά πλεονεκτήματα σε σύγκριση με τους προαναφερθέντες αισθητήρες, κομμάτι το οποίο θα αναλυθεί εκτενώς αργότερα στην εργασία. Συνοπτικά, το πιο σημαντικό πλεονέκτημα είναι ότι πρόκειται για πολύ φθηνό εξοπλισμό, εύκολο στη χρήση, ο οποίος προσφέρει υψηλής ποιότητας αποτελέσματα.

Οι αισθητήρες δύναμης-πίεσης, αποτελούν αισθητήρες που επιτρέπουν την ανίχνευση της φυσικής πίεσης, τη συμπίεση και το βάρος που τους ασκείται. Πρόκειται για αισθητήρες, οι οποίοι είναι εξαιρετικά απλοί στη χρήση τόσο στον τρόπο σύνδεσης, όσο και στον προγραμματισμό τον οποίο απαιτούν για τη χρήση τους σε συνδυασμό με τη χρήση μικροελεγκτή. Το Σχήμα 2.11 παρουσιάζει μια φωτογραφία ενός τέτοιου αισθητήρα, συγκεκριμένα του μοντέλου Interlink 402, το οποίο αποτελεί το πλέον χρησιμοποιημένο μοντέλο αυτού του είδους των αισθητήρων. Πρόκειται για πολύ μικρούς αισθητήρες, με ένα βασικό μοντέλο όπως το παρακάτω (Σχήμα 2.11) να έχει διαστάσεις 1.5"x1.5", γεγονός που όμως δεν τους μειώνει σε καμία περίπτωση την ακρίβεια μετρήσεων.



Σχήμα 2.11. Η μορφολογία ενός αισθητήρα δύναμης πίεσης

Ο αισθητήρας δύναμης-πίεσης, χωρίζεται σε δύο διαφορετικά στρώματα, μεταξύ των οποίων υπάρχει μια επιπλέον διαχωριστική επιφάνεια. Όση περισσότερη πίεση ασκείται στην άνω επιφάνεια του αισθητήρα, τόσο μεγαλύτερη ενεργή περιοχή έρχεται σε επαφή με τον ημιαγωγό του, με αποτέλεσμα να μειώνεται αντιστοίχως η αντίστασή του. Οι αισθητήρες δύναμης-πίεσης συνεπώς, αποτελούν μια μεταβλητή

αντίσταση που διαφοροποιεί την τιμή της, αναλόγως με το πόσο πίεση εφαρμόζεται στον αισθητήρα (Σχήμα 2.11).

Η λογική πίσω από την λειτουργία των αισθητήρων αυτών βρίσκεται πίσω από την εκμετάλλευση δύο φαινομένων: Του φαινομένου της διήθησης και του φαινομένου της κβαντικής σήραγγας.

Και τα δύο παραπάνω φαινόμενα, εκμεταλλεύονται με τη σειρά τους το στρώμα αγώγιμου πολυμερούς του αισθητήρα. Το φαινόμενο διήθησης εμφανίζεται όταν η συγκέντρωση των σωματιδίων είναι πάνω από το όριο διήθησης $\boldsymbol{\varphi_c}$. Τότε εμφανίζεται στο πολυμερές μια αύξηση της ηλεκτρικής αντίστασης **R** του αισθητήρα, που είναι συνέπεια της αύξησης της εφαρμοζόμενης πίεσης σε αυτόν. Για κάποια τιμή πίεσης, η ικανότητα ηλεκτρικής αντίστασης **ρ** του αισθητήρα μπορεί να υπολογιστεί από την σχέση [23], [24], [25]:

$$\boldsymbol{\rho = \rho_o(\varphi - \varphi_c)^{-x}}$$

Εξ.20

όπου το $\boldsymbol{\rho_o}$ είναι ένας παράγοντας που αλλάζει ανάλογα με τις ιδιότητες μεταφοράς του αγώγιμου πολυμερούς και ο παράγοντας **x** είναι ο κρίσιμος εκθέτης αγωγιμότητας. Με το φαινόμενο της διήθησης τα σωματίδια στην επιφάνεια του πολυμερούς διαχωρίζονται μεταξύ τους υπό την μηχανική καταπόνηση, προκαλώντας ταυτόχρονα αύξηση της αντίστασης του αισθητήρα [26].

Το φαινόμενο της κβαντικής σήραγγας όμως, είναι ο βασικότερος λόγος λειτουργίας των αισθητήρων δύναμης-πίεσης. Κατά το φαινόμενο της κβαντικής σήραγγας, ο μέσος διαχωρισμός των φορτισμένων σωματιδίων στην επιφάνεια του πολυμερούς μειώνεται όταν αυτό υποβάλλεται σε πίεση. Έχουν αναπτυχθεί πολλά μοντέλα για την αγωγιμότητα κβαντικής σήραγγας, τα οποία βασίζονται σε εξισώσεις που αφορούν την μετάδοση των σωματιδίων αυτών σε ένα ορθογώνιο δυνητικό φράγμα. Η μείωση της αγωγιμότητας αυτής, προκαλεί μια ταυτόχρονη αύξηση της πιθανότητας τα σωματίδια αυτά να μεταδοθούν, βάσει των εξισώσεων Simmons [27]:

$$I(U,s) = \frac{3A\sqrt{2mV_a}}{2s}\left(\frac{e}{h}\right)^2 Ue^{(-\frac{4\pi s}{h})\sqrt{2mV_a}}$$

Εξ.21

όπου $\mathbf{V_a}$ το άνω όριο του ορθογώνιου δυνητικού φράγματος, **e** και **m** είναι το φορτίο και η μάζα του ηλεκτρονίου αντίστοιχα και **h** η σταθερά Planck. Η παραπάνω εξίσωση Simmons είναι θεμελιώδης για την λειτουργία των αισθητήρων δύναμης-πίεσης και σχετίζει τον αισθητήρα **U** με το ρεύμα **I** που περνά από την αντίσταση του αισθητήρα.

Προκειμένου ο αισθητήρας να λειτουργήσει βάσει του φαινομένου της κβαντικής σήραγγας η συγκέντρωση των σωματιδίων του πολυμερούς πρέπει να διατηρηθεί κάτω από το προαναφερθέν κατώφλι διήθησης, όπως και στο ομώνυμο φαινόμενο.

Σε κατάσταση που η αγώγιμη επιφάνεια πολυμερούς είναι πλήρως φορτισμένη οι σχέσεις που δείχνουν τον διαχωρισμό των σωματιδίων **s**, που προκαλεί την μείωση της αντίστασης στον αισθητήρα ανάλογα και με την πίεση που εφαρμόζεται σε αυτόν, είναι οι:

$$s_o = D\left[\left(\frac{\pi}{6\varphi}\right)^{\frac{1}{3}} - 1\right]$$

Εξ.22

όπου το $\mathbf{s_o}$ υποδεικνύει τον διαχωρισμό σε κατάσταση ηρεμίας του πολυμερούς, δηλαδή χωρίς εφαρμοζόμενη πίεση στον αισθητήρα και **D** η διάμετρος του σωματιδίου.

$$s = s_o\left(1 - \frac{\sigma}{M}\right)$$

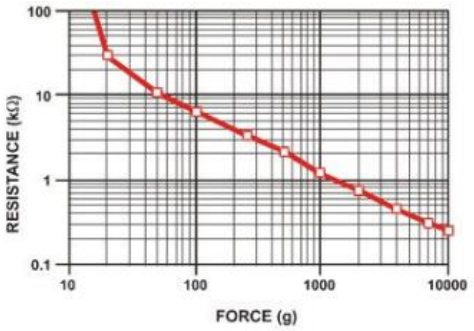
Εξ.23

όπου **M** είναι ο συντελεστής αγωγιμότητας ενός πολυμερούς.

Χρησιμοποιώντας, τις τρεις παραπάνω εξισώσεις, το πιο ευρέως χρησιμοποιούμενο μοντέλο στους αισθητήρες δύναμης-πίεσης, για τον υπολογισμό της μεταβλητής αυτής αντίστασης της πολυμερούς επιφάνειας, είναι το παρακάτω [28]:

$$R_{pol} = \frac{D\left[\left(\frac{\pi}{6\varphi}\right)^{\frac{1}{3}} - 1\right]\left(1 - \frac{\sigma}{M}\right)}{A\sqrt{2mV_a}}\left(\frac{h}{e}\right)^2 e^{\left(\frac{4\pi D}{h}\left[\left(\frac{\pi}{6\varphi}\right)^{\frac{1}{3}} - 1\right]\left(1 - \frac{\sigma}{M}\right)\sqrt{2mV_a}\right)} \quad \text{Εξ.24}$$

Η τιμή της αντίστασης κυμαίνεται μεταξύ 100ΚΩ, σε ελαφρές πιέσεις στον αισθητήρα, μέχρι και 300Ω στη μέγιστη πίεση που μπορεί να αντέξει. Στο παρακάτω διάγραμμα (Σχήμα 2.12) φαίνεται η διακύμανση της συγκεκριμένης αντίστασης, με σημαντική σημείωση ότι η αντίσταση είναι ουσιαστικά άπειρη όταν δεν ασκείται πίεση στον αισθητήρα.



Σχήμα 2.12. Διάγραμμα πίεσης-μεταβλητής αντίστασης στον αισθητήρα

2.2.3 Λόγοι επιλογής των αισθητήρων δύναμης-πίεσης

Οι αισθητήρες δύναμης-πίεσης αποτελούν ίσως την καλύτερη επιλογή όσον αφορά την αναλογία ποιότητας αποτελεσμάτων και κόστους. Πρόκειται για εξοπλισμό πολύ φθηνό, ειδικά λαμβάνοντας υπόψη πως για το μέγεθός του αλλά και για την απλότητά του, οι δυνατότητες του ξεπερνούν τις προσδοκίες.

Στην σχεδίαση της κυκλωματικής διάταξης έγιναν διάφορες δοκιμές μέχρι να σχεδιαστεί το τελικό πρωτότυπο, τόσο σε υλικό, όσο και σε λογισμικό. Χρησιμοποιήθηκαν λοιπόν στα αρχικά στάδια της σχεδίασης απλοί πιεζοηλεκτρικοί αισθητήρες. Το πρόβλημα που εμφανίστηκε, ήταν ότι οι αισθητήρες αυτοί πραγματοποιούν διαφορική μέτρηση, με αποτέλεσμα η απόκρισή τους στην εφαρμοζόμενη δύναμη πάνω στην επιφάνειά τους να είναι πάρα πολύ αργή. Κατά συνέπεια, ήταν αδύνατη η σωστή μέτρηση της πίεσης με απόκριση πραγματικού χρόνου, που είναι και ζητούμενο της διάταξης. Η επιλογή των αισθητήρων δύναμης-πίεσης, έλυσε αυτό το πρόβλημα, χωρίς να απαιτηθεί ουσιαστική αλλαγή στην κυκλωματική διάταξη.

Βασικό ρόλο στην επιλογή των εν λόγω αισθητήρων έπαιξε και η μορφολογία τους. Το γεγονός δηλαδή ότι είναι πολύ μικροί, επίπεδοι και απαιτούν ελάχιστη καλωδίωση, αποτελεί εξαιρετικά σημαντικό κομμάτι, καθώς οι αισθητήρες αυτοί θα τοποθετηθούν σε αθλητικό παπούτσι και άρα πρέπει να είναι όσο το δυνατόν λιγότερο επεμβατικοί. Οι πιεζοηλεκτρικοί αισθητήρες σε αντίθεση, έχουν μια δομή (βλέπε Σχήμα 2.10), που τους καθιστά δύσχρηστους σε μια διάταξη σαν αυτή της εν λόγω πτυχιακής εργασίας.

2.3 Το Arduino

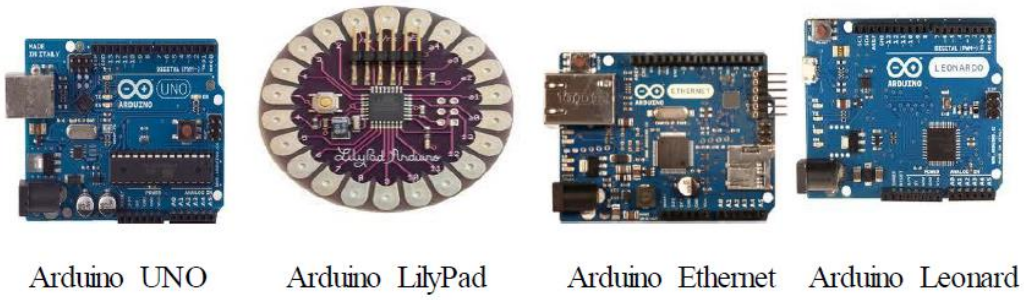
2.3.1 Υλικό

Μια πλακέτα Arduino αποτελείται από ένα μικροελεγκτή Atmel, με ένα σύνολο επιπρόσθετων στοιχείων που διευκολύνουν χρήστες μη οικείους με τον προγραμματισμό, στο κομμάτι της συγγραφής κώδικα και στην ενσωμάτωση της πλακέτας σε άλλα κυκλώματα. Σημαντικό κομμάτι του Arduino είναι η επεκτασιμότητά του. Ειδικά πρόσθετα (shields) μπορούν να τοποθετηθούν σε αυτό και να προσφέρουν επιπλέον λειτουργίες. Μερικά πρόσθετα επικοινωνούν με το Arduino με απευθείας σύνδεση πάνω σε αυτό. Το Arduino έχει χρησιμοποιήσει τη σειρά μικροελεγκτών megaAVR και συγκεκριμένα τους ATmega8, ATmega168, ATmega328, ATmega1280 και ATmega2560. Η πλειοψηφία των μικροελεγκτών που χρησιμοποιούν τα διαφορετικά μοντέλα Arduino λειτουργούν στα 5V, στη συχνότητα των 16MHZ. Το σύνολο των πλακετών Arduino προγραμματίζονται μέσω USB, με τη χρήση ενσωματωμένου προσαρμογέα USB.

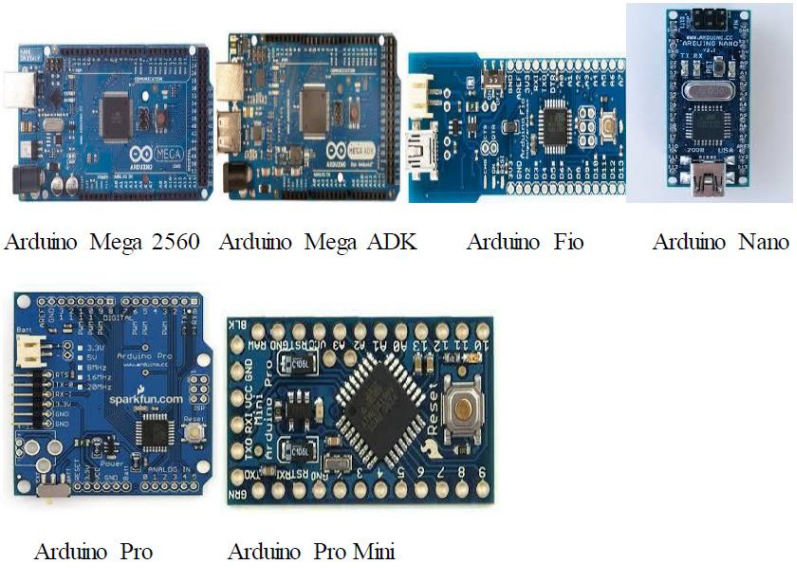
Υπάρχουν πάρα πολλές πλακέτες παρόμοιες με το Arduino, ή και συμβατές με αυτό. Μερικές είναι λειτουργικά ισοδύναμες με ένα Arduino και μπορεί να χρησιμοποιηθούν εναλλακτικά. Οι παραλλαγές των πλακετών αυτών μπορούν να βρίσκονται τόσο σε επίπεδο λογισμικού όσο και σε επίπεδο υλικού, με διαφορετικούς επεξεργαστές παραδείγματος χάριν [29].

2.3.2 Μοντέλα Arduino

Τα μοντέλα Arduino που κυκλοφορούν στην αγορά, παρουσιάζονται στα παρακάτω Σχήματα (Σχήματα 2.12.1 & 2.12.2). Το Arduino UNO είναι το μοντέλο που επιλέχθηκε επειδή καλύπτει όλες τις ανάγκες της συγκεκριμένης εργασίας, καθώς τα pins που διαθέτει επαρκούν για όλες τις λειτουργίες του πελματογράφου. Ακόμα, μαζί με το Arduino Lilypad, είναι τα μοντέλα τα οποία χρησιμοποιούνται κατά βάση σε wearable εφαρμογές, όπως είναι και η προτεινόμενη διάταξη.



Σχήμα 2.12.1. Μοντέλα Arduino (I)



Σχήμα 2.12.2. Μοντέλα Arduino (II)

Μια σημαντική σημείωση που πρέπει να γίνει είναι ότι, όπως φαίνεται και από τις παραπάνω εικόνες, υπάρχει πληθώρα επιλογών του μικροελεγκτή Arduino, επιλογή που κάθε φορά εξαρτάται από τον τύπο της εφαρμογής.

2.3.3 Τα χαρακτηριστικά του Arduino

- **Μνήμη**

Οι μικροελεγκτές ATmega, που είναι η ομάδα μικροελεγκτών που χρησιμοποιεί το Arduino, διαθέτουν μνήμη flash, στην οποία αποθηκεύονται τα προγράμματα, SRAM (static random access memory), στην οποία δημιουργείται κάθε διεργασία και αποθηκεύει και χρησιμοποιεί τις μεταβλητές της όταν εκτελείται και EPPROM, στην οποία αποθηκεύονται δεδομένα που πρόκειται να χρησιμοποιηθούν επανειλημμένα ακόμα και μετά από την διακοπή παροχής ρεύματος στον μικροελεγκτή. Επειδή οι μικροελεγκτές ATmega είναι αρκετοί θα αναφερθούμε μόνο στα χαρακτηριστικά των 3 μνημών του επεξεργαστή του Arduino Uno, που είναι και το μοντέλο που χρησιμοποιήθηκε.

Η SRAM είναι μνήμη, χωρητικότητας 2KB, που χρησιμοποιείται από τα προγράμματα για να αποθηκεύονται όλες οι μεταβλητές που χρησιμοποιούνται από το εκάστοτε πρόγραμμα, όταν αυτό εκτελείται. Η μνήμη χάνει τα δεδομένα της όταν η παροχή ρεύματος στον Arduino σταματήσει ή πατηθεί το κουμπί επανεκκίνησης.

Έπειτα, υπάρχει η 32KB μνήμη flash. Από το σύνολο της χωρητικότητάς της, 2KB χρησιμοποιούνται από το υλικό του Arduino, το οποίο αφορά την εγκατάσταση όλων των προγραμμάτων που φορτώνονται στο μικροελεγκτή μέσω της θύρας USB. Τα υπόλοιπα 30KB της μνήμης Flash χρησιμοποιούνται για την αποθήκευση των προγραμμάτων, αφού πρώτα μεταγλωττιστούν στον υπολογιστή. Η μνήμη flash, αντίστοιχα με την EEPROM, δεν χάνει τα περιεχόμενά της με απώλεια τροφοδοσίας ή με επανεκκίνηση. Είναι συνεπώς και αυτή μνήμη που χρησιμοποιείται για δεδομένα που χρησιμεύουν μακροχρόνια σε οποίον προγραμματίζει το Arduino.

Τέλος, υπάρχει και η EEPROM, που μπορεί να χρησιμοποιηθεί για εγγραφή ή ανάγνωση δεδομένων από τα προγράμματα που τρέχουν στον μικροελεγκτή. Σε αντίθεση με την SRAM, δεν χάνει τα περιεχόμενά της με απώλεια τροφοδοσίας ή επανεκκίνηση. Πρόκειται λοιπόν, για τη μνήμη που χρησιμοποιούν οι διάφορες διεργασίες του Arduino, καθώς αυτές εκτελούνται.

- **Οι ακροδέκτες του Arduino**

Καταρχήν το Arduino διαθέτει σειριακό τρόπο επικοινωνίας με τον χρήστη. Ο μικροελεγκτής που διαθέτει συνδέεται σειριακά και επικοινωνεί με τον υπολογιστή μέσω θύρας USB . Η σύνδεση αυτή συνεπώς χρησιμοποιείται για την φόρτωση των προγραμμάτων που σχεδιάζονται στο προγραμματιστικό περιβάλλον του Arduino αλλά και για αμφίδρομη επικοινωνία του Arduino με τον υπολογιστή μέσα από το πρόγραμμα την ώρα που εκτελείται.

