



A G H

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI,
INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

KATEDRA BIOCYBERNETYKI I INŻYNIERII BIOMEDYCZNEJ

Praca dyplomowa inżynierska

*System do monitorowania przebiegu ćwiczeń siłowych w oparciu o
analizę biopotencjałów*

*System for monitoring of strength training based on biopotential
analysis*

Autor:

Piotr Połcik

Kierunek studiów:

Inżynieria Biomedyczna

Opiekun pracy:

dr inż. Eliasz Kańtoch

Kraków, 2018

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpozna bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystycznego wykonania albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, videogram lub nadanie.”, a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.): „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej «sądem koleżeńskim».”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

*Serdecznie dziękuję wszystkim osobom, które
przyczyniły się do powstania niniejszej pracy.*

Spis treści

1. Wstęp	7
2. Podstawy teoretyczne	9
2.1. Podstawy fizjologiczne i metodologia pomiarów EMG	9
2.1.1. Fizjologia pomiarów EMG.....	9
2.1.2. Problemy występujące podczas pomiaru EMG.....	12
2.2. Maszyna wektorów wspierających SVM.....	13
2.2.1. Liniowa maszyna wektorów nośnych.....	13
2.2.2. "Kernel trick"	16
2.3. Przegląd rozwiązań	16
2.3.1. Cele omawianych rozwiązań.....	16
2.3.2. Wykorzystane układy pomiarowe i oprogramowanie	17
2.3.3. Metody przetwarzania, analizy i klasyfikacji sygnałów.....	18
2.3.4. Porównanie protokołów eksperymentów	19
2.3.5. Wyniki pomiarów	20
3. Opis systemu	21
3.1. Hardware.....	21
3.1.1. Mikrokontroler	21
3.1.2. Czujnik EMG	22
3.2. Oprogramowanie.....	26
3.2.1. Arduino IDE oraz Qt	26
3.2.2. Oprogramowanie mikrokontrolera	26
3.2.3. Aplikacja komputerowa.....	28
3.3. Klasyfikacja sygnału	36
3.3.1. Protokół pomiarowy	36
3.3.2. Cechy sygnału i ich ekstrakcja	38
3.3.3. Uczenie klasyfikatora	40
4. Testy i omówienie wyników	41
5. Podsumowanie i perspektywy rozwoju.....	45
Bibliografia	47
A. Dodatek - zawartość dołączonej płyty CD	51

1. Wstęp

Sporty siłowe, choć nie są zaliczane do tzw. sportów ekstremalnych, niosą ze sobą pewne ryzyko. Osoby je uprawiające są często narażone na niekiedy dotkliwe kontuzje, które mogą uniemożliwić dalszy trening na pewien czas lub nawet całkowicie wyeliminować możliwość wykonywania określonych ćwiczeń. Przyczynami kontuzji może być złe dopasowanie ciężaru do możliwości fizycznych [1], brak asekuracji ze strony partnera treningowego, zły stan techniczny używanego sprzętu, nieprzystosowanie miejsca do wykonywania ćwiczeń [2], ale przede wszystkim niewłaściwa technika ich wykonywania [1, 2, 3, 4]. Szczególnie narażone na kontuzje są osoby początkujące, które nie umieją wykonywać ćwiczeń poprawnie oraz nie potrafią dopasować ciężaru do swoich możliwości [1].

Amerykańska Komisja ds. Bezpieczeństwa Produktów Konsumenckich po przestudiowaniu danych pochodzących od służb ratowniczych odnotowała ponad 25 tys. kontuzji powiązanych z treningiem siłowym i na tej podstawie wyliczono, że w ciągu roku średnio ponad 970 tys. wizyt na oddziałach ratunkowych na rok powiązanych jest z ćwiczeniami z wykorzystaniem ciężarów, co daje średnią ok. 1 wizyty na 45 osób w samych Stanach Zjednoczonych w skali roku [4]. Jednakże liczby te są najprawdopodobniej zaniżone, gdyż, jak pokazują ankiety, ok. 54% osób uprawiających tę formę treningu leczy kontuzje we własnym zakresie, bez konsultacji z lekarzem [4]. Najczęściej do kontuzji dochodzi w wyniku naglego upuszczenia ciężaru (65.5%), z kolei przetrenowanie i stosowanie zbyt dużego ciężaru przyczynia się do ok. 81% kontuzji, a nieprawidłowa technika wykonywania ćwiczeń - do ok. 31% [4].

Sygnał elektromiograficzny, który towarzyszy skurczom mięśni, mógłby być wykorzystany do monitorowania, czy ćwiczenia wykonywane są w sposób poprawny, oraz czy ciężar zastosowany przez osobę ćwiczącą jest odpowiednio dobrany do jej możliwości. Pozwolić może na to fakt, że amplituda rejestrowanego sygnału jest zależna od generowanej przez mięsień siły, a właściwie od ilości zrekrutowanych jednostek motorycznych, których liczebność bezpośrednio przekłada się na siłę skurcza [5].

Posiadając wiedzę na temat tego, które grupy mięśniowe oraz w jakim stopniu powinny być aktywne podczas wykonywania danego ćwiczenia przy użyciu danego ciężaru, oraz wykorzystując odpowiednio wytrenowany klasyfikator i klasyfikację sygnału w czasie rzeczywistym, można by wspomóc amatorów sportów siłowych w nabywaniu odpowiedniej techniki wykonywania ćwiczeń, zmniejszając tym samym ryzyko kontuzji, nie tylko wśród początkujących ale także w przypadku tych bardziej doświadczonych.

Zaletą takiego rozwiązania jest prostota wykonywania badania, które sprowadzałoby się praktycznie tylko do umieszczenia elektrod na powierzchni ciała badanego i uruchomienia programu. Ocena wykonywanego ćwiczenia dokonywana byłaby automatycznie w czasie rzeczywistym, co pozwalałoby

na korygowanie techniki na bieżąco, aż do osiągnięcia właściwych parametrów dla obserwowanych grup mięśniowych.

Celem pracy było stworzenie uproszczonej wersji systemu do nadzorowania przebiegu ćwiczeń fizycznych, opisanego wcześniej. Proces tworzenia systemu składał się z kilku etapów.

Pierwszym z nich był dobór odpowiednich komponentów sprzętowych, jak czujniki oraz mikroprocesor. Drugim etapem było oprogramowanie systemu, zarówno części sprzętowej jak i stworzenie aplikacji realizującej następujące czynności:

- odczyt sygnału przesyłanego za pośrednictwem USB
- wizualizacja sygnału
- zapis danych do pliku
- odczyt danych uczących z pliku i wytrenowanie klasyfikatora
- przetwarzanie sygnału w czasie rzeczywistym
- klasyfikacja sygnału w czasie rzeczywistym
- prezentacja wyników ćwiczenia

Trzecim etapem było zebranie sygnałów z grupy badawczej 15 osób, z których następnie należało wyekstrahować odpowiednie cechy, służące do wytrenowania klasyfikatora używanego w systemie.

Kolejny rozdział pracy odnosi się do zagadnień fizjologiczno-technicznych związanych z rejestracją sygnału EMG, zawiera także omówienie zastosowanego w systemie klasyfikatora oraz przegląd rozwiązań w pewnym stopniu zbliżonych do zaproponowanego w pracy. W rozdziale trzecim zamieszczony został opis systemu oraz procedury zbierania danych potrzebnych do jego implementacji. Czwarty rozdział stanowi prezentację i omówienie wyników testów systemu. W rozdziale ostatnim zamieszczono podsumowanie pracy oraz możliwości dalszego rozwoju zaproponowanego systemu.

2. Podstawy teoretyczne

Poniższy rozdział dotyczy teoretycznych zagadnień związanych z tematem realizowanego projektu, takich jak fizjologia i metodologia pomiarów EMG oraz zasada działania zastosowanego klasyfikatora. Ponadto w rozdziale zamieszczony został przegląd kilku prac zbliżonych tematyką do niniejszej pracy.

2.1. Podstawy fizjologiczne i metodologia pomiarów EMG

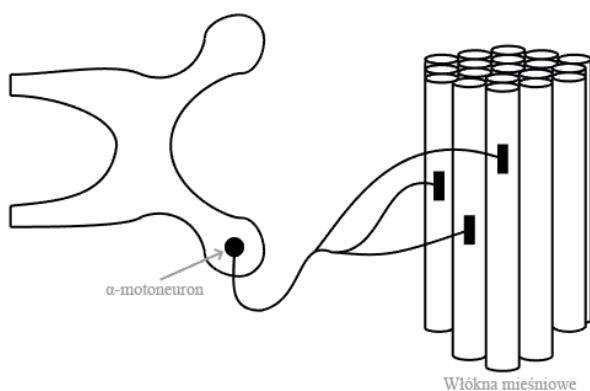
2.1.1. Fizjologia pomiarów EMG

"Elektromiografia (EMG) jest techniką eksperymentalną związaną z uzyskiwaniem, nagrywaniem i analizą sygnałów mioelektrycznych. Sygnały mioelektryczne powstają wskutek zmian fizjologicznych w stanie błon komórkowych włókien mięśniowych." [6]

Powyższy cytat stanowi kwintesencję tego, czym są pomiary elektromiograficzne. Jednak przed za- głębiением się w metodykę pomiarów EMG, warto zatrzymać się na chwilę nad tym, jaki dokładnie sygnał stanowi cel pomiaru, oraz w jaki sposób jest on generowany.

Mechanizm skurcza mięśnia rozpoczyna się od impulsu nerwowego napływającego do pojedynczego miocytu, a generowanego przez α -motoneurony wychodzące z rogu przedniego rdzenia kręgowego. Akson motoneuronu dzieli się na kilka odnóg, z których każda łączy się z pojedynczym miocytom w połowie jego długości, formując tym samym złącze nerwowo-mięśniowe. Komórka ruchowa (α -motoneuron) wraz ze wszystkimi unerwianymi przez nią włóknami mięśniowymi tworzy **jednostkę motoryczną**. Ilość miocytów wchodzących w skład pojedynczej jednostki motorycznej wahana się od 10 do nawet 800, i zależy przede wszystkim od tego, jak dużą precyzją ruchów charakteryzuje się mięsień. Jednostki motoryczne, w których skład wchodzi mała ilość miocytów, zdolne są do szybkich i krótkotrwałych skurczów, podczas gdy duże jednostki motoryczne mają możliwość wykonywania długotrwałych i powolnych skurczów o znacznie większej sile. Schematycznie przedstawienie jednostki motorycznej zaprezentowano na **Rys.2.1** [7, 8].

Impuls nerwowy motoneuronu dociera do złącza nerwowo-mięśniowego. Odgałęzienia aksonu komórki ruchowej tworzą element presynaptyczny synapsy nerwowo-mięśniowej. Na skutek docierającego do niego impulsu następuje uwolnienie transmittera - acetylocholiny - do szczeliny synaptycznej. Po stroenie płytka końcowej znajdują się receptory, umożliwiające przyłączenie się transmitemera. W momencie

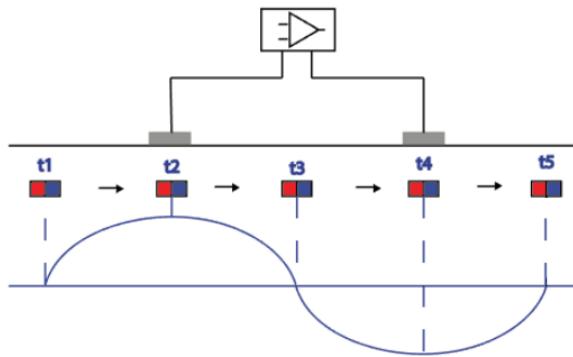


Rys. 2.1. Schemat jednostki motorycznej. Zmodyfikowana ryc. 6.28. z pozycji [7] str. 329 i ryc. 5 z pozycji [6] str. 6.

przyłączenia acetylocholiny do receptora błony postsynaptycznej następuje zwiększenie przepuszczalności płytka dla jonów Na^+ i K^+ , a następnie gwałtowny napływ jonów sodowych, powodujący jej depolaryzację. Ta zmiana potencjału motorycznej płytka końcowej nosi nazwę **potencjału płytka końcowej**. Wsteczna wymiana jonów, w wyniku której dochodzi do odtworzenia potencjału spoczynkowego płytka końcowej, wynoszącego około -90mV, nosi nazwę repolaryzacji. Powstająca depolaryzacja płytka końcowej jest wystarczająco duża, by móc wywołać miejscowy przepływ prądu między płytka końcową a spolaryzowaną błoną miocytu, powodując tym samym depolaryzację błony komórkowej. Napływające do wnętrza komórki jony potasu powodują gwałtowną zmianę potencjału błonowego. W przypadku, gdy impuls będzie na tyle silny, by przekroczyć wartość napięcia progowego (-45mV), następuje wyzwolenie potencjału czynnościowego. Potencjał ów, powstający w obrębie złącza nerwowo-mięśniowego znajdującego się w połowie długości miocytu, przemieszcza się spontanicznie wzdłuż komórki mięśniowej depolaryzując dalsze obszary błony komórkowej, w obu kierunkach miocytu. Przemieszczający się potencjał powoduje wzrost przepuszczalności retikulum endoplazmatycznego, znajdującego się we wnętrzu miocytów, dla jonów Ca^{2+} , co powoduje ich uwalnianie do wnętrza komórki i zapoczątkowuje skurcz mięśnia [6, 7].

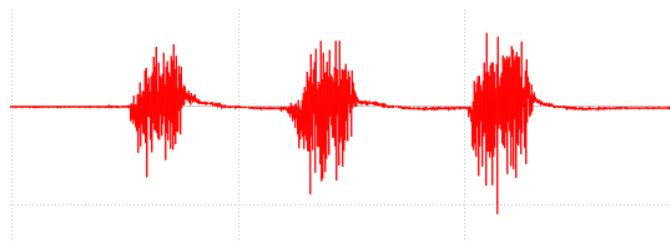
Do skurcza mięśnia dochodzi w wyniku pojawienia się potencjałów czynnościowych miocytów i to właśnie te potencjały, będące efektem opisanych wyżej procesów, są źródłem sygnału rejestrowanego podczas badania EMG. Depolaryzacja w danej chwili czasowej obejmuje określoną długość pojedynczego włókna mięśniowego, a obszar ten nazywany jest strefą depolaryzacji. Jego szerokość wynosi zwykle $1-3mm^2$, natomiast prędkość przemieszczania się wzdłuż włókna zawiera się w przedziale $2-6\frac{m}{s}$. Następująca niemal natychmiast po depolaryzacji repolaryzacja tworzy swego rodzaju cykl, który określa się mianem fali depolaryzacji [6, 9].

Sygnal elektromiograficzny rejestrowany jest najczęściej przy użyciu elektrod dwubiegunowych oraz wzmacniacza różnicowego. **Rys. 2.2.** przedstawia schemat powstawania sygnału EMG. Fala depolaryzacji powstaje w chwili t_1 i początkowo nie jest rejestrowana przed żadną z elektrod. Następnie



Rys. 2.2. Schematyczne przedstawienie powstawania sygnału EMG. Zmodyfikowana ryc. 9 z pozycji [6] str. 8.

fala przesuwa się pod pierwszą elektrodą, a rejestrowany sygnał różnicowy osiąga maksimum. W chwili t3 rejestrowany sygnał osiąga wartość zero, by w chwili t4 ponownie osiągnąć maksimum, o przeciwej wartości niż maksimum w chwili t2 ze względu na wzmacnienie różnicowe. W zależności od odległości między elektrodami różnica potencjałów może pochodzić równocześnie od więcej niż jednego przesuwającego się dipola elektrycznego. Powyższy model wyjaśnia, dlaczego jednobiegunkowy potencjał formuje dwubiegunkowy sygnał. Elektrody rejestrują sygnał pochodzący równocześnie ze wszystkich włókien wchodzących w skład jednostki motorycznej. Suma potencjałów pochodzących ze wszystkich włókien jednostki motorycznej tworzy tzw. **MUAP - Motor unit action potential** (potencjał czynnościowy jednostki motorycznej). Kształt i wielkość MUAP zależą od geometrycznej orientacji włókien w stosunku do ułożenia elektrod. W rzeczywistości podczas badania SEMG (Surface EMG - powierzchniowe EMG) nie jest możliwe rejestrowanie sygnału pochodzącego z pojedynczej jednostki motorycznej. W związku z tym zarejestrowany sygnał to w rzeczywistości nakładające się na siebie potencjały czynnościowe wszystkich jednostek motorycznych, wykrywanych tam gdzie przyłożone zostały elektrody. Ze względu na sposób jego rejestrowania, sygnał ten jest symetryczny o wartości średniej równej zeru. Jest to tak zwany interferencyjny zapis EMG [6, 9].



