AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

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

KATEDRA INFORMATYKI STOSOWANEJ

Praca dyplomowa magisterska

Opracowanie prototypu interfejsu dla gier wykorzystujących pętlę afektywną

Development of a prototype interface for games based on the affective loop

Autor:	*Kamil Osuch*
Kierunek studiów:	*Informatyka*
Opiekun pracy:	*prof. dr hab. inż. Grzegorz J. Nalepa*

Kraków, 2019

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 rozpowszechnia 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, wideogram 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 profesorowi Grzegorzowi J. Nalepie oraz doktorowi Krzysztofowi Kuttowi za cierpliwość i poświęcony czas podczas tworzenia tej pracy. Jestem wdzięczny także rodzicom, bez wsparcia których nie dotarłbym tu gdzie jestem.

Spis treści

1. Wstęp

W ciągu ostatnich lat pojęcie sztucznej inteligencji przestało być tylko fenomenem, który dla większości społeczeństwa istniał pod postacią filmów lub powieści dotyczących maszyn posiadających świadomość. Od tamtego momentu człowiek zaczął znajdować zastosowanie sztucznej inteligencji w coraz większej liczbie dziedzin. Od maszyn przetwarzających w sposób automatyczny ogromne ilości informacji, aż po systemy gromadzące dane dotyczące użytkowników i na ich podstawie generujące reguły, dzięki którym możliwe jest znalezienie rozwiązań i porad dopasowanych do danego użytkownika.

Najistotniejszy jest w tym fakt, że człowiek otacza się sztuczną inteligencją, nawet tego nie zauważając. Systemy rekomendujące, dopasowujące produkty z każdej tematyki do preferencji użytkowników [1], urządzenia nasobne monitorujące nasz stan zdrowia dzięki pomiarom parametrów życiowych i zbieraniu informacji na temat naszych nawyków [2], czy coraz popularniejsze systemy autonomicznej jazdy [3]. Sztuczna inteligencja z dnia na dzień coraz bardziej wnika w każdy aspekt życia człowieka.

Jednym z elementów odgrywającym ważną rolę w życiu człowieka, który coraz mocniej oparty jest o sztuczną inteligencję, są gry komputerowe. Choć początkowo były one traktowane wyłącznie jako rozrywka, to dziś coraz częściej są określane nawet jako forma nauki konkretnych umiejętności [4]. Przykładem mogą być tutaj gatunek gier strategicznych, które uczą taktyki i zarządzania, a także gry logiczne pozwalające na rozwój logicznego myślenia. Warto również wspomnieć o grach poważnych [5], które nie skupiają się na aspektach rozrywkowych, a bardziej są formą edukacji i szkoleń przedstawionych w formie interaktywnej symulacji.

Pojęciem, które jest coraz szerzej widoczne w kontekście sztucznej inteligencji związanej przede wszystkim z tematyką komunikacji człowiek-komputer jest informatyka afektywna. Chociaż samo pojęcie istnieje już od ponad 20 lat [6], to dopiero w ciągu kilku ostatnich badania w tej tematyce stały się popularne [7]. Początkowo główną ideą tej dziedziny były systemy zbierające i analizujące dane na temat stanów emocjonalnych użytkowników. Ponieważ emocje nie mogą być kontrolowane poprzez działania człowieka, a jednocześnie można je opisać między innymi przy pomocy zmian fizjologicznych w ludzkim ciele, wykorzystanie informatyki afektywnej w systemach inteligentnych takich jak aplikacje rekomendujące czy systemy badające stan zdrowotny użytkowników pozwala na zwiększenie ich skuteczności działania.

W podobny sposób powstała próba powiązania dziedziny informatyki afektywnej z grami komputerowymi, tworząc nowy rodzaj gier, nazywanych grami afektywnymi. Główną ich ideą jest pomiar stanów emocjonalnych wywoływanych na użytkowniku w trakcie rozgrywki oraz dostosowywanie gry w czasie

rzeczywistym do odczytanych reakcji gracza, tak aby zwiększyć doznania płynące z gry [8]. Dzięki temu każda gra może zostać w pewien sposób spersonalizowana na podstawie indywidualnych cech użytkownika.

Celem pracy jest opracowanie dwuczęściowego interfejsu umożliwiającego pomiar sygnałów pozwalających na określenie zmian stanów emocjonalnych gracza. Interfejs ma posłużyć do opracowania prototypów gier zawierających pętlę afektywną. W skład interfejsu będą wchodzić:

– zbiór urządzeń umożliwiających pomiary sygnałów wykorzystanych do określenia zmian emocji gracza

– moduł przygotowany w środowisku do tworzenia gier, który na podstawie zgromadzonych z urządzeń pomiarowych sygnałów będzie określał stan emocjonalny i zachowania użytkownika.

Ważnym elementem pracy jest stworzenie gry zawierającej pętlę afektywną [9], która będzie wykorzystywała przygotowany interfejs. Po wykryciu zachowań oraz zmian stanów emocjonalnych użytkownika stan gry jest aktualizowany.

Niniejsza praca składa się z 8 rozdziałów. W rozdziale 2 zostały przedstawione podstawy teoretyczne dotyczące informatyki afektywnej. Szczególny nacisk położono na tematykę gier afektywnych, prezentując wybrane istniejące rozwiązania, problemy i kierunki badań z tego zakresu. Rozdział 3 zawiera opis oraz analizę dostępnych sprzętowych platform pomiarowych, możliwych mechanizmów wnioskowania oraz narzędzi wykorzystywanych do budowy gier komputerowych. Ważnym elementem jest przedstawienie wad i zalet każdego z rozwiązań w kontekście tematyki pracy. Rozdział 4 jest podsumowaniem analizy platform sprzętowych z poprzedniego rozdziału. Przedstawione tu zostały podstawowe założenia, jakie powinny być spełnione przez wybraną grupę urządzeń pomiarowych, a także zdefiniowano sprzęt wybrany podczas końcowej implementacji. W rozdziale 5 opisany został proces budowy modelu do rozpoznawania emocji. Omówione zostały wybrane zbiory danych, sposób ich przetwarzania, oraz budowa i wybór końcowego modelu na podstawie działania z dostępnymi danymi. Rozdział 6 jest jedną z najistotniejszych części pracy. Przedstawiono w nim proces implementacji utworzonej gry komputerowej z pętlą afektywną. Skupiono się na opisie interfejsu łączącego rozwiązania wybrane w poprzednich rozdziałach i sposobie jego wykorzystania wewnątrz gry. Następnie przedstawione zostały mechaniki pokazujące, w jaki sposób przygotowany moduł może wpłynąć na rozgrywkę tak, by sprzężenie zwrotne mogło zostać zamknięte. W rozdziale 7 opisany został sposób ewaluacji stworzonego rozwiązania oraz charakterystyka i proces przeprowadzonych eksperymentów. Ostatni rozdział stanowi podsumowanie niniejszej pracy. Zawiera wnioski dotyczące przygotowanego projektu, jego mocne i słabe strony. Opisane zostały także możliwe kierunki dalszego rozwoju stworzonego rozwiązania.

2. Informatyka afektywna

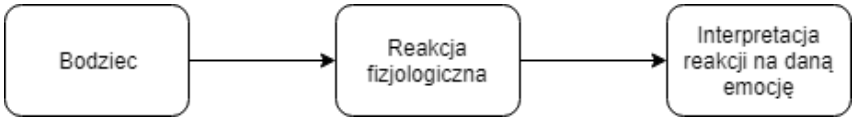
Informatyka afektywna (ang. *affective computing*) jest dziedziną informatyki, w której obliczenia są powiązane z emocjami lub bezpośrednio na nie wpływają [6]. Głównym celem informatyki afektywnej jest rozpoznawanie oraz analiza emocji ludzkich, możliwość ich symulacji przez komputer, a także wpływanie na emocje użytkownika poprzez konkretne bodźce. Aby uzyskać taki efekt, informatyka afektywna jest silnie powiązana z dziedzinami takimi jak psychologia, fizjologia, czy kognitywistyka [10].

2.1. Emocje

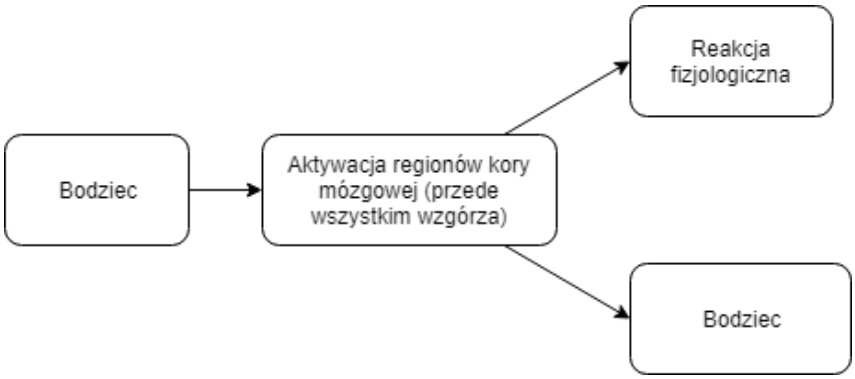
Choć do dzisiaj nie ma jednej, uniwersalnej definicji czym są emocje, to środowisko naukowe ciągle przeprowadza badania na tematy powiązane z emocjami. Już w XIX wieku powstała teoria opracowana niezależnie przez Williama Jamesa oraz Carla Lange'a, gdzie emocję zdefiniowano jako interpretację reakcji cielesnej na zaobserwowany bodziec [11]. Obaj badacze przyjęli, że na konkretny czynnik człowiek reaguje najpierw reakcją fizjologiczną. Następnie zostaje ona przez niego przypisana do wzorca odpowiadającego danej emocji. Dla przykładu, jeśli człowiek znajdzie się w sytuacji, w której zobaczy zagrożenie, zaczyna się trząść i pocić. Jednocześnie jego tętno gwałtownie wzrasta, mózg natomiast interpretuje to jako strach.

Teoria Jamesa-Lange'a została zakwestionowana w latach dwudziestych XX wieku przez dwójkę naukowców, Waltera Cannona i Phillipa Barda. Zasugerowali oni, że odczuwanie emocji nie jest zależne od reakcji fizjologicznych, a raczej są to reakcje zachodzące jednocześnie jako odpowiedź na dany bodziec [12]. Koncepcja ta była bezpośrednim zakwestionowaniem badań przeprowadzonych przez Jamesa i Lange'a. Jej autorzy przeprowadzili badania na kotach, na podstawie których przedstawili, iż to wzgórze jest obszarem mózgu odpowiedzialnym za reakcje emocjonalne na doświadczane bodźce. Cannon zauważył, że całkowite odcięcie wszystkich układów od mózgu nie zmienia zachowania emocjonalnego zwierząt, co kłóciło się z teorią Jamesa-Lange'a, według której koty powinny przestać wykazywać jakiekolwiek reakcje emocjonalne.

Teoria Jamesa-Lange'a została ponownie podjęta przez Prinza, który przedstawił więcej dowodów na potwierdzenie tej teorii, jednocześnie opisując argumenty pokazujące, że reakcje fizjologiczne nie zawsze są wystarczające lub w ogóle niepotrzebne do odczuwania emocji [13]. Zwrócił on między innymi uwagę na badania Hohmanna nad pacjentami z urazami rdzenia kręgowego [14], w których zauważono



(a) Teoria Jamesa-Lange'a, źródło: opracowanie własne na podstawie [11]



(b) Teoria Canona-Barda, źródło: opracowanie własne na podstawie [12]

Rys. 2.1. Graficzne porównanie teorii emocji Jamesa-Lange'a i Canona-Barda

efekty zarówno potwierdzające, jak i podające w wątpliwość teorię Jamesa-Lange'a. Hohmann zaobserwował, że u osób z urazem można dostrzec redukcję w odczuwaniu emocji, ale niektóre z nich wciąż były widoczne. Prinz komentuje to jako zdolność mózgu do przewidzenia reakcji fizjologicznej na dany bodziec, co z kolei będzie skutkować odczuwaniem emocji. Jest to możliwe dzięki wcześniejszym doświadczeniom. Tak jak człowiek tracący wzrok, jest w stanie wyobrazić sobie przedmiot, który kiedyś widział, tak samo mózg potrafi określić jaka reakcja fizjologiczna mogła nastąpić na dany bodziec.

2.2. Modele emocji

Ponieważ emocje są pojęciem abstrakcyjnym, ich pomiar oraz analiza w informatyce afektywnej nie jest prosta i wymaga przyjętego modelu pozwalającego na jednoznaczny pomiar emocji. Na przestrzeni lat powstało wiele modeli opisujących emocje, które można podzielić na dwa główne typy [15]:

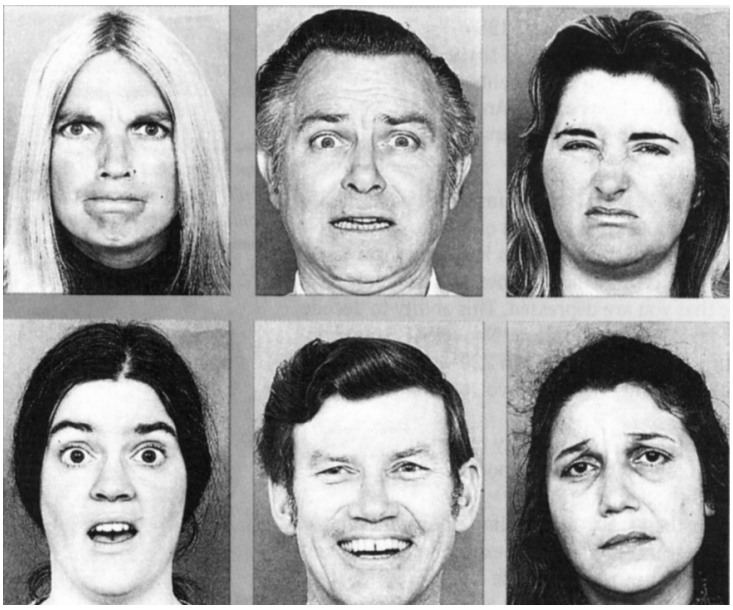
– modele kategoryzowane, w których określony jest zbiór emocji reprezentowanych przez konkretne oznaczenia

– modele wymiarowe, w których emocje są reprezentowane przy pomocy zbioru miar określających ich własności.