Υπάρχουν δύο σειρές από pins στη πλακέτα του Arduino. Η μια σειρά αριθμημένη από 0 έως 13 που αντιστοιχεί στις ψηφιακές εισόδους και εξόδους της πλακέτας. Λειτουργούν στα 5V και καθένα μπορεί να παρέχει ή να δεχτεί το πολύ 40mA. Ως ψηφιακή έξοδος, ένα από αυτά τα pins, μπορεί να τεθεί από το πρόγραμμα σε κατάσταση HIGH ή LOW, οπότε το Arduino θα ξέρει αν πρέπει να διοχετεύσει ή όχι ρεύμα στο συγκεκριμένο pin. Η κατάσταση ενός pin σε HIGH ή LOW μπορεί να πραγματοποιηθεί και εξωτερικά αλλά και μέσω του κώδικα. Έκτος της λειτουργίας των pins αυτών ως ψηφιακές είσοδοι/έξοδοι, αυτά τα pins πραγματοποιούν και επιπλέον πολύ σημαντικές λειτουργίες και εν προκειμένω:

▪ Τα pins 0 και 1 λειτουργούν ως RX και TX της σειριακής σύνδεσης, όταν το πρόγραμμά μας ενεργοποιεί την σειριακή θύρα. Έτσι, όταν το πρόγραμμά μας στέλνει δεδομένα στην σειριακή, αυτά προωθούνται και στην θύρα USB αλλά και στο pin 0, για να τα διαβάσει ενδεχομένως μια άλλη συσκευή (π.χ. ένα μοντέλο Bluetooth). Αυτό φυσικά σημαίνει ότι αν στο πρόγραμμά μας ενεργοποιήσουμε το σειριακό λογισμικό, χάνουμε 2 ψηφιακές εισόδους/εξόδους.

▪ Τα pins 2 και 3 λειτουργούν και ως εξωτερικές διακοπές (διακοπή 0 και 1 αντίστοιχα). Με άλλα λόγια, μπορούμε να τα ρυθμίσουμε μέσα από το πρόγραμμά μας ώστε να λειτουργούν αποκλειστικά ως ψηφιακές είσοδοι στις οποίες όταν συμβαίνουν συγκεκριμένες αλλαγές, η κανονική ροή του προγράμματος σταματάει άμεσα και εκτελείται μια συγκεκριμένη συνάρτηση. Οι εξωτερικές διακοπές είναι ιδιαίτερα χρήσιμες σε εφαρμογές που απαιτούν συγχρονισμό μεγάλης ακρίβειας.

▪ Τα pins 3, 5, 6, 9, 10 και 11 μπορούν να λειτουργήσουν και ως ψευδοαναλογικές έξοδοι με το σύστημα PWM (PuIse Width Moodulation), για τιμές 0-255. Είναι σημαντικό το ότι το PWM δεν είναι πραγματικά αναλογικό σύστημα και ότι θέτοντας

στην έξοδο την τιμή 127, δεν σημαίνει ότι η έξοδος θα είναι 2.5V αντί της κανονικής τιμής των 5V, αλλά ότι θα παράγεται ένας παλμός που θα εναλλάσσεται με μεγάλη συχνότητα και για ίσους χρόνους, μεταξύ των τιμών 0 και 5V.

Υπάρχει και μια ακόμη σειρά από pins, αριθμημένα από το 0 ως το 5. Το καθένα από αυτά λειτουργεί ως αναλογική είσοδος, κάνοντας χρήση του ADC (Analog to Digital Converter) που είναι ενσωματωμένος στον μικροελεγκτή. Για παράδειγμα, μπορούμε να τροφοδοτήσουμε ένα από αυτά με μια τάση την οποία μπορούμε να μεταβάλουμε με ένα ποτενσιόμετρο από ON ως μια τάση αναφοράς Vref, η οποία, αν δεν κάνουμε κάποια αλλαγή, είναι προρυθμισμένη στα 5V. Τότε, μέσα από το πρόγραμμά μας μπορούμε να «διαβάσουμε» την τιμή του pin ως ένα ακέραιο αριθμό ανάλυσης 10-bit, από 0 (όταν η τάση στο pin είναι ON) μέχρι 1023 (όταν η τάση στο pin είναι 5V). Η τάση αναφοράς μπορεί να ρυθμιστεί με μια εντολή σε όποια τάση επιθυμούμε (μεταξύ 2 και 5V) τροφοδοτώντας εξωτερικά με αυτή την τάση το pin με την σήμανση AREF που βρίσκεται στην απέναντι πλευρά της πλακέτας. Έτσι, αν τροφοδοτήσουμε το pin AREF με 3.3V και στην συνέχεια δοκιμάσουμε να διαβάσουμε κάποιο pin αναλογικής εισόδου στο οποίο εφαρμόζεται τάση 1.65V, το Arduino θα μας επιστρέψει την τιμή 512. Τέλος, καθένα από τα 6 αυτά pins, με κατάλληλη εντολή μέσα από το πρόγραμμα, μπορεί να μετατραπεί σε ψηφιακό pin εισόδου/εξόδου, όπως τα 14 που βρίσκονται στην απέναντι πλευρά και τα οποία περιγράφηκαν πριν. Σε αυτή την περίπτωση τα pins μετονομάζονται από 0~5 σε 14~19 αντίστοιχα.

- **Τροφοδοσία**

Το Arduino τροφοδοτείται είτε από εξωτερική τροφοδοσία είτε απευθείας από την θύρα USB. Η επιλογή της πηγής γίνεται αυτόματα. Ως εξωτερική τροφοδοσία ορίζεται είτε μια μπαταρία, είτε ένα τροφοδοτικό 7-12 VDC. Η τροφοδοσία μπορεί να συνδεθεί στο pin Vin ή στην ειδική υποδοχή που υπάρχει.

Αν το Arduino τροφοδοτηθεί με λιγότερα από 7 Volt, τα pin εξόδου 5Volt δεν θα καταφέρουν να εξάγουν την αντίστοιχη τάση. Αν από την άλλη το τροφοδοτήσουμε με πάνω από 12 Volt, θα υπερθερμανθεί ο σταθεροποιητής τάσης στην πλακέτα και ενδεχομένως να καταστραφεί. Συνεπώς μια ιδανική τάση είναι τα 9 Volt. Οι ακροδέκτες τροφοδοσίας είναι οι εξής: i) Vin: Ακροδέκτης για μη σταθεροποιημένη τάση. Συνήθως εδώ συνδέεται μια εξωτερική πηγή τροφοδοσίας, ii) 5V: Ακροδέκτης σταθεροποιημένης τάσης 5Volt. Χρησιμοποιείται για την τροφοδοσία του

μικροελεγκτή ή άλλων ηλεκτρονικών στοιχείων, iii) 3.3V: Το ολοκληρωμένο FDTI που βρίσκεται στην πλακέτα του Arduino παράγει τάση των 3.3V με μέγιστο ρεύμα 50mA, iv) GND: Ακροδέκτες Γείωσης.

- **Ενσωματωμένες Λειτουργίες**

Επάνω στην πλακέτα υπάρχουν 4 μικροσκοπικά LEDs επιφανειακής στήριξης καθώς και ένας διακόπτης reset. Τα δύο LEDs με τις σημάνσεις TX και RX, χρησιμοποιούνται ως ένδειξη λειτουργίας της σειριακής επικοινωνίας του Arduino. Λειτουργούν παραδείγματος χάριν κατά την επικοινωνία του Arduino μέσω Bluetooth. Ένα από τα LEDs, το οποίο φέρει και την ένδειξη POWER προφανώς ενημερώνει ότι το Arduino έχει τροφοδοσία και είναι σε λειτουργία.. Η βασική δοκιμή λειτουργίας του Arduino είναι να του αναθέσουμε να αναβοσβήνει ένα LED. Για να μπορούμε να το κάνουμε αυτό από την πρώτη στιγμή, χωρίς να συνδέσουμε τίποτα πάνω στο Arduino, οι κατασκευαστές του έχουν ενσωματώσει ένα LED στην πλακέτα, το οποίο είναι συνδεδεμένο με το ψηφιακό pin 13. Έτσι, ακόμα και αν δεν έχουμε συνδέσει τίποτα πάνω στο φυσικό pin 13, αναθέτοντάς του την τιμή HIGH μέσα από το πρόγραμμά μας, θα ανάψει το ενσωματωμένο LED [29], [30].

2.3.4 Λογισμικό

Το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) του Arduino (Σχήμα 2.13) είναι μια εφαρμογή γραμμένη σε Java, εφαρμόσιμη σε πολλές διαφορετικές πλατφόρμες και προέρχεται από το IDE για τη γλώσσα προγραμματισμού Processing. Το πρόγραμμα της Processing, για οπτικοποίηση δεδομένων από το Arduino είναι και αυτό που χρησιμοποιήθηκε για την πειραματική διάταξη. Θα αναφερθούμε σε αυτό όμως αναλυτικότερα παρακάτω. Έχει σχεδιαστεί για να εισαγάγει στον προγραμματισμό άτομα τα οποία δεν είναι εξοικειωμένα με την ανάπτυξη λογισμικού και ασχολούνται με εφαρμογές και projects που αφορούν οπτικοποίηση δεδομένων. Χαρακτηριστικό παράδειγμα της ευκολίας που προσφέρει, είναι η δυνατότητα να μεταγλωττίζει και να φορτώνει προγράμματα στην πλακέτα με το πάτημα ενός κουμπιού. Περιλαμβάνει ένα πρόγραμμα επεξεργασίας κώδικα, με χαρακτηριστικά ειδικά σχεδιασμένα για να βοηθούν "άπειρους προγραμματιστικά" χρήστες, όπως είναι η επισήμανση σύνταξης, ο συνδυασμός αγκύλων και η αυτόματη εσοχή. Με τον όρο «σχέδιο» αναφερόμαστε

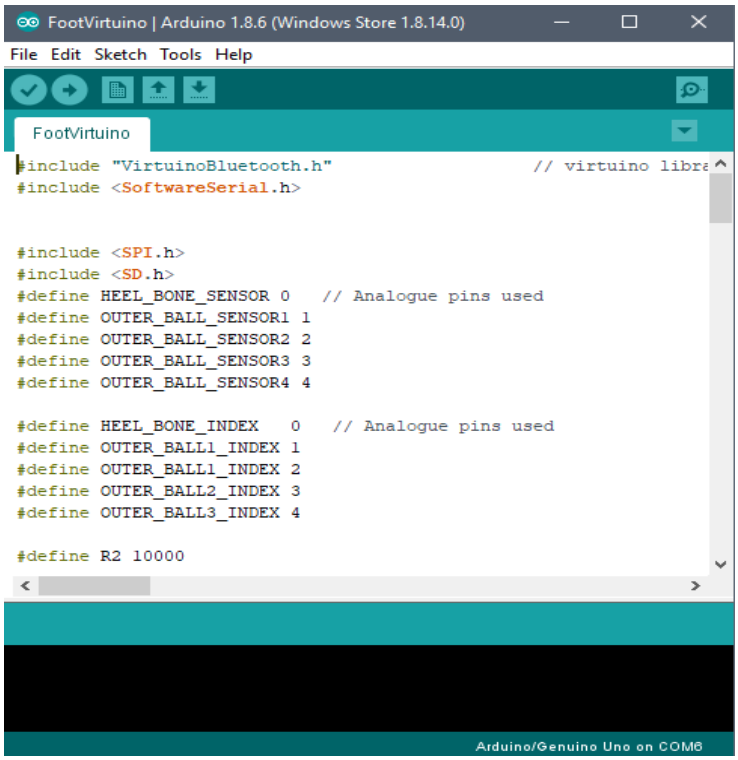
σε ένα πρόγραμμα ή κώδικα που γράφτηκε για Arduino.

Τα σχέδια του Arduino είναι γραμμένα σε C ή C++. Το προγραμματιστικό περιβάλλον Arduino IDE, είναι πακέτο με την προσθήκη μιας βιβλιοθήκης λογισμικού που ονομάζεται "Wiring", γεγονός που καθιστά πολλές κοινές λειτουργίες εισόδου / εξόδου πολύ πιο εύκολες για τον χρήστη. Για την εγγραφή ενός κώδικα απαιτείται η χρήση και εγγραφή δύο βασικών μόνο συναρτήσεων [30], [31].:

setup(): μια συνάρτηση που εκτελείται μία φορά κατά την έναρξη ενός προγράμματος, η οποία μπορεί να αρχικοποιεί τις ρυθμίσεις.

loop(): μια συνάρτηση η οποία καλείται συνέχεια μέχρι η πλακέτα να απενεργοποιηθεί.

Τα πλεονεκτήματα του προγραμματιστικού περιβάλλοντος του Arduino είναι πάρα πολλά, με βασικότερο το ότι απαιτεί πολύ βασικές γνώσεις προγραμματισμού για τη δημιουργία κάποιου σχεδίου. Βέβαια, δεν προσφέρεται για οπτικοποίηση των δεδομένων και για αυτό άλλωστε δεν επελέγη για το κομμάτι αυτό.



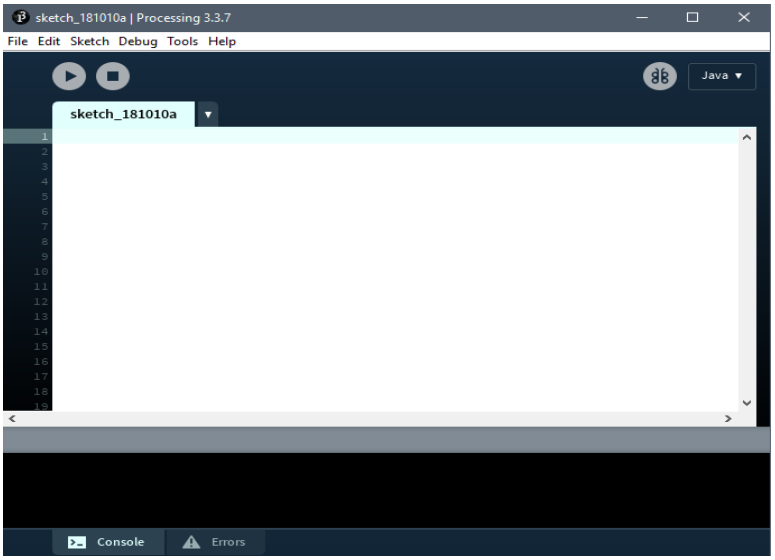
Σχήμα 2.13. Προγραμματιστικό περιβάλλον Arduino IDE

2.4 Λογισμικό οπτικοποίησης και αξιολόγησης δεδομένων

Όπως αναφέρθηκε και νωρίτερα, αν και το προγραμματιστικό περιβάλλον του Arduino είναι πολύ εύκολο στη χρήση, οι δυνατότητες που προσφέρει σε επίπεδο οπτικοποίησης των δεδομένων, που λαμβάνει παραδείγματος χάριν από έναν αισθητήρα, είναι τρομερά περιορισμένες. Συγκεκριμένα, δίνεται η δυνατότητα οπτικοποίησης των δεδομένων μόνο με γραφικές παραστάσεις, οι οποίες ανταποκρίνονται σε πραγματικό χρόνο, αλλά προφανώς δεν είναι αρκετές για οπτική ανάλυση, ειδικά για μια εφαρμογή όπως το πελματογράφημα. Επιλέχτηκε λοιπόν και χρησιμοποιήθηκε το πρόγραμμα Processing.

2.4.1 Processing

Το Processing είναι μια ανοικτή γλώσσα προγραμματισμού και ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) που χρησιμοποιείται για εφαρμογές οπτικού σχεδιασμού και χρησιμεύει ως θεμέλιο για προγράμματα που γράφονται σε προγραμματιστικά περιβάλλοντα όπως το IDE του Arduino. Η δημιουργία του έγινε το 2001 από τους Casey Reas και Ben Fry, οι οποίοι βρίσκονταν αρχικά στο MIT Media Lab. Ένας από τους στόχους του Processing είναι να λειτουργήσει ως ένα εργαλείο για άτομα μη οικεία με τον προγραμματισμό, μέσω της πολύ δυνατής οπτικής επικοινωνίας που παρέχει. Η γλώσσα Processing βασίζεται στις γραφικές δυνατότητες της γλώσσας προγραμματισμού Java, απλοποιώντας τα χαρακτηριστικά της και προσθέτοντας καινούργια [33].



Σχήμα 2.14. Το προγραμματιστικό περιβάλλον του Processing

Το προγραμματιστικό περιβάλλον του Processing περιλαμβάνει έναν οργανωτή των σχεδίων, ο οποίος αποτελεί μια εναλλακτική λύση για την οργάνωση σε σχέση με αυτήν του IDE της Java. Το Processing διαθέτει ήδη πάνω από 100 διαφορετικές βιβλιοθήκες, γεγονός που του προσφέρει τεράστια ευελιξία σε πληθώρα εφαρμογών [33].

Κάθε σχέδιο του Processing είναι στην πραγματικότητα μια υποκατηγορία της PApplet Java κλάσης, που αποτελεί και την κλάση η οποία υλοποιεί τα περισσότερα από τα χαρακτηριστικά του Processing. Κατά τον προγραμματισμό σε αυτό το προγραμματιστικό περιβάλλον, όλες οι επιπλέον κλάσεις που καθορίζονται θα πρέπει να αντιμετωπίζονται ως εσωτερικές κλάσεις όταν ο κώδικας μεταφράζεται σε Java πριν από την σύνταξη (compile). Αυτό σημαίνει, ότι η χρήση στατικών μεταβλητών και των μεθόδων των κλάσεων απαγορεύεται, εκτός αν θέσει κανείς ρητά στο Processing ότι ο κώδικας του σχεδίου είναι σε Java. Επιτρέπεται έτσι, σε σύνθετους τύπους δεδομένων να μπορούν να περιλαμβάνουν οποιοδήποτε αριθμό από ορίσματα και αποφεύγονται οι περιορισμοί της αποκλειστικής χρήσης προτύπων τύπων δεδομένων, όπως χρώματος (RGB, HSB) που μάλιστα είναι και τύπος δεδομένων που μας ενδιαφέρει στην εν λόγω πτυχιακή εργασία [33].

2.4.2 Πλεονεκτήματα Processing

Το Processing επιλέχτηκε τελικά για την οπτικοποίηση των δεδομένων από το σύστημα πελματογράφου το οποίο σχεδιάστηκε. Κατ' αρχάς, όπως αναφέρθηκε και νωρίτερα, αποτελεί ένα πρόγραμμα που μαζί με το προγραμματιστικό περιβάλλον του Arduino είναι σχεδιασμένο, ώστε να χρησιμοποιείται από χρήστες που είναι λιγότερο εξοικειωμένοι με προγραμματισμό. Η ευκολία που παρέχει στη σύνταξη κώδικα, στη μεταγλώττιση και στην επικοινωνία με το Arduino είναι ένας από τους βασικότερους λόγους για τον οποίο επελέγη το συγκεκριμένο πρόγραμμα για την οπτικοποίηση των δεδομένων της διάταξης.

Όπως σε όλα τα κομμάτια της εν λόγω κατασκευαστικής εργασίας, έτσι και εδώ καταλήξαμε στην επιλογή του Processing αφού πρώτα δοκιμάσαμε μια σειρά άλλων προγραμμάτων για την οπτικοποίηση των δεδομένων. Ενδεικτικά, αναφέρουμε πως δοκιμάστηκε το προγραμματιστικό περιβάλλον του Arduino, καθώς διαθέτει και μια δυνατότητα οπτικοποίησης, που όμως, όπως αναφέρθηκε και νωρίτερα, είναι πολύ χαμηλών δυνατοτήτων και συγκεκριμένα διαθέτει μόνο την επιλογή οπτικοποίησης μέσω δισδιάστατων γραφικών παραστάσεων που προφανώς δεν επαρκούν. Χρησιμοποιήθηκε και το Matlab, όμως σε αυτή τη περίπτωση οι επιλογές οπτικοποίησης ήταν πολλές, αλλά ο βαθμός δυσκολίας στη σύνταξη του κώδικα και η πολυπλοκότητα, κρίθηκαν δυσανάλογα για το αποτέλεσμα που το Matlab προσέφερε. Η επιλογή του Processing έδωσε εν τέλει, ένα πάρα πολύ ποιοτικό αποτέλεσμα στην οπτικοποίηση, κατάλληλο για την διάταξη της συγκεκριμένης εργασίας.