Rys. 2.3. Sygnał EMG zarejestrowany dla trzech skurczów mięśnia dwugłowego ramienia.

Rys. 2.3. przedstawia surowy zapis SEMG, będący efektem nakładania się potencjałów czynnościowych jednostek motorycznych. W trakcie gdy mięsień pozostaje rozluźniony, rejestrowany sygnał przyjmuje formę linii podstawowej, charakteryzującej się występowaniem szumu, którego uśredniona wartość

powinna wynosić 1-5 mV. Ważną właściwością rejestrowanego sygnału EMG jest jego przypadkowy kształt - piki powstające podczas skurczu mięśnia nie mogą zostać dokładnie powtórzone przy ponownym skurczu. Podczas skurczu nieustannie zmienia się układ zrekrutowanych jednostek motorycznych, co jest bezpośrednią przyczyną losowości sygnału. W celu "uogólnienia" rejestrowanego sygnału oraz zmniejszenia wpływu losowych składowych możliwe jest wyznaczenie tzw. EMG envelope (obwiedni sygnału EMG). Zakres częstotliwości sygnału mieści się w zakresie 0 do 500 Hz, przy czym większość energii przenoszona jest w paśmie 20-150 Hz [5, 6, 9].

2.1.2. Problemy występujące podczas pomiaru EMG

Pomiary prowadzone w ramach niniejszej pracy mają charakter dynamiczny, w związku z czym to właśnie od problemów z nimi związanych rozpoczyna się niniejszy podrozdział.

1. Przemieszczańcie się brzuśca mięśnia.

W trakcie skurczu następuje względne przemieszczenie się brzuśca mięśnia. W przypadku pomiarów wykonywanych w niniejszej pracy, a więc skurzu mięśnia dwugłowego ramienia, konieczne jest umieszczenie elektrod centralnie, w oddaleniu od czynnej masy mięśnia, w osi ramienia od jego wewnętrznej strony. Dodatkowy problem stanowi odkształcanie się skóry podczas skurzu mięśnia oraz zmiany kształtu samego mięśnia. Na uwadze należy mieć możliwość odklejania się elektrod oraz ich wzajemnego napierania na siebie, gdyż nie ulegają one naturalnym deformacjom, jakim podlega skóra [5, 6, 9].

2. Unieruchomienie przewodów pomiarowych.

Właściwe umocowanie przewodu pomiarowego podczas testów dynamicznych pozwala na uniknięcie artefaktów mających swoje źródło w ruchu kabla, a także zmniejsza ryzyko odklejenia się elektrod podczas ruchu. Warto także rozważyć możliwość zastosowania bezprzewodowego przesyłu danych, tak by zminimalizować długość przewodów, zmniejszając tym samym szansę na ich przemieszczenie [5, 6].

3. Opór elektryczny tkanek ludzkich.

Poszczególne tkanki ciała ludzkiego posiadają różny opór elektryczny. Na jego wpływ ma nie tylko typ tkanki, ale również grubość, zachodzące procesy fizjologiczne, temperatura oraz wilgotność. Z tych przyczyn nie jest zalecane bezpośrednie porównywanie parametrów ilościowych surowego sygnału EMG nie tylko pomiędzy dwiema osobami, ale też pomiędzy pomiarami wykonywanymi dla tej samej osoby w różnych miejscach ciała. Ponadto, w celu uzyskania rzetelnych wyników pomiaru, konieczne jest także odpowiednie przygotowanie skóry do pomiaru (w przypadku pomiaru SEMG) [5, 6, 9].

4. Interferowanie z sygnałem EKG.

W trakcie rejestracji sygnału EMG możliwe jest występowanie interferencji z sygnałem EKG,



(a) Ilustracja interferencji EMG z sygnałem EKG. (b) Ilustracja zakłóceń pochodzących z sieci. Fragment o niewielkim szumie powstał po odpięciu kabla zasilającego od laptopa.

Rys. 2.4. Zakłócenia sygnału EMG.

szczególnie w miejscach pomiaru takich jak mięśnie obręczy barkowej czy tułowia. Zaproponowanych zostało kilka metod usuwania tych interferencji, wykorzystujących np. filtry górnoprzepustowe czy filtrowanie adaptacyjne. Okazuje się to jednak dosyć problematyczne, ze względu na nakładanie się sygnałów w dziedzinach czasu i częstotliwości. Przykład sygnału EMG zakłóconego przez sygnał EKG przedstawia Rys. 2.4(a) [6, 10, 11].

5. Szумy pochodzenia zewnętrznego.

Podczas pomiaru EMG, ze względu na jego stosunkowo niewielką amplitudę, konieczne jest uwzględnienie wpływu szumów zewnętrznych, których źródłem jest promieniowanie elektromagnetyczne tła, pochodzące od np. fal radiowych, przewodów, żarówek. Na szczególną uwagę zasługują szумy pochodzące z sieci zasilania. W celu ich wyeliminowania (a także przede wszystkim zachowania zasad BHP), konieczne jest zastosowanie odpowiedniej bariery, np. logicznej (układ pomiarowy zasilany z akumulatora/baterii). Przykład zakłóceń pochodzących z sieci zasilania przedstawia Rys.2.4(b) [5, 6].

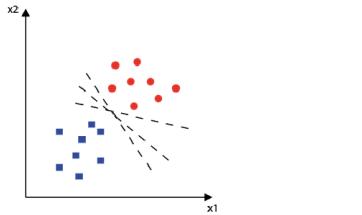
6. "Cross talk"

Pomiar sygnału EMG może być obarczony błędem związanym z rejestracją sygnałów EMG pochodzących od sąsiadujących grup mięśniowych, dlatego tym bardziej należy zwrócić uwagę na odpowiednie umieszczenie elektrod. Sygnały te nie przekraczają zazwyczaj 10-15% całkowitej zawartości sygnału, jednak są trudne do wykrycia w przypadku SEMG [6, 9].

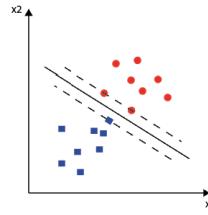
2.2. Maszyna wektorów wspierających SVM

2.2.1. Liniowa maszyna wektorów nośnych

SVM czyli z angielskiego support vector machine to metoda klasyfikacji danych z nadzorem zaliczana do dziedziny machine learning'u. Podstawowym założeniem SVM jest odnalezienie dwóch równoległych do siebie hiperpłaszczyzn rozdzielających dane treningowe (w przypadku niniejszego projektu przez dane rozumiemy wektory wykstrahowanych cech sygnału EMG). Idealny klasyfikator SVM



(a) Losowa liniowa klasyfikacja zbioru danych.

(b) Klasyfikacja zbioru danych za pośrednictwem SVM.
Zmodyfikowana Fig. 10.1. z [12] str. 344**Rys. 2.5.** Przedstawienie LSVM

rozdziela bezbłędnie dane na podstawie wektora decyzji, który jest w każdym punkcie równoodległy od obu hiperpłaszczyzn. Wyznaczenie granicy rozdziału zostanie omówione na przykładzie LSVM (Linear SVM - Liniowa maszyna wektorów nośnych) [12].

Rozważmy zbiór danych przedstawiony na **Rys.2.5.** Linie przedstawione na pozycji (a) w istocie rozdzielają w pełni dwa przedstawione zbioru danych. Instynktownie jednak można stwierdzić, że linia ciągła z pozycji (b) jest "bezpieczniejszą" granicą rozdziału, o maksymalizowanej odległości od najbliższych wektorów treningowych. Te właśnie najbliższe wektory, przez które przechodzą teoretyczne hiperpłaszczyzny zilustrowane linią przerywaną, są nazywane wektorami wspierającymi. "Strefa bezpieczeństwa" może być określona jako najmniejsza odległość pomiędzy pojedynczą hiperpłaszczyzną a granicą rozdziału [13].

Załóżmy, że wektory, przez które przebiegają przerywane linie na **Rys. 2.5(b)** stanowią zbiór p par uczących.

$$(x_i, d_i) \quad \text{dla } i = 1, 2, \dots, p \quad (2.1)$$

x_i - wektor danych wejściowych

$d_i \in \{-1; +1\}$ - dyskryminowane klasy

Zakładając, że obie klasy da się odseparować liniowo, to równanie granicy rozdziału oznaczonej ciągłą linią na **Rys.2.5.(b)** może być wtedy zapisane jako (wzór 2.2) [12, 14]

$$f(x) = w^T x + b = 0 \quad (2.2)$$

gdzie:

w - wektor wag

x - wektor danych wejściowych

b - polaryzacja

Można zatem określić równania decyzyjne [12, 14]

$$w^T x + b \geq 0 \quad \text{wtedy } d_i = +1 \quad (2.3)$$

$$w^T x + b \leq 0 \quad \text{wtedy } d_i = -1 \quad (2.4)$$

które zapisane w postaci pojedynczej nierówności przedstawia wzór 2.5 [12, 14]

$$d_i(w^T x + b) \geq 1 \quad (2.5)$$

Pary punktów (x_i, d_i) spełniające powyższe równanie określone są mianem właśnie **wektorów wspierających**. Decydują one o tym gdzie znajduje się granica rozdziału i jak szeroka jest "strefa bezpieczeństwa", określana mianem marginesu separacji. W celu poprawienia rozdzielczości klasyfikatora SVM, margines separacji jest następnie maksymalizowany. Szerokość marginesu separacji ρ (zakładając, że wektory wspierające posiadają wagę równą ± 1) dana jest wzorem 2.6 [12, 14]

$$\rho = (x^+ - x^-) \frac{w}{\|w\|} = \frac{2}{\|w\|} \quad (2.6)$$

gdzie:

$(x^+ - x^-)$ - różnica odległości wektorów nośnych z przeciwnych klas

$\|w\| = \|w\|_2$ - norma euklidesowa wektora

Aby zmaksymalizować dany powyższą zależnością margines separacji, należy zminimalizować $\|w\|$ co jest równoznaczne ze wzorem 2.7 [12, 13, 14]

$$\min_w \left\{ \frac{1}{2} \|w\|^2 \right\} \quad (2.7)$$

Powyższy problem optymalizacji nosi nazwę *problemu optymalizacji kwadratowej*. Problemy optymalizacji tego typu rozwiązuje się za pomocą metod tzw. optymalizacji wypukłej, jednakże temat ten nie będzie poruszany w niniejszej pracy [12, 13].

Często zdarza się, że dane nie są w pełni liniowo separowalne lub można otrzymać większy margines separacji, jeśli umożliwi się klasyfikatorowi niepoprawne zaklasyfikowanie niektórych z wektorów treningowych. Taki margines określa się nazwą *miękkiego marginesu* (ang. *soft margin*) w przeciwieństwie do omawianego wcześniej *sztywnego (twardego)* marginesu (ang. *hard margin*). Częstokroć okazuje się, że margines miękki osiąga lepsze wyniki aniżeli margines sztywny. W takim przypadku formuły (2.5) i (2.7) rozszerzane są o dodatkowy czynnik, uwzględniający margines miękki i umożliwiający umieszczenie części wektorów treningowych w marginesie separacji lub w niewłaściwej grupie [12, 13, 14]

$$\min_w \left\{ \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \right\} \quad (2.8)$$

$$d_i(w^T x + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad (2.9)$$

gdzie:

ξ_i - zmienne pozwalające na umieszczenie wektora w marginesie lub złej klasie (ang. *slack variables*).

C - czynnik modyfikujący "sztywność" marginesu (wpływ sumy *slack variables* na margines).

2.2.2. "Kernel trick"

Istnieją jednak sytuacje, w których nieregularność danych sprawia, że znacznie lepiej sprawdziłby się klasyfikator nieliniowy. Klasyfikatory liniowe mają jednak mocną stronę w postaci prostoty algorytmów uczących. Najlepiej byłoby, gdyby istniała możliwość utworzenia nieliniowej granicy rozdziału używając liniowego klasyfikatora. Rozwiązań tak postawionego problemu okazuje się być zastosowanie tzw. **kernel trick**, czyli "triku jądrowego". Polega on na rzutowaniu każdego wektora treningowego do innej przestrzeni, przy użyciu odpowiednich funkcji jader. W efekcie możliwe jest, aby w nowej przestrzeni uzyskać liniową funkcję dyskryminującą (granicę rozdziału), która w pierwotnej przestrzeni jest funkcją nieliniową, jeśli funkcja jądra jest nieliniowa. Najpopularniejsze funkcje jader przedstawiają wzory 2.10, 2.11, 2.12, 2.13 [13, 14, 15].

– liniowe:

$$(K(x_i, x_j) = x_i^T x_j) \quad (2.10)$$

– wielomianowe:

$$(K(x_i, x_j) = (\gamma x_i^T x_j)^d, \quad \gamma > 0) \quad (2.11)$$

– radialna funkcja bazowa:

$$(K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \quad \gamma > 0) \quad (2.12)$$

– sigmoidalna:

$$(K(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)) \quad (2.13)$$

gdzie:

γ, r, d - parametry jader.

2.3. Przegląd rozwiązań

W poniższym podrozdziale omówione zostały najważniejsze aspekty rozwiązań odnoszących się do podobnego problemu co niniejszy projekt, z ostatnich pięciu lat. Omawiane pozycje charakteryzują się różnym podejściem do problemu poprzez zastosowanie zróżnicowanych systemów pomiarowych czy technik analizy i klasyfikacji sygnału. I choć wszystkie z omawianych artykułów wykorzystują klasyfikację EMG, żaden z nich pod kątem proponowanego w niniejszej pracy zastosowania.

2.3.1. Cele omawianych rozwiązań

Jak wspomniano, wszystkie z przywoływanych tutaj artykułów wykorzystywały klasyfikację sygnału EMG, ale do różnych celów. W niektórych, przedmiotem badań było określenie wywieranej siły na podstawie sygnału elektromiograficznego, ([16, 17, 18]) ale do różnych zastosowań. W [17] rzeczywistym

celem było przewidywanie samej siły na podstawie sygnału, z kolei [16] skupiał się na aspekcie sportowym, mającym zastosowanie w podnoszeniu ciężarów. W przypadku [18] siła była określana w celach kontroli procesu rehabilitacji, określania stopniowego powrotu do pełni sił.

W kilku przypadkach testowano możliwość zastosowania sygnału EMG do rozpoznawania gestów ([19, 20, 21, 22, 23, 24, 25]). Niektóre proponowały rozwiązania mające przede wszystkim na celu usprawnienie protez dloni korzystających z sygnału SEMG ([19, 22, 24, 25]). Z kolei w [20] skupiano się przede wszystkim na porównaniu kilku parametrów i metod klasyfikacji sygnału SEMG, a w [21] opracowano nową metodą klasyfikacji. Szczególny przypadek stanowi [23], w którym rozpoznawanie gestów ma służyć do kontroli sprzętów domowych.

W [26, 27] sygnał elektromiograficzny badany był pod kątem rozpoznawania zmęczenia, a w przypadku [27] również do rozpoznawania bólu odcinka lędźwiowego kręgosłupa. Jedynym wyjątkiem jest pozycja [28], która dotyczyła konstrukcji odpowiedniego systemu do pomiaru sygnału SEMG podczas ćwiczeń siłowych. W [29] zastosowano SEMG do pomiaru ciężaru podnoszonego przez człowieka, wykorzystując również czujniki inercyjne. Z kolei [30] skupiał się na przetestowaniu regresji logistycznej jako techniki klasyfikacji sygnału. Przegląd powyższych możliwości i proponowanych zastosowań może uzmysłosić, jak wiele przydatnych zastosowań mogą mieć pomiary sygnału elektromiograficznego.