2.2.1. Modele kategoryzowane

Do typu modeli kategoryzowanych zaliczają się przede wszystkim modele emocji bazowych. Na przestrzeni lat powstało wiele modeli różniących się od siebie ilością oraz rodzajami emocji. Przykładem

może być tutaj model zaproponowany przez Oatley'a i Johnsona-Lairda, w którym proponują oni pięć podstawowych emocji: gniew, lęk, odraza, radość i smutek [16]. Jednym z popularniejszych modeli emocji bazowych, jest ten przedstawiony przez Ekmana. Razem z Friesenem przeprowadzili badania na plemieniu z Papui-Nowej Gwinei. Członkowie plemienia byli w stanie zidentyfikować sześć emocji: strach, gniew, wstręt, smutek, szczęście i zaskoczenie (rys. 2.2) [17]. Kilka lat później do tej grupy Ekman dodał pogardę, aby odróżnić ją od wstrętu.

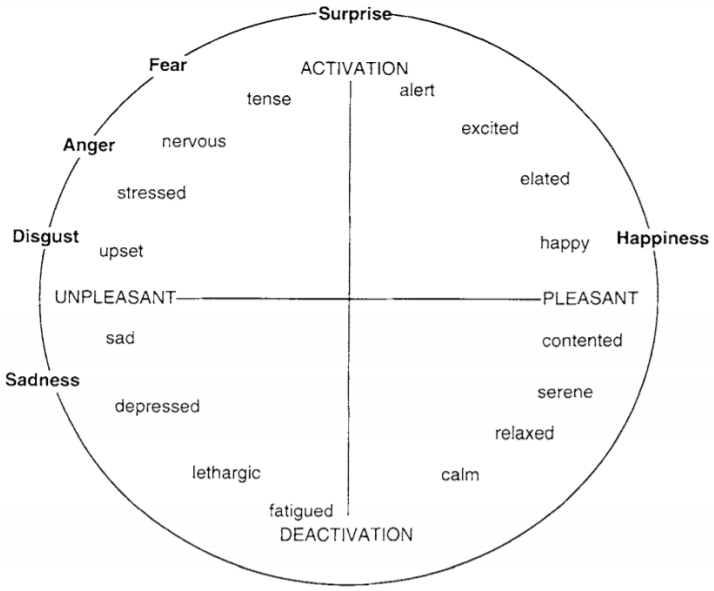


Rys. 2.2. Wyrazy twarzy odpowiadające sześciu emocjom zaproponowanym przez Ekmana, źródło: [17]

W modelach kategoryzowanych często nazywanych także modelami dyskretnymi emocje są niezależne od siebie. Ich dużą zaletą jest to, że automatycznie reprezentują ludzkie emocje dzięki łatwym do zrozumienia etykietom. Jednak w swojej pracy James Russell i Lisa Barret przedstawiają problemy, jakie można napotkać w tym typie modeli [18]. Pierwszym są różnice występujące w nazwach kategorii w zależności od języka. Każdy ze stanów emocjonalnych może być opisany na różne sposoby w zależności od języka. Różnice mogą wystąpić nie tylko w samym tłumaczeniu, ale nawet w ilości słów opisujących daną emocję. Kolejnym problemem jest fakt, że granice pomiędzy danymi klasami emocji często są rozmyte. Te same stany emocjonalne można wyrazić za pomocą różnych kategorii w zależności od różnic kulturowych, środowiskowych czy osobowościowych [15]. Trzecim przedstawionym problemem jest to, że każdy z typów emocji, takich jak gniew czy strach, składa się z zestawu uporządkowanych w czasie powiązanych ze sobą zdarzeń. W związku z tym takie emocje powinno traktować jako złożone procesy, które nie powinny być traktowane jako elementy atomowe.

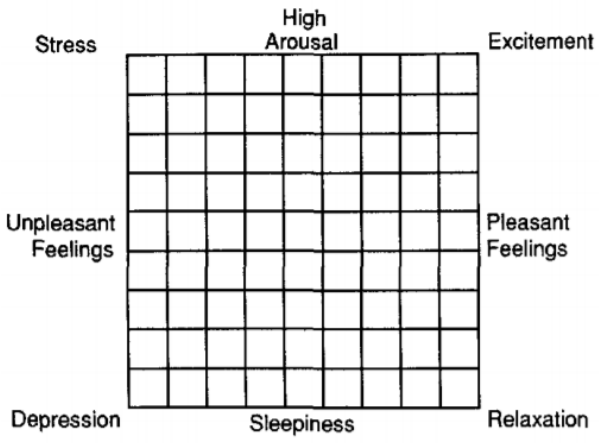
2.2.2. Modele wymiarowe

Russell w swoich badaniach zauważył jeszcze jeden istotny problem modeli dyskretnych. Skoro opisują one stany emocjonalne w sposób jednoznaczny, nie można określić poziomu odczuwania emocji. Jako przykład podaje porównanie strachu w trakcie przejażdżki kolejką górską a sytuacjami zagrażającymi życiu. W przypadku modeli dyskretnych w obu sytuacjach można jedynie określić odczuwanie emocji strachu, pomimo różniących się reakcji fizjologicznych. W ramach swoich badań Russel zaproponował model w kształcie okręgu, składający się z dwóch wymiarów (rys. 2.3). Pierwszym z nich jest poziom odczuwania przyjemności w modelu nazwany wartościowością (ang. *valence*). Odzwierciedlał on także pozytywność lub negatywność emocji. Niskie wartości odpowiadają emocjom takim jak smutek, stres, czy odraza, natomiast szczęście lub ekscytacja są opisane poprzez wysokie wartości wartościowości. Drugim wymiarem jest pobudzenie (ang. *arousal*), które opisuje natężenie danej emocji. Odczucia takie jak senność, depresja czy zrelaksowanie charakteryzują się niskimi wartościami pobudzenia, natomiast wartościom wysokim odpowiadają emocje takie jak ekscytacja, zaskoczenie, czy złość. Model został przedstawiony w formie koła, na którego brzegach opisane zostały podstawowe emocje. Warto wspomnieć, że nie jest to pierwszy model o strukturze kołowej. Sam Russel, chcąc pokazać uniwersalność swojego modelu, porównuje go między innymi z teoriami zaproponowanymi przez Watsona i Tellegena [19]. Tym, co wyróżniało model Russela, były przeprowadzone badania [20]. Russel wybrał 28 słów reprezentujących konkretne stany afektywne i przy pomocy trzech różnych technik przeskalował je, otrzymując zbliżone wyniki. W celu potwierdzenia swojej teorii przeprowadził eksperyment z grupą 36 osób, która miała przyporządkować wybrane słowa do ośmiu kategorii określających podstawowe stany emocjonalne, a następnie ułożyć te kategorie na modelu kołowym w taki sposób, aby przeciwne emocje znalazły się po przeciwnych stronach koła.



Rys. 2.3. Model wymiarowy zaproponowany przez Russela, źródło: [18]

Kilka lat później Russel wraz z Anną Weiss oraz Geraldem Mendelsohnem przedstawił propozycję jednopunktowej skali nazwaną Affect Grid, która miała służyć jako sposób szybkiej i prostej do analizy oceny stanów afektywnych wzdłuż wymiarów znanych z modelu Russela, wartościowości oraz pobudzenia [21]. Skala jest przedstawiona w formie siatki, gdzie oba wymiary określane są w zakresach od 1 do 9 w sposób ciągły. Początkowo miała ona mieć formę kołową, podobną do kołowej struktury modeli emocji, jednak ostatecznie zrezygnowano z niej na rzecz znacznie prostszej do wytłumaczenia badanym osobom siatki. W ramach sprawdzenia, czy przedstawiona skala w poprawny sposób odzwierciedla stany emocjonalne, przeprowadzono badania, w których wykorzystano między innymi słowa wybrane przy wcześniejszych badaniach Russela oraz zbiór zdjęć z wyrazami twarzy odpowiadającymi konkretnym stanom afektywnym. Zadaniem badanych było oznaczenie na skali poziomu wartościowości i pobudzenia najlepiej odzwierciedlających dane słowo lub wyraz twarzy. Następnie wyniki porównano z innymi skalami opisującymi poziomy wartościowości i zauważono wysoki stopień podobieństwa między nimi.

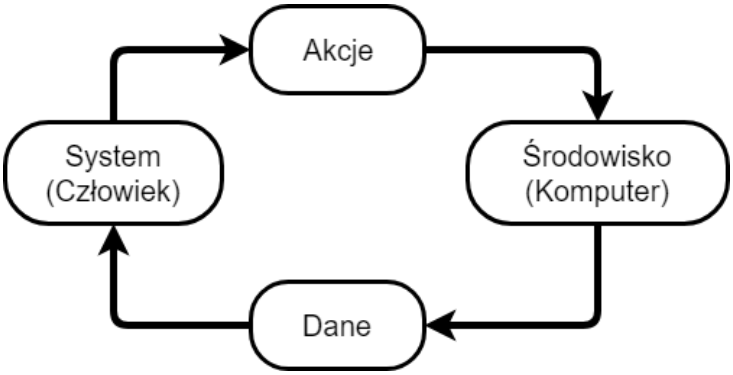


Rys. 2.4. *Affect Grid*, źródło: [21]

Model emocji przedstawiony przez Russela oraz skala opisana w *Affect Grid* stały się jednymi z bardziej popularnych form przedstawiania emocji w informatyce afektywnej. Ogromną ich zaletą jest przede wszystkim łatwość przetwarzania danych opartych na dwuwymiarowej skali, które określają dane stany emocjonalne. Skalę tą można zauważyć między innymi w zbiorach danych zawierających powiązania między reakcjami fizjologicznymi a odpowiednimi wartościami wartościowości i pobudzenia [22, 23].

2.3. Pętla afektywna

Do realizacji celów zakładanych przez informatykę afektywną [6] wykorzystywana jest pętla afektywna. To pochodzące z dziedziny Automatyki i Robotyki pojęcie zakłada, że system oraz środowisko są zaangażowane w ciągłą interakcję między sobą. Schemat pętli został przedstawiony na rysunku 2.5.



Rys. 2.5. Cykl pętli afektywnej, źródło: opracowanie własne na podstawie [9]

Kristina Höök przedstawiła definicję pętli afektywnej w formie trzech kroków [9]:

1. Użytkownicy wyrażają emocje poprzez pewne interakcje obejmujące ich ciało, na przykład gesty czy inne akcje wpływające na środowisko.

2. Następnie odpowiada środowisko generujące elementy afektywne, używając na przykład kolorów, animacji czy innej formy, która może wpłynąć na użytkowników.

3. Elementy afektywne wpływają na reakcję użytkowniów, co prowadzi do coraz większego ich zaangażowania w interakcję ze środowiskiem.

Jak widać, istotnym elementem pętli jest sposób określenia stanów emocjonalnych danej osoby. Spoglądając na rozwiązania praktyczne, można zauważyć, że duże znaczenie ma wykorzystanie sztucznej inteligencji do przewidywania emocji na podstawie reakcji fizjologicznych [24]. Wykorzystywane są tutaj przede wszystkim wcześniej opisane modele Ekmana [17] oraz Russela [20], które przedstawiają emocję w formie zrozumiałej dla komputera. Same emocje są natomiast przewidywane na podstawie odczytów reakcji fizjologicznych użytkowników, a także ich interakcji ze środowiskiem. Określenie stanu emocjonalnego użytkownika pozwala na generację w środowisku sytuacji, które wpłyną na użytkownika tak, aby ten mógł zareagować w odpowiedni sposób i zwiększać swoje zaangażowanie w interakcję ze środowiskiem.

Dobrym przykładem systemów wykorzystujących pętlę afektywną są systemy świadome kontekstu, takie jak afektywne odtwarzacze muzyki [25, 26], które na podstawie sygnałów odczytanych od użytkownika modyfikują listę odtwarzanych piosenek, tak, aby dopasować ją do aktualnego stanu emocjonalnego użytkownika i w wybrany sposób na niego wpłynąć.

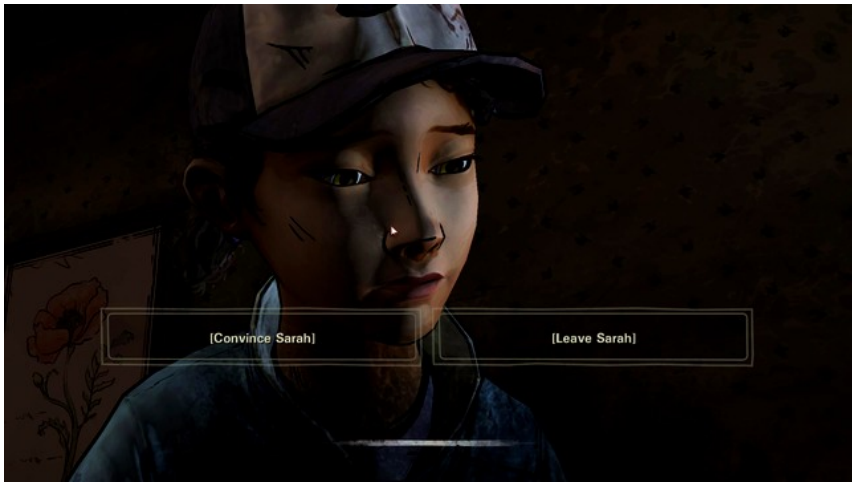
2.4. Gry z pętlą afektywną

W ciągu ostatnich lat znacznie wzrosło zainteresowanie grami komputerowymi zawierającymi elementy informatyki afektywnej. Głównym celem tworzenia gier jest dostarczenie użytkownikowi rozrywki poprzez zwiększanie poziomu immersji [27], często również dzięki zapewnieniu wrażeń emocjonalnych. Końcowy odbiór zależny jest między innymi od takich elementów jak fabuła, mechaniki wykorzystane w trakcie rozgrywki, czy też szczegóły wizualne. Aby jak najlepiej zadowolić użytkownika, twórcy gier muszą także wykorzystywać najnowsze technologie, które mogą sprawić, że gracz będzie w stanie jeszcze mocniej odczuwać wrażenia płynące z rozgrywki.