Κεφάλαιο 3. Αναπτυχθείσα πειραματική διάταξη

3. Αναπτυχθείσα πειραματική διάταξη

3.1 Υλικό

3.1.1 Σύνολο στοιχείων κυκλωματικής διάταξης

Παρακάτω βρίσκεται ο πίνακας (Πίνακας 3.1) που έχει όλα τα εξαρτήματα που χρησιμοποιήσαμε για την υλοποίηση της πειραματικής διάταξης.

Πίνακας 3.1: Μέρη πειραματικής διάταξης

Στοιχείο	Τύπος	Ποσότητα
Μικροελεγκτής	Arduino Uno	1
Αντίσταση	10kΩ	5
Αντίσταση	1kΩ	3
Bluetooth	HC – 05	2
MicroSD θήκη	V1.0 Module	1
Αισθητήρες	FSR	5
Πλακέτα δοκιμών	-	1
Καλώδιο	USB 2.0 Type A/B	1
Μπαταρία	9V	1
Καλώδιο	9V - Arduino	1

Η επιλογή των αισθητήρων δύναμης-πίεσης, έχει δικαιολογηθεί πλήρως στο προηγούμενο κεφάλαιο. Οι αντιστάσεις, αποτελούν συμπληρωματικά στοιχεία, για να μπορεί να σχεδιαστεί κυκλωματικά η διάταξη. Όσο για τα μοντέλα Bluetooth, κάρτας MicroSD, πρόκειται για μοντέλα τα οποία χρησιμοποιούνται κατά κόρον σε εφαρμογές με Arduino. Διαθέτουν πολύ μικρή κατανάλωση και μέγεθος, και επειδή είναι συμβατά πρόσθετα, ήταν εύκολο να ενταχθούν στη διάταξη σε κυκλωματικό αλλά και λογισμικό επίπεδο.

Σημαντικό είναι να προσθέσουμε πως επελέγησαν δύο διαφορετικά μεγέθη αισθητήρων (Σχήμα 3.1). Ο λόγος βρίσκεται στη διαφορετική κατανομή της πίεσης στην πελματιαία επιφάνεια. Συγκεκριμένα, στην πτέρνα ασκείται το μεγαλύτερο ποσοστό πίεσης και γι’ αυτό επιλέχτηκε όπως φαίνεται και στο παρακάτω σχήμα, ένας αισθητήρας μεγαλύτερης επιφάνειας, ώστε να επιτρέπει την ομοιόμορφη κατανομή της πίεσης για τη μέτρηση. Στις υπόλοιπες 4 περιοχές, τοποθετήθηκαν αισθητήρες

δύναμης-πίεσης σε μικρότερο μέγεθος, που όμως θεωρήθηκε επαρκές για την ακριβή μέτρηση της πίεσης.

Σχήμα 3.1. Τα δύο μεγέθη αιθητήρων FSR που επελέγησαν στη διάταξη

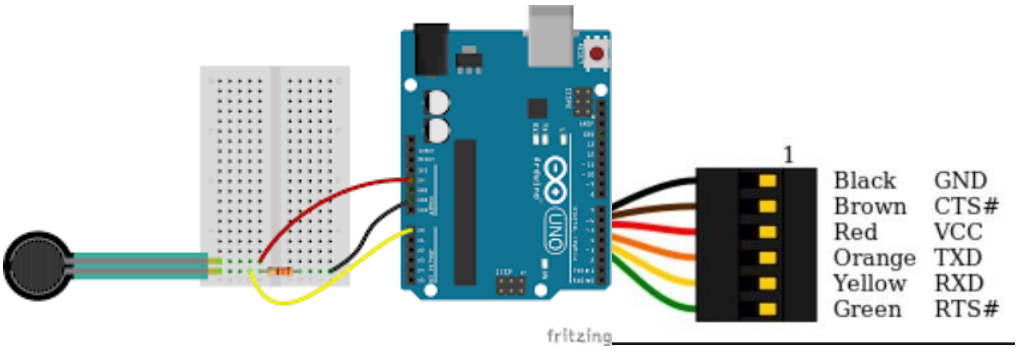
Όσον αφορά το Arduino Uno, οι λόγοι επιλογής του προκύπτουν από την περιγραφή του, που έχει γίνει σε προηγούμενο κεφάλαιο. Σημαντικό πλεονέκτημά του αποτελεί το μέγεθος, αλλά και η τάση λειτουργίας των 5V. Επίσης, έχει χαμηλή κατανάλωση, χαρακτηριστικό πολύ βασικό για τέτοιου είδους διατάξεις. Τέλος, διαθέτει όλα τα pins που είναι απαραίτητα για να ικανοποιούνται όλες οι λειτουργίες που χρειάζονται στη διάταξη.

3.1.2 Διάταξη ηλεκτρονικού κυκλώματος

Χρησιμοποιήθηκε πλακέτα δοκιμών για τη σχεδίαση του κυκλώματος, καθώς πρόκειται για πειραματικό πρωτότυπο μοντέλο. Οι βραχυκυκλωμένες σειρές θέσεων της πλακέτας δοκιμών, με σημάνσεις ‘+’ και ‘ - ’, χρησιμοποιήθηκαν αντίστοιχα για την τροφοδοσία των 5V και τη γείωση. Αρχικά, σε ένα σχέδιο πέλματος τοποθετήθηκαν οι 5 αισθητήρες δύναμης-πίεσης στις περιοχές που επιλέχτηκαν, όπως φαίνεται και στο Σχήμα 2.8. Καθένας από τους πέντε αισθητήρες, διαθέτει δύο άκρα, συνεπώς δύο εξόδους. Με μια καλωδιοταινία, το ένα άκρο του κάθε αισθητήρα οδηγήθηκε σε κοινό σημείο στη τροφοδοσία των 5V του κυκλώματος και το άλλο οδηγήθηκε πάλι σε άλλο κοινό σημείο στη πλακέτα δοκιμών.

Από το δεύτερο σημείο αυτό, οι έξοδοι των πέντε αισθητήρων οδηγήθηκαν αντίστοιχα

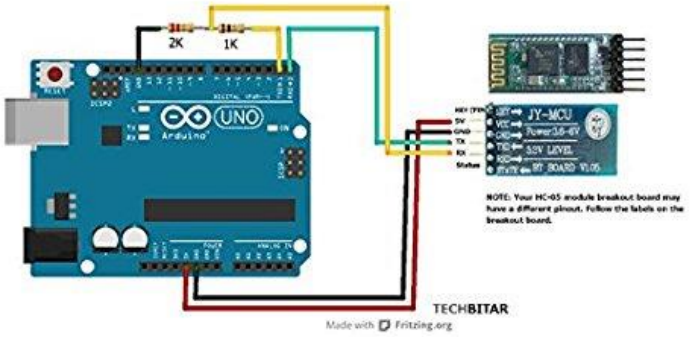
στις πέντε αναλογικές εισόδους A0 ~ A4 του μικροελεγκτή, μέσω πέντε όμοιων αντιστάσεων 10kΩ η καθεμία, οι οποίες με τη σειρά τους είχαν το ένα άκρο γειωμένο και το άλλο στην μη γειωμένη έξοδο κάθε αισθητήρα. Η σύνδεση αυτή φαίνεται στο παρακάτω Σχήμα, για έναν αισθητήρα.



Σχήμα 3.2. Σύνδεση ενός FSR με το Arduino **Σχήμα 3.3.** Σύνδεση USB - Serial

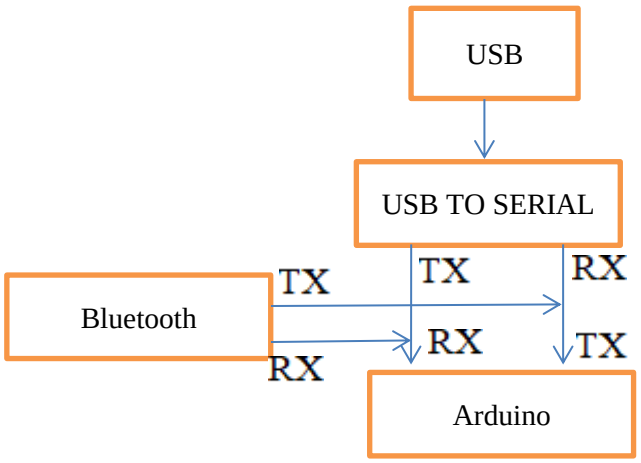
Ύστερα συνδέθηκαν τα δύο πρόσθετα επικοινωνίας μέσω Bluetooth, στο κύκλωμα. Αρχικά, το ένα πρόσθετο μένει μόνιμα συνδεδεμένο στον υπολογιστή μέσω ενός καλωδίου μετατροπής USB σε σειριακή επικοινωνία. Το καλώδιο διαθέτει τέσσερα διαφορετικά μικρότερα καλώδια με θηλυκές άκρες για να συνδεθούν στα pins του Bluetooth. Ο σωστός τρόπος σύνδεσης φαίνεται στο Σχήμα 3.3.

Το δεύτερο πρόσθετο, όπως και το πρώτο, διαθέτει έξι διαφορετικά pins, εκ των οποίων χρησιμοποιήθηκαν τα τέσσερα. Εν προκειμένω, χρησιμοποιήθηκαν οι δύο έξοδοι RX, TX, που προφανώς χρησιμοποιούνται για την σειριακή επικοινωνία, και οι έξοδοι γείωσης και τροφοδοσίας, με προφανή λειτουργία. Η τροφοδοσία οδηγείται με καλώδιο στο αντίστοιχο σημείο της πλακέτας δοκιμών, όπως και η γείωση αντίστοιχα. Η έξοδος RX του Bluetooth, μέσω αντίστασης 1kΩ, οδηγείται στην είσοδο TX του μικροελεγκτή και μέσω δύο αντιστάσεων 1kΩ συνδεδεμένων εν σειρά, οδηγείται στη γείωση. Όσο για την έξοδο TX, αυτή οδηγείται με καλώδιο απευθείας στην είσοδο RX του Arduino.



Σχήμα 3.4. Σύνδεση Bluetooth

Στο σημείο αυτό είναι σημαντικό να σημειωθούν δύο βασικές παρατηρήσεις: Το Arduino διαθέτει εσωτερικά ένα ολοκληρωμένο σειριακής επικοινωνίας, προκειμένου να μεταφράσει τη σειριακή πληροφορία σε USB, το οποίο και χρησιμοποιεί για την επικοινωνία με τον υπολογιστή. Το ίδιο τσιπ χρησιμοποιείται και για να επικοινωνήσει το πρόσθετο Bluetooth με το Arduino. Στην περίπτωση που θέλουμε να φορτώσουμε τον κώδικα στο Arduino ασύρματα μέσω Bluetooth, τότε έχουμε ξανά την παραπάνω συνδεσμολογία, με την διαφορά όμως ότι η έξοδος RX οδηγείται στην είσοδο RX και επίσης η έξοδος TX στην είσοδο TX. Τα παραπάνω εμφανίζονται στο επόμενο διάγραμμα:



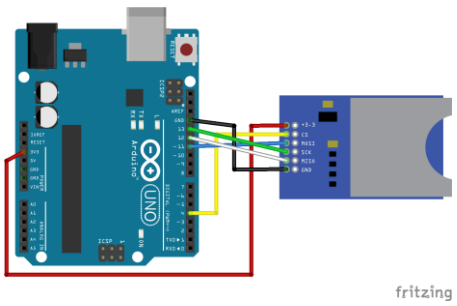
Διάγραμμα 3.1. Συνδεσμολογία Bluetooth και USB to serial με το Arduino

Φαίνεται, λοιπόν, ότι το ολοκληρωμένο σειριακής σε USB επικοινωνίας συνδέει ταυτόχρονα το TX άκρο του με τα RX άκρα και του USB και του Arduino, και ομοίως με το RX. Άρα, όταν αποστέλλεται μια εντολή, την λαμβάνουν ταυτόχρονα το Bluetooth και ο μικροελεγκτής. Επικοινωνεί, δηλαδή, ταυτόχρονα και με τα δύο. Όταν όμως χρειάζεται άμεση επικοινωνία του Arduino με το Bluetooth πρέπει προφανώς η

σύνδεση να αλλάξει.

Επίσης, για την επικοινωνία με το δεύτερο πρόσθετο Bluetooth, το οποίο συνδέεται στον υπολογιστή, απαιτήθηκε να γίνει ζεύξη μεταξύ των δύο. Συνοπτικά, το πρόσθετο που συνδέθηκε στον υπολογιστή χρησιμοποιήθηκε σε προγραμματιστική λειτουργία, και ακολουθώντας τις οδηγίες από το datasheet του προσθέτου, έγινε η σύζευξη των δύο με μια σειρά εντολών στη σειριακή οθόνη του Arduino IDE. Με τη σύζευξη αυτή, κάθε φορά που η διάταξη λαμβάνει τροφοδοσία, αναγνωρίζει αυτόματα το δεύτερο πρόσθετο Bluetooth και δεν υπάρχει ανάγκη για εκ νέου σύνδεση.

Τέλος, συνδέθηκε στη διάταξη και το πρόσθετο για κάρτα μνήμης microSD. Στην μνήμη αυτή αποθηκεύονται αρχεία τύπου .csv με τις μετρήσεις που λαμβάνουμε κάθε φορά από τους αισθητήρες πίεσης, ώστε με κατάλληλο πρόγραμμα να μπορούμε να κάνουμε αξιολόγηση αυτών. Το πρόσθετο της microSD που χρησιμοποιήθηκε διαθέτει 5 υποδοχές. Δύο αντιστοιχούν στη γείωση και στη τροφοδοσία του πρόσθετου, που οδηγούνται με καλώδιο στα αντίστοιχα σημεία της πλακέτας δοκιμών. Τα υπόλοιπα τρία, με σημάνσεις MOSI (master out-slave in), SS (chip select), SCK (serial clock), MISO (master in-slave out) οδηγούνται με καλώδια αντίστοιχα στις ψηφιακές εισόδους 10 ~13 του Arduino, όπως φαίνεται στο Σχήμα 3.5.



Σχήμα 3.5. Σύνδεση προσθέτου microSD με το Arduino

3.1.3 Λειτουργία ηλεκτρονικού κυκλώματος

Αρχικά, σε κατάσταση λειτουργίας της διάταξης, η μπαταρία συνδέεται με το Arduino, μέσω του καλωδίου. Καλώδια από τα pins 5V και GND του Arduino οδηγούνται στα προαναφερθέντα σημεία της πλακέτας δοκιμών, και από εκεί

λειτουργούν σαν τροφοδοσία και γείωση για όλο το κύκλωμα. Σε προγραμματιστική κατάσταση, όταν δηλαδή φορτώνουμε κώδικα στον Arduino, η μπαταρία δεν είναι απαραίτητη, καθώς συνδέεται με καλώδιο USB στον υπολογιστή, ο οποίος παρέχει τελικά την τροφοδοσία στον μικροελεγκτή.

Το βασικό κομμάτι της λειτουργίας, είναι η αντιστοίχιση της πίεσης που ασκείται σε κάθε αισθητήρα, με μια τιμή στην αναλογική είσοδο του μικροελεγκτή. Καθένας από τους πέντε αισθητήρες λοιπόν, ανάλογα με την εφαρμοζόμενη πίεση σε αυτόν αποδίδει την πίεση αυτή σε μια αντίστοιχη τιμή δύναμης σε Newton. Η αντιστοίχιση αυτή γίνεται σε επίπεδο λογισμικού. Αρχικά, δημιουργείται μια τάση, που εν τέλει αντιστοιχίζεται σε μια τιμή δύναμης. Το πώς λειτουργεί αυτό σε επίπεδο ηλεκτρονικής είναι στη πραγματικότητα πολύ απλό. Κάθε σύνδεση αισθητήρα με την αντίστοιχη αναλογική είσοδο του μικροελεγκτή, δημιουργεί έναν διαιρέτη τάσης με είσοδο την τροφοδοσία των 5V και έξοδο την τάση της αντίστασης των 10kΩ, που αντιστοιχεί σε μια αναλογική είσοδο του μικροελεγκτή (Σχήμα 3.6). Από εκεί ο μετατροπέας αναλογικού σήματος σε ψηφιακό, μεταφράζει την τιμή αυτή για να την επεξεργαστεί ο μικροελεγκτής.

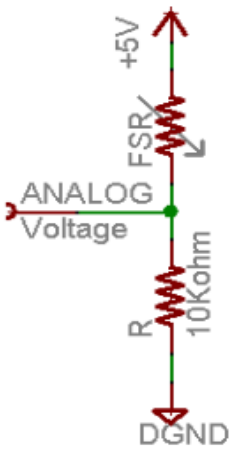
Καθώς η μεταβλητή αντίσταση του αισθητήρα μειώνεται, ανάλογα με την πίεση που δέχεται, η συνολική αντίσταση του αισθητήρα δύναμης-πίεσης και της αντίστροφης αντίστασης μειώνεται από περίπου 100kΩ σε 10kΩ. Αυτό σημαίνει ότι αυξάνεται το ρεύμα που διέρχεται στην αναλογική είσοδο του μικροελεγκτή και από τις δύο αντιστάσεις, γεγονός που με τη σειρά του προκαλεί αύξηση της τάσης κατά μήκος της σταθερής αντίστασης των 10kΩ.

Η τάση εξόδου τελικά, υπολογίζεται από τη σχέση:

$$\boldsymbol{V_{out} = V_{cc}\left(\frac{R}{R + FSR}\right)}$$

Εξ.25

καθώς μιλάμε για έναν διαιρέτη τάσης.



Σχήμα 3.6. Διαιρέτης τάσης στην σύνδεση του αισθητήρα

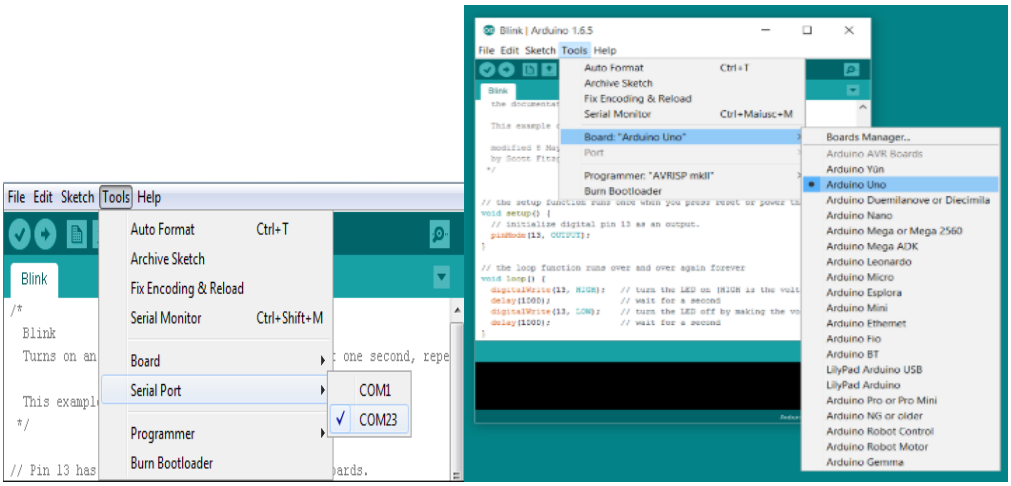
Όσον αφορά τα δύο πρόσθετα Bluetooth, όπως προαναφέρθηκε, το ένα είναι συνεχώς συνδεδεμένο στον υπολογιστή και το δεύτερο συνδέεται με την προαναφερθείσα σύνδεση με το Arduino και επικοινωνούν τελικά μεταξύ τους. Σημαντικό για την επεξήγηση της λειτουργίας του κυκλώματος είναι να αναλύσουμε γιατί χρησιμοποιήθηκαν 2 πρόσθετα Bluetooth, αντί της απευθείας χρήσης του Bluetooth που πιθανώς να διαθέτει ο υπολογιστής. Το Bluetooth λειτουργεί με ένα σύστημα master-slave. Τα προγράμματα οδήγησης του Bluetooth του υπολογιστή δεν επιτρέπουν τη λειτουργία ως master, σε αντίθεση με ένα κινητό τηλέφωνο. Για την ακρίβεια, είναι δυνατή η χρήση του Bluetooth για τη σύζευξη, αλλά το Processing, που χρησιμοποιείται σαν πρόγραμμα οπτικοποίησης των δεδομένων στη διάταξη, δεν έχει πρόσβαση στο ενσωματωμένο Bluetooth του υπολογιστή. Αυτό δεν ισχύει για τα κινητά τηλέφωνα, καθώς η πρόσβαση παρέχεται από έναν ενδιάμεσο κώδικα, αυτόν της εφαρμογής που θα χρησιμοποιείται για την οπτικοποίηση. Έτσι όταν ο πελματογράφος λειτουργεί σε επικοινωνία με τον υπολογιστή, χρειάζονται 2 πρόσθετα, με το ένα να έχει το ρόλο του slave και το άλλο να αναλαμβάνει έναν διπλό ρόλο master και slave, το οποίο αρχικοποιεί και τη σύζευξη των δύο προσθέτων.