2.3.2. Wykorzystane układy pomiarowe i oprogramowanie

Pojawiające się w omawianych artykułach układy pomiarowe zazwyczaj nie powtarzają się w nieljszym zestawieniu. Jedynym wyjątkiem od tej sytuacji jest opaska *Myo Thalmic Labs*, wykorzystana w pozycjach [22, 23, 29]. Opaska posiada 8-kanałów do pomiaru SEMG, częstotliwość próbkowania 200Hz oraz moduł Bluetooth, przy stosunkowo niskiej cenie, co może być przyczyną jej popularności. W dwóch przypadkach ([16, 28]) zastosowano własne systemy pomiarowe. W [28] system składał się z sensorów powierzchniowych Kendall, Ref. 31050522 (Covidien, Mansfield, MA), wzmacniacza różnicowego *AMPo₄FPZ* (Analog Devices), wzmacniacza *TLo₈4* (Texas Instruments, Inc. Dallas, TX), a rolę (ang.) *Data Acquisition Unit* (DAQ) pełnił oscyloskop cyfrowy TDS2002C (Tektronix, Beaverton, OR). Układ zastosowany w [16] wyglądał bardzo podobnie, jednak nie zostały podane dokładne modele komponentów. W pozostałych przypadkach zastosowane układy pomiarowe są bardzo zróżnicowane. Można tutaj wymienić urządzenia firmy DelSys (DelSys Inc., Boston MA) jak sensory DE-2.1 oraz system EMG Bagnoli-4 zastosowane w [19], system EMG-830C od EMG System do Brasil ([30]), układ BIOFORCEN (Mr Intelligent technology Co., Ltd.) ([18]), DAQ wyprodukowane przez firmę National Instruments (PCI-6220 i DAQ-6009) użyte w [19, 30], Raspberry PI 2 ([23]) a nawet Arduino Mega 2560 ([27]).

Jeśli chodzi o parametry akwizycji, najczęściej pojawiającą się częstotliwością próbkowania jest 1000Hz ([16, 17, 18, 19, 21, 25, 30]), ale pojawiają się też częstotliwości nawet 2500Hz ([28]), lub 100Hz ([24]) czy 500 Hz ([20]).

Równie zróżnicowana co zastosowane urządzenia pomiarowe jest liczba rejestrówanych i analizowanych kanałów, która waha się od 1 kanału ([21, 27, 28]) aż do 28 ([26]). Często pojawiającą się liczbą kanałów jest również 4 ([17, 20]) a także 8 ([16, 26, 30]) i 16 ([18, 25]) kanałów.

Jeśli natomiast chodzi o wykorzystywane oprogramowanie, to zdecydowanie dominującym programem jest MATLAB (Mathworks Inc.) stosowany w różnych wersjach w [16, 18, 21, 22, 24, 25, 27, 28]. Z kolei w [19, 30] zastosowano oprogramowanie zaimplementowane w LabView (*National InstrumentsTM*) ze względu na zastosowanie komponentów firmy National Instruments. W kilku przypadkach zastosowano oprogramowanie zaimplementowane przez autorów, w językach R ([23, 26]) oraz w Javie ([20]).

2.3.3. Metody przetwarzania, analizy i klasyfikacji sygnałów

Wstępne przetwarzanie, jakiemu poddany został sygnał, opisane zostało tylko w kilku przypadkach. W niektórych miało ono przede wszystkim za zadanie poprawić jakość samego sygnału ([16, 17, 18, 28], a w przypadkach [21, 24, 30] stanowiło istotną część procesu ekstrakcji cech (choć tu również nie można pominąć aspektu ogólnej poprawy jakości sygnału). I tak w w przypadku [16] wstępne przetwarzanie miało na celu odszumienie sygnału, podobnie jak w [17] gdzie zastosowano w tym celu pierwszorzędowe filtry górnoprzepustowe (*cut-off frequency* 20, 200 i 400 Hz) oraz dolnoprzepustowe (*cut-off frequency* 2 Hz) i w [18], gdzie użyty został filtr eliptyczny. W [28], gdzie celem było stworzenie układu pomiarowego, wstępne przetwarzanie, a więc wzmacnianie, filtracja pasmowa, prostowanie i filtracja dolnoprzepustowa, były realizowane analogowo. W [30], operacjami, którym wstępnie poddano sygnał było jego prostowanie, normalizacja i segmentacja, natomiast w [24] filtracja, segmentacja, uśrednianie oraz normalizacja. Operacje wstępne przetwarzania sygnału w przypadku [21] były, na tle pozostałych przykładów, dość nietypowe. Na podstawie wartości progowej z sygnału wybierano piki ją przekraczające, pozostałe wartości zastępując zerem. Dla tak otrzymanych zespołów pików wykonywano następnie analizę składowych głównych (ang. *Principal Component Analysis*, PCA) i klastrowanie metodą *Gaussian Mixture Model* (GMM).

W omawianych pracach pojawiła się duża liczba cech, które ekstrahowano z sygnałów w celu użycia ich do późniejszej klasyfikacji. Część z nich pojawiła się jednak w więcej niż jednej pracy - średnia wartości bezwzględnej ([20, 21]), wariancja sygnału ([19, 30]), RMS ([22, 27, 30]), średnia częstotliwość ([22, 27]), mediana częstotliwości ([22, 27, 30]) oraz energia ([20]) i moc sygnału ([18]). Część z cech była jednak unikalna dla danego rozwiązania (spośród rozpatrywanych). Możemy tutaj wymienić (z bardziej interesujących, nietypowych) model autoregresyjny 4-tego rzędu ([16]), *Hudgins' time-domain features* oraz współczynnik autokorelacji i korelacji krzyżowej zastosowane w [19], *approximate entropy* (przybliżona entropia, miara nieregularności i nieprzewidywalności sygnału), mediana wartości niezerowych oraz jej indeks użyte w [21] czy kurtoza ([30]). W niektórych przypadkach zastosowane wektory cech zostały także poddane analizie składowych głównych (PCA) w celu zredukowania ilości zmiennych ([22, 24]).

W przypadku klasyfikatorów (metod klasyfikacji) również można wyróżnić takie, które stosowane były wielokrotnie, oraz unikalne dla danej pracy. Do tej pierwszej grupy można zaliczyć przede wszystkim popularne klasyfikatory - k-NN (ang. *k-Nearest Neighbours*, k-najbliższych sąsiadów) zastosowane w [20, 27], LDA (ang. *Linear Discriminant Analysis*, liniowa analiza dyskryminacyjna) użyte w [18, 19, 20, 21, 22], QDA (ang. *Quadratic Discriminant analysis*, kwadratowa analiza dyskryminacyjna) pojawiające się w [18, 20], SVM z różnymi typami jąder ([20, 22, 23, 25, 27]) oraz ANN (ang. *Artificial Neural Network*, sztuczna sieć neuronowa) o różnej ilości warstw ukrytych, ilości neuronów oraz różnego typu, zastosowane w [17, 25, 27, 29]. Unikalne dla niektórych prac metody klasyfikacji to przede wszystkim zastosowanie dekompozycji sygnału algorytmem MODWT (ang. *Maximal overlap discrete wavelet transform*) oraz falką Daubechies-4 w celu zamodelowania funkcji siły mięśnia w zależności od jego poziomu aktywacji ([16]), w pracy [26] porównywano pięć różnych algorytmów wykrywania zmęczenia, i można tu wymienić algorytm średniej częstotliwości, stosunku momentów widmowych, oparty na falkach algorytm WIRM1551, *recurrence quantification analysis* (analiza kwantyfikacji rekuencyjnej), oraz dwa algorytmy opierające się na analizie entropii sygnału. Z kolei w [22] zastosowano także klasyfikatory *Naive Bayes* oraz *Random Forest*.

2.3.4. Porównanie protokołów eksperymentów

Podstawowym aspektem protokołów eksperymentów porównywanych prac jest ilość osób badanych, które brały udział w eksperymencie, oraz ich wiek. Ilość tych osób kształtuje się różnie, spośród omawianych prac najczęściej liczba osób badanych zawiera się w zakresie 5 - 7 osób ([19, 20, 17, 29, 22, 23]), choć pojawiają się również bardziej liczne grupy badanych jak 12 osób ([26]), czy 15 ([27]). W przypadkach [16, 18] badana była tylko jedna osoba, a w przypadku [24] korzystano z bazy sygnałów, w efekcie korzystając z badań przeprowadzonych na 27 osobach. Jeżeli chodzi o wiek, większość badanych mieściła się w przedziale wiekowym 18-28 lat ([16, 20, 17, 29, 22, 30, 25]). Jedyny wyjątek stanowi [26], w którym badani byli w wieku od 17 do 56 lat.

Kolejnym aspektem, jaki należy rozważyć jest odpowiednie przygotowanie miejsca, w którym mają zostać przyczepione elektrody. Skórę golono ([16, 19, 17, 18]), przemywano za pomocą alkoholu lub innej substancji ([16, 19, 20, 17, 18]), oraz\lub dokonywano abrazji ([16, 19]). W przypadku [20] na skórę nałożono żel przewodzący. Warto zaznaczyć, że nie we wszystkich przypadkach podano informacje odnośnie przygotowania skóry.

Innym ważnym elementem jest, w przypadku badań dynamicznych, sposób wykonywania ruchu. Dla [19], dany gest wykonywany był dziesięciokrotnie przez każdą osobę, utrzymywany przez 5 sekund, z 6 sekundowymi przerwami pomiędzy każdym wykonaniem. Z kolei w [20] każdy gest wykonywano piećdziesiąt razy tak, by ruch trwał jedynie 0,5 sekundy, z przerwą pomiędzy każdym wykonaniem około 1 - 2 sekund. W przypadku [17], gdzie mierzono zależność wywieranej siły nacisku i generowanego sygnału SEMG, próba pomiarowa wykonywana była 3-krotnie z jednominutowymi odstępami pomiędzy każdą z prób. Natomiast w pracy [29], gdzie wykonywanym ruchem było uginanie ramienia stojąc, wykorzystano 4 ciężary dla każdej osób (2, 4, 6, 8kg), a pojedynczy cykl ruchu kształtał się jako 2 sekundy

unoszenia ciężaru, 1 sekunda pauzy, 2 sekundy opuszczania ciężaru i 1 sekunda pauzy. Porównując powyższe metody można stwierdzić, że sposób wykonywania ruchów w dużej mierze zależy od samego ruchu, a także od docelowego zastosowania rozwiązania opracowanego w pracy i nie da się jednoznacznie stwierdzić, jak powinien badany ruch przebiegać.

2.3.5. Wyniki pomiarów

Przez wyniki pomiarów rozumie się dokładność (skuteczność) klasyfikacji przy użyciu danego klasyfikatora. W większości przypadków wyniki prezentują się bardzo dobrze, osiągnięta dokładność jest na wysokim poziomie. I tak w przypadku [16], w celu określenia jakości wyznaczonego modelu, określono współczynnik korelacji pomiędzy opracowaną metodą a metodą wzorcową, *Crude Monte Carlo*. Tak obliczony współczynnik korelacji osiągnął wartość minimalną na poziomie 0,986, ze średnią 0,99, przy czym opracowana metoda jest prostsza i wydajniejsza. W przypadku [19] dokładność jaką osiągnięto wynosiła 95.94 - 98.12% dla testów off-line, a średnia dokładność testów on-line wyniosła 97.95%. Interesującym przypadkiem jest [20], w którym porównywano różne klasyfikatory oraz cechy sygnałów, w celu wybrania najlepszego zestawu. W tym przypadku dokładność dla testów off-line wynosi dla metody k-NN 96.17%, dla LDA 95.5%, QDA 95.67%, z kolei dla SVM średnia wartość przy zastosowaniu różnych jąder wyniosła 97%. Ostatecznie w testach on-line zdecydowano się na użycie QDA i z cech sygnały wybrano *Hudgins' time domain features*, a ustalenie to osiągnęło dokładność powyżej 90%. W pracy [17] osiągnięto dla sieci neuronowej dokładność dla różnych osób badanych w przedziale 94.7-98.2%. Gorsze od pozostałych wyniki otrzymano w [18], gdzie dokładność dla LDA wyniosła 55%, a dla QDA 65%. W przypadku [29], gdzie testowano dwie sieci (z 3 neuronami oraz z 50 neuronami w warstwie ukrytej), otrzymane wyniki prezentują się podobnie. Odwzorowanie momentu skręcającego nie jest zbyt zadowalające, ale możliwe jest odróżnienie od siebie ciężarów. W [21] osiągnięto średnią dokładność klasyfikacji za pomocą LDA na poziomie 52.6%, natomiast w [26] wyniki dla testowanych metod nie różnią się od siebie znacząco, stąd też trudno jest stwierdzić, która może być skuteczniejsza. Porównanie przeprowadzone w [22] prezentuje się interesująco: dokładność wszystkich użytych metod jest wysoka - dla SVM 92.25%, Random Forest 87.50%, Naive Bayes 86.83%, LDA 91.67%. Podobnie sytuacja wygląda w [23], gdzie przeprowadzono porównanie dla SVM z zastosowaniem różnych jąder. I tak dokładność dla jądra *Radial Basis* wyniosła 79.36%, wielomianowego - 55.94%, a sigmoidalnego - 53.53%. W [30] osiągnięto dokładność klasyfikacji za pomocą regresji logistycznej na poziomie 90.2%. W przypadku [24] porównywano wyniki klasyfikacji przy użyciu dwóch oraz trzech składowych głównych (po PCA), i w przypadku dwóch dokładność wyniosła 66.6% a w przypadku trzech - 77.3%. Porównanie klasyfikacji za pomocą SVM oraz ANN przeprowadzone w [25] wypadło na korzyść SVM, którego średnia dokładność wyniosła 94.265%. Dokładność dla ANN wyniosła 77.143%. Wynik dla [27], ostatniej z omawianych prac, uwzględnia rozpoznawanie bólu odcinka lędźwiowego, w przypadku którego dokładność dla SVM wyniosła 71.11%, dla ANN 70.37% a dla metody k-NN 61.33%. Z kolei w przypadku rozpoznawania zmęczenia mięśni przedramienia wyniki są nieco lepsze - dokładność dla SVM wyniosła 84.72%, ANN 72.73%, a dla k-NN 73.61%.

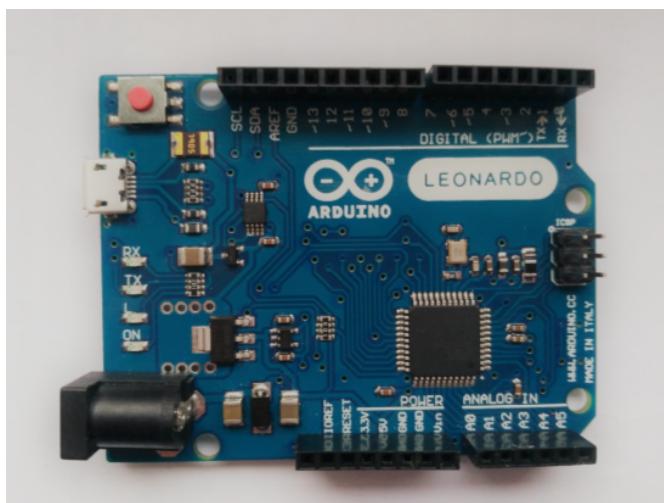
3. Opis systemu

W niniejszym rozdziale omówione zostaną wszystkie elementy wchodzące w skład systemu. Składają się na niego część sprzętowa, czyli hardware, część oprogramowania, czyli program wgrywany do mikrokontrolera i aplikacja komputerowa wraz z klasyfikatorem i pomiarami niezbędnymi do jego wytrenowania.

3.1. Hardware

3.1.1. Mikrokontroler

Mikrokontrolerem zastosowanym w omawianym rozwiązaniu jest Arduino Leonardo. Jego zdecydowanymi zaletami jest łatwość oprogramowania, ze względu na zainstalowany *bootloader*, który pozwala zaprogramować urządzenie posiadając jedynie kabel micro USB oraz IDE (ang. *Integrated Development Environment*) dostarczane przez producenta, oraz niski koszt. Ponadto Arduino, ze względu na swoją popularność, posiada również wiele bibliotek typu *open source*, które znaczco ułatwiają obsługę niektórych peryferiów. **Rys.3.1.** przedstawia zdjęcie płytki, wykorzystanej w opisywanym systemie [31].



Rys. 3.1. Płytki Arduino Leonardo wykorzystana do konstrukcji systemu.