Jednym z takich rozwiązań jest właśnie informatyka afektywna i wykorzystanie emocji do zwiększenia możliwości wpływania na grę. Eva Hudlicka w swoich pracach [28] wskazuje na kluczowe znaczenie emocji w projektowaniu gry. Wywołanie ich u gracza jest możliwe dzięki istotnym momentom w fabule gry, interakcji z postaciami, do których użytkownik może się w pewien sposób odnieść emocjonalnie, czy nawet poprzez same mechaniki zaimplementowane w grze. Ten sam efekt można także uzyskać przez wygląd środowiska, w którym zostaje umieszczony gracz. Odpowiednie nacechowanie wykorzystanych kolorów, dźwięki otoczenia, czy budowa miejsca, gdzie znajduje się postać prowadzona przez gracza, mogą wpłynąć na to, jakie emocje odczuwa gracz. Idealnym przykładem są tutaj modyfikacje graficzne takich tytułów jak Minecraft czy The Elder Scrolls V: Skyrim. Zapewniają one tekstury wyższej jakości, zbliżone jak najbardziej do tego, co gracz może zobaczyć za oknem w realnym świecie. Pozwala to na zwiększenie immersji poprzez dostarczanie graczowi doświadczeń zbliżonych do rzeczywistości.

Powyższy sposób wpływania na emocje Hudlicka nazywa otwartą pętlą. W tym podejściu nie jest wymagane badanie reakcji gracza. Głównym celem jest wpłynięcie na emocje przy pomocy elementów świata gry. Istnieje wiele rozwiązań komercyjnych, które, choć wprost nie są nazywane grami afektywnymi, realizują opisane wyżej założenia. Doskonałym przykładem są tutaj gry przygodowe od studia Telltale Games, gdzie rozgrywka jest formą interaktywnego filmu. W trakcie rozgrywki gracz podejmuje kluczowe decyzje (rys. 2.6) wpływające na to, jak potoczy się dalsza fabuła gry. Taka forma przedstawienia historii pozwala na zaangażowanie gracza w poznanie postaci i przywiązanie się do nich [29]. Dzięki temu, w kluczowych momentach, takich jak na przykład odejście czy śmierć bohatera, w graczu mogą zostać wywołane adekwatne do wydarzenia emocje.

Drugim sposobem jest wykorzystanie odczuwanych przez gracza emocji do modyfikacji systemów zaimplementowanych w grze tak, aby dostosować je do potrzeb użytkownika i utrzymać jego zaangażowanie. Przykładem zamkniętej pętli, jak nazywana jest ta metoda przez Hudlicką, jest zmiana stopnia trudności na podstawie samopoczucia gracza. Jeżeli jest on zdenerwowany czy przestraszony, poziom skomplikowania rozgrywki zmniejsza się, natomiast gdy zaczyna on odczuwać nudę, stawiane są przed nim nowe wyzwania, które wzbudzą jego zainteresowanie grą. Zamknięta pętla może być także wykorzystywana w kontekście gier poważnych. Dla przykładu gry wykorzystywane w terapiach mogą na bieżąco monitorować stan emocjonalny użytkownika i modyfikować rozgrywkę w taki sposób, aby osiągnąć konkretny rodzaj emocji.



Rys. 2.6. Przykład kluczowej decyzji w grze The Walking Dead, źródło: [30]

Ważnym aspektem, na który należy zwrócić uwagę, jest odpowiednie dobranie elementów sprzętowych wykorzystanych do określenia stanu emocjonalnego użytkownika. Ponieważ środowisko odbiorców gier komputerowych wykracza daleko poza aspekty naukowe, istotne jest to, aby urządzenia nie przeszkadzały w rozgrywce, a jednocześnie pozwalały na dokładne określenie, jakie emocje odczuwa gracz. W kolejnych rozdziałach niniejszej pracy omówiono przykłady takich urządzeń, wraz z ich zaletami i wadami w kontekście odczytywania emocji, oraz przystępności dla graczy poza warunkami laboratoryjnymi.

Choć świadome wykorzystanie gier afektywnych jest bardziej widoczne w środowisku naukowym i akademickim [31, 32], to producenci gier starają się powoli wprowadzać kontekst emocjonalny w komercyjnych rozwiązaniach jako dodatkowy element urozmaicający rozgrywkę [33, 34]. Jednym z głównych problemów jest niewielka ilość opracowań, w których omówione by zostały dostępne rozwiązania sprzętowe umożliwiające predykcję emocji, oraz to, w jaki sposób zaimplementować odczytywanie i reakcje na stany emocjonalne w środowiskach do tworzenia gier. Z tego właśnie powodu powstała idea opracowania platformy, która w prosty sposób może zostać wcielona do istniejącej już gry stworzonej w określonym oprogramowaniu do tworzenia gier.

3. Specyfikacja komponentów interfejsu do rozpoznawania emocji

Celem pracy jest opracowanie prototypu interfejsu umożliwiającego pomiar sygnałów pozwalających na określenie zmian stanów emocjonalnych gracza. W związku z tym na jeden z elementów niniejszej pracy składa się przegląd możliwych rozwiązań użytych w poszczególnych komponentach interfejsu. W tym rozdziale skupiono się na przedstawieniu urządzeń do pomiaru sygnałów umożliwiających określenie stanu emocjonalnego użytkownika, mechanizmach wnioskowania, które mogą zostać użyte podczas budowania modelu do rozpoznawania emocji, oraz silnikach do budowy gier, przy pomocy których zostanie wykonany końcowy interfejs wraz z grą.

3.1. Platformy pomiarowe

Odczyt sygnałów umożliwiających rozpoznawanie emocji jest jednym z najważniejszych elementów w pętli afektywnej. Obecnie dostępnych jest wiele platform umożliwiających wykonanie takich pomiarów i można je podzielić na dwie główne kategorie: urządzenia służące do mierzenia reakcji fizjologicznych użytkownika oraz sprzęt umożliwiający odczyt sygnałów pośrednich, na podstawie których można określić stan użytkownika.

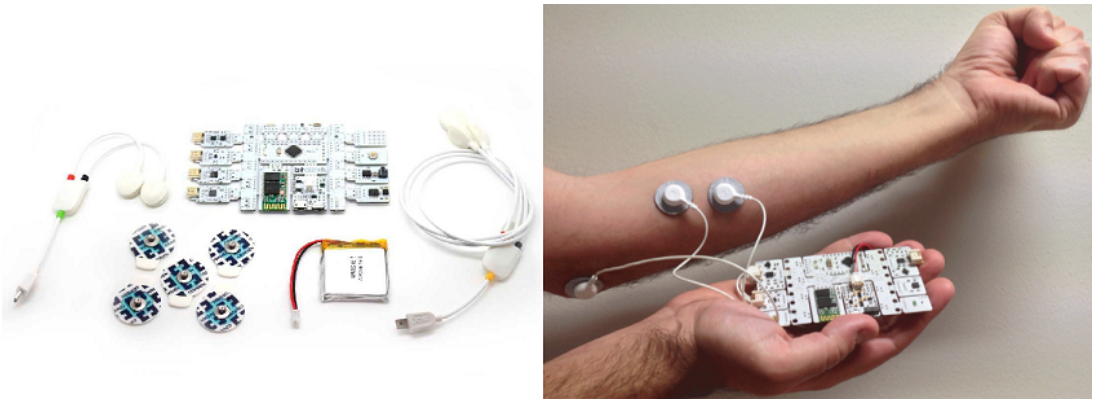
Na pierwszą grupę składają się między innymi platformy klasy medycznej umożliwiające dokładne pomiary z ludzkiego ciała. Jednym z takich urządzeń jest Neurobit Optima. Jest to przenośny sprzęt umożliwiający pomiar sygnałów takich jak praca serca, mózgu, ruchów mięśni, reakcji elektrodermalnej czy nawet temperatury skóry [35]. Ogromną zaletą Neurobit Optimy są wielofunkcyjne kanały pomiarowe, które użytkownik może dostosować do swoich potrzeb [35]. Widocznym atutem z perspektywy wykorzystania w informatyce afektywnej jest gotowe oprogramowanie dostępne od producenta oraz wbudowany interfejs Bluetooth dający możliwość przenoszenia urządzenia. Pomiary z Neurobit Optima charakteryzuje wysoka dokładność i stabilność pomiarów. Podobnym do niego sprzętem jest NeXus-10, który także pozwala na pomiar pracy serca i reakcji elektrodermalnej, posiada wbudowany interfejs Bluetooth oraz jest do niego dołączane gotowe oprogramowanie. Niestety ogromną wadą jest jego waga i rozmiar wpływające negatywnie na wygodę podczas użytkowania. W kontekście gier afektywnych potencjalną problemem urządzeń klasy medycznej może być także ich inwazyjność. Choć w ciągu ostatnich lat środowisko sprzętowe wyszło daleko poza standardowy komputer czy konsole, to część z graczy może

Rys. 3.1. Urządzenie Neurobit Optima, źródło: [35]

nie zaakceptować platform kojarzonych z medycyną jako części stanowiska do gier. Jednym z problemów są tutaj szeroko wykorzystywane elektrody, które, choć umożliwiają dokładne pomiary sygnałów fizjologicznych, są kojarzone jednak głównie ze środowiskiem medycznym.

W kontekście platform do pomiarów sygnałów fizjologicznych w informatyce afektywnej można zauważyć wzrost zainteresowania urządzeniami nasobnymi [36]. Ich wyraźną zaletą jest rozmiar i przenośność, dzięki czemu ruchy użytkownika nie są w żadnym stopniu ograniczone. Najbardziej rozpowszechnionym, a jednocześnie jednym z tańszych rozwiązań, są inteligentne opaski, takie jak Xiaomi Mi Band czy Microsoft Band [36]. Oba z wymienionych urządzeń posiadają wbudowany optyczny sensor pulsu [37, 38], drugie natomiast posiada także czujnik do pomiaru reakcji elektrodermalnej [38]. Inną, mniej popularną opaską, jest Empatica E4, która poza sensorami dostępnymi w Xiaomi Mi Band i Microsoft Band umożliwia pomiar temperatury skóry [39]. Warto wspomnieć też o wbudowanych w urządzenia czujnikach ruchu używanych między innymi do zliczania kroków mogących posłużyć jako sygnał pośredni, na podstawie którego można wykryć poziom aktywności użytkownika. Niestety ogromną wadą inteligentnych opasek jest duża niedokładność pomiarów w porównaniu do odczytów ze sprzętów klasy medycznej. Wskazania z opasek charakteryzują się wartościami mocno odbiegającymi od tych wykonanych przy pomocy sprzętu medycznego [36, 40].

Innymi platformami nasobnymi charakteryzującymi się większą dokładnością, do których często porównuje się odczyty z innych urządzeń do pomiaru pracy serca [36, 40], są monitory tętna Garmin HRM-Run oraz Polar H10. W przeciwieństwie do optycznego sensora wbudowanego w inteligentne opaski są one wyposażone w suche elektrody zamontowane w pasku zakładanym na klatkę piersiową[41, 42]. Jest to rozwiązanie, które nie jest inwazyjne dla użytkownika, a jednocześnie pozwala na dokładny pomiar pracy serca. Ważną zaletą obu rozwiązań jest nie tylko wsparcie dla standardu Bluetooth, ale także dla



Rys. 3.2. BITalino (r)evolution kit oraz przykład podłączenia sensora do pomiaru ruchu mięśni, źródło: [35]

protokołu ANT+ wykorzystywanego w coraz większej liczbie urządzeń do pomiarów aktywności użytkownika[1], do którego twórcy udostępniają również gotowe implementacje interfejsów do odczytywania pomiarów. Dzięki temu w prosty sposób możliwe jest zbudowanie aplikacji interpretujących wysyłane dane.

Rozwiązaniem będącym pomostem między kosztownymi platformami medycznymi a często niedokładnymi urządzeniami nasobnymi jest platforma BITalino (r)evolution kit [43]. Jest to narzędzie modułowe, składające się z płytki stanowiącej rdzeń urządzenia, do której mogą zostać podpięte oddzielne moduły do pomiarów sygnałów fizjologicznych (rys. 3.2). Na liście dostępnych sensorów znajdują się te odpowiadające za pomiar akcji serca, reakcji elektrodermalnej skóry, czynności ruchowej mięśni oraz aktywności mózgu. Każdy z modułów może zostać podłączony do dowolnego kanału urządzenia, natomiast pomiary odbywają się poprzez podłączenie do drugiej strony modułów kabli, na których końcu znajdują się elektrody. Co więcej, poza modułami służącymi do pomiaru reakcji fizjologicznych użytkownika, na liście dostępnych segmentów znajdują się także akcelerometr, sensor światła, dioda LED, brzęczyk, oraz przycisk, które mogą posłużyć do odczytu pomiarów pośrednich czy informowania użytkownika o zdarzeniach. Dzięki temu platforma BITalino może służyć nie tylko jako urządzenie wykorzystywane do pomiarów, ale również do interakcji z grą. Bardzo dużą zaletą BITalino z perspektywy informatyki afektywnej jest dostępność narzędzi przygotowanych przez twórców, a także ogromna ilość interfejsów do komunikacji z urządzeniem. Twórcy na swojej stronie udostępniają biblioteki dla wielu popularnych języków programowania takich jak Python, C# czy Java oraz konkretne implementacje z przykładami dla środowisk takich jak silnik do gier Unity. Dodatkowym plusem jest fakt, że większość tych rozwiązań jest oprogramowaniem otwartym, w związku z czym mogą być one rozszerzane i naprawiane przez społeczność korzystającą z platformy.