Τέλος, απομένει να αναλυθεί και η λειτουργία του προσθέτου κάρτας μνήμης microSD στη διάταξη. Τα pins στα οποία συνδέεται το πρόσθετο της microSD λειτουργούν και σαν ψευδοαναλογικές είσοδοι. Η επικοινωνία ουσιαστικά γίνεται μέσω του κώδικα του Arduino. Κυκλωματικά, αυτό που πρέπει να γίνει είναι να συνδεθεί ο αντάπτορας στα σωστά pins. Όλα τα υπόλοιπα τα αναλαμβάνει ο κώδικας κατά την επικοινωνία.

3.2 Λογισμικό

3.2.1 Κώδικας Arduino IDE (Foot)

Είναι σημαντικό να σημειώσουμε πως, παρόλο που στο κομμάτι αυτό της πτυχιακής θα αναλυθεί ο τρόπος λειτουργίας του κώδικα που δημιουργήθηκε για την λειτουργία της διάταξης, θα υπάρχουν μόνο ενδεικτικά κομμάτια του κώδικα και όχι ολόκληρος ο κώδικας. Αυτός θα υπάρχει σε παράρτημα στο τέλος της συγκεκριμένης εργασίας. Πρώτο βήμα είναι να ορίσουμε στο Arduino IDE κάποια στοιχεία για να μπορούμε να προγραμματίσουμε στον μικροελεγκτή. Συγκεκριμένα, πρέπει να ορίσουμε τη σωστή θύρα USB, στην οποία είναι συνδεδεμένο το Arduino. Ακόμα πρέπει να επιλεχτεί το σωστό μοντέλο πλακέτας, εν προκειμένω το Arduino Uno, τα οποία και τα δύο φαίνονται στο Σχήμα 3.7. Στο πρώτο κομμάτι του κώδικα λοιπόν, πριν από τις δύο βασικές συναρτήσεις που απαιτεί το IDE του Arduino (setup() και loop()), γίνονται οι απαραίτητες αρχικοποιήσεις και ορισμοί των σταθερών και μεταβλητών στις οποίες θα αποθηκεύονται οι τιμές που μετράμε στον κάθε αισθητήρα.



Σχήμα 3.7. Επιλογή θύρας COM και πλακέτας στο Arduino IDE

```
#define HEEL_BONE_SENSOR 0   // Analogue pins used
#define OUTER_BALL_SENSOR1 1
#define OUTER_BALL_SENSOR2 2
#define OUTER_BALL_SENSOR3 3
#define OUTER_BALL_SENSOR4 4

#define HEEL_BONE_INDEX   0   // Analogue pins used
#define OUTER_BALL1_INDEX 1
#define OUTER_BALL1_INDEX 2
#define OUTER_BALL2_INDEX 3
#define OUTER_BALL3_INDEX 4

#define R2 10000


#define FILTER_COEF1 6 //ensure that both sum to 8, increas FILTER_COEF2 value to make the filter storngger.
#define FILTER_COEF2 2
// Variables for storing sensor values
float unHeelBone = 0;
float unOuterBall0 = 0;
float unOuterBall1 = 0;
float unOuterball2 = 0;
float unOuterball3 = 0;

float unHeelBoneOld = 0;
float unOuterBall0Old = 0;
float unOuterBall1Old = 0;
float unOuterball2Old = 0;
float unOuterball3Old = 0;
```

Η setup() συνάρτηση, που τρέχει μόνο μια φορά κάθε φορά που ξεκινά να λειτουργεί το Arduino, εκτελεί μόνο την εντολή serial.begin(9600).

```
void setup()
{

Serial.begin(9600);

}
```

Αυτή θέτει μόνο την τιμή μεταφοράς των δεδομένων από το Arduino στον υπολογιστή στα 9600 bits / δευτερόλεπτο. Σημαντικό είναι, να τεθεί και η ίδια τιμή επικοινωνίας στον υπολογιστή, κάτι το οποίο γίνεται με εντολή στο Arduino IDE. Η τιμή αυτή για τα bits / δευτερόλεπτο επικοινωνίας μεταξύ Arduino και υπολογιστή ονομάζεται baud rate και μπορεί να έχει τις τιμές 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 ή 115200, τιμή που επιλέγεται από τον προγραμματιστή ανάλογα με την εκάστοτε εφαρμογή. Επελέγησαν τα 9600kbps, επειδή προσέφεραν την καλύτερη δυνατή απόκριση στην εφαρμοζόμενη πίεση στους αισθητήρες μου.

Ύστερα, γίνεται μια χαρτογράφηση (mapping) των τιμών που μετρώνται από τους αισθητήρες. Γίνεται, δηλαδή, μια μετατροπή των τιμών που μετρώνται, από ένα σύνολο τιμών (min1, max1) σε ένα νέο σύνολο (min2, max2). Το mapping αυτό απαιτεί μια σειρά από μαθηματικές μετατροπές, η οποία δίνεται από το datasheet των

αισθητήρων δύναμης-πίεσης. Η αντιστοίχιση αυτή, ουσιαστικά μετατρέπει τις τιμές της τάσης εξόδου σε τιμές δύναμης σε Newton. Αυτό γίνεται με υπολογισμό της τάσης, χρησιμοποίηση του υπολογισμού αυτού για να μετρηθεί η αντίσταση και αντιστοίχιση τελικά της αντίστασης αυτής σε μια τιμή δύναμης, βάσει της γραφικής παράστασης που φαίνεται και στο Σχήμα 2.12. Ο υπολογισμός γίνεται βάσει του τύπου που έχει γραφεί παραπάνω, λυμένο τελικά ως προς την μεταβλητή αντίσταση FSR, δηλαδή:

$$\boldsymbol{FSR = \frac{\left((V_{cc} - V)R\right)}{V}}$$

Εξ.26

Υπενθυμίζεται ότι έχουμε σταθερές τις τιμές της V_{cc} = 5V= 5000mV και της αντίστασης R =10kΩ.

```
for (int i = 0; i<5; i++)
{
  mySensorsVolArr[i] = map(analogRead(i),0,1023,0,5000);

}

for (int i = 0; i<5; i++)
{

  mySensorsForceArr[i] = (5000-mySensorsVolArr[i]);
  mySensorsForceArr[i] *=R2;
  mySensorsForceArr[i] /= mySensorsVolArr[i];
//  mySensorsForceArr[i] = 250;
  //Serial.println(mySensorsForceArr[i]);

  mySensorsForceArr[i] = 1000000 / mySensorsForceArr[i];
  mySensorsForceArr[i] = (constrain(mySensorsForceArr[i],1,1000000));

  if (mySensorsForceArr[i] <= 1000)
  {

   mySensorsForceArr[i] = mySensorsForceArr[i] / 80;

  }
  else
  {
  mySensorsForceArr[i] = mySensorsForceArr[i] - 1000;
  mySensorsForceArr[i] /= 30;

  }
}
```

Έπειτα, οι υπολογισμένες τιμές περνούν από ένα αναδρομικό χαμηλοπερατό φίλτρο για να γίνει το σήμα που θα δέχεται η είσοδος από την εφαρμοζόμενη πίεση στον αισθητήρα, πιο ομαλό, με λιγότερες απότομες κορυφές. Αυτό επιτυγχάνεται πολλαπλασιάζοντας την τελευταία τιμή μέτρησης με τέσσερα, προσθέτοντας την

επόμενη, επίσης πολλαπλασιασμένη με τέσσερα και τελικά διαιρώντας το άθροισμα τους με 8. Έτσι, σε τιμές με σχετικά μεγάλη απόκλιση μεταξύ τους η διαφορά αυτή μειώνεται για να μην εμφανίζονται απότομες αλλαγές στο σήμα. Παραδείγματος χάριν, με προηγούμενο σήμα 8N και επόμενο 10N, η φιλτραρισμένη τιμή θα είναι 7.7N ~ 7.5N. Άρα θα αποφευχθεί, αυτό το μεγάλο άλμα και θα πάμε στην τιμή των 10N με 2 βήματα αλλά πιο ομαλά.

```
for (int i = 0; i<5; i++)
{

  mySensorsForceFilteredArr[i] = (mySensorsForceFilteredArr[i]*4 + mySensorsForceArr[i]*4)/8;
}
```

Σε περίπτωση που παρατηρηθεί ότι το σήμα μας έχει περισσότερο θόρυβο, είναι δυνατή η προσαρμογή των συντελεστών βαρύτητας των προηγούμενων και των νέων σημάτων. Φυσικά, όσο μεγαλύτερη αλλαγή πραγματοποιηθεί, τόσο πιο αργή απόκριση θα έχει το σύστημά μας. Γι' αυτό και επιλέχθηκαν οι συγκεκριμένοι συντελεστές βαρύτητας, καθώς εμφάνιζαν άμεση απόκριση και ελάχιστες παραμορφώσεις στο σήμα.

Στο τελικό τμήμα του κώδικα, γίνεται η σειριακή επικοινωνία με τον υπολογιστή. Συγκεκριμένα, προβάλλονται στη σειριακή οθόνη του Arduino IDE τα δεδομένα που λαμβάνονται από τους αισθητήρες, σαν σειρά από έξι bytes, τιμές τις οποίες παίρνει για επεξεργασία ο κώδικας του Processing. Πρώτα, γράφει την τιμή 255 και μετά με τη σειρά τις τιμές σε Newton, των πέντε αισθητήρων.

```
Serial.write(255);
for (int i = 0; i<5; i++)
{
  //Serial.write((i*20)+20);
  //Serial.println(String(int((mySensorsForceArr[i]))));
  Serial.write(int(constrain(mySensorsForceFilteredArr[i],0,98)));

}
```

Ο λόγος που αποστέλλεται πρώτα η τιμή 255 και ύστερα οι τιμές για τους πέντε αισθητήρες έχει να κάνει με την επικοινωνία του συγκεκριμένου προγράμματος, με το Processing. Το Arduino στέλνει δεδομένα συνεχώς. Επομένως, ο κώδικας του Processing δεν μπορεί να γνωρίζει πότε πρέπει να λάβει τα δεδομένα αυτά σαν τιμές πίεσης για να τα επεξεργαστεί περαιτέρω. Συνεπώς, αποστέλλεται ο χαρακτήρας 255 ακριβώς πριν αποσταλούν σειριακά οι μετρήσεις. Μόλις αναγνωρίσει το 255, ο κώδικας του Processing αναμένει πως τα πέντε bytes που ακολουθούν αποτελούν τις

μετρήσεις από τους αισθητήρες. Το 255 επιλέχτηκε τυχαία, καθώς θα μπορούσε να χρησιμοποιηθεί οποιοσδήποτε άλλος χαρακτήρας σαν σήμα, μεταξύ των τιμών 99 και 255. Δεν μπορεί να χρησιμοποιηθεί τιμή από 0 έως 98 επειδή αυτές χρησιμοποιούνται για τις τιμές της δύναμης σε Newton. Για την περίπτωση πιο περίπλοκων μετρήσεων, εάν είχαμε για παράδειγμα μεγαλύτερο εύρος τιμών σε Newton για κάθε αισθητήρα, θα μπορούσαμε να χρησιμοποιήσουμε πληροφορία των 8 byte.

3.2.2 Κώδικας Processing

Ο κώδικας που έχει αναπτυχθεί στο Processing διαβάζει τις τιμές που εμφανίστηκαν στη σειριακή οθόνη του Arduino IDE και τις επεξεργάζεται κατάλληλα για να τις οπτικοποιήσει σε μια μορφή χάρτη με χρωματική κλίμακα, ανάλογα με την εφαρμοζόμενη πίεση, αλλά και τις αποθηκεύει σε μια εξωτερική κάρτα μνήμης, ώστε με άλλο κώδικα που αναπτύχθηκε, να μπορούν να αξιολογηθούν.

3.2.3 Κώδικας οπτικοποίησης δεδομένων πίεσης (FootGraphics)

Αρχικά, χρησιμοποιήθηκε μια κλάση με ονομασία Blob που έπαιξε βασικό ρόλο στον κώδικα που αναπτύχθηκε [33]. Για την οπτικοποίηση των δεδομένων, καταλήξαμε πως μια απεικόνιση όπως αυτή του Σχήματος 2.6 είναι ιδανική, ώστε να προσομοιωθεί ένας χάρτης με χρωματική κλίμακα, ανάλογη της εφαρμοζόμενης δύναμης στον αισθητήρα,. Κάθε ένας από αυτούς τους κύκλους αλλάζει το χρώμα των γύρω pixels, ανάλογα με την αύξηση της εφαρμοζόμενη πίεσης. Αρχικά λοιπόν, δημιουργήθηκε μια κλάση σαν ένα διάνυσμα 2 διαστάσεων, που ορίζει τους κύκλους αυτούς και ονομάστηκε Blob. Πρόκειται για κύκλους σε ένα δισδιάστατο παράθυρο του υπολογιστή, το οποίο ονομάζουμε καμβά. Αρχική τιμή της ακτίνας του κύκλου αυτού, ορίστηκε η μηδενική, καθώς αρχικά σε όλους τους αισθητήρες δεν εφαρμόζεται καθόλου πίεση.

```
class Blob {

  PVector pos;

  float r;

  PVector vel;

  Blob(float x, float y) {

    pos = new PVector(x, y);

    vel = PVector.random2D();

    vel.mult(random(0, 0));

   r = 0;

  }
```

Αφού ορίσαμε τους κύκλους έπρεπε να δημιουργηθεί μια συνάρτηση που θα τους σχεδιάζει. Η λογική της σχεδίασης είναι αρκετά εύκολη. Σκεφτόμαστε αρχικά την εικόνα πάνω στην οποία θέλουμε να σχεδιάσουμε τους κύκλους σαν έναν καμβά, στον οποίο κάθε σημείο ορίζεται από συντεταγμένες (x, y). Στον καμβά αυτόν το χρώμα καθενός pixel αντιστοιχεί σε μια συνάρτηση των x και y. Πρέπει, λοιπόν, η σχεδιαστική συνάρτηση να σχεδιάζει κάθε φορά τον κύκλο στο σημείο που ορίζεται από τις συντεταγμένες του κάθε αισθητήρα πάνω στον καμβά. Αρχικοποιούμε στην συνάρτηση κάποια δεδομένα απαραίτητα για τη σχεδίαση σχημάτων στο Processing και συγκεκριμένα το χρώμα και το πάχος του συνόρου του σχήματος που σχεδιάζουμε.

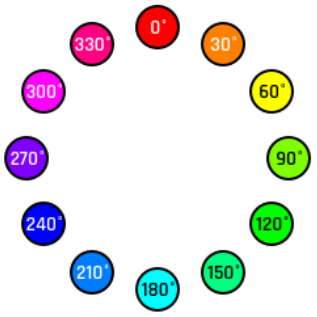
```
  void show() {

    noFill();

    stroke(0);

    strokeWeight(1);

    ellipse(pos.x, pos.y, r*2, r*2);
    text(str(r)+"N",pos.x+r/4,pos.y-r/2);

  }
```

Στον βασικό κώδικα οπτικοποίησης, λοιπόν, αρχικοποιούμε, ορίζουμε και δημιουργούμε στην αρχή το σύνολο των μεταβλητών που θα χρησιμοποιήσουμε. Συγκεκριμένα, δημιουργούμε πέντε αντικείμενα της κλάσης Blob όπως αυτή επεξηγήθηκε παραπάνω και ορίζουμε πέντε ακέραιες μεταβλητές για να

χρησιμοποιούμε για τις τιμές των πέντε αισθητήρων. Ορίζουμε ακόμα την μεταβλητή εξόδου, όπου θα εμφανίζονται τελικά τα δεδομένα που θέλουμε να οπτικοποιήσουμε και τον αριθμό του δείγματος των μετρήσεων που παίρνουμε. Δημιουργούμε, τέλος, ένα αντικείμενο σειριακής κλάσης για την επικοινωνία με την σειριακή οθόνη του Arduino IDE.

```
PImage bg;
int y;
import processing.serial.*;

int s1;
int s2;
int s3;
int s4;
int s5;
PShader blur;
Serial myPort;  // Create object from Serial class
int val;      // Data received from the serial port
int number=0;
int force;
Blob[] blobs = new Blob[5];
PShape foot;
int sample = 0;
PrintWriter output;
```

Το επόμενο κομμάτι είναι η συνάρτηση setup() που εκτελείται μια φορά, κάθε φορά που ανοίγουμε και εκτελούμε τον κώδικα του Processing. Σε αυτήν αρχικοποιούμε κάποια βασικά τμήματα της οπτικοποίησης. Ορίζουμε αρχικά το μέγεθος του καμβά πάνω στο οποίο θα οπτικοποιήσουμε τα δεδομένα, φροντίζοντας να ταιριάζει η εικόνα του πέλματος που θα χρησιμοποιηθεί. Ορίζουμε ακόμα, τον αριθμό των εικόνων που θα μπορεί ο καμβάς αυτός να αναγνωρίζει ανά δευτερόλεπτο (frame rate) και ακόμα το πώς θα αντιλαμβάνεται και θα επεξεργάζεται τις τιμές των pixels (τα χρώματα σε κάθε σημείο (x,y) του καμβά δηλαδή), επιλέγοντας την κωδικοποίηση μεταβλητών HSB.



Είναι σημαντικό να δοθεί μια επεξήγηση της κωδικοποίησης χρωματισμού pixels, HSB. Είναι τα αρχικά των λέξεων: “Hew”, “Saturation” και “Brightness”, δηλαδή “Λάμψη”, “Κορεσμός” και “Φωτεινότητα”. Κάθε χρώμα ενός pixel αντιστοιχεί σε τρείς τιμές. Το H παίρνει τιμές από 0 έως 360 αντιστοιχίζοντας κάθε τιμή σε ένα χρώμα του φάσματος.

Το S παίρνει τιμές από 0 έως 100 και προσαρμόζει το πόσο έντονο ή απαλό θα είναι το χρώμα που καθορίστηκε από το H. Εάν για παράδειγμα το H έδειξε το κόκκινο χρώμα, με τιμή του S, 100, θα πάρουμε το πιο έντονο κόκκινο που είναι δυνατόν και με τιμή μηδέν το πιο απαλό, ουσιαστικά γκρι.

Τέλος, το B χρησιμοποιεί επίσης το χρώμα που ορίστηκε από το H και ορίζει τη φωτεινότητά του. Όσο αυξάνεται η τιμή του, τόσο πιο φωτεινό είναι και το χρώμα που έχει οριστεί, με όρια το λευκό και το μαύρο.