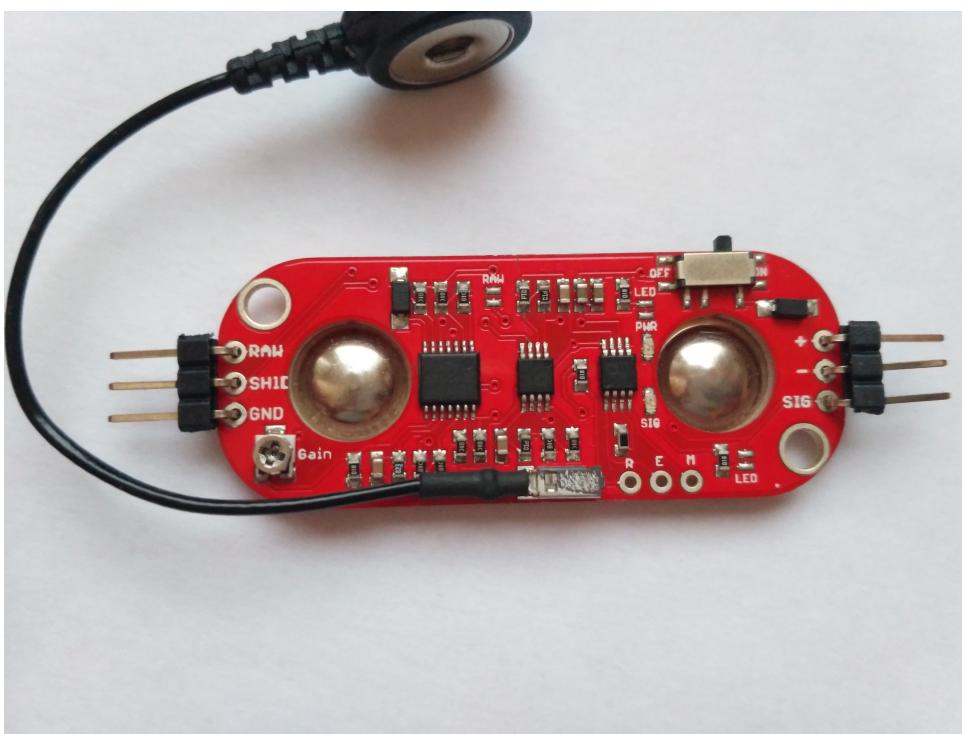
Arduino Leonardo posiada 20 cyfrowych złącz wejścia/wyjścia, z czego 7 z nich posiada możliwość modulacji szerokości impulsu, pozwalając tym samym na zmianę wypełnienia sygnału, a 12 można

użyć jako wejścia analogowe, o 10-bitowej rozdzielczości. Na płytce znajdują się dwa wyjścia zasilania, o napięciu wyjściowym 3.3V oraz 5V [31].

Płytką skonstruowaną została w oparciu o procesor ATmega32u4. Mikroprocesor pracuje na napięciu zasilania zawierającym się w przedziale 7-12V. Prąd pojawiający się na złączach wejścia/wyjścia oraz wyjściach zasilania zawiera się w przedziale 40-50mA. Zastosowany procesor posiada 32kB pamięci Flash, przy czym 4kB są na stałe zajęte przez wspomniany *bootloader*. Ponadto posiada 2.5kB pamięci SRAM (ang. *Static Random Access Memory*) oraz 1kB pamięci EEPROM (ang. *Electrically Erasable Programmable Read-only Memory*). Taktowanie procesora wynosi 16MHz. Atmega32u4 wyposażony jest w cztery timery - dwa 8-bitowe oraz dwa 16-bitowe [31].

3.1.2. Czujnik EMG

W projekcie wykorzystano czujniki elektromiograficzny *MyoWare™ MuscleSensor* (AT – 04 – 001) firmy Advancer Technologies. Podstawową zaletą tego czujnika jest dobra jakość, wystarczająca by spełnić wymagania postawione w założeniach projektu, przy równocześnie stosunkowo niskiej cenie. Zdjęcie czujnika prezentują Rys.3.2. oraz Rys.3.3.



Rys. 3.2. Wierzch czujnika. Widać piny (+/-) napięcia zasilania oraz wyprowadzenia sygnałów.

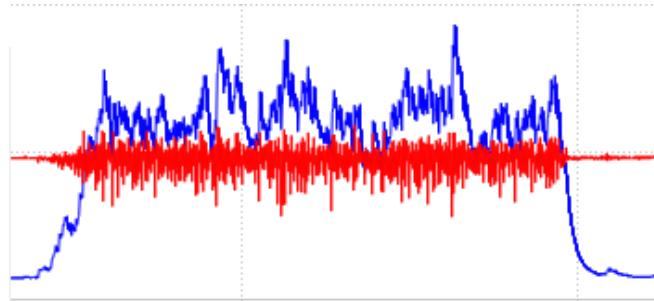
Czujnik posiada trzy miejsca na przypięcie elektrod - dwa znajdujące się na płytce, które doprowadzają sygnał EMG do wzmacniacza różnicowego, oraz pojedynczy przewód, który stanowi odprowadzenie dla elektrody referencyjnej. Napięcie zasilania czujnika zawiera się w przedziale od +3.1V do +6.3V. Ważnym parametrem czujników EMG (jako, że jest to pomiar napięcia) stanowi impedancja wejściowa



Rys. 3.3. Spód czujnika, widoczne są luty oraz miejsca do przyczepienia elektrod.

czujnika, którą sensor MyoWare posiada na dobrym poziomie - producent podaje wartość impedancji wejściowej równą $110\text{G}\Omega$ [32].

MyoWare posiada dwa wyprowadzenia rejestrowanego sygnału. Domyślne wyprowadzenie pozwala na zarejestrowanie obwiedni sygnału EMG (ang. *EMG envelope*). Surowy sygnał jest wzmacniany, prostowany a następnie całkowany (analogowo). W efekcie otrzymywana jest obwiednia sygnału, która pozwala na zaobserwowanie ogólnego zachowania rejestrowanego przebiegu. Drugie wyprowadzenie pozwala na zarejestrowanie wzmacnionego, surowego sygnału EMG. Rys.3.4. przedstawia przykład obwiedni sygnału EMG wraz z odpowiadającym jej sygnałem surowym, zarejestrowane przy pomocy czujnika [32].



Rys. 3.4. Sygnał surowy oraz obwiednia pochodzące z czujnika.

Producent nie podaje dokładnych parametrów układu, przy pomocy którego wyznaczana jest obwiednia sygnału. Z tego powodu projekt opiera się na analizie rejestrowanego surowego sygnału EMG, o znany wzmocnieniu (co zostanie omówione w dalszej części pracy).

Jak zostało wspomniane wcześniej, sygnał EMG jest rejestrowany również i za pomocą tego czujnika jako sygnał różnicowy. Może zatem przyjmować wartości zarówno dodatnie jak i ujemne. Pojawia się więc problem, ponieważ standardowy przetwornik analogowo-cyfrowy wbudowany w mikrokontroler nie jest w stanie rejestrować napięć o wartości poniżej zera. Jako, że czujnik MyoWare został zaprojektowany by współpracować z mikrokontrolerami tego typu, surowy sygnał EMG jest w każdym punkcie powiększony o $\frac{1}{2}V_{zasilania}$, a więc leży w połowie zakresu, gdyż możliwe jest rejestrowanie napięć od 0 do $V_{zasilania}$.

Istnieje możliwość dopasowania wzmocnienia sygnału za pośrednictwem potencjometru umiejscowionego w lewym dolnym rogu płytki. Wzmocnienie (Gain) sygnału dane jest wzorem [32]

$$G = \frac{201R_{gain}}{1k\Omega} \quad (3.1)$$

gdzie:

G -wzmocnienie

R_{gain} - opór potencjometru

Przetwornik analogowo-cyfrowy mikrokontrolera posiada 10-bitową rozdzielcość, a zatem na wejściu rejestrowane są wartości 0-1023. Przeliczenie tej wartości na wartość zmierzonego napięcia, dane jest wzorem:

$$V_{in} = Val \frac{5}{1023} \quad (3.2)$$

gdzie: $Val \in \{0, 1, 2, \dots, 1022, 1023\}$

Czynnik $\frac{5}{1023}$ pozwala na przeliczenie wartości z zakresu 0-1023 na wartość zmierzonego napięcia, która należy do zakresu 0-5V (a w rzeczywistości 0 - $V_{zasilania}$). Aby z kolei przeliczyć tę wartość na rzeczywistą wartość napięcia, należy skorzystać z poniższego wzoru:

$$V_{real} = \frac{V_{in}}{G} \quad (3.3)$$

gdzie: $V_{in} \in <0, V_{zasilania}>$

V_{real} - wartość rzeczywista napięcia

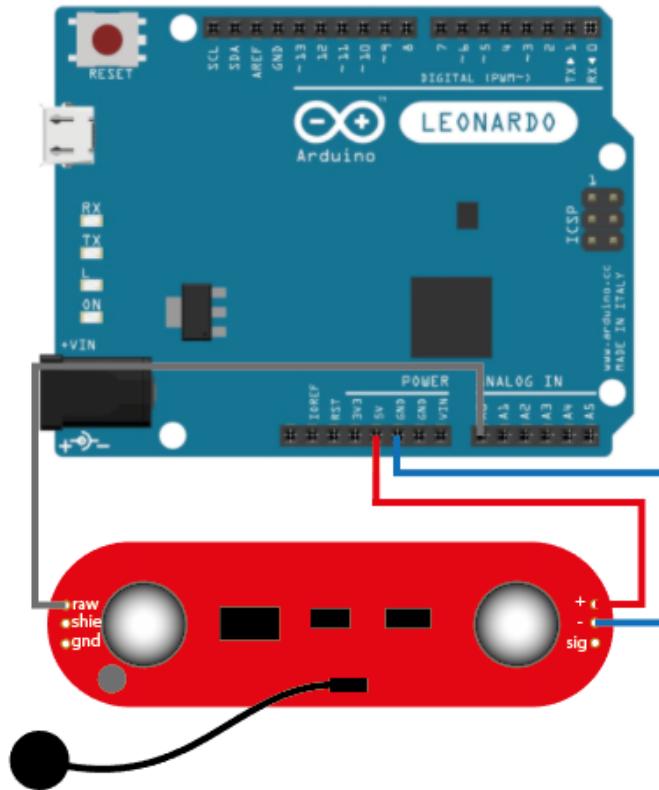
W przypadku niniejszego projektu, nie było konieczne korzystanie z rzeczywistych wartości rejestrowanego napięcia, nie przeliczano zatem zmierzonego napięcia przy użyciu wzoru (3.3), a jedynie korzystając z formuły (3.2).

Dodatkowo, w zależności od potrzeb konstrukcji, istnieje możliwość zamontowania do czujnika dłuższych wyprowadzeń dla elektrod, bądź tzw. *shields*, które mogą służyć jako miejsce przyczepu zewnętrznych elektrod, źródło zasilania (miejsce na baterie), płytka do zmontowania własnego układu "na" sensorze czy indykatorki amplitudy zarejestrowanego sygnału (matryca diod led).

Czujnik nie posiada zamontowanych fabrycznie wyprowadzeń, a widoczne na Rys.3.2. piny zostały do niego przylutowane ręcznie. Ważnym aspektem było zatem staranne wykonanie lutów, gdyż wszelkie skazy mogą zakłócić otrzymywany sygnał, sprawiając, że system nie będzie działał prawidłowo.

MyoWare posiada wbudowany swojego rodzaju indyktor skurczu, pozwalający stwierdzić poprawne ułożenie czujnika jeszcze przed badaniem. Na Rys.3.2. widoczne są dwie diody LED - oznaczone napisami "PWR" oraz "SIG". Dioda "PWR" świeci się w momencie podłączenia zasilania do czujnika, z kolei dioda SIG zapala się gdy zarejestrowany zostanie jakiś sygnał, przy czym intensywność światła odpowiada amplitudzie wykrywanego sygnału.

Schemat połączeń zastosowanych w układzie pomiarowym przedstawiono na Rys.3.5.



Rys. 3.5. Schemat połączeń układu pomiarowego.

Źródło schematu mikrokontrolera: <https://techtaught.files.wordpress.com/2014/09/leonardo1.jpg>, term. wzi. 2017-01-10.

3.2. Oprogramowanie

3.2.1. Arduino IDE oraz Qt

Do zaprogramowania mikrokontrolera skorzystano z dedykowanego Arduino IDE 1.8.5. Jak zostało już wspomniane, zaletą Arduino oraz samego Arduino IDE jest prostota użytkowania - po wybraniu odpowiedniego typu płytka oraz wykryciu jej na porcie USB, można wgrać napisany program za pomocą jednego kliknięcia. Ponadto IDE wyposażone jest we wbudowany monitor portu szeregowego, co pozwala na zaobserwowanie informacji wysyłanych przez płytke. Do napisania programu wykorzystano zalecany język, opierający się na C/C++.

Qt jest to *framework* służący do tworzenia wieloplatformowego oprogramowania, wykorzystujący przede wszystkim języki C++ oraz *Qt Meta Language* (QML). W projekcie zdecydowano się na wykorzystanie tego narzędzia ze względu przede wszystkim na możliwość korzystania z języka C++, który jest językiem wydajnym i w którym istnieje duża ilość bibliotek typu *open-source*, które znaczaco rozszerzają jego możliwości i ułatwiają korzystanie z niego. Ponadto Qt wyposażony jest w *Qt Creator*, będący graficznym narzędziem typu przeciągnij i upuść do tworzenia GUI (ang. *Graphical User Interface*), znacznie przyspieszającym proces kształtowania interfejsu użytkownika. Qt rozszerza możliwości języka C++ o mechanizm sygnałów i slotów, który pozwala na natychmiastową reakcję w przypadku wystąpienia danego zdarzenia. Ponadto istnieje możliwość pobrania i używania Qt za darmo w wersji *open-source* na zasadach licencji GPL & GPLv3. Do stworzenia aplikacji zastosowano Qt w wersji 5.9.2 oraz *Qt Creator* w wersji 4.4.1.

3.2.2. Oprogramowanie mikrokontrolera

Podstawowymi wymogami programu do obsługi mikrokontrolera było uzyskanie odpowiedniej częstotliwości próbkowania, oraz przesyłanie danych sformatowanych w taki sposób, by możliwe było ich poprawne odebranie po stronie aplikacji komputerowej.

Wzorując się na omawianych w poprzednim rozdziale pracach, w których najczęściej występującą częstotliwością próbkowania był 1kHz, również w projektowanym systemie była to docelowa częstotliwość. Jak zostało wspomniane wcześniej, częstotliwości występujące w sygnale EMG zawierają się w przedziale 0 - 500Hz. Zatem przy częstotliwości próbkowania 1kHz jest możliwe pełne odtworzenie sygnału, gdyż spełnia ona twierdzenie Kotielnikowa-Shannona (twierdzenie o próbkowaniu) [33]. W celu jej jak najdokładniejszego osiągnięcia posłużono się mechanizmem przerwań.

Przerwania to mechanizm pozwalający na wstrzymanie aktualnie wykonywanej operacji i rozpoczęcie innej w momencie wystąpienia określonej sytuacji, a po jej zakończeniu powrót do wcześniejszych wykonywanych działań. Gdy rozpoczyna się procedura obsługi przerwania, aktualny stan rejestrów zapisywany jest na stosie, następnie wykonywana jest procedura obsługi przerwania, a po jej zakończeniu wartości odczytywane są ze stosu i poprzedni proces jest kontynuowany. Warto zwrócić tutaj uwagę na konieczność zachowania szczególnej ostrożności w momencie, gdy podczas obsługi przerwania może

ulec zmianie wartość zmiennej zapisanej na stosie. Konieczne może być w takiej sytuacji nadanie owej zmiennej modyfikatora *volatile*, w przeciwnym wypadku po zakończeniu przerwania wartość zmiennej zostanie odczytana ze stosu, a zmiana jakiej w procedurze obsługi przerwania zostanie nadpisana.