Do wspomnianej na początku drugiej kategorii urządzeń, z których możliwe jest odczytanie sygnałów pośrednich wykorzystanych do określenia stanu użytkownika, można przede wszystkim zaliczyć

[1] *https://www.thisisant.com/directory*

sprzęty wykorzystywane przez graczy. Mowa tutaj między innymi o myszkach, klawiaturach czy padach, z których możliwe jest odczytanie intensywności kliknięć lub odczyt szybkości poruszania myszką na podstawie jej pozycji na ekranie [44]. Szczególną uwagę należy poświęcić kontrolerowi DualShock 4 od firmy Sony, który w przeciwieństwie do większości spopularyzowanych kontrolerów do gier posiada wbudowany akcelerometr oraz żyroskop, które mogą posłużyć jako dodatkowe źródło informacji do określenia stanu emocjonalnego użytkownika [45]. Niestety ze względu na umowy licencyjne oprogramowanie umożliwiające odczyt tych parametrów z pada, jest płatne, co można zaliczyć jako wadę tego rozwiązania z perspektywy twórcy aplikacji wykorzystującej funkcjonalności DualShocka.

3.2. Możliwe mechanizmy do rozpoznawania emocji

Aby określić emocje występujące u użytkownika na podstawie danych zebranych przy pomocy wybranych urządzeń, należy zaimplementować mechanizm sztucznej inteligencji, który jak najlepiej będzie umieć określić stan emocjonalny gracza.

Poniżej przedstawiono i opisano kilka wybranych metod wnioskowania, które mogłyby posłużyć jako algorytmy do modelu rozpoznającego stan emocjonalny użytkownika na podstawie zebranych danych fizjologicznych.

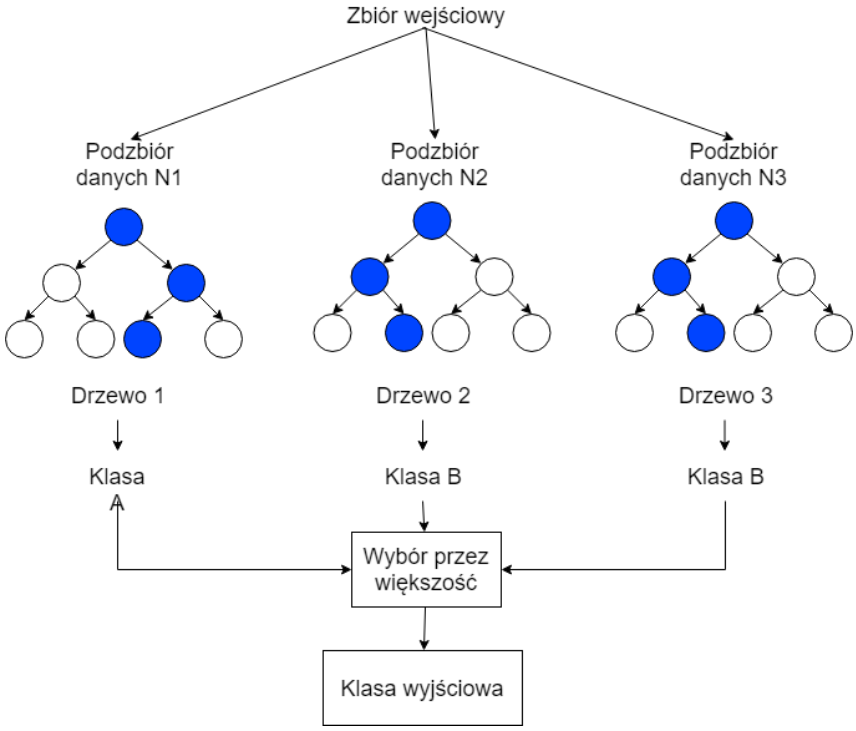
3.2.1. Lasy losowe

Lasy losowe są metodą uczenia maszynowego, która polega na zbudowaniu wielu drzew decyzyjnych, wykonaniu na nich klasyfikacji, a następnie wybraniu klasy poprzez zasadę większości (rys. 3.3). Każde takie drzewo składa się z wierzchołków, z których wewnętrzne, nazywane węzłami, zawierają pytania i warunki dotyczące sprawdzanych cech, natomiast zewnętrzne wierzchołki nazywane liśćmi zawierają decyzję o klasyfikacji obiektów. Z każdego węzła wychodzi tyle gałęzi, ile jest możliwych wyników warunku sprawdzanego na danym węźle.

Dla każdego z drzew decyzyjnych wybierany jest podzbiór wejściowego zbioru danych za pomocą metody bootstrap, polegającej na losowaniu ze zwracaniem elementów z podanego zbioru [46]. Liczność podzbiorów jest taka sama jak wejściowego zbioru danych, co oznacza, że niektóre elementy mogą znaleźć się w nich więcej niż raz, a inne w ogóle nie zostaną uwzględnione. Dla każdego z drzew wybierany jest także podzbiór cech, które będą brane pod uwagę przy tworzeniu wierzchołków drzewa. W każdym węźle podział jest wybierany na podstawie tego, jak wiele informacji na temat wyboru klasy można uzyskać nam wybrany podzbiór cech.

3.2.2. Drzewa ekstremalnie losowe

Metoda ta jest bardzo podobna do lasów losowych, z dwoma istotnymi różnicami. Pierwszą jest brak tworzenia podzbiorów trenujących dla drzew. Każde z nich jest uczone na tym samym zbiorze danych. Drugą różnicą jest sposób wyboru punktu podziału, który jest zupełnie losowy. Taka metoda



Rys. 3.3. Uproszczony przykład działania metody lasów losowych

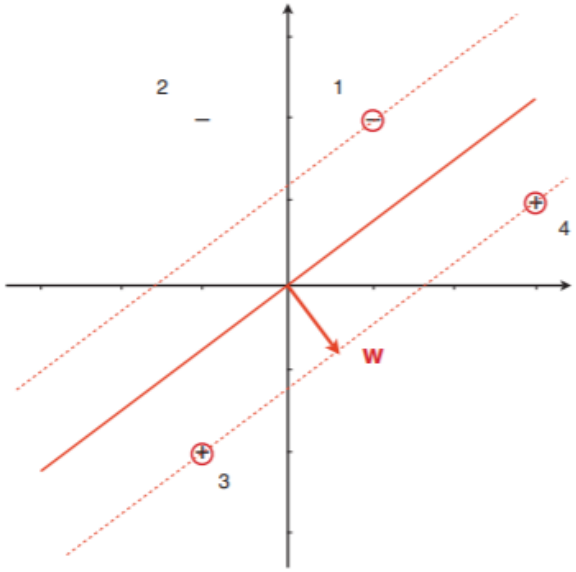
zapewnia dużo mniejszy koszt obliczeń w porównaniu do lasów losowych, ponieważ nie tracimy czasu na sprawdzenie, który z podziałów daje najlepsze rezultaty.

3.2.3. Maszyna z wektorem wspierającym

Jest to metoda wykorzystywana głównie w przypadku klasyfikacji. Głównym celem jest znalezienie linii lub hiperpłaszczyzny, która podzieli badany zbiór danych na dwie różne klasy. Ponieważ takich rozwiązań może być nieskończenie wiele, wprowadzone zostało pojęcie marginesu. Poprzez równoległe przesuwanie granicy do pierwszych punktów z obu klas otrzymuje się dwie hiperpłaszczyzny. Marginesem natomiast określa się odległość między nimi. Zadaniem maszyny z wektorem wspierającym jest znalezienie takiej hiperpłaszczyzny, która maksymalizuje margines. (rys. 3.4). Przypadki uczące, na których oparte są równoległe hiperpłaszczyzny, nazywane są wektorami wsparcia.

3.3. Narzędzia do budowy gier

Tworzenie gier komputerowych jest złożonym procesem, wymagającym pracy w ramach wielu dziedzin naukowych. W przypadku przygotowywania wysokobudżetowych produkcji przy pracy nad nimi potrafi pracować nawet kilkaset osób. Od projektantów poziomów, przez osoby zajmujące się przygotowaniem sceny fabularnej, grafików, kompozytorów muzyki aż po twórców oprogramowania, którzy łączą wcześniej przygotowane elementy w jedną całość. To właśnie ci ostatni odpowiadają za ostateczną wersję gry, jej wygląd oraz optymalizację.



Rys. 3.4. Przykład optymalnej linii maksymalizującej margines. Przypadki zaznaczone czerwonymi okręgami są wektorami wsparcia, źródło: [46]

Programiści tworzący końcową wersję gry, podczas tworzenia oprogramowania odpowiadającego za poszczególne jej elementy, muszą wziąć pod uwagę to, z jakim rodzajem gry mają do czynienia. Każdy z gatunków charakteryzuje się konkretnymi wymaganiami, które programista musi jak najlepiej odwzorować w przygotowywanym kodzie gry. Dla przykładu gra wojenna powinna charakteryzować się balistyką pocisków oprogramowaną za pomocą wzorów fizycznych, realistycznym biegiem bohatera posiadającego określony poziom wytrzymałości, czy chociażby skutkami wybuchu granatu przedstawionymi jako uruchomione w odpowiednim momencie elementy dźwiękowe i graficzne. W ostatnich latach dużą popularnością cieszą się gry z gatunku Battle Royale [47], w których kilkudziesięciu graczy ściera się ze sobą w rozgrywce wieloosobowej. W przypadku takich gier programiści muszą zwracać uwagę nie tylko na elementy związane z rozgrywką, ale także na jak najdokładniejszą synchronizację klientów gry z serwerem.

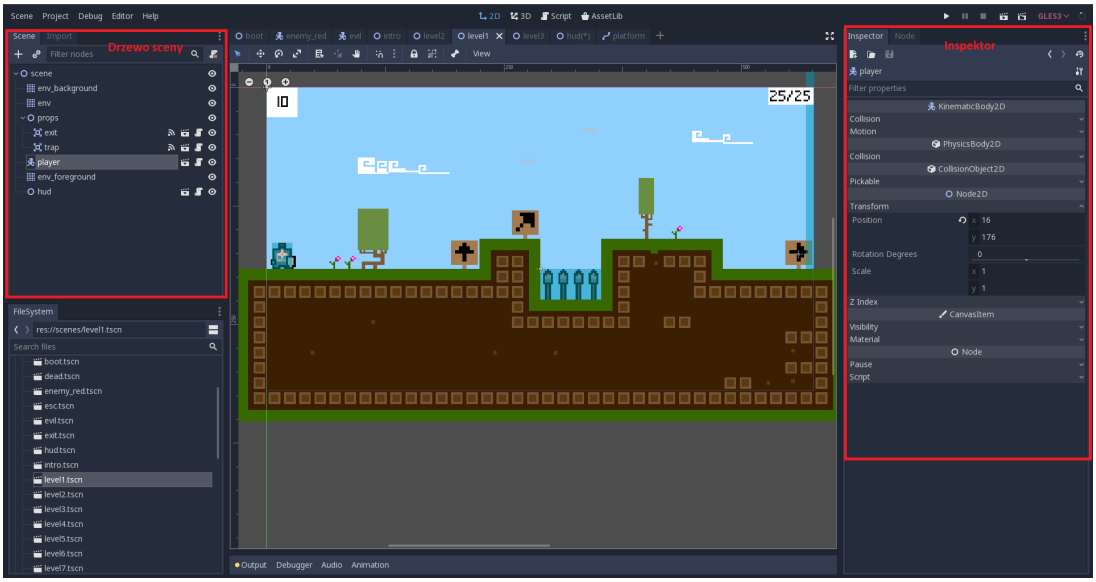
Ilość oraz złożoność elementów, które muszą zostać uwzględnione w kodzie gry, powoduje, że stworzenie jej wyłącznie przy pomocy języków programowania jest niemal niemożliwe. Właśnie w takim celu powstają oprogramowania nazywane silnikami do tworzenia gier. Umożliwiają one obsługę sterowania, grafiki, animacji, interakcji między obiektami, a nawet sztucznej inteligencji, która może być jednym z elementów gry. Silniki zwykle są dostępne w formie środowisk programistycznych wprowadzających ułatwienia do zarządzania wcześniej wymienionymi elementami, jednocześnie przy tym nie rezygnując z możliwości języków programowania.

Istnieje mnóstwo narzędzi do tworzenia gier, jednak większość z nich stanowią rozwiązania wykorzystywane jedynie w obrębie danej firmy, która na potrzeby gry lub całej serii stworzyła silnik spełniający wymagania i usprawniający proces tworzenia gier produkowanych przez daną spółkę. Przykładem może być rozwijany od lat IW engine, przy pomocy którego powstała seria gier Call of Duty. Z tego

właśnie powodu większość publikacji naukowych na temat gier komputerowych oparta jest o darmowe silniki do gier. Kilka z takich rozwiązań omówiono poniżej, zwracając uwagę na popularność, funkcje dostępne w danym środowisku, języki, w których można tworzyć aplikacje oraz na jakie platformy można wydać gry stworzone przy pomocy danego silnika.

3.3.1. Godot

Godot [48] jest narzędziem umożliwiającym tworzenie gier na platformy takie jak Windows, macOS czy Linux, a także na środowiska mobilne oraz webowe. Dużą zaletą Godota jest mały rozmiar edytora oraz dostępność na większość popularnych systemów operacyjnych. Skrypty oprogramowujące mechaniki gry mogą być tworzone przy pomocy języków C#, C++, oraz wysokopoziomowego języka GDScript przygotowanego specjalnie dla tej platformy. Alternatywnym rozwiązaniem do tworzenia logiki gry są także skrypty wizualne (ang. *visual scripting*) polegające na budowaniu kodu przy pomocy gotowych bloków, które będą zrozumiałe nie tylko dla programistów, ale również dla grafików czy animatorów. Dużym plusem, szczególnie dla osób, które dopiero zaczynają pracę z Godotem, jest liczba przykładowych projektów, nie tylko w formie gotowych gier, ale również pojedynczych elementów, takich jak sposoby padania światła czy to, w jaki sposób stworzyć realistyczną wodę w grze.