Φορτώνουμε, λοιπόν, ένα αρχείο τύπου .svg το οποίο είναι και η εικόνα του πέλματος πάνω στην οποία οπτικοποιούνται οι πελματιαίες πιέσεις (Σχήμα 2.8) και δημιουργούμε πέντε αντικείμενα blobs στις συντεταγμένες της εικόνας που αντιστοιχούν στην θέση όπου έχουν τοποθετηθεί οι πραγματικοί αισθητήρες στη διάταξη.

```
void setup() {
  frameRate(50);
  size(480, 795);
  colorMode(HSB);

  String portName = Serial.list()[0];
  myPort = new Serial(this, portName, 9600);

  // The background image must be the same size as the parameters
  // into the size() method. In this program, the size of the image
  // is 640 x 360 pixels.
    blobs[0] = new Blob(270-100,100);
    blobs[1] = new Blob(270-100,290);
    blobs[2] = new Blob(375-100,290);
    blobs[3] = new Blob(475-100,320);
    blobs[4] = new Blob(385-100,660);

  //bg = loadImage("foot.png");
  //background(bg);
  foot = loadShape("foot.svg");
```

Προφανώς μπορούν να προστεθούν όσα ακόμα στοιχεία κλάσης blob θέλουμε, σε περίπτωση που σχεδιάσουμε μια διάταξη με περισσότερους αισθητήρες. Ακόμα, με μια απλή αλλαγή των συντεταγμένων των στοιχείων που έχουμε δημιουργήσει μπορούμε να μετακινήσουμε την σχεδίαση του blob σε οποιοδήποτε σημείο της εικόνας θέλουμε, εάν αντίστοιχα έχουμε μετακινήσει έναν αισθητήρα στην πραγματική διάταξη. Επίσης, η τιμή του frame rate είναι προσαρμόσιμη στις ανάγκες της εκάστοτε εφαρμογής. Στον κώδικα που σχεδιάστηκε και για τις απαιτήσεις της

οπτικοποίησης, οι 50 εικόνες ανά δευτερόλεπτο ήταν επαρκείς. Σε περίπτωση όμως, πιο απαιτητικών μετρήσεων, εύκολα ο αριθμός αυτός μπορεί να αλλάξει.

Στο τελικό κομμάτι της setup(), δημιουργούμε έναν φάκελο τύπου .csv, ώστε να αποθηκεύονται τα δεδομένα των πιέσεων για κάθε αισθητήρα.

```
int m = minute();  // Values from 0 - 59
int h = hour();    // Values from 0 - 23
int d = day();
int mm = month();
String fileName = "Gait_"+str(mm)+"_"+str(d)+"_"+str(h)+"_"+str(m)+".csv";
output = createWriter(fileName);
output.println("Time, Sample number, Sensor1, Sensor2, Sensor3, Sensor4, Sensor5");
output.flush(); // Writes the remaining data to the file
```

Αυτό το τελευταίο κομμάτι είναι βασικό για τον επόμενο κώδικα που έχει αναπτυχθεί, ο οποίος λαμβάνει τις τιμές αυτές από το αρχείο .csv για ανάλυση και αξιολόγηση.

Το επόμενο, είναι το βασικό μέρος του κώδικα. Είναι η συνάρτηση που τρέχει διαρκώς, παίρνει σειριακά τα δεδομένα από το Arduino και σε πραγματικό χρόνο, τα οπτικοποιεί πάνω στην εικόνα της πελματιαίας επιφάνειας.

Σχήμα 3.8. Η εικόνα πέλματος που χρησιμοποιεί ο κώδικας

Αρχικά, όπως αναφέρθηκε, ο τρόπος με τον οποίο θέλουμε να σχεδιάζεται το blob είναι η τιμή κάθε pixel του καμβά να είναι μια συνάρτηση των συντεταγμένων (x, y) κάθε σημείου του καμβά. Το μοντέλο μπορεί να εξελιχθεί ως εξής: Τα blobs να σχεδιάζονται και να τοποθετούνται στα σωστά σημεία της εικόνας του πέλματος στην

αρχή και όσο εφαρμόζεται πίεση σε κάποιον αισθητήρα, να μεγαλώνει η ακτίνα του αντίστοιχου blob, αλλά και να αλλάζουν τα χρώματα γύρω από την ακτίνα. Ουσιαστικά, χρειαζόμαστε τις συντεταγμένες του καμβά εντός της εικόνας του πέλματος. Τελικά, πρέπει να δημιουργηθεί μια συνάρτηση που θα προσαρμόζει τα χρώματα, συγκρίνοντας την απόσταση των σημείων αυτών με τις συντεταγμένες του blob, αλλά και ανάλογα με την ακτίνα του.

Ακόμα, πρέπει να οπτικοποιείται το εξής φαινόμενο της πελματιαίας πίεσης: Όταν η πίεση σε δύο διαφορετικά σημεία του πέλματος, τα οποία βρίσκονται κοντά είναι μεγάλη, στην περιοχή που βρίσκεται μεταξύ τους εμφανίζεται ένα άθροισμα των πιέσεων και από τα δύο σημεία. Πρέπει, λοιπόν, η συνάρτηση που αναφέραμε να αλλάζει τα χρώματα και βάσει ενός τέτοιου αθροίσματος, εάν με την αυξημένη εφαρμοζόμενη πίεση σε διαφορετικά blobs μεγαλώσουν τόσο οι ακτίνες τους, που οι κύκλοι μεταξύ τους να τέμνονται. Το επόμενο κομμάτι του κώδικα, ουσιαστικά υλοποιεί αυτή τη συνάρτηση σχεδίασης.

```
void draw() {
  background(51);
  shape(foot,-100,30,630,790);
  loadPixels();
  for (int x = 50; x < width-50; x++) {
    for (int y = 20; y < height-10; y++) {
      int index = x + y * width;
      float sum = 0;
      for (Blob b : blobs) {
        float d = dist(x, y, b.pos.x, b.pos.y);
        sum += 30 * b.r / d;
      }
      if(pixels[index] >= color(1,1,1))
      {
      pixels[index] = color(sum, 255, 255);
      }
    }
  }
```

Αναλυτικότερα, ο παραπάνω κώδικας αρχικά χρωματίζει το φόντο του καμβά και τοποθετεί στο κέντρο την εικόνα του πέλματος, στο σημείο που θέλουμε. Λαμβάνει το σύνολο των συντεταγμένων που μας ενδιαφέρουν, εσωτερικά της εικόνας του πέλματος δηλαδή, χρησιμοποιώντας έναν διπλό βρόχο για τις x και y συντεταγμένες αντίστοιχα. Δημιουργεί ένα διάνυσμα, με τιμές για κάθε δυνατό συνδυασμό των x και y, όπου κάθε μια αποτελεί συνάρτηση μόνο των συντεταγμένων x και y. Αν δεν ασκηθεί καμία πίεση στην πελματιαία επιφάνεια, απλά χρωματίζει όλα τα pixels που

αντιστοιχούν στις τιμές αυτές με κόκκινο χρώμα.

Στον διπλό βρόχο υπολογίζει, ακόμα, την απόσταση μεταξύ των x και y και των συντεταγμένων του κάθε blob και προσθέτει σε ένα άθροισμα μια συνάρτηση της αντίστασης αυτής και της ακτίνας του αντίστοιχου blob. Όταν ασκείται λοιπόν πίεση σε κάποιον αισθητήρα δεν χρησιμοποιεί το απλό διάνυσμα που αναφέρθηκε νωρίτερα, αλλά προσαρμόζει τα χρώματα γύρω από την περιοχή αυτή βάσει της νέας αυτής συνάρτησης, με το σύστημα χρωμάτων HSB.

Κάποιος μπορεί να παρατηρήσει πως η τιμή της ακτίνας κάθε blob παραμένει μηδέν συνεχώς. Αυτό όμως που γίνεται είναι ότι οι τιμές των ακτινών αυτές, αντιστοιχίζονται μέσω της σειριακής επικοινωνίας του Processing με το Arduino IDE, στις τιμές της εφαρμοζόμενης δύναμης σε κάθε αισθητήρα.

```
  if ( myPort.available() >= 6) {  // If data is available,

while(myPort.read()!=255);

blobs[0].r = myPort.read();         // read it and store it in val
blobs[1].r = myPort.read();
blobs[2].r = myPort.read();
blobs[3].r = myPort.read();
blobs[4].r = myPort.read();
String time = str(millis());
myPort.clear();
output.println(time +", "+str(sample)+", "+str(blobs[0].r)+", "+str(blobs[1].r)+", "+str(blobs[2].r)+", "+str(blobs[3].r)+", "+str(blobs[4].r));
output.flush(); // Writes the remaining data to the file

sample++;


}
```

Μπορεί να προσέξει κανείς ακόμα, πως ο κώδικας εκτελεί δύο ελέγχους πριν διαβάσει από την σειριακή επικοινωνία. Πρώτα ελέγχει αν υπάρχουν διαθέσιμα δεδομένα προς ανάγνωση, και μετά περιμένει πρώτα να λάβει τον χαρακτήρα 255, που όπως σημειώσαμε νωρίτερα είναι η σήμανση πως τα επόμενα 5 bytes είναι τιμές που αντιστοιχούν σε τιμές πίεσης. Τα πέντε επόμενα bytes, διαβάζονται και γράφονται και στο αρχείο .csv.

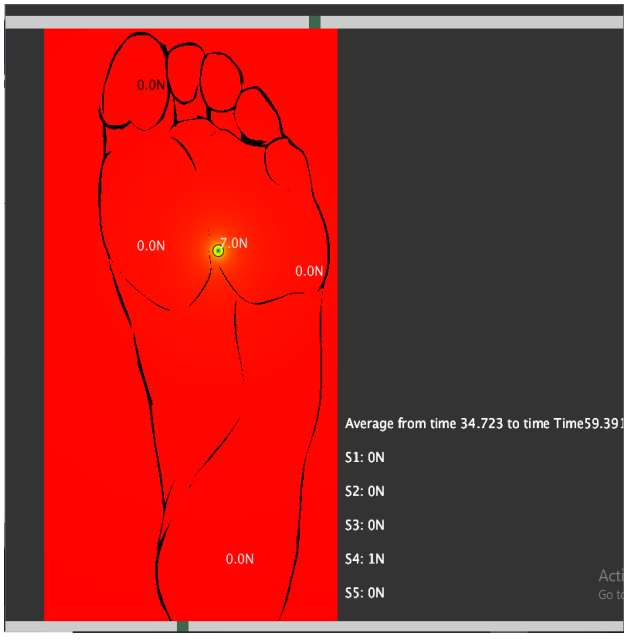
Τελικά, οι τιμές των blobs και των pixels ανανεώνονται βάσει των πιέσεων που δέχτηκαν οι αισθητήρες και με τα blobs σχεδιάζονται στην εικόνα του πέλματος.

```
updatePixels();
println(frameRate);
for (Blob b : blobs) {
  b.update();
  b.show();
```

Όλη αυτή διαδικασία φυσικά, τρέχει συνεχόμενα και η σχεδίαση γίνεται με ταχύτητα 50 εικόνων ανά δευτερόλεπτο ώστε να μπορούμε να έχουμε την οπτικοποίηση τελικά των πελματιαίων πιέσεων με μια απόκριση πραγματικού χρόνου, που είναι το ζητούμενο.

3.2.4 Κώδικας αξιολόγησης δεδομένων πίεσης (Analytics)

Η βασική λειτουργία του δεύτερου κώδικα με το Processing, είναι να λαμβάνει δεδομένα από το αρχείο .csv στο οποίο έχουμε αποθηκεύσει τις μετρήσεις μας καθ' όλη τη διάρκεια ενός πελματογραφήματος και να μπορεί να τις παρουσιάζει στο σύνολό τους, ώστε να δίνεται η ευκαιρία να αξιολογηθούν ακόμα και πολύ αργότερα της εξέτασης.



Σχήμα 3.9. Πρόγραμμα Analytics

Όπως και στον προηγούμενο κώδικα Processing, χρησιμοποιήθηκε η κλάση Blob, αλλά και μια ακόμα κλάση με όνομα HScrollbar. Λειτουργία της κλάσης αυτής είναι

η δημιουργία δύο μπαρών και μέσα σε αυτές δύο γραμμών κύλισης βάσει των οποίων θα μπορούμε να κοιτάμε τα δεδομένα που έχουμε από μια μέτρηση, από το σημείο εκκίνησης το οποίο θέλουμε και για όσο διάστημα από την μέτρηση θέλουμε (Σχήμα 3.9).

Σε πρώτη φάση ορίζουμε όλα τα χαρακτηριστικά της κλάσης όπως ύψος, πάχος, μέγιστη και ελάχιστη τιμή της γραμμής κύλισης, αν έχει επιλεχτεί από τον κέρσορα κ.ο.κ. Δημιουργούμε ύστερα, μια μπάρα της κλάσης HScrollbar, βάσει μεταβλητών που παίρνουμε από τον κυρίως κώδικα και τα αντιστοιχούμε στα βασικά χαρακτηριστικά που ορίσαμε νωρίτερα, για την μπάρα που μόλις δημιουργήθηκε.

```
class HScrollbar {
  int swidth, sheight;    // width and height of bar
  float xpos, ypos;       // x and y position of bar
  float spos, newspos;    // x position of slider
  float sposMin, sposMax; // max and min values of slider
  int loose;              // how loose/heavy
  boolean over;           // is the mouse over the slider
  boolean locked;
  float ratio;

  HScrollbar (float xp, float yp, int sw, int sh, int l) {
    swidth = sw;
    sheight = sh;
    int widthtoheight = sw - sh;
    ratio = (float)sw / (float)widthtoheight;
    xpos = xp;
    ypos = yp-sheight/2;
    spos = xpos + swidth/2 - sheight/2;
    newspos = spos;
    sposMin = xpos;
    sposMax = xpos + swidth - sheight;
    loose = l;
  }
```

Σχεδιάστηκαν ύστερα μια σειρά από συναρτήσεις που χρησιμοποιεί ο βασικός κώδικας, ώστε να πραγματοποιεί απαραίτητες λειτουργίες. Συγκεκριμένα, οι συναρτήσεις αφορούν την ανανέωση των μεταβλητών χαρακτηριστικών της γραμμής κύλισης της μπάρας, όταν μετακινούμε τον κέρσορα πάνω σε αυτήν. Αλλά και την συνάρτηση για την οπτικοποίηση τη γραμμής κύλισης και όλων των ενεργειών που γίνονται πάνω σε αυτήν.

Δημιουργήθηκαν λοιπόν, οι συναρτήσεις overEvent(), update(), getPos() και setPos() που ασχολούνται με τη θέση του δείκτη του ποντικιού, αν αυτός αλλάζει τελικά την θέση της γραμμής κύλισης σε κάθε μπάρα και την ανανέωση της θέσης αυτής αν

πρέπει. Η overEvent(), ελέγχει αν ο δείκτης του ποντικιού είναι μέσα στο πεδίο της μπάρας και αν είναι αρχικοποιεί μια boolean μεταβλητή σε true. Διαφορετικά την αρχικοποιεί false.

Η update() παίρνει το αποτέλεσμα της overEvent() και σε περίπτωση που ο κέρσορας βρίσκεται στο πεδίο της μπάρας, τότε ελέγχει αν έχει πατηθεί το ποντίκι και αν ναι μετακινεί τη γραμμή κύλισης στην σωστή θέση. Ανανεώνει τελικά τα όρια της γραμμής κύλισης, βάσει της νέας θέσης της.

```
void update() {
  if (overEvent()) {
    over = true;
  } else {
    over = false;
  }
  if (mousePressed && over) {
    locked = true;
  }
  if (!mousePressed) {
    locked = false;
  }
  if (locked) {
    newspos = constrain(mouseX-sheight/2, sposMin, sposMax);
  }
  if (abs(newspos - spos) > 1) {
    spos = spos + (newspos-spos)/loose;
  }
}

float constrain(float val, float minv, float maxv) {
  return min(max(val, minv), maxv);
}

boolean overEvent() {
  if (mouseX > xpos && mouseX < xpos+swidth &&
     mouseY > ypos && mouseY < ypos+sheight) {
    return true;
  } else {
    return false;
  }
```

Όσο για τις συναρτήσεις getPos() και setPos(), η πρώτη αναλαμβάνει να μετατρέψει τη θέση της γραμμής κύλισης από μια δυάδα δισδιάστατων δεδομένων (x,y) που αντιστοιχούν στο σημείο που βρίσκεται στον καμβά, σε μια τιμή μεταξύ του 0 και του μέγιστου άκρου της γραμμής κύλισης σε πάχος. Και η δεύτερη θέτει τη τιμή αυτή στη μεταβλητή που χρησιμοποιεί η συνάρτηση update() για να πραγματοποιήσει τις προαναφερθείσες λειτουργίες της.

```
float getPos() {
  // Convert spos to be values between
  // 0 and the total width of the scrollbar
  return spos * ratio;
}
void setPos(int pos) {
  // Convert spos to be values between
  // 0 and the total width of the scrollbar
  spos = pos;
}
```

Τέλος, υπάρχει και η συνάρτηση display(), που ασχολείται με την οπτικοποίηση τόσο της μπάρας όσο και της γραμμής κύλισης μέσα σε αυτή. Η συνάρτηση αυτή καθορίζει αρχικά, κάποια βασικά χαρακτηριστικά της μπάρας, όπως το γκρι χρώμα και το γεγονός ότι δεν έχει σχεδιασμένα τα σύνορα της. Και βέβαια δημιουργεί και το σχήμα της, ένα ορθογώνιο παραλληλόγραμμο δηλαδή στο ύψος της εικόνας του πέλματος που τη θέλουμε. Αλλάζει μετά το χρώμα της και εμφανίζει και το σχήμα της γραμμής κύλισης μέσα στη μπάρα, ανάλογα με το που βρίσκεται ο κέρσορας του ποντικιού και αν πατήθηκε.

```
void display() {
  noStroke();
  fill(204);
  rect(xpos, ypos, swidth, sheight);
  if (over || locked) {
    fill(0, 0, 0);
  } else {
    fill(102, 102, 102);
  }
  rect(spos, ypos, sheight, sheight);
}
```

Ο κύριος κώδικας δεν διαφέρει πολύ σε λογική σε σχέση με αυτόν της οπτικοποίησης των δεδομένων. Λαμβάνει τα δεδομένα από τα αποθηκευμένα αρχεία .csv και όχι φυσικά από την σειριακή οθόνη του Arduino IDE. Χρησιμοποιεί πίνακες λοιπόν, για όλα τα δεδομένα από τους πέντε αισθητήρες. Επίσης, εκτελεί και όλες τις ενέργειες για τις 2 μπάρες και τις αντίστοιχες γραμμές κύλισης τους.

Σημαντικό είναι να προστεθεί πως αφού αλλάζουμε εμείς τα δεδομένα που βλέπουμε μέσω της μιας μπάρας και δεν πρόκειται για δεδομένα που προέρχονται από μια πραγματικού χρόνου μέτρηση πελματογραφήματος, είναι ικανοποιητικό ένα χαμηλότερο frame rate από ότι χρησιμοποιήσαμε στον προηγούμενο κώδικα.