Aby uzyskać częstotliwość próbkowania jak najbliższą 1kHz posłużono się przerwaniami pochodząymi od 16-bitowego timera. Licznik każdego timera zwiększa się o 1 na każdy takt procesora. 16-bitowy timer może przechowywać maksymalnie wartość $2^{16} = 65\,536$ (a w zasadzie 65 535, ponieważ liczenie rozpoczyna się od zera). W momencie przekroczenia tej wartości następuje tzw.*overflow* i licznik zostaje wyzerowany. Aby uzyskać częstotliwość próbkowania równą 1kHz należy przeskalać to, jak często inkrementowany jest licznik, czyli zastosować tzw. *prescaler*. Biorąc pod uwagę częstotliwość taktowania procesora (16MHz), częstotliwość występowania przerwań w zależności od dobranego prescalera będzie dana wzorem:

$$f_p = \frac{f_{taktowania_procesora}}{P \cdot N} \quad (3.4)$$

gdzie:

f_p - częstotliwość przerwań

P - prescaler

N - liczba wielokrotności prescalera odpowiadająca danej częstotliwości

W niniejszym systemie zdecydowano się zastosować wartość prescalera równą 64. Oznacza to, że inkrementacja licznika timera nastąpi co 64 takty procesora. W związku z tym, przekształcając powyższą zależność:

$$N = \frac{f_{taktowania_procesora}}{P \cdot f_p} = \frac{16 \cdot 10^6}{64 \cdot 10^3} = 250$$

Jest to wartość, po osiągnięciu której timer, przy tak ustawionym prescalerze, spowoduje wywołanie przerwania (liczenie rozpoczyna się od 0, więc w rzeczywistości wartość ta wynosi 249).

W pierwszej kolejności konieczne było prawidłowe skonfigurowanie timera 1 wewnątrz funkcji *void setup()*, która jest pierwszą funkcją wykonywaną w momencie wgrania programu, tak aby spełniał powyższe założenia. Kod konfigurujący wygląda następująco:

```
// Używany jest timer 1
TCCR1A = 0; TCR1B = 0; TCNT1 = 0; // wyzerowanie rejestru TCCR1A, TCCR1B
// inicjalizacja licznika zerem
OCR1A = 249; // Ustawienie wartości porównywanej z licznikiem na 249
// Włączenie opcji Clear Timer on Compare Match
// timer zostanie wyczyszczony, gdy osiągnie zadaną wartość
TCCR1B |= (1 << WGM12);
TCCR1B |= (1 << CS11) | (1 << CS10); // Ustawienie bitów tak, by otrzymać prescaler 64
TIMSK1 |= (1 << OCIE1A); // Umożliwienie timerowi 1 wywoływanie przerwań
```

Zgodnie z ustaleniami, takie skonfigurowanie timera spowoduje wywoływanie przerwania pochodzącego od tego timera z częstotliwością 1kHz. Aby uzyskać należytą częstotliwość próbkowania należy więc w trakcie pojedynczego przerwania odczytać aktualną wartość sygnału podawanego przez czujnik na piny mikrokontrolera. Przerwanie pochodzące z timera 1 wywołuje domyślną funkcję **ISR(TIMER1_COMPA_vect)**, wewnątrz której następuje próbkowanie sygnału, nadanie próbce znacznika oraz przesłanie za pomocą portu szeregowego. Te czynności wykonuje poniższy kod:

```
float voltageRaw = analogRead(A1); // Odczytanie wartości na pinie A1
voltageRaw = voltageRaw * (5.0 / 1023.0); // Przeliczenie wartości na napięcie
String dataBuffor = String(voltageRaw,DEC); // Konwersja wartości zmiennoprzecinkowej na String
// Usunięcie zbędnych zer powstających po konwersji
if(dataBuffor.length() > 6) {
    dataBuffor.remove(dataBuffor.indexOf('')+3, dataBuffor.length()- dataBuffor.indexOf('')+2);
}
dataBuffor = dataBuffor + "r"; // Dodanie znacznika
Serial.println(dataBuffor); // Przesłanie danych portem szeregowym
```

Znacznik "r" umożliwia odróżnienie od siebie próbek sygnału surowego od próbek sygnału EMG envelope po przesłaniu ich przez port szeregowy. Analogiczne operacje wykonuje się w celu odebrania sygnału EMG envelope, jednakże domyślnie w systemie nie jest on przesyłany. Podzielenie bufora danych ułatwia fakt, że funkcja *Serial.println()* umieszcza dodatkowo na końcu przesyłanej wartości znacznik końca linii. Trudność w tym wypadku stanowi sam przesył danych, ponieważ funkcja *Serial.println()* zajmuje znaczną ilość czasu. W przypadku przesyłania tylko wartości sygnału surowego, jest ona wystarczająca i nie spowalnia działania układu, jednak przy przesyłaniu dodatkowo wartości sygnału EMG envelope, częstotliwość próbkowania maleje. Stąd ważnym aspektem było ustalenie odpowiedniej wartości *baud rate*, czyli prędkości przesyłu danych w bitach na sekundę. Została ona wyznaczona eksperymentalnie, gdyż w zależności od jej wartości można było zauważać zmiany w prędkości transmisji, tzn. nie następowała ona z częstotliwością 1kHz. W związku z tym przyjęto wartość *baud rate* równą 115200, dla której nie odnotowano spadku prędkości przesyłu danych.

Jako, że całość potrzebnych działań odbywa się w procedurze obsługi przerwania, główna funkcja Arduino IDE, *void loop()*, pozostaje pusta.

3.2.3. Aplikacja komputerowa

3.2.3.1. Obsługa aplikacji

Program, jak wspomniano wcześniej, został zaimplementowany przy użyciu Qt, w związku z czym podstawowym plikiem programu jest plik o rozszerzeniu .pro. Plik ten zawiera wszystkie informacje, których potrzebuje tzw. *qmake* do zbudowania aplikacji. Sam *qmake* z kolei automatyzuje i upraszcza

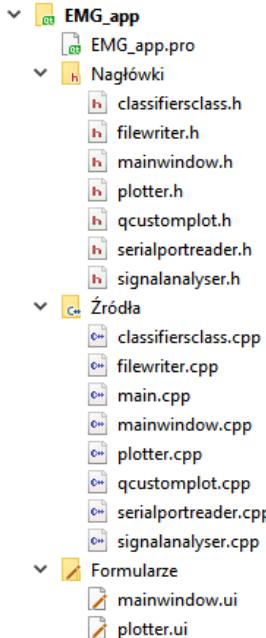
generowanie tzw. *Makefiles*, potrzebnych w procesie komplikacji programu. W pliku .pro przekazywanych jest kilka informacji. Podstawowymi są informacje odnośnie używanych w aplikacji modułów, plikach nagłówkowych i źródłowych aplikacji znajdujących się w tej samej lokalizacji co plik .pro, plikach zawierających informacje odnośnie interfejsu graficznego oraz ścieżek do np. zewnętrznych bibliotek używanych w projekcie. Przykładowy plik .pro został zaprezentowany na **Rys.3.6. (a)**.

```

1 #-----
2 #
3 # Project created by QtCreator 2017-12-06T09:59:4
4 #
5 #-----
6
7 QT      += core gui
8
9 greaterThan(QT_MAJOR_VERSION, 4): QT += widgets
10
11 TARGET = example
12 TEMPLATE = app
13
14 DEFINES += QT_DEPRECATED_WARNINGS
15
16 SOURCES += \
17     main.cpp \
18     mainwindow.cpp
19
20 HEADERS += \
21     mainwindow.h
22
23 FORMS += \
24     mainwindow.ui

```

(a) Przykładowy plik .pro



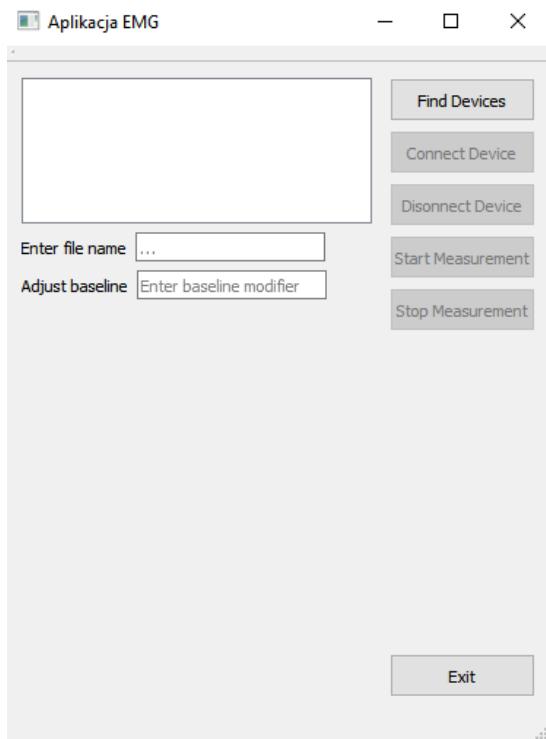
(b) Struktura aplikacji.

Rys. 3.6

Na stworzoną aplikację składają się dwa widoki stanowiące interfejs użytkownika oraz sześć klas: *Classifier*; *FileWriter*; *Plotter*; *SerialPortReader*; *SignalAnalyser* oraz *MainWindow*, zajmujące się obsługą interfejsu oraz przetwarzaniem i klasyfikacją sygnału. **Rys.3.6. (b)** przedstawia strukturę aplikacji.

W katalogu "Źródła" na Rys.3.5. (b) można zauważyc obecność pliku main.cpp. W pliku tym znajduje się funkcja *main()*, która odpowiada za start całego programu i wyświetlenie części interfejsu, wiadomościowej w strukturze projektu, o nazwie "mainwindow.ui", będącej pierwszym widocznym po uruchomieniu oknem. Jego wygląd został przedstawiony na **Rys. 3.7**.

W funkcji *main()* w pierwszej kolejności tworzona jest instancja klasy *MainWindow*, co jest równoznaczne z wywołaniem jej konstruktora. W konstruktorze tworzona jest instancja głównego okna, tworzone jest po jednym obiekcie dla każdej z pozostałych klas głównych wymienionych wcześniej, oraz instancja klasy *QSerialPort*, służąca do komunikacji z portem szeregowym. Wszystkie przyciski zamieszczone w interfejsie po kliknięciu emitują sygnał *clicked()* i każdy z nich połączony jest z osobnym slotem *on_xxxButton_clicked()*, w których wykonywane są odpowiednie operacje. W konstruktorze



Rys. 3.7. Główne okno programu.

wykonywana jest jeszcze jedna ważna operacja - instancja klasy *FileWriter* odczytuje z pliku "extracted_features.txt" dane treningowe dla klasyfikatora. Dane te są następnie wykorzystywane jako argument wejściowy konstruktora obiektu klasy *ClassifierClass*.

Po uruchomieniu programu użytkownik może wcisnąć przycisk *Exit*, który spowoduje wyłączenie programu, lub przycisk *Find Devices*. W momencie wybrania drugiego z wymienionych przycisków, następuje znalezienie wszystkich aktualnie dostępnych urządzeń podłączonych do portów USB komputera oraz wypisanie ich na liście według ich nazw. Do tego celu używana jest klasa *QSerialPortInfo* umożliwiająca uzyskanie podstawowych informacji dotyczących podłączonego urządzenia. W przypadku, gdy znalezione zostało jakieś urządzenie, przycisk *Connect Devices* staje się dostępny. Wciśnięcie go (możliwe dopiero po wybraniu pozycji z listy) powoduje nawiązanie połączenia pomiędzy urządzeniem a aplikacją. Ustawiane są tutaj takie parametry jak nazwa portu, tryb w jakim następuje otwarcie połączenia (w tym przypadku *ReadOnly*), *BaudRate*, *Parity*, *StopBits* czy *Flow Control*. Udane nawiązanie połączenia jest sygnalizowane informacją na pasku stanu.

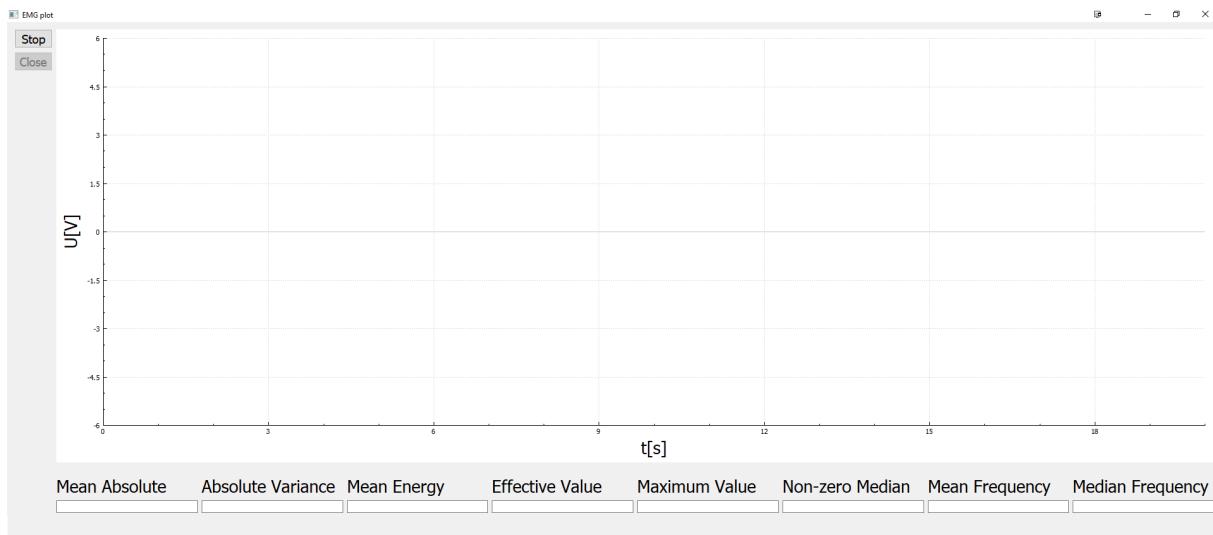
Następnie możliwe staje się wciśnięcie przycisków *Start Measurement* oraz *Disconnect Device*. Wciśnięcie drugiego z wymienionych przycisków powoduje rozłączenie urządzeń. Po wciśnięciu pierwszego z kolei następuje połączenie odpowiednich sygnałów oraz slotów, co umożliwia wreszcie rozpoczęcie pomiaru.

Najważniejszym z połączeń jest połączenie pomiędzy sygnałem *readyRead()* emitowanym przez instancję klasy *QSerialPort* w momencie, gdy w buforze pojawią się dane do odczytu, ze slotem *readSerial()* obiektu klasy *MainWindow*. Wewnątrz tego slotu odbierane są dane przesyłane za pośrednictwem

portu szeregowego i wywoływanie są metody obiektów innych klas odpowiadających za rysowanie wykresu, obliczanie parametrów sygnału oraz jego klasyfikację w czasie rzeczywistym.

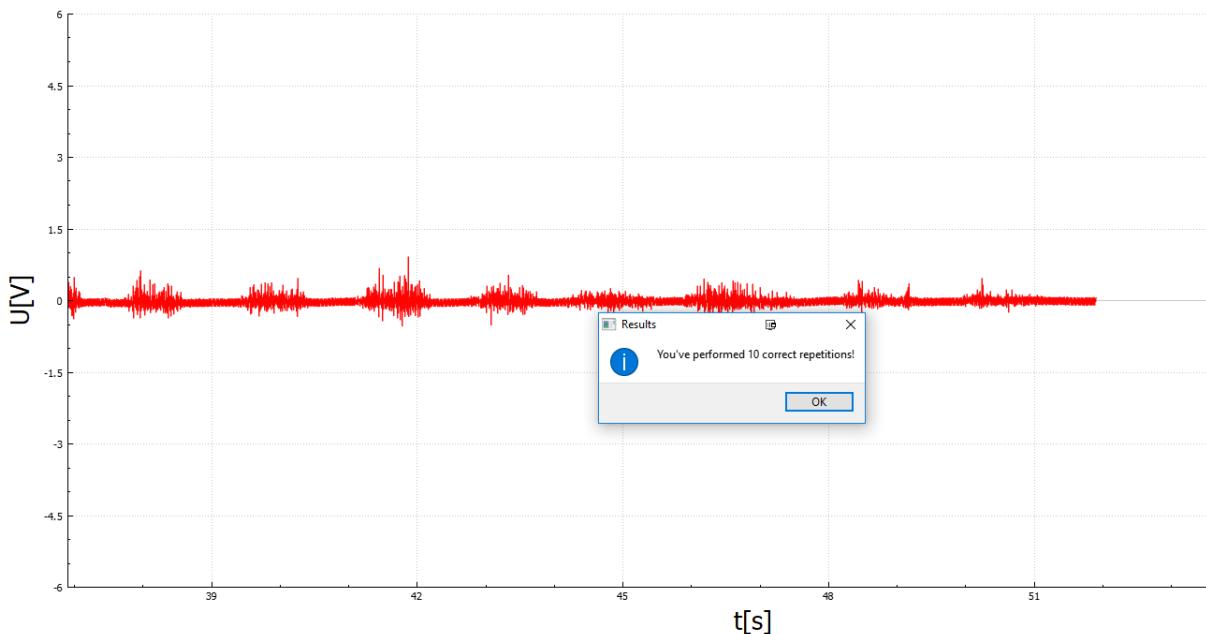
Ostatnim elementem okna głównego są pola oznaczone jako "Enter file name" oraz "Adjust baseline". W pierwszym należy umieścić nazwę pliku, do którego zapisane zostaną dane pomiarowe. Drugi z kolei jest szczególnie istotny z perspektywy uzyskania dobrych wyników klasyfikacji sygnału. Jak zostało wspomniane, surowy sygnał EMG pochodzący z czujnika posiada dodaną składową stałą równą $\frac{1}{2}V_{zasilania}$. Niestety, w rzeczywistości napięcie zasilania z USB niekoniecznie jest równe 5V i może się różnić pomiędzy urządzeniami. Pole *Adjust baseline* pozwala na ustalenie przez użytkownika wartości składowej stałej, która zostanie odjęta od rejestrowanego sygnału. Jej wyznaczenie jest stosunkowo proste - wystarczy przed wykonaniem właściwych pomiarów wykonać pomiar testowy, w którym nie będzie wykonywana żadna aktywność ruchowa, równocześnie wpisując w pole *Adjust baseline* wartość 0. W efekcie uzyskany zostanie przebieg o stałej wartości napięcia, która to jest równa składowej stałej sygnału.