Rys. 3.5. Interfejs silnika Godot, po lewej stronie widać drzewo aktualnej sceny, źródło: opracowanie własne na podstawie [48]

Projektowanie gier przy pomocy silnika Godot opiera się na budowaniu poszczególnych obiektów nazywanych scenami, które mogą następnie być łączone ze sobą na kolejnych scenach. Dzięki temu kolejne poziomy, a nawet cały projekt mogą ostatecznie być opisane w formie grafu powiązań. Takie rozwiązanie powoduje, że projekty przygotowane przy pomocy Godota są proste w utrzymaniu podczas wspólnej pracy wielu osób. Każdy programista może w danej chwili zająć się oddzielną sceną, nie powodując konfliktów z elementami przygotowanymi przez inne osoby.

3.3.2. Unity

Unity jest silnikiem umożliwiającym tworzenie gier na o wiele większą liczbę platform niż Godot [49]. Poza opisanymi wcześniej Windowsem, Linuxem, macOS'em oraz platformami mobilnymi i webowymi, przy pomocy Unity można tworzyć gry na konsole współczesnej generacji oraz okulary wirtualnej i rozszerzonej rzeczywistości. Sama platforma jest dostępna na systemy Windows oraz Mac, a od niedawna twórcy starają się dostosować ją także na dystrybucje oparte o jądro Linux.

Projektowanie gier w środowisku Unity polega na tworzeniu obiektów, do których przypisane są konkretne właściwości w zależności od rodzaju stworzonego obiektu. Do każdego z takich elementów mogą również zostać przyporządkowane skrypty odpowiadające za logikę działania obiektu, które tworzone są przy pomocy języka C#. W momencie tworzenia niniejszej pracy twórcy zapowiedzieli także dodanie skryptów wizualnych, co stanowi alternatywę dla osób niezaznajomionych z programowaniem lub też z samym językiem. Przygotowane elementy są umieszczane w środowisku gry nazywanym sceną. Interfejs Unity pozwala na bardzo proste manewrowanie pomiędzy obiektami znajdującymi się na scenie, zarządzenie ich parametrami oraz powiązaniami między nimi.

Rys. 3.6. Kerbal Space Program, jedna z najpopularniejszych gier stworzonych przy pomocy silnika Unity, źródło: [50]

Unity jest obecnie jedną z najpopularniejszych platform do tworzenia gier komputerowych. Przy jego pomocy powstało wiele znanych na świecie produkcji takich jak gry karciane Hearthstone i Gwint, a także gra symulacyjna Kerbal Space Program (rys. 3.6). Popularność ta przejawia się przede wszystkim w postaci ogromnej liczby poradników dostępnych w sieci, a także aktywnej społeczności, która stale komunikuje się między sobą w celu rozwiązania problemów występujących podczas tworzenia gier komputerowych. Dzięki temu, poza standardową dokumentacją, która także jest mocno rozbudowana, użytkownik ma dostęp do dodatkowych informacji uzyskanych od bardziej doświadczonych programistów. Cechą platformy, która mogła wpłynąć na jej popularność, jest także specjalnie przygotowany

Rys. 3.7. Interfejs edytora Unreal Engine w wirtualnej rzeczywistości, źródło: [51]

sklep zawierający zarówno płatne, jak i darmowe dodatki rozszerzające możliwości silnika lub po prostu ułatwiające pracę przy konkretnym gatunku gry.

3.3.3. Unreal Engine

Unreal Engine jest narzędziem, które przez wiele lat było wykorzystywane jedynie do tworzenia dużych, komercyjnych gier. Do momentu wydania wersji trzeciej, projektowanie gier przy jego pomocy było ograniczone odpłatną licencją [51]. Dopiero w roku 2009 twórcy wydali oprogramowanie Unreal Development Kit, będące darmową wersją trzeciej wersji silnika, z uszczuploną bazą gotowych modeli oraz bez dostępu do kodu źródłowego [52]. Dopiero kilka lat po wydaniu Unreal Engine 4 twórcy zdecydowali się na udostępnienie pełnej wersji oprogramowania za darmo.

Spośród opisywanych rozwiązań, Unreal Engine jest najbardziej rozbudowanym. Podobnie do Unity umożliwia tworzenie gier na większość dostępnych platform, w tym także okularów wirtualnej rzeczywistości. Językiem programowania wykorzystywanym do projektowania gier w tym silniku jest C++, co można traktować jako wadę, ponieważ jest to język trudny i wymagający dużo czasu na opanowanie. Z drugiej strony jednak gry napisane przy pomocy C++ potrafią działać szybciej i płynniej od rozwiązań stworzonych przy pomocy języków wysokopoziomowych. Ogromną zaletą silnika, która wyróżnia go na tle innych rozwiązań, jest dostęp do kodu źródłowego. Dzięki temu, twórcy gier mogą dostosować, a nawet rozszerzyć możliwości platformy.

Dla osób niepracujących z kodem Unreal Engine udostępnia rozbudowany interfejs, pozwalający na stworzenie gry bez konieczności pisania kodu. Podobnie jak w przypadku Unity, projekt gry składa się z powiązanych ze sobą scen, tutaj nazywanych poziomami. Na każdej ze scen umieszczone są obiekty, nazywane aktorami, pomiędzy którymi zaprogramowane są interakcje specyficzne dla danej sceny. Osoba tworząca grę może zaprogramować jej logikę przy pomocy skryptów wizualnych. Tutaj

warto opisać kolejny atut wyróżniający ten silnik na tle innych, którym jest możliwość korzystania z edytora przy pomocy okularów wirtualnej rzeczywistości podczas tworzenia gier na tę platformę (rys. 3.7). Twórca dostaje możliwość dokładnego zaprojektowania świata gry, patrząc na nią od razu z perspektywy gracza.

Unreal Engine jest obecnie drugą, tuż po Unity, najpopularniejszą otwartą platformą do gier. Przewaga Unity wynika tutaj nie z większych możliwości silnika, ale z jego otwartości od początku istnienia. Ponieważ jednak przez wiele lat Unreal Engine był wykorzystywany głównie w rozwiązaniach komercyjnych, posiada rozbudowaną, złożoną dokumentację, która opisuje każdy fragment możliwości edytora. Po udostępnieniu darmowej wersji oprogramowania pojawiła się także duża ilość materiałów na temat tworzenia gier przy pomocy Unreal Engine, a także urosła społeczność twórców gier, którzy wykorzystują ten silnik i wymieniają się między sobą cennymi informacjami.

4. Architektura

Na podstawie analizy możliwych rozwiązań sprzętowych w rozdziale 3 oraz przedstawionych poniżej założeń architektury platformy dokonano wyboru urządzeń, które będą stanowiły część sprzętową tworzonego interfejsu. W tym rozdziale przedstawiono ich specyfikacje oraz argumenty, które przemawiają za wyborem każdego z nich. Opisano także sposób montażu sprzętu.

4.1. Założenia architektury sprzętowej

Aby wybrać odpowiedni dla przygotowywanego prototypu interfejsu zestaw urządzeń, określone zostały założenia i wymagania, jakie powinny one spełniać:

1. Platforma sprzętowa powinna być lekka, aby używanie jej przez gracza nie powodowało dyskomfortu, który może wpłynąć negatywnie na jakość odbieranych danych.

2. Dokładność dostępnych sensorów powinna być jak największa, aby mieć pewność odczytywanych zachowań użytkownika.

3. Urządzenie powinno być łatwe w obsłudze, zarówno w kwestii jego zamontowania, jak i uruchomienia odczytów. Jednocześnie powinno charakteryzować się jak najmniejszą inwazyjnością podczas pomiaru, aby zminimalizować dyskomfort użytkownika.

4. Dla wybranego sprzętu powinno być dostępne oprogramowanie umożliwiające odczyt sygnałów w wybranym środowisku do tworzenia gier. Najlepiej, gdyby rozwiązanie było dostępne w postaci biblioteki oferowanej przez twórcę narzędzia lub rozszerzenia silnika, które umożliwi odbieranie danych.

5. Urządzenie powinno mieć możliwość połączenia bezprzewodowego, najlepiej przy pomocy technologii Bluetooth lub innego protokołu umożliwiającego bezprzewodowy przesył danych.

6. Przynajmniej jedno z urządzeń powinno umożliwiać odczyt pracy serca w postaci pulsu lub sygnału z elektrokardiogramu. Pomiar tętna jest jedną z podstawowych metod umożliwiających określenie zmian w stanie emocjonalnym użytkownika.

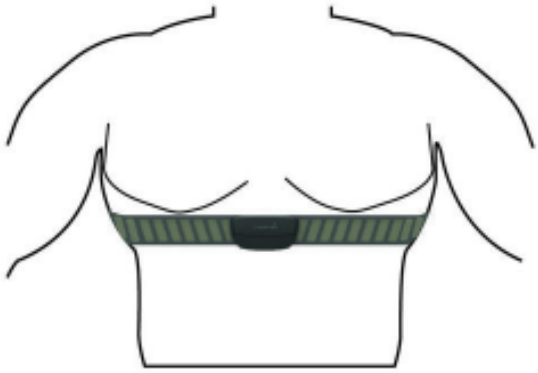
7. Przynajmniej jedno z urządzeń powinno umożliwiać odczytywanie ruchów mięśni. Uzyskane sygnały mogą być wykorzystane do wpływania na rozgrywkę w zależności od aktywności mięśniowej użytkownika w danej partii ciała.

4.2. Garmin HRM-Run

Garmin HRM-Run jest jednym z wielu dostępnych czujników tętna produkowanych przez firmę Garmin. Urządzenie zbudowane jest z modułu zawierającego elektronikę odpowiedzialną za interpretację oraz wysyłanie odczytywanych danych. Zasilany jest on baterią CR2032 i zamontowany jest na elastycznej opasce, umożliwiającej zamontowanie opaski na klatce piersiowej. Sama opaska zawiera dwie elektrody odpowiadające za odczyt sygnałów fizjologicznych, które następnie są przekazywane w formie informacji na temat pracy serca.

Czujnik, poza podstawowym pomiarem liczby uderzeń na minutę, pozwala na pozyskiwanie dodatkowych informacji na temat zmienności rytmu zatokowego w postaci odstępów czasowych pomiędzy kolejnymi załamkami R, czyli szczytami kompleksów QRS. Na podstawie tej informacji można określić stan zdrowotny użytkownika lub zmiany jego stanu emocjonalnego. Dla przykładu nagłe spadki w zmienności rytmu zatokowego mogą sygnalizować możliwość odczuwania stresu przez użytkownika. HRM-Run umożliwia także zbieranie danych na temat dynamiki biegu, takich jak liczba kroków na minutę, czas kontaktu z podłożem czy długość kroku. Ze względu na charakter pracy, która skupia się głównie na tematyce gier komputerowych, pomiary te nie były brane pod uwagę.

Urządzenie HRM-Run zostało wybrane, ponieważ stanowi pewien kompromis pomiędzy dokładnością odczytów a rozmiarami i wygodą użytkowania. W porównaniu do inteligentnych opasek, charakteryzujących się najwyższym poziomem komfortu spośród omówionych w rozdziale 3 platform sprzętowych umożliwiających pomiar tętna, Garmin HRM-Run pozwala na o wiele dokładniejsze odczyty, nie powodując przy tym dyskomfortu podczas użytkowania. Jest to możliwe dzięki wykorzystaniu elastycznej opaski ze wbudowanymi suchymi elektrodami, które, w przeciwieństwie do fotopletyzmografu odpowiadającego za optyczny odczyt tętna w inteligentnych opaskach, odbierają sygnały elektryczne bezpośrednio z ciała użytkownika. Informacje te są następnie interpretowane do danych w postaci wykrycia kolejnych uderzeń serca. Opaski natomiast charakteryzują się dużymi opóźnieniami w stosunku do aktualnego tętna użytkownika, co w przypadku rozpoznawania emocji w sposób ciągły całkowicie eliminuje je jako akceptowalne rozwiązanie. Jednocześnie elastyczna opaska, do której przymocowany jest moduł, w bardzo prosty sposób jest zakładana przez użytkownika na klatce piersiowej. Takie umiejscowienie i brak kabli łączących elektrody z głównym modułem sprawia, że użytkownik nie czuje dyskomfortu podczas noszenia urządzenia. W podobny sposób Garmin HRM-Run można porównać z omawianym wcześniej BITalino (r)evolution kit. W tym przypadku można zauważyć sytuację odwrotną niż przy porównaniu z inteligentnymi opaskami. BITalino oferuje bardzo dokładny pomiar pracy serca, nie tylko w formie liczby uderzeń na minutę, ale także pełnego elektrokardiogramu umożliwiającego o wiele szerszą analizę pracy serca. Niestety, odczyt wymaga zamocowania przyklejanych elektrod, które połączone



Rys. 4.1. Miejsce montażu czujnika tętna Garmin HRM-Run, źródło: [42]

są przewodami z sensorem, a następnie z samym BITalino. Taki sposób montażu może wpłynąć negatywnie na komfort podczas gry, co nie wystąpi w przypadku urządzenia HRM-Run.

Innym aspektem decydującym o wyborze tej platformy sprzętowej, jest obsługa dwóch protokołów bezprzewodowego przesyłania danych. Poza standardowym połączeniem dzięki technologii Bluetooth HRM-Run wykorzystuje także protokół ANT+ rozwijany przez firmę Garmin. Jego główną zaletą jest brak ograniczeń co do ilości urządzeń mogących odbierać dane z czujnika. W przeciwieństwie do technologii Bluetooth, która ogranicza połączenie do jednego lub w przypadku Bluetooth Smart, dwóch urządzeń, pomiar przy pomocy protokołu ANT+ umożliwia jednoczesne odczytywanie danych w grze i monitorowanie pracy serca przy pomocy innych narzędzi korzystających z tego protokołu. Twórcy ANT+ udostępniają szeroką dokumentację oraz biblioteki umożliwiające odczyt w wielu językach programowania, od C++, C#, aż po dedykowane implementacje dla systemu Android.