Αρχικά λοιπόν ο βασικός κώδικας. εκτελεί αρχικοποίηση όλων των μεταβλητών και πινάκων που θα χρησιμοποιεί, ορίζοντας και όσα αντικείμενα θα χρησιμοποιήσει από τις δύο κλάσεις που σχεδιάστηκαν. Ορίζει τέλος και το αρχείο .csv με το οποίο θα επικοινωνεί.

```
PImage bg;
int y;
HScrollbar hs1;
HScrollbar hs2;
import processing.serial.*;
Table table;

int [] sensorAvg = new int[5];
int [] sensorAvgCurrent = new int[5];

PShader blur;
Serial myPort;  // Create object from Serial class
int val;      // Data received from the serial port
int number=0;
int force;
Blob[] blobs = new Blob[5];
PShape foot;
int sample = 0;
PrintWriter output;
int rows;
int newScrollBarPos;
int newScrollBarPos2;

String fileName;
```

Ακολουθεί η setup() συνάρτηση. Αρχικά ορίζουμε πάλι τα βασικά, δηλαδή τον χρωματικό κώδικα που χρησιμοποιούμε (HSB), το μέγεθος του καμβά, το κεντράρισμα της εικόνας και το χρώμα στο φόντο του καμβά. Μετά ζητείται από τον χρήστη να επιλεγεί το αρχείο .csv με το οποίο ο κώδικας θα επικοινωνεί. Δημιουργούνται ομοίως με πριν, τα πέντε αντικείμενα blobs, φορτώνεται η εικόνα του πέλματος και επιπρόσθετα δημιουργούνται και οι δύο μπάρες με τις αντίστοιχες γραμμές κύλισης.

```
selectInput("Select a file to process:", "fileSelected");

while(fileName == null)
{
 delay(500);
 println("here");
}
//String portName = Serial.list()[0];
//myPort = new Serial(this, portName, 9600);
//selectInput("Select a file to process:", "fileSelected");
```

```
  blobs[0] = new Blob(270-100,100);
  blobs[1] = new Blob(270-100,290);
  blobs[2] = new Blob(375-100,290);
  blobs[3] = new Blob(475-100,320);
  blobs[4] = new Blob(385-100,660);

foot = loadShape("foot.svg");


hs2 = new HScrollbar(0, height-60, width, 15, 15);
hs1 = new HScrollbar(0, 20, width, 15, 15);
```

Για το κομμάτι της επιλογής του αρχείου .csv σχεδιάστηκε μια μικρή συνάρτηση η οποία αναμένει την επιλογή του χρήστη στο αρχείο. Εάν αυτή η επιλογή είναι κενή εμφανίζει αντίστοιχο μήνυμα troubleshooting. Διαφορετικά, εμφανίζεται πάλι μήνυμα με την επιλογή του αρχείου που έγινε και το πρόγραμμα αποθηκεύει το μονοπάτι μέχρι το αρχείο αυτό και κατά συνέπεια μέσω αυτού, το όνομα του αρχείου, ώστε να γνωρίζει από πού να πάρει τις πληροφορίες που χρειάζεται.

```
void fileSelected(File selection) {
  if (selection == null) {
    println("Window was closed or the user hit cancel.");
  } else {
    println("User selected " + selection.getAbsolutePath());
    fileName = selection.getAbsolutePath();
  }
```

Ύστερα, δημιουργείται ένας πίνακας στον οποίο εισάγονται οι τιμές των μετρήσεων από το αρχείο .csv, διαμορφώνοντάς τον κατάλληλα σε μέγεθος. Υπολογίζεται βάσει των τιμών αυτών σε κάθε στιγμή μέτρησης μια μέση τιμή πίεσης στην εκάστοτε περιοχή και αυτή είναι και η τιμή που εμφανίζεται στο πρόγραμμα αξιολόγησης.

```
table = loadTable(fileName, "csv");
println(table.getRowCount() + " total rows in table");
println(table.getColumnCount());
rows = table.getRowCount();

  textSize(16);

  long sum = 0L;

  for(int i = 0; i < 5; i++)
  {
  sum = 0L;
  for (TableRow row : table.rows()) {
  sum += int(row.getFloat(i+2));
  }
  sensorAvg[i] = int(sum/rows);
  }
```

Στην συνάρτηση που επαναλαμβάνεται συνεχώς, έχουμε αρχικά πάλι κάποιες βασικές

αρχικοποιήσεις και μετά εκτελείται η λογική σχεδίασης των blobs με τον ίδιο τρόπο ακριβώς όπως κάναμε και στον προηγούμενο κώδικα με μια ειδοποιό διαφορά. Πλέον, οι τιμές ακτίνων των κύκλων διαβάζονται από έναν πίνακα αρχείου .csv και όχι σε πραγματικό χρόνο, ανάλογα με την εφαρμοζόμενη πίεση στους πέντε αισθητήρες.

```
void draw() {
  background(51);
  frameRate(30);
  shape(foot,-100,30,630,790);
  loadPixels();
  for (int x = 50; x < 480-50; x++) {
    for (int y = 20; y < height-10; y++) {
      int index = x + y * width;
      float sum = 0;
      for (Blob b : blobs) {
        float d = dist(x, y, b.pos.x, b.pos.y);
        sum += 30 * b.r / d;
      }
      if(pixels[index] >= color(1,1,1))
      {
      pixels[index] = color(sum, 255, 255);
      }
    }
  }
```

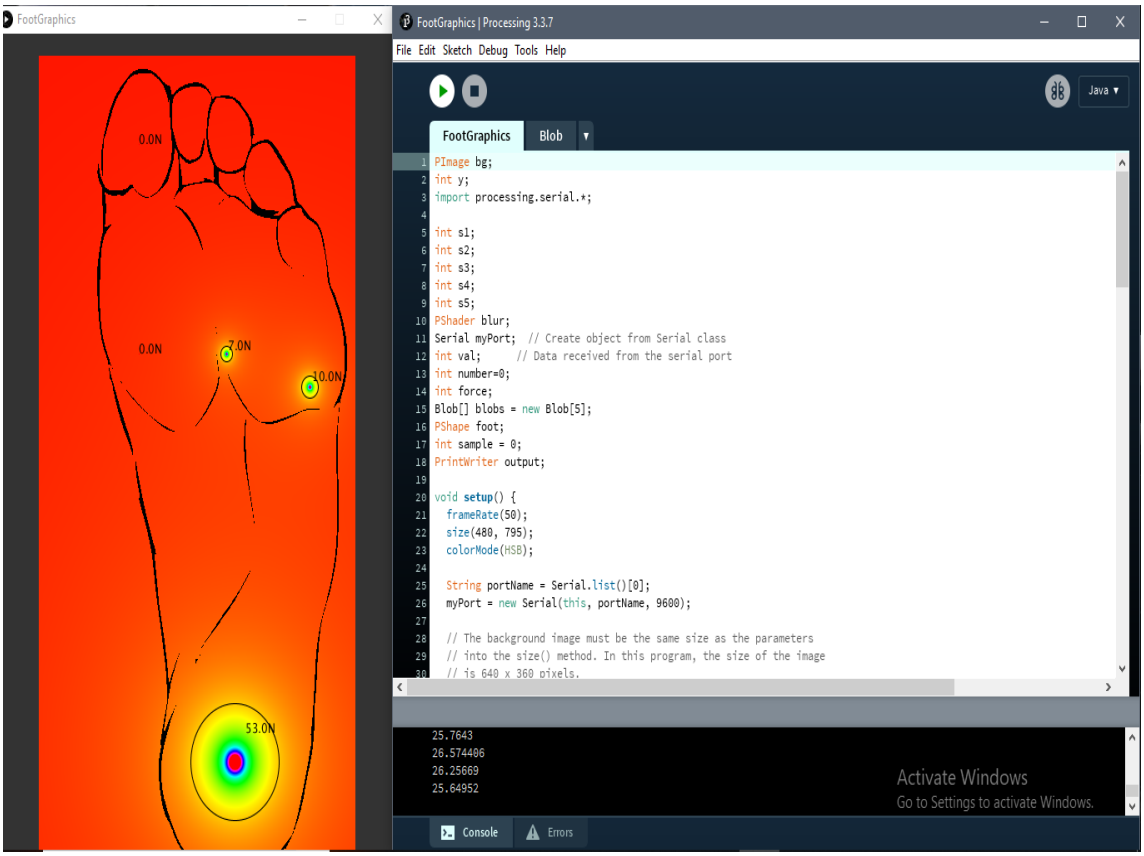
Φυσικά, υπάρχει ο κώδικας που ανανεώνει τα δεδομένα όλων των blobs και των δύο μπαρών και τις σχεδιάζει στον καμβά μας.

```
  updatePixels();
  //println(frameRate);
  fill(1);
  for (Blob b : blobs) {
    b.update();
    b.show();

    hs1.update();
    hs1.display();

    hs2.update();
    hs2.display();
```

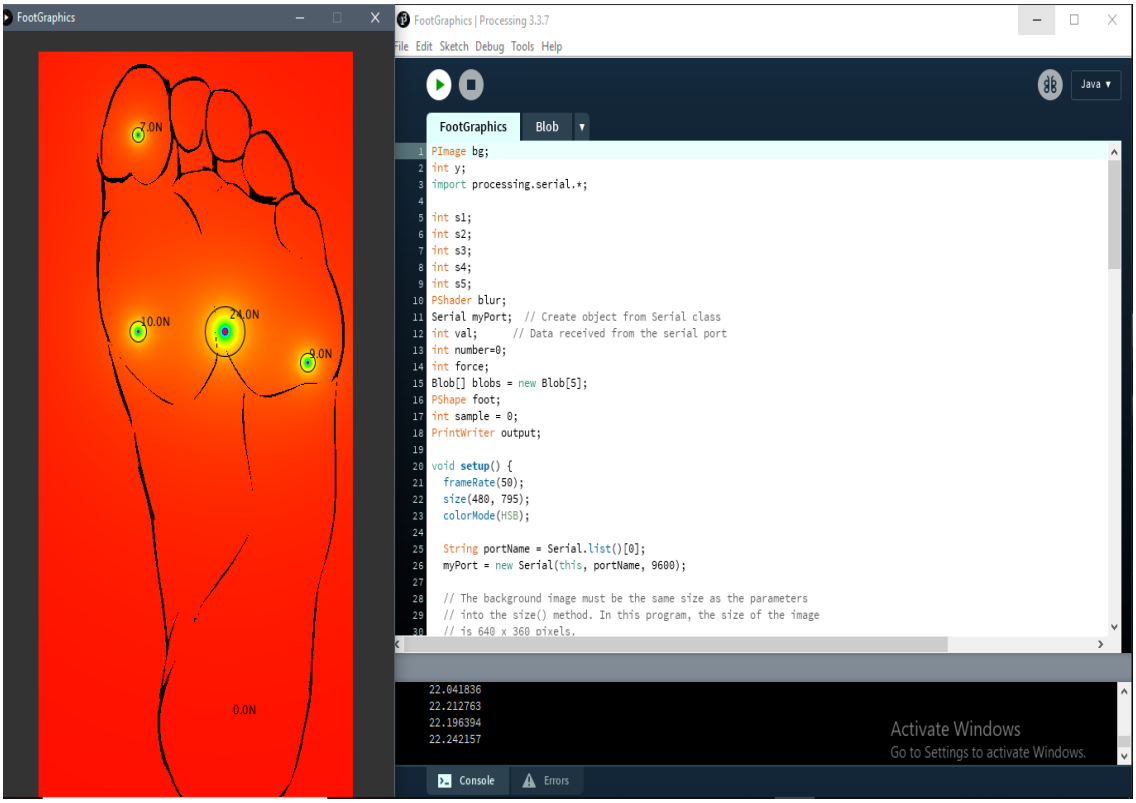
Τέλος, υπάρχει και το κομμάτι του κώδικα που αναλαμβάνει να προβάλει κείμενο που δείχνει τη μέση τιμή πίεσης σε κάθε αισθητήρα, στο χρονικό διάστημα που ελέγχεται και το οποίο αλλάζει ανάλογα με την μετακίνηση της γραμμής κύλισης.

```
fill(255);
text("Average from time " +str((table.getFloat(newScrollBarPos2,0)/1000))+" to time "+"Time"+str((table.getFloat(newScrollBarPos,0)/1000))+"s",440,500);
text("S1:"+" "+str(sensorAvgCurrent[0])+"N",440,540);

text("S2:"+" "+str(sensorAvgCurrent[1])+"N",440,580);

text("S3:"+" "+str(sensorAvgCurrent[2])+"N",440,620);

text("S4:"+" "+str(sensorAvgCurrent[3])+"N",440,660);

text("S5:"+" "+str(sensorAvgCurrent[4])+"N",440,700);
```

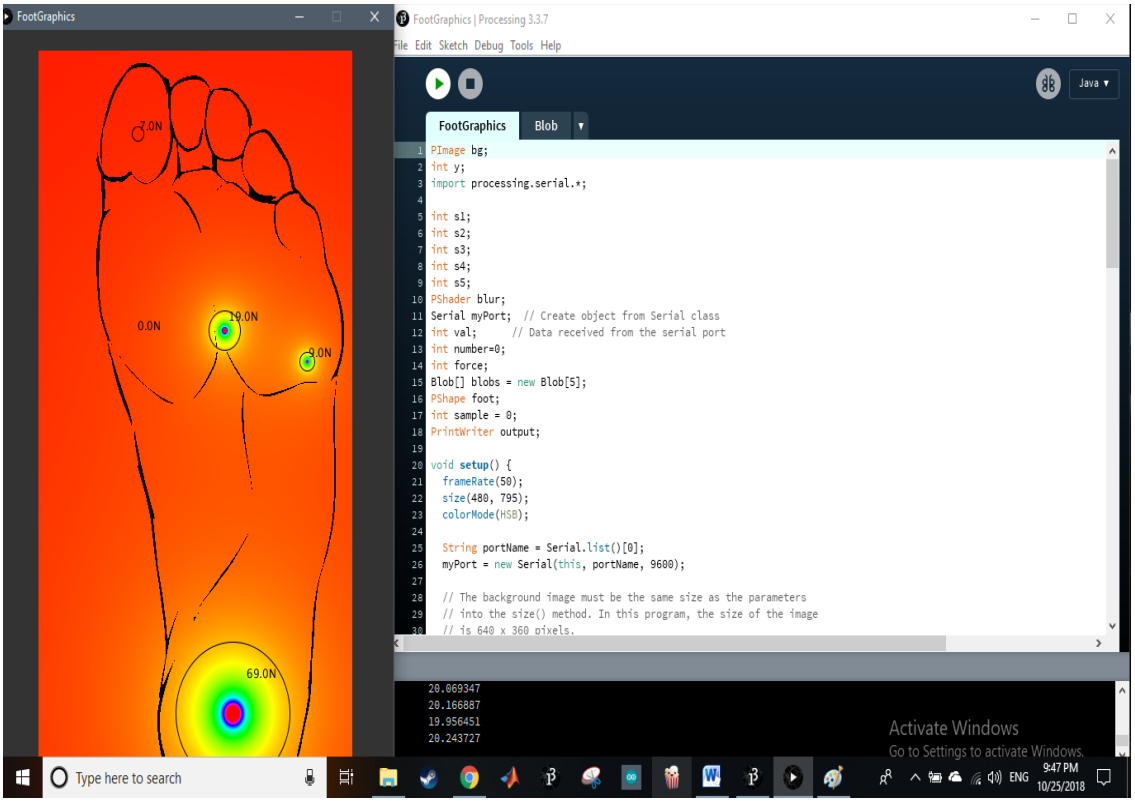
Στο σημείο αυτό έχοντας εξηγήσει πλήρως και το λογισμικό το οποίο δημιουργήθηκε για την πειραματική διάταξη, θεωρήθηκε σκόπιμο να δοθούν κάποιες εικόνες (Σχήμα 3.10, Σχήμα 3.11 & Σχήμα 3.12), που θα προσφέρουν μια καλύτερη οπτική του προγράμματος σε λειτουργία. Οι τρεις εικόνες που ακολουθούν, ελήφθησαν από προσομοιώσεις που πραγματοποιήθηκαν για την εξέταση της ορθότητας της λειτουργίας του κυκλώματος και του προγράμματος.



Σχήμα 3.10 Προσομοίωση όρθιας στάσης στη πτέρνα



Σχήμα 3.11 Προσομοίωση στήριξης στο εμπρόσθιο κομμάτι του πέλματος κατά τη βάδιση



Σχήμα 3.12 Τυχαία στιγμή μέτρησης σε προσομοίωση λειτουργίας της διάταξης

Κεφάλαιο 4. Προτεινόμενες εφαρμογές της διάταξης

4. Προτεινόμενες εφαρμογές της διάταξης

Στο θεωρητικό κομμάτι της εργασίας έχουμε ήδη παρουσιάσει τα προτερήματα της συγκεκριμένης διάταξης και τα πλεονεκτήματά της έναντι άλλων διατάξεων παρόμοιας λειτουργίας. Τα χαρακτηριστικά αυτά που την διαφοροποιούν από τις υπόλοιπες διατάξεις τους είδους της είναι αυτά που οδηγούν στις παρακάτω προτάσεις για εφαρμογή της διάταξης.

4.1 Αθλητικές εφαρμογές

Η διάταξη του πελματογράφου είναι σχεδιασμένη έτσι ώστε το μέγεθός της να είναι προσαρμόσιμο στο πέλμα το οποίο εξετάζεται. Ακόμα, είναι φορητή. Αποτελεί άρα, το ιδανικό εργαλείο προπόνησης για αθλήματα στα οποία η σωστή κατανομή της πίεσης στην πελματιαία επιφάνεια με τον αθλητή εν κινήσει, είναι βασικό κομμάτι της βελτίωσης του. Αθλητές ατομικά, αλλά και ολόκληρες ομάδες αθλητών, μπορούν να χρησιμοποιούν τον πελματογράφο εν ώρα προπόνησης και η προπονητική τους ομάδα να χρησιμοποιεί τα δεδομένα για την ατομική βελτίωση του κάθε αθλητή, κατά τη διάρκεια της προπόνησης, αλλά και αργότερα. Φυσικά, η προπονητική ομάδα μπορεί να συλλέγει και δεδομένα κατά τη διάρκεια αγώνων ή αθλητικών επιδείξεων, ώστε να διαθέτει και μετρήσεις με τους παλμούς του αθλητή σε συνθήκες μέγιστης απόδοσης. Ολόκληρο το τμήμα των εργομετρικών εξετάσεων στις οποίες υποβάλλονται αθλητές κάθε χρόνο, μπορεί να αντικατασταθεί από την χρήση της προτεινόμενης διάταξης. Η διάταξη του πελματογράφου λοιπόν, προσφέρει μια άμεση εφαρμογή στη βιομηχανία της επαγγελματικής άθλησης, στον τομέα της προπόνησης, στον οποίο αφιερώνεται τεράστιο ποσοστό του προϋπολογισμού ομάδων και αθλητών ατομικών αθλημάτων.

Φυσικά, ο πελματογράφος αυτός δεν βρίσκει εφαρμογή μόνο σε επαγγελματίες αθλητές. Η νέα τεχνολογική τάση, είναι οι έξυπνες συσκευές άθλησης. Ήδη κυκλοφορούν στην αγορά, χιλιάδες συσκευές που βελτιώνουν την καθημερινή άθληση του χρήστη τους, με λειτουργίες όπως μέτρημα χιλιομέτρων που διανύθηκαν, μέτρηση παλμών της καρδιάς εν ώρα άσκησης κ.ο.κ. Η διάταξη που σχεδιάστηκε,

βρίσκει λοιπόν εφαρμογή σαν μια νέα τέτοια συσκευή, που μπορεί να πωλείται σε χαμηλή τιμή, ώστε να είναι προσιτή σε οποιονδήποτε, με σκοπό να βελτιώσει την ποιότητα της καθημερινής του άσκησης.

Τέλος, ίσως η μεγαλύτερη εφαρμογή σε επίπεδο αθλητισμού είναι η παρακάτω: Στη σημερινή εποχή, όλα μετρώνται και κρίνονται από αριθμούς. Φυσικά, αυτό δεν είναι διαφορετικό και στην βιομηχανία του αθλητισμού. Υπάρχουν εταιρίες, οι οποίες παίρνουν στατιστικά δεδομένα από εκατοντάδες αθλήματα και χιλιάδες αθλητές, τα οποία ύστερα χρησιμοποιούνται για έρευνα, πωλούνται σε τρίτους και γενικά εκμεταλλεύονται και αναλύονται με κάθε τρόπο. Η διάταξη μπορεί να παρέχει σε πραγματικές συνθήκες αγώνα και μάλιστα με απόκριση πραγματικού χρόνου, δεδομένα που μπορούν να χρησιμοποιηθούν σε πληθώρα στατιστικών κατηγοριών. Σε έναν αγώνα δρόμου, παραδείγματος χάριν, μπορεί να υπολογιστεί βάσει της πελματιαίας πίεσης κάθε δρομέα, ο αθλητής που έκανε την καλύτερη εκκίνηση. Σε έναν αγώνα μπάσκετ, μπορεί να υπολογιστεί η δύναμη την οποία άσκησε το πέλμα ενός αθλητή, κατά τη διάρκεια ενός άλματος. Γίνεται κατανοητό λοιπόν, πως και στο πεδίο αυτό υπάρχουν τεράστιες δυνατότητες εφαρμογής και εκμετάλλευσης της διάταξης.