Jak zostało wspomniane, wcisnięcie przycisku *Start Measurement* zainicjalizuje pomiar. W efekcie wyświetlony zostanie drugi widok aplikacji, przedstawiony na **Rys.3.8.**, na którym w czasie rzeczywistym rysowany jest wykres.



Rys. 3.8. Okno z wykresem.

Po wciśnięciu przycisku *Stop* w oknie wykresu następuje zatrzymanie pomiaru i wyświetlenie okna prezentującego wyniki, to znaczy ile poprawnych powtórzeń ćwiczenia udało się wykonać badanej osobie. Okno wynikowe zostało przedstawione na **Rys.3.9**.



Rys. 3.9. Okno z wynikami ćwiczeń.

3.2.3.2. Funkcjonalność klas oraz schemat blokowy

Każda z klas służy do wykonywania innych czynności. Zostaną one opisane krótko w kilku kolejnych akapitach.

Klasa *FileWriter* odpowiada za odczyt danych treningowych używanych następnie do wytrenowania klasyfikatora, do czego służy metoda *readTrainingData()*. Ponadto przy użyciu tej klasy tworzone są pliki .txt zawierające dane pomiarowe z konkretnego pomiaru, do czego służą metody *makeNewFile()* oraz *writeToFile()*.

Klasa *Plotter* jest klasą odpowiadającą za utworzenie (*makePlot()*) oraz rysowanie wykresu (*updatePlot()*, *shiftPlot()*). Dodatkowo metoda *updateSignalParams()* jest odpowiedzialna za odświeżanie aktualnych parametrów sygnału wyświetlanych pod wykresem. Do tworzenia oraz zarządzania wykresem w czasie rzeczywistym wykorzystano bibliotekę *QCustomPlot* na licencji *GNU General Public License*, dostarczającą szereg funkcji znacznie ułatwiających obsługę wykresów w Qt w czasie rzeczywistym. Jako, że w celu uproszczenia transmisji za pomocą portu szeregowego przesyłane są tylko wartości sygnału pochodzące z czujnika bez ich oznaczenia czasowego, wektor czasu generowany jest sztucznie, z odstępem 1 milisekundy dla próbki, co daje częstotliwość równą 1kHz.

Klasa *SerialPortReader* służy do odpowiedniego formatowania danych wejściowych za pomocą metody *parseSerialData()*. Metoda ta jest niezwykle ważna z punktu widzenia programu, ponieważ to wewnątrz niej dane są odpowiednio formatowane i dodawane do bufora, który następnie służy do zapisu danych do pliku. Ponadto aktualizuje ona zawartość wektorów na podstawie których rysowany jest wykres, oraz to wewnątrz niej wywoływana jest metoda odpowiadająca za obliczanie chwilowych parametrów sygnału.

Klasa *SignalAnalyser* ma za zadanie obliczanie chwilowych parametrów sygnału w danym oknie czasowym. Jako, że w trakcie obliczania parametrów konieczne jest skorzystanie z podstawowego algorytmu FFT (ang. *Fast Fourier Transform*), konieczne było, aby okno czasowe zawierało ilość próbek równą potęzej liczby 2. Szerokość okna ustalono eksperymentalnie jako 512 próbek. Podstawową metodą niniejszej klasy jest *getSignalParams()*, która odpowiada za policzenie wszystkich chwilowych parametrów, wywołując kolejno metody odpowiedzialne za liczenie każdego z nich i zapisując je do wektora parametrów. Więcej na temat obliczanych parametrów zostanie powiedziane w rozdziale 3.3.

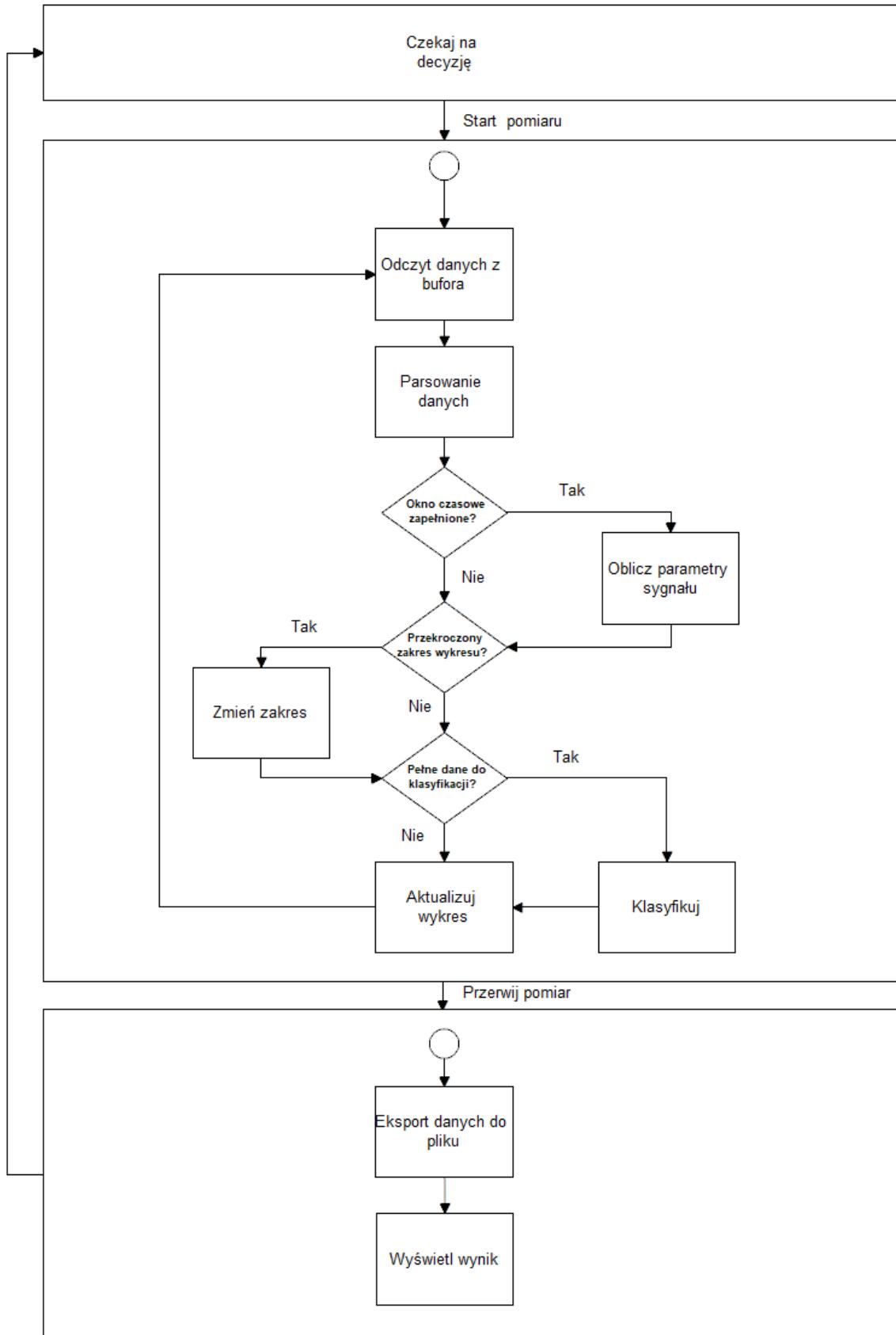
Ostatnia klasa, *ClassifierClass*, służy do klasyfikowania sygnału w czasie rzeczywistym. Podstawę tej klasy stanowi biblioteka *open-source* o nazwie *dlib*. W bibliotece tej zaimplementowanych zostało wiele klasyfikatorów wraz z niezbędnymi mechanizmami, jak funkcje uczące czy normalizujące, posiada również szereg funkcji służących do przetwarzania obrazów, obsługi wielu wątków, parsowania tekstów, optymalizacji. Dlib jest używany przez wielu użytkowników i instytucji, jak np. przez MIT (The Massachusetts Institute of Technology). Do klasyfikacji wykorzystano klasyfikator SVM z funkcją jądra w postaci radialnej funkcji bazowej, ze względu na dobre wyniki klasyfikatora tego typu w omawianych wcześniej pracach. Wewnątrz konstruktora klasy *ClassifierClass* następuje wytrenowanie klasyfikatora oraz obliczenie funkcji normalizującej wektory danych wejściowych, w celu uniknięcia sytuacji, w której jeden z parametrów będzie dominował nad innymi. Zawarte w klasie metody *normalizeSamples* oraz *classifySignalUsingSVM* służą odpowiednio do znormalizowania wektora danych wejściowych a następnie jego zaklasyfikowania. Zestawienie wszystkich klas wraz z ich najważniejszymi metodami i funkcjonalnością przedstawia **Tabela 3.1**.

Dodatkowo w programie zaimplementowano mechanizm, który utrudnia zaklasyfikowanie pojedynczego skurcza jako dwóch powtórzeń ćwiczenia - zapamiętuje on wyniki trzech poprzednich klasyfikacji, i jeśli któryś z nich został zaklasyfikowany jako przypadek +1, uniemożliwia inkrementację licznika powtórzeń. Mając na uwadze że rzeczywisty czas wykonywania pojedynczego powtórzenia wynosi ok. 1,5s, taka blokada nie powinna wpływać negatywnie na wyniki pomiarów.

Przedstawione w powyższej tabeli zestawienie jest tylko uogólnieniem funkcjonalności poszczególnych klas. Nie zawiera wszystkich metod należących do poszczególnej klasy oraz nie opisuje w pełni ich funkcjonalności. Na diagramie z **Rys. 3.10.** przedstawiono uogólniony schemat blokowy programu, z pominięciem niektórych czynności prowadzących do uruchomienia pomiaru.

Tabela 3.1. Zestawienie klas i ich metod wraz z zastosowaniem.

Klasa	Metoda	Zastosowanie
FileWriter	makeNewFile()	Utworzenie nowych plików, w których zapisane będą dane wejściowe
	writeToFile()	Zapis danych wejściowych do pliku
	readTrainingData()	Odczyt danych treningowych dla klasyfikatora
Plotter	makePlot()	Utworzenie wykresu o zadanych parametrach
	updatePlot() / shiftPlot()	Rysowanie wykresu w czasie rzeczywistym / skalowanie zakresu
	updateSignalParams()	Odświeżanie tymczasowych parametrów sygnału wyświetlanych pod wykresem
SerialPortReader	parseSerialData()	Parsowanie danych i zapis do bufora, aktualizacja danych wykresu
SignalAnalyser	getSignalParams()	Obliczanie parametrów sygnału w czasie rzeczywistym dla zadanego okna
ClassifierClass	ClassifierClass() (konstruktor)	Obliczenie funkcji normalizującej, wytrenowanie klasyfikatora
	classifyWithSVM()	Klasyfikowanie wektora danych przy pomocy SVM



Rys. 3.10. Schemat blokowy programu.

3.3. Klasyfikacja sygnału

Niezwyczajnym elementem systemu dla jego poprawnego działania jest wybór cech sygnału, na podstawie których będzie on klasyfikowany, dobór klasyfikatora oraz przeprowadzenie pomiarów, które pełnić będą rolę zbioru uczącego. Niniejszy podrozdział dokładnie omawia wszystkie wspomniane elementy.

3.3.1. Protokół pomiarowy

W pozyskaniu sygnałów wzorcowych wzięła udział grupa piętnastu ochotników płci męskiej. Od każdego z nich zebrano parametry opisowe, którymi były: wiek, wzrost, waga, BMI (ang. *Body Mass Index*), obwód rozluźnionego ramienia, obwód ramienia przy napiętym bicepsie oraz obwód rozluźnionego przedramienia. Zestawienie parametrów osób badanych przedstawia **Tabela 3.2**.

Tabela 3.2. Parametry osób badanych.

Nr	Wiek [lata]	Wzrost [cm]	Waga [kg]	BMI	Obwód ramienia [cm]		Obwód przedramienia [cm]
					relaxed	flexed	
1	49	178	74	23,35	29	33,5	27
2	23	175	78	25,47	34	38	30
3	21	183	80	23,89	30	34	26
4	21	174	66	21,8	29	33	27
5	21	176	72	23,24	26	29	25
6	21	180	65	20,06	25	28	24
7	23	174	68	22,46	30	33	28
8	22	174	89	29,4	35	38	31
9	24	180	79	24,38	31	36	29
10	23	179	75	23,41	32	34	28
11	22	186	85	24,56	29	33	28
12	22	180	67	20,68	26	28	25
13	28	193	94	25,24	30	35	28
14	22	187	100	28,6	34	36	28
15	22	183	90	26,87	38	41	31

Przed rozpoczęciem jakichkolwiek pomiarów, od osoby badanej zbierano parametry opisowe. Określono wiek w latach, zmierzono wzrost z dokładnością do 1cm, wagę z dokładnością do 1kg, oraz zmierzono obwody z dokładnością do 1cm. BMI obliczono na podstawie wzoru 3.5 [34].

$$BMI = \frac{masa}{wzrost^2} \quad (3.5)$$

gdzie

$$[masa] = 1\text{kg}$$

$$[wzrost] = 1\text{m}$$

Następnie badanej osobie demonstrowano ćwiczenie, które miała za zadanie wykonywać, czyli ugiwanie ramienia ze sztangielką stojąc, z jednosekundowym okresem unoszenia i jednosekundowym okresem opuszczania ciężaru, z niewielką przerwą pomiędzy powtórzeniami, dziesięciokrotnie. Zwracano uwagę na poprawną technikę wykonywania ćwiczenia, szczególnie na to, by łokieć w trakcie ruchu przylegał do żeber, oraz by ruch przebiegał od pełnego wyprostu do kąta prostego między ramieniem i przedramieniem. Początkowa i końcowa faza ruchu została przedstawiona na **Rys.3.11**.



(a) Początkowa faza ruchu

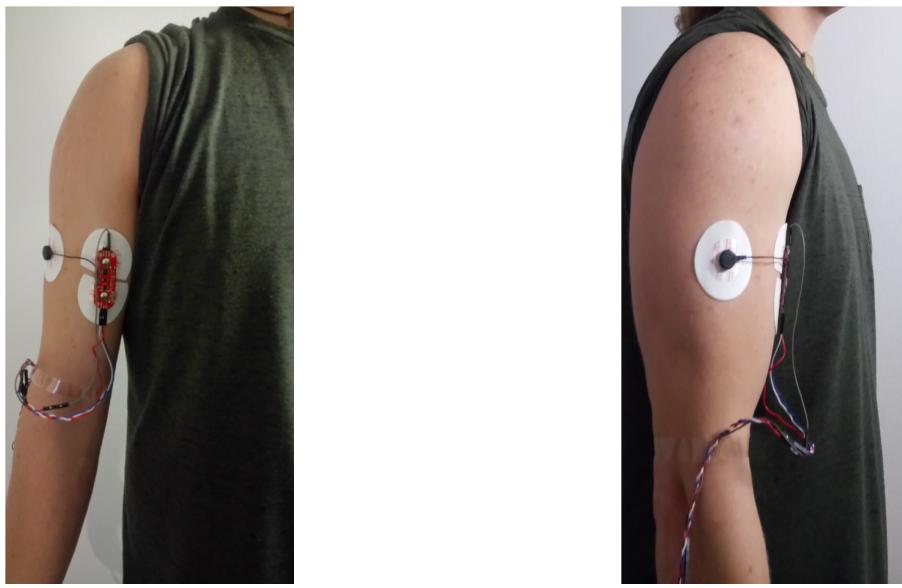


(b) Końcowa faza ruchu

Rys. 3.11. Przebieg ćwiczenia wykonywanego w trakcie pomiaru.