4.3. BITalino (r)evolution kit

BITalino (r)evolution Plugged Kit BT jest urządzeniem produkowanym przez firmę PLUX Wireless Biosignals przeznaczonym do tworzenia platform, w których zawarte są elementy wymagające informacji na temat sygnałów fizjologicznych. Główny element narzędzia stanowi płytka zbudowana z następujących części [43]:

– 10 złącz UC-E6, w ramach których można wyróżnić: 6 wejść analogowych, 1 wejście cyfrowe, 1 wyjście cyfrowe, 1 złącze mogące pracować w trybie wejścia lub wyjścia cyfrowego oraz 1 złącze PWM, do którego może zostać podłączony przetwornik DAC lub dioda LED. Złącza są podzielone na 2 grupy w oddzielnych segmentach.

– Mikrokontroler z mikroprocesorem oraz stykami umożliwiającymi połączenie każdego z segmentów w sposób inny niż standardowy

– Segment zasilający, zawierający włącznik, gniazdo ładownia w formie złącza micro-USB oraz złącze JST, do którego podłączana jest bateria 3.7V zasilająca całą płytkę.

– Moduł Bluetooth odpowiadający za przesył danych z płytki.

W ramach przygotowywanego interfejsu, z dostępnych sensorów opisanych w rozdziale 3 wybrany został jedynie moduł zawierający elektromiogram mierzący napięcie mięśni podskórnych. Głównym powodem takiego wyboru są dwie główne wady urządzenia dotyczące komfortu użytkowania. Są to przewody łączące płytkę z sensorem oraz elektrody mocowane z drugiej strony przewodów. Choć z perspektywy eksperymentów naukowych użycie elektrod żelowych przyklejanych do skóry nie jest problemem, to w przypadku środowiska gier komputerowych standardowy użytkownik może odczuwać dyskomfort lub w ogóle zrezygnować z używania urządzenia ze względu na jego inwazyjność.

Głównym powodem wyboru BITalino oraz samego modułu EMG jest wykorzystanie sygnałów fizjologicznych do kontrolowania gry w świadomy sposób. Odczyt zmian napięcia mięśni podskórnych może pozwolić na wykrycie, kiedy i jak mocno są one używane, co może następnie zostać wykorzystane do zaimplementowania mechanik w grze uruchamianych na podstawie konkretnych reakcji mięśniowych. Kolejnym powodem wykorzystania platformy sprzętowej jest możliwość sprawdzenia, w jakim stopniu użytkownik tak naprawdę odczuwa dyskomfort podczas używania elektrod przyklejanych do skóry. Pozwoli to stwierdzić, czy urządzenia takie jak BITalino, które umożliwiają wykonanie dokładnych pomiarów dzięki wykorzystaniu elementów bliższych rozwiązaniom medycznym, mogłyby przyjąć się jako stały element stanowiska do gier.

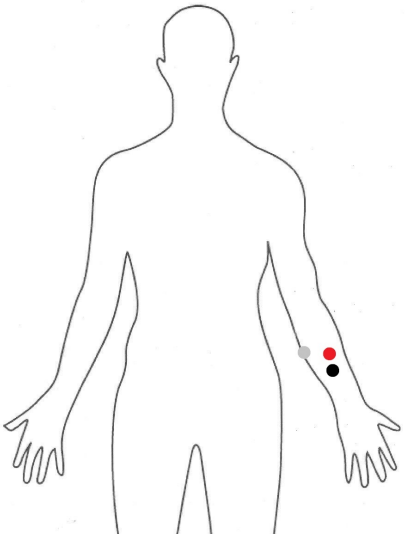
Ponieważ odczyty z elektromiogramu mają posłużyć jako pewien element kontroli nad grą, jako miejsce, do którego podłączone będą elektrody, wybrano przedramię, ponieważ ruch mięśni znajdujących się w tamtej części ciała może być precyzyjny. Dzięki temu użytkownik w prosty sposób, poprzez ruchy nadgarstka lub dłoni, może kontrolować napięcie mięśni w trakcie rozgrywki. Sposób montażu elektrod został przedstawiony na rysunku 4.2. Elektrody czerwona i czarna powinny znajdować się na wewnętrznej części przedramienia, równolegle do niego, stosunkowo blisko siebie. Elektroda referencyjna, w BITalino oznaczona kolorem białym, powinna znajdować się na wystającej kości łokciowej.

4.4. DualShock 4

DualShock 4 jest urządzeniem produkowanym przez firmę Sony, stworzonym początkowo jako kontroler do konsoli PlayStation 4. Na przyciski dostępne na jego powierzchni składają się [45]:

– 2 gałki analogowe, gdzie dla każdej z nich odczyty interpretowane są jako ciągły sygnał wejściowy przedstawiony w dwóch wymiarach.

– Pad kierunkowy (ang. *d-pad*) stanowiący cyfrowy odpowiednik gałki analogowej. Lewy i prawy przycisk odpowiadają wartościom granicznym osi horyzontalnej, górny i dolny są przypisane w podobny sposób do osi wertykalnej.



Rys. 4.2. Sposób przypięcia elektrod dla sensora EMG, kolor czerwony oznacza elektrodę dodatnią, czarny ujemną, a szary elektrodę referencyjną

– 9 przycisków cyfrowych, na które składają się: 4 przyciski akcji (trójkąt, kwadrat, krzyżyk, kółko), L3 i R3 zamontowane pod gałkami analogowymi oraz przyciski PS, SHARE i OPTIONS, domyślnie służące do zarządzania opcjami konsoli.

– 2 przyciski L1 i R1, dla których wykrycie wciśnięcia opiera się na sile nacisku na przycisk.

– 2 analogowe spusty L2 i R2, których wciśnięcie jest odczytywane jako sygnał w określonym zakresie wartości.

– Dwupunktowy panel dotykowy, posiadający także możliwość kliknięcia go.

Innymi elementami widocznymi na urządzeniu są wbudowany głośnik, wejście słuchawkowe Jack 3.5mm, umieszczony na froncie pasek świetlny RGB oraz złącze micro-USB służące do ładowania i podłączenia kontrolera w przypadku braku połączenia Bluetooth. Kontroler, wraz ze wszystkimi opisanymi powyżej elementami, został przedstawiony na rysunku 4.3.

Poza widocznymi elementami Dualshock 4 posiada także wbudowane moduły wibrujące, żyroskop oraz akcelerometr. To właśnie dwa ostatnie z wymienionych elementów wpłynęły na wybór urządzenia jako części przygotowywanego interfejsu. W porównaniu do innych dostępnych kontrolerów, poza klasycznym wykorzystaniem narzędzia jako sposobu kierowania postacią w grze, akcelerometr wbudowany w DualShocka może posłużyć jako źródło danych, które w pośredni sposób mogą określać pewne elementy stanu emocjonalnego użytkownika. Bardziej pobudzony gracz, może zupełnie nieświadomie poruszać kontrolerem, natomiast brak zmian w odczytach może sugerować, że użytkownik jest skupiony lub znudzony. W porównaniu do określania ilości kliknięć na myszy i klawiaturze w danym czasie, wykorzystanie akcelerometru jest lepszym rozwiązaniem, ponieważ ruchy dłoni gracza trzymającego kontroler są instynktowne w sytuacjach wywołujących pobudzenie. Sceny wywołujące nagły strach, mogą

Rys. 4.3. Moduły sprzętowe wykorzystywane w platformie. Od lewej: kontroler DualShock 4, czujnik tętna Garmin HRM-Run, BITalino z modułem EMG

doprowadzić do ruchu całego ciała, natomiast sytuacje powodujące uczucie złości lub desperacji mogą wywołać subtelne przesunięcie pozycji rąk na kontrolerze, co wywoła zmianę wartości na akcelerometrze. Interpretacja tych danych oraz odczytów pracy serca pozwoli na dokładniejsze określenie, jaką emocję odczuwa gracz w danym momencie.

Problemem, który warto zaznaczyć w przypadku DualShocka, jest ograniczenie dostępności oprogramowania pozwalającego odczytywać pomiary z akcelerometru. Z powodu licencji narzuconych przez producenta, większość dodatków i bibliotek wykorzystywanych w silnikach do gier jest dostępnych wyłączenie w wersji płatnej, a oficjalne oprogramowanie od twórców jest udostępniane wyłącznie osobom zatwierdzonym przez Sony jako projektanci gier na konsolę PlayStation 4. Jednakże, pomimo wspomnianego ograniczenia, zdecydowano się na wybór Dualshocka 4 jako części przygotowywanego interfejsu.

Na rysunku 4.3 przedstawiony został pełen zestaw urządzeń wykorzystanych w przygotowywanym interfejsie. Pozyskane z nich pomiary zostaną wykorzystane do rozpoznania stanu emocjonalnego gracza, a także do kontrolowania mechanik wykorzystywanych w grze. Dokładny opis sposoby odczytu danych oraz jak będzie wyglądał model do predykcji emocji użytkownika, zostaną opisane w rozdziałach 5 i 6.

5. Mechanizm predykcji emocji

Jednym z zadań niniejszej pracy jest opracowanie mechanizmu umożliwiającego odczyt emocji na podstawie danych zebranych przy pomocy urządzeń przedstawionych w rozdziale 4. Zdecydowano się na przygotowanie modelu uczenia maszynowego, który na podstawie cech wyciągniętych z sygnałów fizjologicznych będzie w stanie określić stan emocjonalny użytkownika. Aby uniezależnić przygotowywany model od silnika, w którym będzie stworzona gra, zdecydowano się przygotować go w języku Python przy pomocy biblioteki scikit-learn służącej do tworzenia modeli uczenia maszynowego. Komunikacja modelu z grą będzie odbywała się poprzez zapytanie HTTP przyjmujące na wejściu dane fizjologiczne odczytane z danego zakresu czasu. Na wyjściu zwracana będzie reprezentacja emocji określona w modelu.

W tym rozdziale opisano proces przygotowywania modelu do rozpoznawania stanów emocjonalnych. Zostaną opisane wykorzystane zbiory danych treningowych, sposób ich wstępnego przetworzenia, wybór cech i klas, oraz proces uczenia i wyboru najlepszego modelu. Rozwiązanie dotyczące komunikacji serwera z grą zostanie opisane w rozdziale 6.

5.1. Zbiory danych

W trakcie poszukiwania zbiorów uczących nie znaleziono niestety takich, które jednocześnie wykorzystywałyby pomiary pracy serca oraz akcelerometru. Zdecydowano więc o wyborze danych zawierających odczyty pracy serca, i to na ich podstawie określony zostanie stan emocjonalny zwracany przez model. Wartości z akcelerometru zostaną wykorzystane jako dodatkowy kontekst korygujący klasę emocji zwracaną przez model, o czym więcej jest wspomniane w rozdziale 6.

5.1.1. DEAP

DEAP[1] [22] to zbiór danych zawierający pomiary przeprowadzone na 32 osobach w wieku 19–37 lat. Każdy z badanych został poddany eksperymentowi, w ramach którego musiał obejrzeć 40 fragmentów teledysków, każdy z nich trwający 60 sekund. Były one dostosowane w taki sposób, aby wywoływać konkretny rodzaj emocji. Uczestnicy oceniali każdy z fragmentów w czterech skalach: *valence* określający przyjemność odczuwaną podczas oglądania, *arousal* opisujący pobudzenie użytkownika, *dominance*

[1] A Database for Emotion Analysis using Physiological Signals

wskazujący poziom kontroli nad odczuwanymi emocjami oraz *liking* warunkujący jak bardzo materiał wideo podobał się osobie badanej. W trakcie badań uczestnicy byli podłączeni do specjalistycznych przyrządów pomiarowych, przy pomocy których zebrano następujące dane fizjologiczne: elektroencefalogram (EEG), puls objętości krwi (BVP), reakcja elektrodermalną (GSR), elektromiogram (EMG), objętość oddechowa, temperatura skóry oraz elektrookulogram. Poza tym, dla 22 uczestników dostępne były również nagrania twarzy.

Dane udostępnione zostały w dwóch wersjach: oryginalnej oraz poddanej wstępnej obróbce. Wersja oryginalna zawierała pełne odczyty, próbkowane z częstotliwością 512 Hz. W wersji przetworzonej, próbkowanie zostało zmniejszone do 128 Hz i usunięte zostały 3 pierwsze sekundy stanowiące czas na przygotowanie się uczestników do badania. Część danych została także przefiltrowana i uśredniona względem pomiarów wszystkich osób badanych.

5.1.2. AMIGOS

AMIGOS[2] [23] jest zbiorem zawierającym dane zebrane podczas dwóch rodzajów eksperymentów:

1. 16 krótkich fragmentów filmów hollywoodzkich wybranych ze zbioru DECAF [53] o długości 51–150 s ocenianych przez grupę 40 osób.

2. 4 długie materiały wideo o długości 14.1–23.58 min ocenianych przez 37 osób.

Podobnie jak w przypadku zbioru DEAP, uczestnicy mieli ocenić fragmenty wideo w kilku skalach: *valence*, *arousal*, *dominance*, *liking* oraz *familiarity* określający poziom znajomości fragmentu wideo. Każdy z badanych miał także zaznaczyć jaką emocję odczuwał podczas oglądania danego materiału, mając do wyboru: emocję neutralną, odrazę, radość, zaskoczenie, złość, strach i smutek. W trakcie badań zostały zebrane następujące dane pomiarowe: elektroencefalogram (EEG), elektrokardiogram (ECG), reakcja elektrodermalna (GSR), nagrania wideo twarzy osób badanych oraz nagrania całego ciała wykonane przy pomocy urządzenia Kinect V1. W przypadku danych z elektrokardiogramu pomiary zostały przeprowadzone na lewym i prawym ramieniu.