4.2 Ιατρικές εφαρμογές

Οι ιατρικές εφαρμογές της διάταξης συναντώνται περισσότερο σε επίπεδο πρόληψης. Μπορεί, λοιπόν, ο πελματογράφος, μέσω εφαρμογής στο κινητό, να ενημερώνει για τη δυσανάλογη κατανομή της πίεσης στην πελματιαία επιφάνεια. Ο χρήστης του πελματογράφου, μπορεί έτσι να προλάβει παθήσεις οι οποίες προκύπτουν από την λανθασμένη κατανομή των πελματικών πιέσεων. Τραυματισμοί μπορούν να αποφευχθούν ακόμα και σε επίπεδο αθλητών που πραγματοποιούν κάποια άσκηση λανθασμένα, καταπονώντας τα οστά στο πέλμα. Στο τμήμα της πρόληψης εντάσσονται ακόμα και προβλήματα πολύ σοβαρής φύσης, όπως το εγκεφαλικό, το οποίο παρουσιάζεται πρώτα με μια στρέβλωση του περπατήματος του ασθενούς.

Έπειτα, ίσως η βασικότερη εφαρμογή της διάταξης είναι η χρήση της από ορθοπεδικούς ιατρούς για διάγνωση. Το γεγονός ότι πρόκειται για πολύ προσιτή οικονομικά διάταξη δίνει την δυνατότητα να χρησιμοποιηθεί για διαγνωστικούς

σκοπούς από οποιοδήποτε ιατρικό κέντρο και όχι πλέον μόνο από συγκεκριμένα φυσικοθεραπευτικά ιατρεία. Οι παθήσεις που μπορούν να διαγνωστούν με την συγκεκριμένη διάταξη ποικίλουν, με παραδείγματα όπως η ισχιαλγία, οσφυαλγία κ.α.

Τέλος βρίσκει μεγάλη εφαρμογή και σαν εργαλείο στον τομέα της φυσικοθεραπείας. Η φυσικοθεραπεία μετά από κάποιον τραυματισμό λειτουργεί ως εξής: Μετά τον τραυματισμό, ο ασθενής επισκέπτεται ένα φυσικοθεραπευτήριο, όπου ακολουθεί αγωγή για να ξεπεράσει τον τραυματισμό του. Επιπλέον της επίσκεψής του, δίνεται και μια σειρά ασκήσεων που πρέπει να πραγματοποιήσει μόνος του, προκειμένου να επιτύχει η θεραπεία. Συνήθως οι ασκήσεις αυτές εκτελούνται λανθασμένα από τον ασθενή με αποτέλεσμα η ίαση να καθυστερεί. Εδώ μπορεί να χρησιμοποιηθεί η προτεινόμενη διάταξη. Φορώντας ο ασθενής τον πελματογράφο, κατά την διεκπεραίωση των ασκήσεων αυτών, δίνει την δυνατότητα στον φυσικοθεραπευτή χρησιμοποιώντας το πρόγραμμα αξιολόγησης που έχει σχεδιαστεί, να επισημαίνει τα όποια λάθη. Η θεραπεία λοιπόν γίνεται γρηγορότερη και πολύ πιο αποτελεσματική.

Κεφάλαιο 5. Σύνοψη και προτάσεις για μελλοντική έρευνα

5. Σύνοψη, συμπεράσματα και προτάσεις για μελλοντική έρευνα

5.1 Σύνοψη και κύρια ευρήματα

Κατά την εκπόνηση της διπλωματικής έγινε μια προσπάθεια επίτευξης των στόχων που τέθηκαν στην αρχή. Οι βασικοί στόχοι αφορούσαν τη δημιουργία μιας νέας διάταξης πελματογράφου, ικανής να αντικαταστήσει την ήδη υπάρχουσα. Παράλληλα, πραγματοποιήθηκε πλήρης ανάλυση όλων των υλικών και λογισμικών που χρησιμοποιήθηκαν, ώστε η παρούσα πτυχιακή να βοηθήσει σε παρόμοιες σχεδιάσεις διατάξεων.

Για κάθε έναν από τους στόχους αυτούς, έγινε ανάλυση και σύγκριση της σχετικής διεθνούς και ελληνικής βιβλιογραφίας και προστέθηκαν πιθανοί τρόποι αντιμετώπισης των προβλημάτων, οι οποίοι επέφεραν μία σειρά αποτελεσμάτων.

Δόθηκε μια συνοπτική και περιεκτική καταγραφή και ανάλυση, όλων των στοιχείων που χρησιμοποιήθηκαν για τη δημιουργία της διάταξης. Τόσο σε επίπεδο υλικού, όσο και σε επίπεδο λογισμικού, αιτιολογήθηκαν οι επιλογές και έγινε μια συνοπτική παρουσίαση των εναλλακτικών. Η συγκεκριμένη εργασία αφήνει μια σοβαρή βιβλιογραφική βάση, αλλά και ένα καλό επίπεδο σχεδίασης της διάταξης, πάνω στην οποία μπορούν να εργαστούν μελλοντικοί ερευνητές για εξέλιξη της ίδιας ή παρόμοιων διατάξεων.

Φυσικά, ο βασικός στόχος της πτυχιακής επετεύχθη, με την σχεδίαση και την δημιουργία της διάταξης. Μάλιστα, όπως αναφέρεται και στα συμπεράσματα παρακάτω, η λειτουργία της διάταξης δοκιμάστηκε και κρίθηκε ορθή.

Υπήρχαν αρκετοί περιορισμοί, κυρίως χρόνου και χρημάτων (για αγορά ποιοτικότερων υλικών) και υπάρχουν σαφή περιθώρια περαιτέρω ανάπτυξης και μελλοντικής βελτίωσης του πελματογράφου, αν και οι στόχοι της εργασίας επιτεύχθηκαν σε ικανοποιητικό βαθμό.

5.1 Συμπεράσματα

Παρακάτω, αναλύονται τα βασικά συμπεράσματα που προέκυψαν από τη συγκεκριμένη εργασία:

Το πρώτο συμπέρασμα προέκυψε από την έμπνευση και μόνο για την εκπόνηση της εν λόγω πτυχιακής. Συγκεκριμένα, υπάρχει τεράστιο περιθώριο ανάπτυξης στο υπάρχον σύστημα πελματογράφου. Αν και πρόκειται για εξαιρετικά ποιοτικό εξοπλισμό, υπάρχουν σαφή περιθώρια βελτίωσης των συστημάτων που χρησιμοποιούνται κατά την εξέταση πελματογραφήματος. Είτε με την αναπτυχθείσα διάταξη, είτε με κάποια άλλη παρόμοια που θα έχει το πλεονέκτημα της φορητότητας, η εξέταση του πελματογραφήματος μπορεί να βελτιωθεί σημαντικά.

Συμπεράσματα προέκυψαν και από κάποιες δοκιμές στις οποίες υποβλήθηκε ο πελματογράφος. Λόγω της πρωτότυπης μορφής του, ήταν αδύνατο να δοκιμαστεί στο εσωτερικό υποδήματος, όπως είναι σχεδιασμένος. Για τον λόγο αυτό εκτελέστηκαν αρκετές προσομοιώσεις εισάγοντας μέσω κώδικα μια εντολή η οποία εφάρμοζε τυχαίες πιέσεις (μεταξύ 0 και 98 N) με τυχαίο τρόπο στους αισθητήρες. Τα κύρια ευρήματα που προέκυψαν από τις προσομοιώσεις είναι βασικά δύο: Πρώτον, δοκιμάστηκε η αντοχή της διάταξης και του κώδικα σε συνεχόμενη χρήση, με μέγιστο χρόνο 2.5 ωρών. Η λειτουργία της συνεχίστηκε χωρίς κανένα απολύτως πρόβλημα. Δεύτερον, η απόκριση της διάταξης σε όλο το εύρος τιμών αλλά και στις απότομες αλλαγές αυτών ήταν όντως απόκριση πραγματικού χρόνου.

Πρέπει να τονιστεί βέβαια στο σημείο αυτό, ότι επειδή οι προσομοιώσεις αυτές, εισάγουν τιμές με τελείως τυχαίο τρόπο, δεν προσομοιώνουν στην πραγματικότητα την πίεση στην πελματιαία επιφάνεια σε κατάσταση βάδισης, τρεξίματος κ.ο.κ. Παρ' όλα αυτά, τα συμπεράσματα που προσέφεραν στο κομμάτι της απόκρισης και παρατεταμένης λειτουργίας είναι σημαντικά.

5.2 Προτάσεις για μελλοντική έρευνα

Τα περιθώρια ανάπτυξης της διάταξης είναι πολλά, κυρίως επειδή αποτελεί ένα πειραματικό πρωτότυπο. Καταρχάς, το πρωτότυπο χρησιμοποιεί 5 αισθητήρες μόνο στις βασικότερες περιοχές πελματιαίας πίεσης. Ο περιορισμός του κόστους για την σχεδίαση του πελματογράφου δεν επέτρεπε την αγορά περισσότερων αισθητήρων ή χρήση άλλων μικρότερου μεγέθους. Η διάταξη όμως είναι σχεδιασμένη έτσι ώστε να

δίνεται η δυνατότητα χρησιμοποίησης ακόμα περισσοτέρων αισθητήρων για πιο πληρέστερη ανάλυση. Ιδανικά, μπορεί να σχεδιαστεί και κατασκευαστεί ένας κατά παραγγελία αισθητήρας δύναμης-πίεσης, σε σχήμα ανθρώπινου πέλματος. Έτσι με έναν και μόνο αισθητήρα και χρησιμοποιώντας ακριβώς την ίδια διάταξη (με μικρές μόνο αλλαγές στο λογισμικό), να είναι δυνατή η ανάλυση των πιέσεων σε ολόκληρη την επιφάνεια του πέλματος.

Ύστερα, χρησιμοποιήθηκε η πλακέτα Arduino Uno, που είναι η μια καλή επιλογή μεταξύ των μοντέλων Arduino για wearable εφαρμογές. Η καλύτερη δυνατή επιλογή είναι το Arduino Lilypad. Το συγκεκριμένο μοντέλο έχει πολύ μικρότερο μέγεθος και εκτελεί ίδιες λειτουργίες με το Uno. Κυκλωματικά δεν είναι απαραίτητο να αλλάξει απολύτως τίποτα για να γίνει η αλλαγή των μικροελεγκτών. Ο λόγος που δεν έγινε, είναι ο περιορισμός του χρόνου αλλά και του υλικού που ήταν διαθέσιμο στο εργαστήριο που σχεδιάστηκε ο πελματογράφος.

Μια επιπλέον πολύ ενδιαφέρουσα ανάπτυξη της διάταξης, θα ήταν η τροποποίηση του κυκλώματος, ώστε να χρησιμοποιηθεί για συγκομιδή ενέργειας. Το ρεύμα που δημιουργείται με την πίεση στους αισθητήρες, μπορεί να ανακυκλώνεται από την συσκευή, αφαιρώντας έτσι την ανάγκη τροφοδοσίας αυτής από μπαταρία. Η ενέργεια αυτή βέβαια μπορεί να οδηγείται και αλλού, για τροφοδοσία εξαρτημάτων εκτός της διάταξης.

Τέλος η βασικότερη πρόταση για μελλοντική έρευνα και ανάπτυξη της εν λόγω εργασίας, είναι η δημιουργία μιας εφαρμογής κινητού τηλεφώνου, αποκλειστικά σχεδιασμένη για τη χρήση του πελματογράφου. Αυτή τη στιγμή ο πελματογράφος είναι σχεδιασμένος ώστε να επικοινωνεί και να οπτικοποιεί τα δεδομένα πίεσης, σε έναν φορητό υπολογιστή με τη χρήση δύο συσκευών Bluetooth. Σχεδιάστηκε ένας κώδικας (FootVirtuino - παράρτημα) ο οποίος μεταφέρει και εμφανίζει τα δεδομένα σε εφαρμογή στο κινητό. Όμως η εν λόγω εφαρμογή (Virtuino) καθώς δεν είναι σχεδιασμένη να δουλεύει για το σκοπό αυτό, αλλά για γενικές οπτικοποιήσεις σε δεδομένα που στέλνονται από το Arduino, είχε σοβαρά προβλήματα. Η απόκριση ήταν αργή και κυρίως η οπτικοποίηση ήταν χαμηλότερης ποιότητας συγκριτικά, με αυτήν που σχεδιάστηκε στο Processing. Αποτελεί, βέβαια, ένα πολύ σημαντικό πλεονέκτημα η δυνατότητα του πελματογράφου να συνδέεται με κινητό και γι’ αυτό προτείνεται η σχεδίαση και ανάπτυξη μιας τέτοιας εφαρμογής.

Βιβλιογραφία

[1] Kellis, E. "Plantar Pressure Distribution during Barefoot Standing, Walking and Landing in Preschool Boys." *Gait & Posture* 14, no. 2 (October 2001): 92–97.

[2] Harris, G. F., and J. J. Wertsch. "Procedures for Gait Analysis." *Archives of Physical Medicine and Rehabilitation* 75, no. 2 (February 1994): 216–25.

[3] Bartlett, R. (2007). Introduction to Sports Biomechanics. London: Routledge.

[4] Robertson, D. G., Caldwell, C. A., Hamill, J., Kamen, G., & Whittlesey, S. N. (2004). Research methods in biomechanics. Champaign, IL: Human Kinetics.

[5] Orlin, M. N., & McPoil, T. G. (2000). Plantar pressure assessment. Physical Therapy

[6] Adamec, J., Novotny, P., & Vaverka, F. (1998). A comparison of various methods for the assessment of vertical jump height. Paper presented at the International symposium on biomechanics in sports, Kontsanz.

[7] Bosco, C., Luntanen, P., & Komi, P. V. (1983). A simple method for measurement of mechanical power in jumping. *European Journal of Applied Physiology, 50*, 273-282.

[8] Amiridis, I. G., Hatzitaki, V., & Arabatzi, F. (2003). Age-induced modifications of static postural control inhumans. *Neuroscience Letters, 350*(3), 137-140. doi: S0304394003008784 [pii]

[9] Healy, A.; Burgess-Walker, P.; Naemi, R.; Chockalingam, N. Repeatability of WalkinSense® in shoe pressure measurement system: A preliminary study. *Foot* 2012, *22*, 35–39.

[10] Bamberg, S.; Benbasat, A.Y.; Scarborough, D.M.; Krebs, D.E.; Paradiso, J.A. Gait analysis using a shoe-integrated wireless sensor system. *IEEE Trans. Inf. Technol. Biomed.* 2008, *12*, 413–423.

[11] Tanwar, H.; Nguyen, L.; Stergiou, N. Force Sensitive Resistor (FSR)-Based Wireless Gait Analysis Device. In *Proceeding of The Third IASTED International Conference on Telehealth*, Montreal, QC, Canada, 31 May–1 June 2007.

[12] Lee, N.; Goonetilleke, R.; Cheung, Y.; So, G. A flexible encapsulated MEMS pressure sensor system for biomechanical applications. *J. Microsyst. Technol.* 2001, *7*, 55–62.

[13] Shu, L.; Hua, T.; Wang, Y.; Li, Q.; Feng, D.; Tao, X. In-shoe plantar pressure measurement and analysis system based on fabric pressure sensing array. *IEEE Trans. Inf. Technol. Biomed.* 2009, *14*, 767–775.

[14] Yang, C.M.; Chou, C.M.; Hu, J.S.; Hung, S.H.; Yang, C.H.; Wu, C.C.; Hsu, M.Y.; Yang, T.L. A Wireless Gait Analysis System by Digital Textile Sensors. In

Proceeding of Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2009 (*EMBC2009*), Minneapolis, MN, USA, 3–6 September 2009; pp. 7256–7260.

[15] Zhu, H.; Wertsch, J.; Harris, G.; Loftsgaarden, J.; Price, M. Foot pressure distribution during walking and shuffling. *Arch. Phys. Med. Rehabil.* 1991, *72*, 390–397.

[16] Benocci, M.; Rocchi, L.; Farella, E.; Chiari, L.; Benini, L. A Wireless System for Gait and Posture Analysis based on Pressure Insoles and Inertial Measurement Units. In *Proceeding of the 3rd International Conference on Pervasive Computing Technologies for Healthcare, 2009. Pervasive Health 2009*, London, UK, 1–3 April 2009; pp. 1–6.

[17] Salpavaara, T.; Verho, J.; Lekkala, J.; Halttunen, J. Wireless Insole Sensor System for Plantar Force Measurements during Sport Events. In *Proceedings of IMEKO XIX World Congress on Fundamental and Applied Metrology*, Lisbon, Portugal, 6–11 September 2009; pp. 2118–2123.

[18] Fundamental Understanding of Piezoelectric Strain Sensors Jayanr Sirohi and Inderjit ChopraJournal of intelligent Material Systems and Structures 2000;11;246

[19] J. W. J. W. Gardner, S. V. Paras, A. A. Mouza, N. Dalaoutē, S. Polymatidou, and A. D. Tziolas, *Μικροαισθητήρες, αρχές και εφαρμογές*. Thessalonikē: Tziolas, 2000.

[20] Elgar, P. 2000 *Αισθητήρες Μέτρησης και Ελέγχου*. Θεσσαλονίκη: Εκδόσεις Τζιόλα

[21] National Instruments. *Strain gauge measurements* - A tutorial, Application Note078.

[22] Bicking R.E. *Fundamentals of Pressure Sensor Technology*, Jan. 1995

[23] Knite, M; Teteris, V; Kiploka, A; Kaupuzs, J (15 August 2003). "Polyisoprene-carbon black nanocomposites as tensile strain and pressure sensor materials". Sens. Actuat. A: Phys. doi:10.1016/j.sna.2003.08.006.

[24] Yi, H; Dongrui, W; Xiao-Man, Z; Hang, Z; Jun-Wei, Z; Zhi-Min, D (24 October 2012). "Positive piezoresistive behavior of electrically conductive alkyl-functionalized graphene/polydimethylsilicone nanocomposites". J. Mater. Chem. C. doi:10.1039/C2TC00114D

[25] Basta, M; Picciarelli, V; Stella, R (1 October 1993). "An introduction to percolation". Europ. J. Phys. Bibcode:1994EJPh...15...97B. doi:10.1088/0143-0807/15/3/001.

[26] Zhou, J; Song, Y; Zheng, Q; Wu, Q; Zhang, M (2 February 2008). "Percolation transition and hydrostatic piezoresistance for carbon black filled poly(methylvinylsilioaxne)vulcanizates". Carbon. 46. doi:10.1016/j.carbon.2008.01.028.

[27] L. Paredes-Madrid, A. Matute, J. O. Bareño, C. A. Parra Vargas, and E. I. Gutierrez Velásquez, “Underlying Physics of Conductive Polymer Composites and Force Sensing Resistors (FSRs). A Study on Creep Response and Dynamic Loading,” *Materials (Basel)*, vol. 10, no. 11, Nov. 2017.

[28] *Xiang-Wu, Z; Yi, P; Qiang, Z; Xiao-Su, Y (8 September 2000). "Time dependence of piezoresistance for the conductor-filled polymer composites". J. Pol. Sci. Part B: Pol. Phys. doi:10.1002/1099-0488(20001101)38:21<2739::AID-POLB40>3.0.CO;2-O*

[29] M. Banzi, *Getting Started with Arduino*, 2. ed. Sebastopol, CA: O’Reilly & Associates, 2011.

[30] M. Banzi, *Getting Started with Arduino*, 2. ed. Sebastopol, CA: O'Reilly & Associates, 2011.

[31] S. F. Barrett, "Arduino Microcontroller: Processing for Everyone! Second Edition," *Synthesis Lectures on Digital Circuits and Systems*, vol. 7, no. 2, pp. 1–371, Jun. 2012.

[32] M. Mosher, "Processing: A Programming Handbook for Visual Designers and artists, Second Edition *Processing: A Programming Handbook for Visual Designers and artists, Second Edition* by Casey Reas and Ben Fry, The MIT Press, Cambridge, MA, U.S.A., 2014. 672 pp., illus. Trade. ISBN: 978-0-262-02828-8.," *Leonardo*, vol. 49, no. 5, pp. 476–477, Oct. 2016.

[33] D. Shiffman, *Learning Processing: a beginner's guide to programming images, animation, and interaction*, Second edition. Amsterdam: Elsevier/Morgan Kaufmann, 2015.