Następnie badany otrzymywał kilka minut na przećwiczenie ruchu z ciężarem 1kg. Kolejnym krokiem było przygotowanie skóry badanego pod elektrody. Wzorując się na protokołach badań przedstawionych w omawianych pracach, skórę najpierw umyto wodą, następnie za pomocą pasty abrazyjnej usunięto martwy naskórek (aż do pojawienia się zaczerwienienia) i przemyto alkoholem. Na tak przygotowane miejsce przyklejono elektrody, zgodnie ze wskazaniami wspomnianymi w rozdziale 2.1, a więc w oddaleniu od czynnej masy mięśnia, w osi ramienia od wewnętrznej strony. Przewody przymocowano za pomocą taśmy klejącej do skóry tak, by nie krępowały one ruchów. Po przyklejeniu elektrod badany miał chwilę czasu na wykonanie analizowanych ruchów, w celu odpowiedniego ułożenia elektrod na skórze. Mocowanie czujnika zostało przedstawione na **Rys.3.12**. Podczas pomiarów korzystano z mokrych elektrod Ag/AgCl, przymocowanych bezpośrednio do odpowiednich miejsc na czujniku. Elektroda referencyjna (na przewodzie) została zamocowana do mięśnia niepracującego podczas ćwiczenia.

Po przeprowadzeniu powyższych przygotowań, rozpoczęto pomiary. Jako, że celem było zebrać sygnałów, a nie ich klasyfikacja, nie było konieczne modyfikowanie linii bazowej (wyznaczenie $1/2V_{zasilania}$). Osoba badana wykonywała ćwiczenie dwukrotnie - najpierw z ciężarem 3kg, a następnie z ciężarem 5kg, z 10 minutami odpoczynku pomiędzy próbami, używając swojej dominującej ręki



Rys. 3.12. Umiejscowienie czujnika w trakcie badania.

(w tym przypadku wszyscy badani byli praworęczni). Tak otrzymane sygnały następnie skatalogowano i ekstrahowano z nich cechy potrzebne do wyuczenia klasyfikatora.

3.3.2. Cechy sygnału i ich ekstrakcja

Po przeanalizowaniu prac omówionych w rozdziale 2.3 zdecydowano się wybrać sześć cech sygnału obliczanych w dziedzinie czasu, oraz dwie cechy z dziedziny częstotliwości, a więc w sumie osiem cech sygnału. Są to:

- Średnia wartość bezwzględnej amplitudy sygnału

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N |x_i| \quad (3.6)$$

- Wariancja sygnału

$$\sigma_x = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (3.7)$$

- Moc średnia sygnału

$$P_x = \frac{1}{N} \sum_{i=1}^N x_i^2 \quad (3.8)$$

- Wartość skuteczna

$$x_{sk} = \sqrt{P_x} \quad (3.9)$$

- Wartość maksymalna

$$x_{max} = \max(x_1, x_2, x_3, \dots, x_N) \quad (3.10)$$

- Mediana wartości niezerowych

$$m_e = \frac{x_{\frac{i}{2}} + x_{\frac{i+1}{2}}}{2} \quad (3.11)$$

gdzie: $2i : i \in \mathbb{N}_+ \wedge x_i \neq 0$

$$m_e = x_{\frac{i+1}{2}} \quad (3.12)$$

gdzie: $2i + 1 : i \in \mathbb{N}_+ \wedge x_i \neq 0$

- Średnia częstotliwość (MNF, ang. *Mean Frequency*) [35]

$$MNF = \sum_{j=1}^M f_j P_j / \sum_{j=1}^M P_j \quad (3.13)$$

- Mediana częstotliwości (MDF, ang. *Median Frequency*) [35]

$$\sum_{j=1}^{MDF} P_j = \sum_{j=MDF}^M P_j = \frac{1}{2} \sum_{j=1}^M P_j \quad (3.14)$$

gdzie:

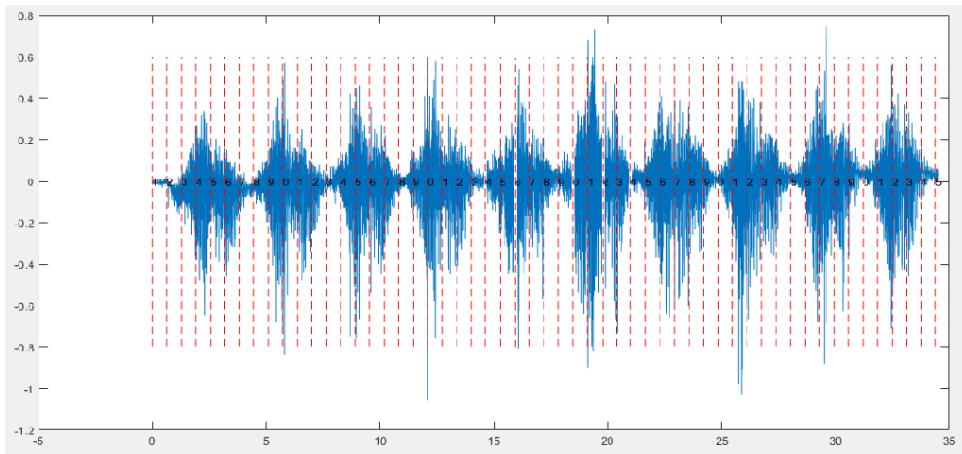
P_j - moc widma dla prążka częstotliwości j

f_j - częstotliwość odpowiadająca prążkowi częstotliwości j

Jak zostało wspomniane w rozdziale 3.2 cechy sygnału są ekstrahowane w klasie *SignalAnalyser*. Każdy parametr obliczany jest przy użyciu osobnej funkcji, wywoływanej w funkcji *getSignalParams()*. Do ekstrakcji zastosowano nienakładające się (ang. *non-overlapping*) okna czasowe o szerokości 512 próbek. Obliczone parametry są przechowywane w wektorze *signalParams*. W celu obliczenia parametrów w dziedzinie częstotliwości, konieczne było wyznaczenie transformaty Fouriera sygnału w czasie rzeczywistym. Do tego celu napisano funkcję obliczającą standardowe DFT (ang. *Discrete Fourier Transform*), jednakże jej zastosowanie powodowało widoczne opóźnienia w działaniu programu. Konieczne było więc zaimplementowanie algorytmu FFT (ang. *Fast Fourier Transform*), opierającego się na algorytmie Cooleya-Tukeya [36]. Jego użycie wyeliminowało występujące opóźnienia.

Mediana częstotliwości sygnału jest to taka wartość częstotliwości, która dzieli moc widma sygnału na dwie równe części [35]. Ze względu na fakt, że sygnał dla którego przeprowadzane są obliczenia jest sygnałem dyskretnym, w wyniku transformaty Fouriera otrzymywane jest również dyskretne widmo częstotliwościowe. Obliczenie dokładnej wartości mediany częstotliwości jest więc trudne, ponieważ z dużym prawdopodobieństwem leży ona pomiędzy dwiema dyskretnymi wartościami, i jej wyznaczenie wymagałoby zastosowania aproksymacji przebiegu widma częstotliwościowego. Celem uproszczenia obliczeń, za medianę częstotliwości przyjmowana jest zatem pierwsza dyskretna wartość, dla której moc widma przekroczyła połowę swojej wartości.

Do ekstrakcji cech wykorzystano program Matlab 2015 (MathWorks), w którym zaimplementowano funkcję do obliczania każdej z cech sygnału w taki sam sposób jak w napisanej aplikacji komputerowej. Sygnał podzielono losowo na odcinki składające się z 512 próbek i dla każdego z nich obliczono parametry. W celu określenia przynależności danego wektora uczącego do odpowiedniej grupy (-1, +1), wyświetlane były wykresy dla sygnałów wraz z nałożonymi ponumerowanymi liniami dzielącymi go na wspomniane odcinki. Następnie ręcznie przyporządkowano wektory do określonej grupy. Przykładowy wykres z nałożonymi liniami podziału przedstawia **Rys.3.13.**



Rys. 3.13. Wykres z nałożonymi liniami podziału.

3.3.3. Uczenie klasyfikatora

Jak wcześniej wspomniano, proces uczenia klasyfikatora wykonuje się w konstruktorze klasy *ClassifierClass*, której instancja tworzona jest w konstruktorze klasy *MainWindow*. Najpierw obiekt klasy *FileReader* odczytuje dane treningowe z pliku. Każdy wektor treningowy jest w nim zapisany w postaci pojedynczej linii, a poszczególne cechy oraz grupa do której dany wektor należy, przedzielone są średnikami. W pierwszej kolejności wprowadzane dane są normalizowane i dopiero na ich podstawie trenowany jest klasyfikator. Następnie funkcja normalizująca przypisywana jest do klasyfikatora, i to ona normalizuje każdy wektor wejściowy. Ostatnim ważnym elementem jest określenie parametru γ oraz parametru C, które mówią o tym, do jakiego stopnia można rozszerzyć margines miękkiego klasyfikatora. Parametry te określone są iteracyjnie za pomocą walidacji krzyżowej. W każdej iteracji wprowadzane są inne parametry i wykonywana jest k -krotna kroswalidacja, polegająca na tym, że zbiór danych wejściowych dzielony jest na k części, następnie jedną z nich wykorzystuje się jako zbiór testowy a pozostałe jako zbiór uczący. W efekcie kroswalidacja wykonywana jest k razy, za każdym biorąc inny podzbiór za zbiór testowy a pozostałe za uczący. Otrzymane wyniki dla każdego podzbioru są uśredniane aby uzyskać pojedynczy wynik. W tym przypadku zastosowano 5-krotną kroswalidację, a do tego celu wykorzystano wszystkie zebrane sygnały. Następnie wybierane są te wartości, dla których klasyfikator osiągnął największą dokładność. W przypadku omawianego systemu, parametr $\gamma = 0.78125$, a parametr $C = 25$. Tak stworzony klasyfikator jest gotowy do pracy i przeprowadzania pierwszych testów.

4. Testy i omówienie wyników

W celu określenia dokładności skonstruowanego systemu, zdecydowano się przeprowadzić dwojakie testy - *off-line*, wykorzystujące wszystkie zebrane sygnały i sprawdzające dokładność klasyfikatora metodą kroswalidacji, określając jak klasyfikator radzi sobie z czystą klasyfikacją sygnału, a więc jaki procent próbek został zaklasyfikowany poprawnie, oraz testy *on-line*, sprawdzające jak dobrze system jest w stanie zliczać ilość wykonanych powtórzeń, a więc jego odporność na błędy typu zaklasyfikowanie pojedynczego skurcza jako dwa lub więcej powtórzeń.

Testy *off-line* podzielone zostały na trzy części. W pierwszej z nich do sprawdzenia klasyfikatora zastosowano jedynie sygnały zebrane podczas wykonywania ćwiczenia z obciążeniem 3kg, w drugiej te z obciążeniem 5kg, a w trzeciej wszystkie pomiary połączono. W każdej z części zbiór danych trzykrotnie podzielono losowo na zbiór uczący, stanowiący 80% ogółu zebranych pomiarów, oraz zbiór walidacyjny, stanowiący 20% zebranych pomiarów, i dla każdego podziału przeprowadzono test, wyniki uśredniając. W tym celu posłużono się funkcjami dostarczonymi wraz ze wspomnianą wcześniej biblioteką dlib, pozwalającymi na losowe ułożenie próbek w wektorze danych oraz przetestowanie przy ich pomocy klasyfikatora. W efekcie otrzymano wyniki dla każdej z trzech części, z których każdy jest średnią trzech wyników. Dla każdego z testów obliczony został procent (dokładność) przypadków +1 (skurcz) zaklasyfikowanych poprawnie, oraz procent przypadków -1 (mięsień w rozkurczu) zaklasyfikowanych poprawnie. Rezultaty testów *off-line* przedstawia **Tabela 4.1**.

Tabela 4.1. Wyniki testów *off-line*

Grupa sygnałów ze względu na obciążenie	Średnia dokładność dla +1 [%]	Średnia dokładność dla -1 [%]
3kg	92,59	86
5kg	95,51	95,01
3kg i 5kg	95,36	93,11

Następnie przeprowadzone zostały testy *on-line*, w których wzięło udział trzech ochotników nienależących do grupy, z której zbierano sygnały. Każdy z nich wykonywał ćwiczenie ośmiokrotnie, w późniejszych konfiguracjach:

- Klasyfikator wyuczony przy użyciu sygnałów z pomiarów z ciężarem 3kg, używając cięzarów 1kg i 3kg
- Klasyfikator wyuczony przy użyciu sygnałów z pomiarów z ciężarem 5kg, używając cięzarów 1kg, 3kg i 5kg
- Klasyfikator wyuczony przy użyciu sygnałów z pomiarów z ciężarami 3kg i 5kg, używając cięzarów 1kg, 3kg i 5kg.

Podczas pomiarów postępowano zgodnie z protokołem pomiarowym przedstawionym w rozdziale 3.3.1., a więc w każdym pomiarze osoba wykonywała 10 powtórzeń analizowanego ćwiczenia. Zbiór uczący klasyfikatora stanowiła całość pomiarów dla danego ciężaru. W tym przypadku za miarę jakości przyjęto formułę daną *wzorem 4.1.*

$$B = \frac{|L - L_R|}{L_R} \cdot 100\% \quad (4.1)$$

gdzie:

B - procent błędów

L - ilość powtórzeń zliczonych przez program

L_R - ilość rzeczywiście wykonanych powtórzeń.

Osoba ćwicząca stawała w pozycji rozluźnionej, z ciężarem trzymanym w ręce dominującej, w taki sposób, aby nie widziała ekranu komputera. Następnie w zadany tempie wykonywała zadaną ilość powtórzeń ćwiczenia, skupiając się na prawidłowej technice jego wykonania. Po wykonaniu ćwiczenia z ciężarem 1kg następowały 2 minuty przerwy (ze względu na znakomity wysiłek potrzebny do jego wykonania), a w przypadku cięzarów 3kg i 5kg badana osoba miała 10 minut na odpoczynek przed następną serią. Po zakończeniu każdej serii zapisywano wynik (zliczoną ilość powtórzeń) wyświetlany na ekranie komputera i porównywano go z rzeczywistą ilością wykonanych powtórzeń, zgodnie ze *wzorem 4.1.*

Wyniki testów *on-line* przedstawią **Tabela 4.2..**

Uzyskane wyniki *off-line* wskazują na dobrą zdolność klasyfikacyjną SVM, przy czym klasyfikator lepiej radzi sobie z rozpoznawaniem skurczu mięśnia, o czym świadczy większa dokładność dla kategorii +1 we wszystkich analizowanych przypadkach. Równocześnie wyniki pomiędzy każdym z rozważanych przypadków nie różnią się w znacznym stopniu, co świadczy o dobrej jakości zebranych sygnałów treningowych. Dużym problemem, zarówno podczas rejestracji sygnałów treningowych jak i testów okazały się artefakty pochodzącego od ruchu kabla oraz przemieszczania się czujnika, które znaczco utrudniały prawidłową ekstrakcję cech z sygnałów oraz miały wpływ na skuteczność klasyfikacji, zarówno w testach *off-line* jak i *on-line*.