Zbiór został udostępniony w dwóch wersjach: oryginalnej, dla której w zależności od odczytywanego sygnału dane są próbkowane z częstotliwością 128 lub 256 Hz, oraz wstępnie przetworzonej. W tej drugiej, próbkowanie sygnałów zostało zmniejszone do 128 Hz, wartości zostały uśrednione względem wszystkich pomiarów, oraz wstępnie przefiltrowane przy pomocy filtrów dolnoprzepustowych.

5.1.3. ASCERTAIN

ASCERTAIN[3] [54] to zbiór danych zebranych podczas badań, w ramach których 56 osób obejrzało 36 materiałów wideo o średniej długości 80 s. Każdy z uczestników oceniał materiały wideo w pięciu skalach: *valence* w zakresie od -3 do 3, *arousal* od 0 do 6, *liking*, *familiarity* oraz *engagement*

[2] A Dataset for Affect, Personality and Mood Research on Individuals and Groups

[3] a multimodal databa**AS**e for impli**C**it p**ER**sonali**T**y and **A**ffect recognit**I**o**N** using commercial physiological sensors

opisujący poziom zaangażowania użytkownika w oglądanie danego fragmentu wideo. Przed przystąpieniem do badania, uczestnicy musieli wypełnić kwestionariusze osobowości BFMS[4]. W trakcie badań zebrano następujące dane pomiarowe: elektrokardiogram (ECG), reakcja elektrodermalna (GSR), elektroencefalogram (EEG) oraz pomiar punktów orientacyjnych twarzy. Podobnie jak w przypadku zbioru ASCERTAIN, odczyt z elektrokardiogramu odbywał się na lewym i prawym ramieniu. Częstotliwości próbkowań były zależne od typu sygnału: EEG — 32 Hz, ECG — 256 Hz, GSR — 128 Hz.

W ramach zbioru udostępnione zostały także cechy wyliczone na podstawie zebranych pomiarów, które mogły zostać wykorzystane jako gotowe dane do uczenia modelu maszynowego. Elementem zawartym w ASCERTAIN, który wyróżnia go na tle innych opisanych zbiorów, jest plik zawierający wyniki wstępnej analizy zebranych wartości, które zostały określone w skali od 1 oznaczającego dane perfekcyjne do 6 opisującego brak danych lub błędy w pomiarach. Udostępniona została także lista eksperymentów zakończonych niepowodzeniem. Dzięki temu, w zależności od wymaganej jakości odczytanych danych, można przeprowadzić ich wstępną filtrację.

5.1.4. DECAF

DECAF[5] [53] jest zbiorem zawierających pomiary zebrane podczas eksperymentów przeprowadzonych na 16 mężczyznach i 14 kobietach o średniej wieku 27 lat. Uczestnicy w trakcie badań oglądali materiały z dwóch grup:

1. 40 fragmentów teledysków wykorzystywanych w opisywanym wcześniej zbiorze DEAP [22].

2. 36 fragmentów wyciętych z hollywoodzkich filmów o średnim czasie trwania 80 sekund.

Podobnie jak w przypadku pozostałych zbiorów, uczestnicy mieli ocenić każdy fragment przy pomocy następujących skal: *valence* w zakresie od -2 do 2, *arousal* od 0 do 4 oraz *dominance* o zakresie takim jak w przypadku *arousal*. W trakcie badań zebrano dane o następujących sygnałach fizjologicznych: magnetoencefalogram (MEG), elektromiogram (EMG), elektrokardiogram (ECG). Pozyskano także nagrania twarzy każdego z uczestników w bliskiej podczerwieni. Sygnały były gromadzone z próbkowaniem o częstotliwości 1000 Hz, co daje dużą ilość danych, jednak może to wprowadzać wyraźne błędy w pomiarach. Poza oryginalnymi odczytami, w ramach zbioru zostały udostępnione wyliczone cechy dla każdego z sygnałów, które mogą być wykorzystane jako gotowy zestaw danych wykorzystywanych do stworzenia modelu. Ze względu na generyczny sposób przetwarzania pomiarów we wszystkich używanych zbiorach, zdecydowano się skorzystać jednak z oryginalnych wartości.

[4] big-five marker scale

[5] MEG-Based Multimodal Database for Decoding Affective Physiological Responses

5.2. Przetworzenie danych

Ponieważ wykorzystane zbiory zostały udostępnione w różnych formatach, a odczyty w nich zawarte różniły się długością pobranych danych, zakresami skal opisujących odczytywane emocje, a w przypadku zbioru DEAP innym rodzajem sygnału opisującego pracę serca, wymagane było przeprowadzenie wstępnej konwersji danych wejściowych. Każdy ze zbiorów udostępniał pomiary w formacie plików MAT, które zostały otwarte przy pomocy biblioteki SciPy[6]. Następnie, dla każdego ze zbiorów przeprowadzono proces przetworzenia danych przedstawiony na rysunku 5.1. Ze względu na dostępność plików zawierającego ocenę jakości sygnałów oraz listę nieudanych eksperymentów, w przypadku zbioru ASCERTAIN został przyjęty dodatkowy krok odrzucający przypadki zakończone niepowodzeniem, oraz te, dla których jakość sygnału z elektrokardiogramu była oceniona na 6.

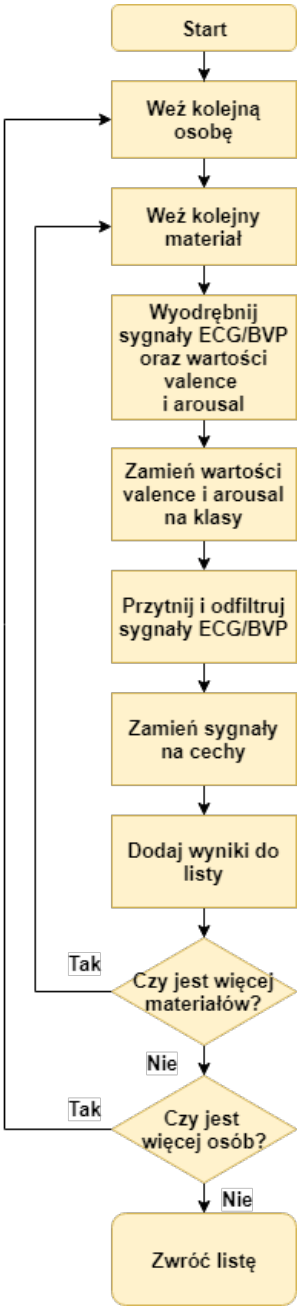
Mając identyfikator osoby i eksperymentu, z wczytanych danych zostały wyodrębnione sygnały z elektrokardiogramu, a w przypadku zbioru DEAP z odczytów pulsu objętości krwi. Odczytano również wartości *valence* i *arousal*, na podstawie których określony zostanie cel przygotowywanego modelu. Aby ujednolicić otrzymane oceny, przeskalowano je do zakresu od 1 do 9, a następnie zamieniono na 9 klas, odpowiadających połączeniu 3 poziomów wartości dla każdej z ocen, tak jak pokazano to na rysunku 5.2. Taki podział pozwoli na objęcie stanów emocjonalnych charakteryzujących się niskim lub wysokim poziomem każdej z ocen, uwzględniając jednocześnie wartości środkowe opisujące stan neutralny. Kolejnym krokiem było dostosowanie sygnałów do przetworzenia ich na cechy. Ponieważ reakcje fizjologiczne, które mogłyby wskazać na zmianę odczuwanej emocji, pojawiają się z opóźnieniem w stosunku do bodźca je wywołującego, wczytane dane zostały obcięte do ostatnich 50 sekund eksperymentu. Następnie, aby pozbyć się ewentualnych szumów i wygładzić przebieg sygnałów, przefiltrowano je przy pomocy pasmowoprzepustowego filtru Butterwortha. Częstotliwości graniczne zostały dobrane na podstawie artykułu Fedotova [55] poświęconego określeniu optymalnych wartości tych parametrów podczas filtrowania sygnału z elektrokardiogramu.

Tak przygotowane dane zostały następnie wykorzystane do obliczenia cech, które zostaną użyte w przygotowywanym modelu. Na początku z sygnałów wyciągnięto interwały pomiędzy kolejnymi uderzeniami serca. Aby określić te wartości, w przypadku sygnału z elektrokardiogramu należało znaleźć położenia kolejnym załamków R, natomiast dla pomiarów pulsu objętości krwi kolejne powtórzenia danej fazy pracy serca. Do wykrycia tych punktów wykorzystana biblioteka PeakUtils[7] oferująca algorytm znajdujący wierzchołki w podanym sygnale. Wymagany parametr minimalnej odległości pomiędzy kolejnych wierzchołkami był wyliczany na podstawie wzoru:

$$\frac{60 * f}{max_{hr}}$$

gdzie f oznaczało częstotliwość danych zbioru, z którego pochodziła próbka sygnału, a max_{hr} przyjętą najwyższą możliwą wartość tętna. Eksperymentalnie została ona ustawiona na 135 uderzeń serca na minutę. Mając na uwadze fakt, że biblioteka mogła wykryć także nieprawidłowe położenia wierzchołków,

[6] *https://scipy.org/*
[7] *https://scikit-learn.org/*



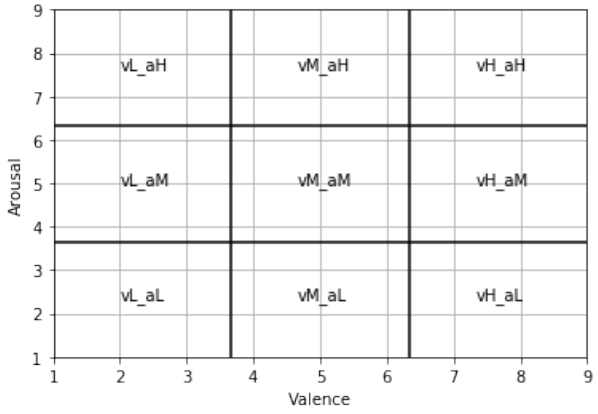
Rys. 5.1. Schemat przedstawiający proces przetworzenia danych do zbioru uczącego

po wyznaczeniu interwałów przefiltrowano je, odrzucając wartości znacznie odbiegające od mediany z odczytów. Na podstawie odfiltrowanych danych obliczono ilości uderzeń serca na minutę, a następnie określono następujące cechy statystyczne brane pod uwagę podczas uczenia modelu:

– Minimalna i maksymalna ilość uderzeń serca na minutę

– Średnia wartość tętna i interwałów między uderzeniami

– Odchylenie standardowe tętna



Rys. 5.2. Podział ocen *valence* i *arousal* na 9 klas

– Kurtoza tętna i interwałów — opisuje, jak bardzo wyniki są skoncentrowane lub oddalone od średniej

– Współczynnik skośności dla tętna i interwałów — opisuje, jak dane rozkładają się wokół średniej.

– Procentowa ilość odczytów tętna i interwałów powyżej wartości: średnia pomiaru + odchylenie standardowe

– Procentowa ilość odczytów tętna i interwałów poniżej wartości: średnia pomiaru − odchylenie standardowe

– Średnie odchylenie bezwględne (MAD) dla wartości interwałów pomiędzy uderzeniami serca.

Analizie została poddana także zmienność rytmu zatokowego, którą rozpatrzono jako parametry opisujące ją w trzech dziedzinach. Pierwszą z nich jest dziedzina czasu, w ramach której można określić następujące parametry:

– odchylenie standardowe interwałów pomiędzy kolejnymi uderzeniami (SDNN)

– średnia kwadratowa różnic pomiędzy kolejnymi interwałami (RMSSD)

– odchylenie standardowe różnic między kolejnymi interwałami (SDSD)

– procentowa ilość interwałów, które różnią się od poprzedniego interwału o więcej niż 20 ms (pNN20)

– procentowa ilość interwałów, które różnią się od poprzedniego interwału o więcej niż 50 ms (pNN50).

Kolejną z dziedzin, w ramach której rozpatrywana jest zmienność rytmu zatokowego, jest częstotliwość. Parametry w ramach tej dziedziny zostały określone poprzez wygenerowanie periodogramu za pomocą metody Welcha [56] dostępnej jako funkcja w bibliotece SciPy, a następnie wyznaczeniu mocy widma w zakresie określonych częstotliwości [57]:

– 0.04–0.15 Hz — zakres LF, dostarczający informacji o krótkich zmianach w tętnie

– 0.15–0.4 Hz — zakres HF, dostarczający informacji o zmianach związanych z cyklem oddechowym.

Jako dodatkowy parametr wykorzystano także stosunek mocy sygnałów LF do HF. Wysoka wartość tej cechy jest widoczna w sytuacjach polegających na walce lub ucieczce, natomiast wartości niskie w przypadku zachowań pozytywnych, takich jak zawarcie przyjaźni.

Trzecią dziedziną, w ramach której można przeprowadzić analizę zmienności rytmu zatokowego, jest dziedzina nieliniowa. Do analizy wyznaczany jest wykres Poincaré, przedstawiający korelację pomiędzy kolejnymi interwałami. Oznacza to, że współrzędne każdego z punktów wykresu są opisane jako sąsiadujące wartości interwałów pomiędzy następującymi po sobie uderzeniami. Cechy wyznaczane w ramach tej analizy to mała i duża oś elipsy okalającej wszystkie punkty wykresu. Mała oś (SD1) jest to odchylenie standardowe odległości punktów od osi $y = x$ i opisuje krótkoczasową zmienność rytmu serca. Duża oś natomiast odpowiada odchyleniu standardowemu odległości każdego z punktów od osi $y = x + avg$, gdzie avg oznacza średnią wartość interwałów. Odczyty z tej osi opisują długoczasową zmienność rytmu serca. Jako dodatkową cechę wykorzystano także stosunek SD1 do SD2.