Παράρτημα

Κώδικας Arduino IDE

```
#define HEEL_BONE_SENSOR 0 // Analogue pins used
#define OUTER_BALL_SENSOR1 1
#define OUTER_BALL_SENSOR2 2
#define OUTER_BALL_SENSOR3 3
#define OUTER_BALL_SENSOR4 4

#define HEEL_BONE_INDEX 0 // Analogue pins used
#define OUTER_BALL1_INDEX 1
#define OUTER_BALL1_INDEX 2
#define OUTER_BALL2_INDEX 3
#define OUTER_BALL3_INDEX 4

#define R2 10000

#define FILTER_COEF1 6 //ensure that both sum to 8, increase FILTER_COEF2
value to make the filter stronger.
#define FILTER_COEF2 2
// Variables for storing sensor values
float unHeelBone = 0;
float unOuterBall0 = 0;
float unOuterBall1 = 0;
float unOuterball2 = 0;
float unOuterball3 = 0;

float unHeelBoneOld = 0;
float unOuterBall0Old = 0;
float unOuterBall1Old = 0;
float unOuterball2Old = 0;
float unOuterball3Old = 0;

float mySensorsVolArr[5];
float mySensorsForceArr[5];
float mySensorsForceFilteredArr[5];

void setup()
{
 Serial.begin(9600);
}

void loop() {
 for (int i = 0; i < 5; i++)
 {
 mySensorsVolArr[i] = map(analogRead(i), 0, 1023, 0, 5000);
 }
 for (int i = 0; i < 5; i++)
 {
 // The voltage = Vcc * R / (R + FSR) where R = 10K and Vcc = 5V
 // so FSR = ((Vcc - V) * R) / V
 mySensorsForceArr[i] = (5000 - mySensorsVolArr[i]);
 mySensorsForceArr[i] *= R2;
 mySensorsForceArr[i] /= mySensorsVolArr[i];
 // mySensorsForceArr[i] = 250;
 //Serial.println(mySensorsForceArr[i]);

 mySensorsForceArr[i] = 1000000 / mySensorsForceArr[i];// we measure in
micromhos
 mySensorsForceArr[i] = (constrain(mySensorsForceArr[i], 1, 1000000));
```

```
 // Use the two FSR guide graphs to approximate the force
 if (mySensorsForceArr[i] <= 1000)
 {
  mySensorsForceArr[i] = mySensorsForceArr[i] / 80;
 }
 else
 {
  mySensorsForceArr[i] = mySensorsForceArr[i] - 1000;
  mySensorsForceArr[i] /= 30;
 }
 }
 for (int i = 0; i < 5; i++)
 {
 mySensorsForceFilteredArr[i] = (mySensorsForceFilteredArr[i] * 4 +
mySensorsForceArr[i] * 4) / 8;
 }
 delay(20);
 Serial.write(255);
 for (int i = 0; i < 5; i++)
 {
 //Serial.write((i*20)+20);
 //Serial.println(String(int((mySensorsForceArr[i]))));
 Serial.write(int(constrain(mySensorsForceFilteredArr[i], 0, 98)));
 }

 //Serial.println("uT");
 //Serial.print("Heel: "); Serial.print(unHeelBoneOld); Serial.print("
"); // Blue
 //Serial.print("Ball1: "); Serial.print(unOuterBall0Old); Serial.print("
"); // Red
 //Serial.print("Ball2: "); Serial.print(unOuterBall1Old); Serial.print("
"); // Green
 //Serial.print("Ball3: "); Serial.print(unOuterball2Old); Serial.print("
"); // Orange
 //Serial.print("Ball4: "); Serial.print(unOuterball3Old); Serial.print("
"); //Purple
 //Serial.println("uT");
}
```

Κώδικας Graphics

```
Κλάση Blob
// Daniel Shiffman (modified slightly for chiprobots alligator test)
// http://codingrainbow.com
// http://patreon.com/codingrainbow
class Blob {
 PVector pos;
 float r;
 PVector vel;
 Blob(float x, float y) {
  pos = new PVector(x, y);
  vel = PVector.random2D();
  vel.mult(random(0, 0));
  r = 0;
 }
 void update() {
  pos.add(vel);
  if (pos.x > width || pos.x < 0) {
  vel.x *= -1;
  }
```

```
  if (pos.y > height || pos.y < 0) {
  vel.y *= -1;
  }
 }
 void show() {
  noFill();
  stroke(0);
  strokeWeight(1);
  ellipse(pos.x, pos.y, r * 2, r * 2);
  text(str(r) + "N", pos.x + r / 4, pos.y - r / 2);
 }
}
```

Βασικός Κώδικας

```
PImage bg;
int y;
import Processing.serial.*;
int s1;
int s2;
int s3;
int s4;
int s5;
PShader blur;
Serial myPort; // Create object from Serial class
int val;  // Data received from the serial port
int number = 0;
int force;
Blob[] blobs = new Blob[5];
PShape foot;
int sample = 0;
PrintWriter output;

void setup() {
 frameRate(50);
 size(480, 795);
 colorMode(HSB);

 String portName = Serial.list()[0];
 myPort = new Serial(this, portName, 9600);

 // The background image must be the same size as the parameters
 // into the size() method. In this program, the size of the image
 // is 640 x 360 pixels.
 blobs[0] = new Blob(270 - 100, 100);
 blobs[1] = new Blob(270 - 100, 290);
 blobs[2] = new Blob(375 - 100, 290);
 blobs[3] = new Blob(475 - 100, 320);
 blobs[4] = new Blob(385 - 100, 660);

 //bg = loadImage("foot.png");
 //background(bg);
 foot = loadShape("foot.svg");

 int m = minute(); // Values from 0 - 59
 int h = hour(); // Values from 0 - 23
 int d = day();
 int mm = month();
 String fileName = "Gait_" + str(mm) + "_" + str(d) + "_" + str(h) + "_"
+ str(m) + ".csv";
 output = createWriter(fileName);
 output.println("Time, Sample number, Sensor1, Sensor2, Sensor3, Sensor4,
Sensor5");
 output.flush(); // Writes the remaining data to the file
}
```

```
void draw() {
 background(51);
 shape(foot, -100, 30, 630, 790);
 loadPixels();
 for (int x = 50; x < width - 50; x++) {
 for (int y = 20; y < height - 10; y++) {
  int index = x + y * width;
  float sum = 0;
  for (Blob b : blobs) {
  float d = dist(x, y, b.pos.x, b.pos.y);
  sum += 30 * b.r / d;
  }
  if (pixels[index] >= color(1, 1, 1))
  {
  pixels[index] = color(sum, 255, 255);
  }
 }
 }
 if ( myPort.available() >= 6) { // If data is available,

 while (myPort.read() != 255);
 blobs[0].r = myPort.read();   // read it and store it in val
 blobs[1].r = myPort.read();
 blobs[2].r = myPort.read();
 blobs[3].r = myPort.read();
 blobs[4].r = myPort.read();
 String time = str(millis());
 myPort.clear();
 output.println(time + ", " + str(sample) + ", " + str(blobs[0].r) + ", "
+ str(blobs[1].r) + ", " + str(blobs[2].r) + ", " + str(blobs[3].r) + ",
" + str(blobs[4].r));
 output.flush(); // Writes the remaining data to the file
 sample++;
 }

 updatePixels();
 println(frameRate);
 for (Blob b : blobs) {
 b.update();
 b.show();
 }

 //float d = map(98, 0, 98, 0, 100);
 //float c;

 //noStroke();
 //c = map(s1, 0, 98, 0, 255);
 //fill(c, 255-(c), 0);
 //ellipse(270, 130, s1, s1);

 //c = map(s2, 0, 98, 0, 255);
 //fill(c, 255-(c), 0);
 //ellipse(270, 270, s2, s2);

 //c = map(s3, 0, 98, 0, 255);
 //fill(c, 255-(c), 0);
 //ellipse(370, 270, s3, s3);

 //c = map(s4, 0, 98, 0, 255);
 //fill(c, 255-(c), 0);
 //ellipse(475, 300, s4, s4);
```

```
 //c = map(s5, 0, 98, 0, 255);
 //fill(c, 255-(c), 0);
 //rect(370-(s5/10), 610-(s5/10), s5, s5);
}
```

Κώδικας Analytics

```
Κλάση HScrollbar
class HScrollbar {
 int swidth, sheight; // width and height of bar
 float xpos, ypos;  // x and y position of bar
 float spos, newspos; // x position of slider
 float sposMin, sposMax; // max and min values of slider
 int loose;    // how loose/heavy
 boolean over;   // is the mouse over the slider
 boolean locked;
 float ratio;

 HScrollbar (float xp, float yp, int sw, int sh, int l) {
  swidth = sw;
  sheight = sh;
  int widthtoheight = sw - sh;
  ratio = (float)sw / (float)widthtoheight;
  xpos = xp;
  ypos = yp - sheight / 2;
  spos = xpos + swidth / 2 - sheight / 2;
  newspos = spos;
  sposMin = xpos;
  sposMax = xpos + swidth - sheight;
  loose = l;
 }

 void update() {
  if (overEvent()) {
  over = true;
  } else {
  over = false;
  }
  if (mousePressed && over) {
  locked = true;
  }
  if (!mousePressed) {
  locked = false;
  }
  if (locked) {
  newspos = constrain(mouseX - sheight / 2, sposMin, sposMax);
  }
  if (abs(newspos - spos) > 1) {
  spos = spos + (newspos - spos) / loose;
  }
 }

 float constrain(float val, float minv, float maxv) {
  return min(max(val, minv), maxv);
 }

 boolean overEvent() {
  if (mouseX > xpos && mouseX < xpos + swidth &&
```

```
   mouseY > ypos && mouseY < ypos + sheight) {
  return true;
  } else {
  return false;
  }
 }

 void display() {
  noStroke();
  fill(204);
  rect(xpos, ypos, swidth, sheight);
  if (over || locked) {
  fill(0, 0, 0);
  } else {
  fill(102, 102, 102);
  }
  rect(spos, ypos, sheight, sheight);
 }

 float getPos() {
  // Convert spos to be values between
  // 0 and the total width of the scrollbar
  return spos * ratio;
 }
 void setPos(int pos) {
  spos = pos;
 }
}
```

Βασικός κώδικας

```
PImage bg;
int y;
HScrollbar hs1;
HScrollbar hs2;
import Processing.serial.*;
Table table;

int [] sensorAvg = new int[5];
int [] sensorAvgCurrent = new int[5];

PShader blur;
Serial myPort; // Create object from Serial class
int val;  // Data received from the serial port
int number = 0;
int force;
Blob[] blobs = new Blob[5];
PShape foot;
int sample = 0;
PrintWriter output;
int rows;
int newScrollBarPos;
int newScrollBarPos2;

String fileName;

void setup() {
 frameRate(50);
 size(800, 795);
 colorMode(HSB);

 selectInput("Select a file to process:", "fileSelected");
```

```
 while (fileName == null)
 {
 delay(500);
 println("here");
 }
 //String portName = Serial.list()[0];
 //myPort = new Serial(this, portName, 9600);
 //selectInput("Select a file to process:", "fileSelected");

 blobs[0] = new Blob(270 - 100, 100);
 blobs[1] = new Blob(270 - 100, 290);
 blobs[2] = new Blob(375 - 100, 290);
 blobs[3] = new Blob(475 - 100, 320);
 blobs[4] = new Blob(385 - 100, 660);

 foot = loadShape("foot.svg");

 hs2 = new HScrollbar(0, height - 60, width, 15, 15);
 hs1 = new HScrollbar(0, 20, width, 15, 15);
 table = loadTable(fileName, "csv");
 println(table.getRowCount() + " total rows in table");
 println(table.getColumnCount());
 rows = table.getRowCount();

 textSize(16);
 long sum = 0L;
 for (int i = 0; i < 5; i++)
 {
 sum = 0L;
 for (TableRow row : table.rows()) {
  sum += int(row.getFloat(i + 2));
 }
 sensorAvg[i] = int(sum / rows);
 }


}

void draw() {
 background(51);
 frameRate(30);
 shape(foot, -100, 30, 630, 790);
 loadPixels();
 for (int x = 50; x < 480 - 50; x++) {
 for (int y = 20; y < height - 10; y++) {
  int index = x + y * width;
  float sum = 0;
  for (Blob b : blobs) {
  float d = dist(x, y, b.pos.x, b.pos.y);
  sum += 30 * b.r / d;
  }
  if (pixels[index] >= color(1, 1, 1))
  {
  pixels[index] = color(sum, 255, 255);
  }
 }
 }

 float pos = hs1.getPos();
 float pos2 = hs2.getPos();
 newScrollBarPos = int(map(pos, 0, width , 1, rows));
 newScrollBarPos2 = int(map(pos2, 0, width , 1, rows));
```

```
 blobs[0].r = table.getFloat(newScrollBarPos, 2);  // read it and store
it in val
 blobs[1].r = table.getFloat(newScrollBarPos, 3);
 blobs[2].r = table.getFloat(newScrollBarPos, 4);
 blobs[3].r = table.getFloat(newScrollBarPos, 5);
 blobs[4].r = table.getFloat(newScrollBarPos, 6);

 //println(newScrollBarPos);
 //println(blobs[4].r);
 long sum2 = 0L;

 for (int i = 0; i < 5; i++)
 {
 sum2 = 0L;
 for (int ii = newScrollBarPos2; ii <= newScrollBarPos; ii++) {
  sum2 += int(table.getFloat(ii, i + 2));
 }
 sensorAvgCurrent[i] = int(sum2 / map(((newScrollBarPos -
newScrollBarPos2)), 0, width, 1, width));
 }

 updatePixels();
 //println(frameRate);
 fill(1);
 for (Blob b : blobs) {
 b.update();
 b.show();
 hs1.update();
 hs1.display();
 hs2.update();
 hs2.display();

 fill(255);
 text("Average from time " + str((table.getFloat(newScrollBarPos2, 0) /
1000)) + " to time " + "Time" + str((table.getFloat(newScrollBarPos, 0) /
1000)) + "s", 440, 500);
 text("S1:" + " " + str(sensorAvgCurrent[0]) + "N", 440, 540);
 text("S2:" + " " + str(sensorAvgCurrent[1]) + "N", 440, 580);
 text("S3:" + " " + str(sensorAvgCurrent[2]) + "N", 440, 620);
 text("S4:" + " " + str(sensorAvgCurrent[3]) + "N", 440, 660);
 text("S5:" + " " + str(sensorAvgCurrent[4]) + "N", 440, 700);
 }
}

void fileSelected(File selection) {
 if (selection == null) {
 println("Window was closed or the user hit cancel.");
 }
 else {
 println("User selected " + selection.getAbsolutePath());
 fileName = selection.getAbsolutePath();
 }
}
```

Κώδικας FootVirtuino

```
#include "VirtuinoBluetooth.h"     // virtuino library 1.47 or highest
#include <SoftwareSerial.h>
```

```
#include <SPI.h>
#include <SD.h>
#define HEEL_BONE_SENSOR 0 // Analogue pins used
#define OUTER_BALL_SENSOR1 1
#define OUTER_BALL_SENSOR2 2
#define OUTER_BALL_SENSOR3 3
#define OUTER_BALL_SENSOR4 4

#define HEEL_BONE_INDEX 0 // Analogue pins used
#define OUTER_BALL1_INDEX 1
#define OUTER_BALL1_INDEX 2
#define OUTER_BALL2_INDEX 3
#define OUTER_BALL3_INDEX 4

#define R2 10000

#define FILTER_COEF1 6 //ensure that both sum to 8, increas FILTER_COEF2
value to make the filter stornger.
#define FILTER_COEF2 2
// Variables for storing sensor values
float unHeelBone = 0;
float unOuterBall0 = 0;
float unOuterBall1 = 0;
float unOuterball2 = 0;
float unOuterball3 = 0;

float unHeelBoneOld = 0;
float unOuterBall0Old = 0;
float unOuterBall1Old = 0;
float unOuterball2Old = 0;
float unOuterball3Old = 0;

String fileName = "datalog.csv";

float mySensorsVolArr[5];
float mySensorsForceArr[5];
float mySensorsForceFilteredArr[5];
long sampleNumber = 1;
SoftwareSerial bluetoothSerial = SoftwareSerial(9, 8); // arduino RX
pin=2 arduino TX pin=3 connect the arduino RX pin to bluetooth module TX
VirtuinoBluetooth virtuino(bluetoothSerial);    // enable this line and
disable all SoftwareSerial lines
// Open VirtuinoBluetooth.h file on the virtuino library folder ->
disable the line: #define BLUETOOTH_USE_SOFTWARE_SERIAL
// Connect HC-05 module to Arduino (MEGA or DUE) Serial1. (pins: 18,19)
File dataFile;
//=========================================================================
= setup
//=========================================================================
=

// The circuit:
// * analog sensors on analog ins 0, 1, and 2
// * SD card attached to SPI bus as follows:
// ** MOSI - pin 11
// ** MISO - pin 12
// ** CLK - pin 13
// ** CS - pin 10 (for MKRZero SD: SDCARD_SS_PIN)
const int chipSelect = 10;

void setup(void) {
 Serial.begin(9600);  // start monitor serial port
 bluetoothSerial.begin(115200);
```

```
 //virtuino.DEBUG=true;

 SD.begin(chipSelect);
 int number = random(0, 100);

 dataFile = SD.open(fileName, FILE_WRITE);
 delay(100);

 dataFile.println("Time, Sample number, Sensor1, Sensor2, Sensor3,
Sensor4, Sensor5");
 dataFile.close();
}

//====================================================================
= setup
//====================================================================
=
void loop(void) {
 virtuino.run();

 virtuino.vDelay(30);

 for (int i = 0; i < 5; i++)
 {
 mySensorsVolArr[i] = map(analogRead(i), 0, 1023, 0, 5000);
 }

 for (int i = 0; i < 5; i++)
 {
 mySensorsForceArr[i] = (5000 - mySensorsVolArr[i]);
 mySensorsForceArr[i] *= R2;
 mySensorsForceArr[i] /= mySensorsVolArr[i];
 // mySensorsForceArr[i] = 250;
 //Serial.println(mySensorsForceArr[i]);

 mySensorsForceArr[i] = 1000000 / mySensorsForceArr[i];
 mySensorsForceArr[i] = (constrain(mySensorsForceArr[i], 1, 1000000));

 if (mySensorsForceArr[i] <= 1000)
 {
  mySensorsForceArr[i] = mySensorsForceArr[i] / 80;
 }
 else
 {
  mySensorsForceArr[i] = mySensorsForceArr[i] - 1000;
  mySensorsForceArr[i] /= 30;
 }
 }
 for (int i = 0; i < 5; i++)
 {
 mySensorsForceFilteredArr[i] = (mySensorsForceFilteredArr[i] * 4 +
mySensorsForceArr[i] * 4) / 8;
 }
 delay(20);
 Serial.write(255);
 for (int i = 0; i < 5; i++)
 {
 //Serial.write((i*20)+20);
 //Serial.println(String(int((mySensorsForceArr[i]))));
 Serial.write(int(constrain(mySensorsForceFilteredArr[i], 0, 98)));
 }
 dataFile.println(String(millis()) + ", " + String(sampleNumber) + ", " +
String(int(constrain(mySensorsForceFilteredArr[0], 0, 98))) + ", " +
```

```
String(int(constrain(mySensorsForceFilteredArr[1], 0, 98))) + ", " +
String(int(constrain(mySensorsForceFilteredArr[2], 0, 98))) + ", " +
String(int(constrain(mySensorsForceFilteredArr[3], 0, 98))) + ", " +
String(int(constrain(mySensorsForceFilteredArr[4], 0, 98))));
 dataFile.close();
 virtuino.vMemoryWrite(0, (constrain(mySensorsForceFilteredArr[0], 0,
98))); // write temperature 1 to virtual pin V0. On Virtuino panel add a
value display or an analog instrument to pin V0
 virtuino.vMemoryWrite(1, (constrain(mySensorsForceFilteredArr[1], 0,
98))); // write temperature 1 to virtual pin V1. On Virtuino panel add a
value display or an analog instrument to pin V1
 virtuino.vMemoryWrite(2, (constrain(mySensorsForceFilteredArr[2], 0,
98)));
 virtuino.vMemoryWrite(3, (constrain(mySensorsForceFilteredArr[3], 0,
98)));
 virtuino.vMemoryWrite(4, (constrain(mySensorsForceFilteredArr[4], 0,
98)));
 sampleNumber++;
 dataFile = SD.open(fileName, FILE_WRITE);
 //bluetoothSerial.println("hello");
}
```