W przypadku pomiarów *on-line* zastosowano trzy różne ciężary: 1kg, 3kg i 5kg. Miało to umożliwić stwierdzenie, czy klasyfikator wyuczony przy użyciu sygnałów pochodzących z ćwiczeń z większymi

Tabela 4.2. Wyniki testów *on-line*

Nr osoby badanej	Grupa sygnałów użyta do wyuczenia klasyfikatora	Ciążar [kg]	Ilość zliczonych powtórzeń	Błąd [%]
1	3kg	1	7	30
		3	10	0
	5kg	1	0	100
		3	10	0
		5	10	0
	3kg i 5kg	1	6	40
		3	10	0
		5	10	0
	3kg	1	6	40
		3	10	0
2	5kg	1	0	100
		3	9	10
		5	10	0
	3kg i 5kg	1	5	50
		3	9	10
		5	10	0
	3kg	1	4	60
		3	10	0
	5kg	1	0	100
		3	10	0
		5	10	0
3	3kg i 5kg	1	6	40
		3	10	0
		5	11	10

obciążeniami, jest w stanie poprawnie klasyfikować również mniejsze obciążenia. Wyniki wskazują jasno, że klasyfikator znacznie lepiej radzi sobie z klasyfikacją ciężarów odpowiadających ciężarom użytym podczas rejestracji sygnałów treningowych. Używając ciężaru 1kg, w żadnym z przypadków powtórzenia nie były rejestrowane w pełni prawidłowo, jak należałoby oczekiwac, przy czym im większa różnica między ciężarem zastosowanym w sygnałach treningowych i ciężarem użytym w czasie testów, tym większy błąd klasyfikacji. Oznacza to, że możliwe jest określenie, na podstawie sygnału EMG, czy zastosowany ciężar jest odpowiedni dla osoby ćwiczącej.

Testy z użyciem ciężarów odpowiadających tym wykorzystanym podczas rejestracji danych sygnałów treningowych osiągnęły niewielki błąd, nieprzekraczający 10%, którego przyczyną mogą wspołmniane wcześniej artefakty.

Otrzymane wyniki testów są porównywalne z wynikami pomiarów, jakie uzyskano w pozycjach literaturowych omawianych w rozdziale 2.3. Zastosowany klasyfikator SVM osiągnął średnią dokładność w testach *off-line* powyżej 92% dla przypadków +1 oraz powyżej 85% dla przypadków -1, co potwierdza zauważoną na podstawie przeglądu literaturowego skuteczność maszyny wektorów wspierających w problemach klasyfikacji sygnału EMG, oraz wskazuje na trafny dobór parametrów sygnału do rozważanego zastosowania.

Założony cel pracy został z powodzeniem zrealizowany - zaimplementowano sprawnie działający system nadzorujący przebieg ćwiczeń fizycznych w czasie rzeczywistym.

5. Podsumowanie i perspektywy rozwoju

Skonstruowany system spełnia wszystkie założenia projektu, co sprawia, że nie odstaje on znacznie od rozwiązań zaprezentowanych w przeglądzie literaturowym. Osiągnięta częstotliwość próbkowania 1kHz jest częstotliwością wystarczającą do zastosowania, ponieważ spełnia twierdzenie Shannona-Kotielnikowa dla sygnału EMG, a równocześnie stosowaną najczęściej we wszystkich omawianych w rozdziale 2.3 pracach. Również płynność wizualizacji wyników jest na zadowalającym poziomie. Rozmiar grupy badawczej, na której przeprowadzono pomiary, jest większy niż w większości omawianych pozycji literaturowych - najczęściej ilość osób badanych zawierała się w przedziale 5-7 osób, a w niniejszej pracy wyniosła ona 15. Otrzymane wyniki testów klasyfikacji *off-line* są porównywalne z tymi w omawianych pracach dla klasyfikatora SVM - wyniosły one powyżej 92% dla wykrywania skurczu, co nie budzi zastrzeżeń, oraz powyżej 85% dla wykrywania rozkurczu, co jest wynikiem gorszym, ale wciąż mieszczącym się w zakresie dokładności maszyny wektorów wspierających w przywoływanych pracach. Jedynym elementem zaproponowanego rozwiązania odstającym od zaprezentowanych prac, jest liczba analizowanych kanałów, czyli jeden. W pozycjach omawianych w rozdziale 2.3 najczęściej pojawiającą się liczbą kanałów było trzy, jednakże do zaproponowanej, uproszczonej wersji systemu, jeden kanał jest wystarczający, choć lepsze odwzorowanie rzeczywistej pracy mięśni można osiągnąć w przypadku większej ilość kanałów.

Największą zaletą proponowanego rozwiązania jest sam pomysł - spośród wszystkich omawianych w rozdziale 2.3 pracach, żadna z nich nie proponuje zastosowania sygnału EMG do celów nauki odpowiedniej techniki wykonywania ćwiczeń z wykorzystaniem ciężarów. Zadowalającym aspektem systemu jest jego wydajna praca w czasie rzeczywistym - wykonywanie obliczeń oraz klasyfikacja sygnału przebiegają w sposób płynny, nie zauważono żadnych opóźnień w pracy programu nawet przy dłuższym czasie pomiaru. W żadnej z przywołanych pozycji literaturowych nie posłużono się w implementacji rozwiązania bardzo wydajnym językiem C++ oraz frameworkiem *Qt* (biorąc pod uwagę tylko te prace, w których zaznaczono z jakich języków programowania/technologii korzystano). Ich zastosowanie wymagało samodzielnego zaimplementowania algorytmów do obliczania wszystkich używanych do klasyfikacji sygnałów parametrów oraz algorytmu FFT.

Dużym problemem okazały się artefakty pochodzące od ruchu przewodów oraz przemieszczania się czujnika. Rozwiązaniem pierwszego z nich mogłoby być zastosowanie przewodów lepszej jakości lub zaimplementowanie systemu w taki sposób, by przesył danych przeprowadzany był bezprzewodowo. Problem czujnika zmieniającego swoje położenie jest trudniejszy do rozwiązania, gdyż nie występuje

u każdej osoby badanej. W przypadku osób o mniejszej masie mięśniowej, zmiana kształtu mięśnia jest na tyle niewielka, że zakłócenia sygnału są pomijalne. U osób o większej masie mięśniowej odkształcenie to jest znaczne i powoduje zdecydowanie większe zakłócenia. Problem ten może wynikać z faktu, że elektrody w zastosowanym czujniku są ze sobą sztywno połączone - wyprowadzenie elektrod za pomocą przewodów w taki sposób, by podczas ćwiczenia zmiana ich położenia była od siebie niezależna, najprawdopodobniej rozwiązałoby powstał problem.

Nie bez znaczenia dla jakości systemu była również grupa badawcza, z której zebrano sygnały - w większości były to osoby nie uprawiające regularnie ćwiczeń z użyciem ciężarów, co bezpośrednio przekładało się na ich siłę (a co za tym idzie amplitudę rejestrowanych sygnałów), oraz technikę wykonywania ćwiczeń. W przypadku chęci rozszerzania systemu, grupę, z której zbierane są sygnały treningowe, powinny stanowić osoby o poprawnej technice wykonywania ćwiczeń oraz świadome swoich możliwości siłowych, co pozwoli na zarejestrowanie sygnałów dobrych pod względem techniki wykonywanego ćwiczenia oraz dobranego ciężaru do możliwości ćwiczącego.

Opracowany system jest jedynie podstawową wersją zamysłu prezentowanego we wstępie niniejszej pracy. Ilość zastosowanych kanałów w układzie pomiarowym pozwala jedynie na wykrycie skurczu oraz rozkurczu mięśnia, co uniemożliwia zastosowanie systemu do sprawdzania poprawności wykonywanych ćwiczeń. W przypadku badanego w niniejszej pracy ćwiczenia, minimalna liczba kanałów potrzebna do docelowego zastosowania to dwa, co pozwalałoby na monitorowanie aktywności mięśnia antagonistycznego (mięśnia trójdługiego ramienia) do mięśnia dwugłowego ramienia. W przypadku większych grup mięśniowych oraz ruchów złożonych konieczne byłoby zastosowanie jeszcze większej liczby kanałów, ponieważ często angażują one większą liczbę grup mięśniowych, których aktywność ma znaczenie dla poprawności techniki wykonywanego ćwiczenia. Bezpośrednio wiąże się z tym konieczność zebrania większej ilości pomiarów, by móc rozszerzyć bazę ćwiczeń, oraz bazę sygnałów treningowych. Rozszerzenie działania systemu wymaga przede wszystkim zmiany części sprzętowej - czujników o większej czułości, filtracji analogowej, oraz DAQ o lepszych parametrach.

Uzyskane wyniki testów świadczą o tym, że dalszy rozwój systemu może okazać się sukcesem i proponowane rozwiązanie może znaleźć rzeczywiste zastosowanie, zgodne z tym zaproponowanym we wstępie pracy.

Bibliografia

- [1] Risser W. L. „Weight-training injuries in children and adolescents.” W: *American Family Physician* 44.6 (1991), s. 2104–2108.
- [2] König M i Biener K. „Sport-specific injuries in weight lifting.” W: *Schweizerische Zeitschrift Fur Sportmedizin* 38.1 (1990), s. 25–30.
- [3] Neviaser T.J. „Weight lifting. Risks and injuries to the shoulder.” W: *Clinics In Sports Medicine* 10.3 (1991), s. 615–621.
- [4] „Upper extremity weightlifting injuries: Diagnosis and management”. W: *Journal of Orthopaedics* 15.1 (2018), s. 24 –27. ISSN: 0972-978X. URL: <http://www.sciencedirect.com/science/article/pii/S0972978X17301332?urldate={2018-01-10},author=>.
- [5] C. J. De Luca. *Surface Electromyography: Detection and Recording*. DelSys Incorporated, 2002. URL: https://www.delsys.com/Attachments_pdf/WP_SEMGintro.pdf (term. wiz. 2017-11-22).
- [6] P. Konrad. *ABC EMG: praktyczne wprowadzenie do elektromiografii kineziologicznej*. Technomex Spółka z o.o., 2007. ISBN: 9788392081814.
- [7] S. Konturek i T.M. Brzozowski. *Fizjologia ogólna, krew i mięśnie*. Fizjologia człowieka. Wydawnictwo Uniwersytetu Jagiellońskiego, 2003. ISBN: 9788323316282.
- [8] P. Augustyniak. *Przetwarzanie sygnałów elektrodiagnostycznych*. AGH Uczelniane Wydawnictwa Naukowo-Dydaktyczne, 2001. ISBN: 9788388408328.
- [9] J.V. Basmajian i De Luca C.J. *Muscles Alive (5th Edition)*. Williams & Wilkins Company, 1985. URL: <https://www.delucafoundation.org/get-involved/read/> (term. wiz. 2017-11-21).
- [10] Redfern M.S, Hughes R.E. i Chaffin D.B. „High-pass filtering to remove electrocardiographic interference from torso EMG recordings”. W: *Clinical Biomechanics* 8.1 (1993), s. 44 –48. ISSN: 0268-0033.
- [11] Guohua Lu i in. „Removing ECG noise from surface EMG signals using adaptive filtering”. W: *Neuroscience Letters* 462.1 (2009), s. 14 –19. ISSN: 0304-3940.
- [12] S.Y. Kung. *Kernel Methods and Machine Learning*. Cambridge University Press, 2014. ISBN: 9781139867634.

- [13] A. Ben-Hur i in. „Support Vector Machines and Kernels for Computational Biology”. W: *PLoS Computational Biology* 4.10: e1000173 (2008), s. 1–10.
- [14] Horzyk A. *Maszyna wektorów nośnych*. Wykład. Akademia Górnictwo-Hutnicza, WEAIIB, 2017. URL: <http://home.agh.edu.pl/~horzyk/lectures/miw/MIW-SVM.pdf> (term. wiz. 2017-11-21).
- [15] Chih-Wei H., Chih-Chung C. i Chih-Jen L. *A Practical Guide to Support Vector Classification*. National Taiwan University, Department of Computer Science, 2016. URL: <https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf> (term. wiz. 2017-11-22).
- [16] Gaofeng W. i in. „A Wavelet-Based Method to Predict Muscle Forces From Surface Electromyography Signals in Weightlifting”. W: *Journal of Bionic Engineering* 9.1 (2012), s. 48 –58. ISSN: 1672-6529.
- [17] Songyuan Z. i in. „Muscle Strength Assessment System Using sEMG-Based Force Prediction Method for Wrist Joint”. W: *Journal of Medical and Biological Engineering* 36.1 (2016), s. 121–131. ISSN: 2199-4757.
- [18] Shuxiang G. i in. „An EMG-based Muscle Force Evaluation Method Using Approximate Entropy”. W: *Proceedings of 2016 IEEE International Conference on Mechatronics and Automation*. (Harbin, Chiny). IEEE, 2016.
- [19] Nianfeng W., Kunyi L. i Xianmin Z. „Design and Myoelectric Control of an Anthropomorphic Prosthetic Hand”. W: *Journal of Bionic Engineering* 14.1 (2017), s. 47 –59. ISSN: 1672-6529.
- [20] Chen X. i Jane Wang Z. „Pattern recognition of number gestures based on a wireless surface EMG system”. W: *Biomedical Signal Processing and Control* 8.2 (2013), s. 184 –192. ISSN: 1746-8094.
- [21] Anbin X. i in. „Classification of Gesture based on sEMG Decomposition: A Preliminary Study”. W: *IFAC Proceedings Volumes* 47.3 (2014). 19th IFAC World Congress, s. 2969 –2974. ISSN: 1474-6670.
- [22] Bian F., Li R. i Liang P. „SVM based simultaneous hand movements classification using sEMG signals”. W: *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*. (Takamatsu, Japonia). 2017.
- [23] Pomboza-Junez G. i Terriza J. H. „Hand gesture recognition based on sEMG signals using Support Vector Machines”. W: *2016 IEEE 6th International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*. (Berlin, Niemcy). 2016.
- [24] Isaković M. S., Miljković N. i Popović M. B. „Classifying sEMG-based hand movements by means of principal component analysis”. W: *2014 22nd Telecommunications Forum Telfor (TELFOR)*. (Belgrad, Serbia). 2014.
- [25] Shen J. i in. „A study on sEMG signals pattern recognition of key hand motions”. W: *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. (Shenzhen, Chiny). 2013.

- [26] Kahl L. i Hofmann U. G. „Comparison of algorithms to quantify muscle fatigue in upper limb muscles based on sEMG signals”. W: *Medical Engineering & Physics* 38.11 (2016), s. 1260 – 1269. ISSN: 1350-4533.
- [27] Sharawardi N. S. A. i in. „Single channel sEMG muscle fatigue prediction: An implementation using least square support vector machine”. W: *Information and Communication Technologies (WICT), 2014 Fourth World Congress on*. (Bandar Hilir, Malezja). 2014.
- [28] Mercado-Medina E. L. i in. „Design of an electronic system for monitoring muscle activity in weight-lifting”. W: *2014 III International Congress of Engineering Mechatronics and Automation (CIIMA)*. (Cartagena, Kolumbia). 2014.
- [29] R. Oboe i in. „Weight estimation system using surface EMG armband”. W: *2017 IEEE International Conference on Industrial Technology (ICIT)*. (Toronto, Kanada). 2017.
- [30] Cene V. H. i Balbinot A. „Upper-Limb Movement Classification Through Logistic Regression sEMG Signal Processing”. W: *2015 Latin America Congress on Computational Intelligence (LACCI)*. (Kurytyba, Brazylia). 2015.
- [31] Oficjalna strona producenta Arduino ze specyfikacją Arduino Leonardo. URL: <https://www.arduino.cc/en/Main/arduinoBoardLeonardo&sa=D&ust=1465255537100000&usg=AFQjCNFoomViIHvbQESYdewaM8sQyjNoWg> (term. wiz. 2017-12-03).
- [32] MyoWare Datasheet. URL: https://github.com/AdvancerTechnologies/MyoWare_MuscleSensor/blob/master/Documents/AT-04-001.pdf (term. wiz. 2017-12-03).
- [33] Richard G. Lyons. *Understanding Digital Signal Processing*. 1st. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1996. ISBN: 0201634678.
- [34] „Body Mass Index”. W: *Encyclopedia of Psychopharmacology*. Wyed. I. P. Stolerman i L. H. Price. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, s. 308–308. ISBN: 978-3-642-36172-2.
- [35] S. Thongpanja i in. „Mean and Median Frequency of EMG Signal to Determine Muscle Force Based on Time-Dependent Power Spectrum”. W: *Elektronika ir Elektrotechnika* 19 (mar. 2013), s. 51–56.
- [36] Uwe Meyer-Baese. „Fourier Transforms”. W: *Digital Signal Processing with Field Programmable Gate Arrays*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, s. 209–238. ISBN: 978-3-662-04613-5.

A. Dodatek - zawartość dołączonej płyty CD

- Tekst pracy w formacie pdf
- Kod źródłowy aplikacji komputerowej
- Kod źródłowy oprogramowania mikrokontrolera
- Kod wykorzystany w programie Matlab
- Zebrane dane pomiarowe
- Zestaw zbiorów uczących wyekstrahowanych z sygnałów pomiarowych