Dla każdego eksperymentu po wyznaczeniu wszystkich wyżej opisanych cech połączono je z wcześniej wyznaczoną klasą i dodano do zbioru uczącego. Przygotowany zbiór następnie przefiltrowano, usuwając przypadki, w których brakowało przynajmniej jednej cechy. Ze względu na działania wykonywane podczas uczenia i dopracowywania modelu, cechy nie zostały znormalizowane na etapie przetwarzania danych.

5.3. Wybór modelu

Do uczenia modelu na podstawie przygotowanego zbioru danych uczących zostały wykorzystane następujące klasyfikatory z biblioteki scikit-learn[8]:

– RandomForestClassifier

– ExtraTreesClassifier

– SupportVectorClassifier (SVC)

– KNeighborsClassifier

– GradientBoostingClassifier

W pierwszej iteracji wszystkie modele były uczone dla domyślnych wartości parametrów. Następnie, aby sprawdzić, czy skuteczności tak przygotowanych modeli są wiarygodne dla pomiarów spoza

[8] *https://scikit-learn.org/*

Tabela 5.1. Skuteczności wybranych modeli przy określonych parametrach

Algorytm	**Skuteczność dla domyślnych parametrów**	**Skuteczność po przeprowadzeniu walidacji krzyżowej**	**Skuteczność po optymalizacji parametrów**
RandomForestClassifier	28%	27%	35%
ExtraTreesClassifier	31%	29%	36%
SVC	25%	25%	28%
GradientBoostingClassifier	24%	24%	-
KNeighboursClassifier	21%	24%	-

zbioru uczącego, przeprowadzono K-krotną walidację krzyżową. Metoda ta polega na podzieleniu danych na K podzbiorów, które następnie są wykorzystane do przeprowadzenia K iteracji uczenia modelu, gdzie jako dane uczące wybieranych jest K-1 podzbiorów, natomiast ostatni jest wykorzystywany jako wartości testowe. W wyniku przeprowadzenia walidacji zostaje wygenerowanych K modeli o określonych skutecznościach, które następnie można wykorzystać do obliczenia średniej skuteczności modelu. W obu przypadkach cechy ostały poddane standaryzacji. Wartości procentowe skuteczności zarówno dla przypadku domyślnego, jak i walidacji krzyżowej przedstawia tabela 5.1.

Aby osiągnąć większą dokładność, należało przeprowadzić optymalizację parametrów. Zdecydowano się ją wykonać jedynie dla modeli o skuteczności wyższej lub równej 25%. Wykorzystana została do tego biblioteka hyperopt-sklearn[9] pozwalająca na odszukanie jak najlepszych parametrów modelu w zależności od zadanych danych wejściowych. Funkcja estymująca na wejściu przyjmuje klasyfikator, dla którego chcemy znaleźć parametry, sposób normalizacji cech, liczbę konfiguracji, dla których ma zostać przeprowadzona analiza oraz maksymalny czas, w jakim każda z prób powinna się zakończyć. Dla każdego z modeli liczba prób została ustalona na 100, natomiast maksymalny czas dla prób ustawiony został na 5 minut. Parametr odpowiadający za normalizację cech ustawiono na wartość dowolną, dzięki czemu estymator będzie mógł także znaleźć funkcję normalizującą cechy wraz z jej właściwościami, która w połączeniu ze znalezionymi parametrami klasyfikatora da najlepsze wyniki. Skuteczności modeli po optymalizacji ich parametrów oraz funkcji do normalizacji cech przedstawione zostały w ostatniej kolumnie tabeli 5.1. Największy przyrost, a jednocześnie najwyższą wartość skuteczności wynoszącą 36% można zauważyć dla klasyfikatora wykorzystujące drzewa ekstremalnie losowe. Dla tego modelu została ponownie przeprowadzona optymalizacja parametrów, tym razem dla dłuższego czasu maksymalnego i liczby prób równej 1200, jednak wynik poprawił się zaledwie o 1%.

Taka skuteczność nie pozwala na pewne rozpoznawanie emocji na podstawie odczytów samej pracy serca. Głównym powodem może być tutaj fakt, że wykorzystane fragmenty sygnałów były na tyle długie, że w trakcie badania użytkownicy mogli odczuwać więcej niż jedną emocję. Ponieważ nie jest jeszcze obecnie możliwe powiązanie wzorców zmian w reakcjach fizjologicznych z konkretnymi emocjami przy

[9] *http://hyperopt.github.io/hyperopt-sklearn/*

pomocy wzorów, nie było także możliwości wyboru odpowiednich fragmentów sygnałów w zależności od eksperymentu. Możliwe, że dołączenie do analizy pomiarów na temat reakcji elektrodermalnej lub odczytów z elektroencefalogramu pozwoliłoby na uzyskanie wyższej skuteczności, jednak wykorzystanie sensorów mierzących te sygnały mogłoby wprowadzić wysoki dyskomfort podczas użytkowania, co kolidowałoby z wymaganiami architektury sprzętowej przygotowywanego interfejsu. Skuteczność rozpoznawania emocji przy pomocy wybranego modelu może zostać poprawiona dzięki odczytom z akcelerometru wbudowanego w kontroler. Ostateczny sposób wyliczania i interpretacja stanów emocjonalnych na podstawie przygotowanego modelu oraz pomiarów z akcelerometru zostanie opisana w rozdziale 6.

6. Implementacja

W ramach niniejszej pracy magisterskiej stworzono grę komputerową wykorzystującą przygotowywany prototyp interfejsu do odczytu zmian stanów emocjonalnych i zachowań gracza. Na podstawie analizy silników do tworzenia gier przeprowadzonej w rozdziale 3 zdecydowano się na wykorzystanie platformy Unity. Głównym powodem takiej decyzji była przede wszystkim ilość materiałów o tematyce tworzenia gier przy pomocy właśnie tego narzędzia, a także dostępność rozwiązań, które pozwalały na prostą komunikację z urządzeniami opisanymi w rozdziale 4.

6.1. Podstawowe założenia

Zanim rozpoczęto tworzenie gry, zostały określone założenia i wymagania implementacyjne oraz projektowe, według których następnie został zbudowany projekt:

1. Gra powinna zawierać mechaniki afektywne modyfikujące rozgrywkę w zależności od zachowań lub stanu emocjonalnego gracza.

2. Jednym z elementów implementacyjnych powinien być interfejs umożliwiający komunikację z urządzeniami opisanymi w rozdziale 4. Moduł musi także komunikować się z serwerem zawierającym model do predykcji emocji oraz w prosty sposób udostępniać zmiany stanu emocjonalnego użytkownika i jego zachowań.

3. Gra powinna mieć możliwość wyboru jednego z dwóch trybów: podstawowego, zawierającego standardowe mechaniki gry, oraz wersję afektywną.

4. Rozgrywka powinna być powtarzalna, aby w trakcie przeprowadzania badań, każdy z uczestników mógł doświadczyć tych samych elementów gry.

6.2. Implementacja gry

Ponieważ przygotowywana gra miała być wykorzystana do przeprowadzenia eksperymentów w celu ewaluacji opracowanego rozwiązania, zdecydowano się na grę jednoosobową. W trakcie rozgrywki gracz wciela się w rolę kapitana statku kosmicznego, który musi walczyć z przeciwnikami, aby przeżyć. Jego

zadaniem jest sterowanie pojazdem i pokonanie jak największej liczby przeciwników przy pomocy dostępnego wyposażenia. Rozgrywka podzielona jest na poziomy, w których trakcie gracz musi zdobyć określoną liczbę punktów. Na każdym z poziomów dookoła statku generowanych jest kilka rodzajów wrogich statków. Schemat i szybkość pojawiania się ich w trakcie gry są zróżnicowane w zależności od postępu. Z każdym kolejnym poziomem pojawiają się nowe rodzaje jednostek, które mogą nie tylko lecieć w kierunku w gracza lub do niego strzelać pojedynczymi pociskami, ale również wybuchać w jego pobliżu, posiadać większą wytrzymałość lub wykorzystywać inne rodzaje pocisków. Każdy z przeciwników ma także przypisaną ilość punktów dodawaną do puli po zniszczeniu go. Aby przejść do kolejnego poziomu, gracz musi osiągnąć odpowiednio wysoki wynik.

Statek posiada ograniczoną liczbę punktów życia, która zwiększa się po przejściu każdego z poziomów. Aby urozmaicić rozgrywkę, dookoła gracza mogą pojawić się także wzmocnienia w postaci innych broni oraz obiektów leczących. Innym elementem mającym mocno wpłynąć na rozgrywkę jest moc specjalna umożliwiająca wypuszczenie fali w kształcie okręgu, która natychmiastowo niszczy wrogie statki. Ponieważ umiejętność ta bardzo ułatwia rozgrywkę, gracz jest w stanie użyć jej jedynie co określony czas.

Aby wyrównać trudność gry i nie znudzić graczy powtarzalną rozgrywką, każdy z poziomów posiada także dodatkowy tryb charakteryzujący się zwiększeniem skomplikowania rozgrywki. Jest on uruchamiany na pewien określony czas po spełnieniu specjalnych warunków, które w zależności od wersji gry są inne. W trakcie trwania trybu trudnego schemat generowania przeciwników jest zmieniany na ich silniejsze wersje, posiadające większą ilość życia i zmodyfikowane umiejętności. Podwojona zostaje także szybkość generowania przeciwników.

Cała gra została stworzona w stylu dwuwymiarowym. Na rysunku 6.1 przedstawiony został fragment przykładowej rozgrywki. Oznaczone elementy interfejsu użytkownika to kolejno:

1. **Pasek wskazujący poziom życia bohatera**. Warto zwrócić uwagę na brak oznaczenia dokładnej ilości punktów życia. Ukrycie tej informacji zmusza użytkownika do kontrolowania wartości tej cechy w trakcie gry, a także do unikania wszystkich przeciwników, niezależnie od tego, jakie obrażenia zadają.

2. **Ikony aktualnie wybranej broni i dostępności mocy specjalnej**. Służą one przede wszystkim jako elementy informacyjne. Wskazanie używanej broni pozwala graczowi na zorientowanie się, czy wzmocnienie w postaci innego typu pocisków jest lepsze od aktualnie posiadanego. Ikona mocy specjalnej natomiast jest nie tylko wskazaniem jej dostępności. W trakcie czasu pomiędzy kolejnymi jej użyciami ikona ta spełnia funkcję orientacyjnego licznika, który wskazuje użytkownikowi, kiedy ponownie będzie mógł z niej skorzystać.

3. **Ilość punktów zdobytych przez gracza oraz pasek postępu dla danego poziomu**. Podobnie jak w przypadku ilości życia, pokazywany jest jedynie orientacyjny postęp gracza na danym poziomie, dzięki czemu użytkownik nie jest do końca pewien, kiedy ukończy poziom. Z drugiej strony

Rys. 6.1. Ekran gry z oznaczonymi elementami interfejsu użytkownika

wyświetlana jest suma punktów zdobytych w trakcie całej rozgrywki, co stanowi pewnego rodzaju motywację dla gracza, aby zdobyć jak najwyższy wynik.

Tak jak zostało wspomniane na początku rozdziału, do implementacji gry wykorzystano silnik Unity. W momencie tworzenia projektu zdecydowano się na użycie wersji 2018.3.0f2 przeznaczonej do budowania gier na systemy operacyjne oparte o architekturę 64-bitową. Ponieważ projektowanie gier przy pomocy tego silnika polega na tworzeniu elementów, które są wprowadzane w interakcje pomiędzy sobą, a logika za to odpowiadająca zawarta jest w skryptach dołączanych do obiektów, zdecydowano się na stworzenie hierarchii, gdzie pierwszym poziomem są poszczególne elementy rozgrywki umieszczane na scenie gry, drugim natomiast są skrypty odpowiadające za logikę danego obiektu lub innych elementów takich jak interfejs użytkownika czy postępy gracza. Do obiektów stanowiących główne elementy świata gry należą:

1. **Player** reprezentujący statek, którym kieruje gracz. Poza elementami graficznymi zawiera on następujące skrypty związane ze sterowanym statkiem:
   - **PlayerController**, opisujący sposób poruszania się kontrolowanego statku. Przy jego pomocy ustawiana są także prędkość oraz czas potrzebny na wyhamowanie pojazdu.
   - **PlayerShooter**, odpowiadający za logikę strzelania wybraną bronią, a także za użycie mocy specjalnej. Jej głównym zadaniem jest zarządzanie tworzeniem pocisków i aktywacji animacji oraz dźwięków związanych ze strzałem lub aktywacją mocy specjalnej. Skrypt odpowiada także za integrację z elementami interfejsu użytkownika odpowiedzialnymi za wyświetlanie aktualnie używanej broni i dostępności mocy specjalnej. Umożliwia on także zmianę wykorzystywanej broni oraz zarządzanie parametrem czasu odnowienia mocy specjalnej.
   - **Player**, odpowiadający za kontrolowanie życia postaci. Mowa tutaj nie tylko o zmniejszaniu jego ilości po przyjęciu obrażeń lub zwiększaniu podczas leczenia bohatera, ale również ustawianiu nowej wartości maksymalnej punktów życia po zakończeniu poziomu. Skrypt

ten zarządza elementem interfejsu użytkownika odpowiedzialnym za wyświetlanie aktualnej ilości życia. Kontroluje także wywołanie animacji i dźwięków uruchamianych po śmierci bohatera, a także oddelegowanie akcji odpowiedzialnych za zatrzymanie gry do obiektów sterujących rozgrywką.

2. **Obiekty reprezentujące przeciwników**. Każdy z nich różnił się przypisanymi elementami graficznymi oraz skryptami, które mogą różnić się w zależności od typu przeciwnika. Do skryptów sterujących logiką przeciwników należą:

   – **Enemy**, odpowiadający przede wszystkim za zarządzanie poziomem życia przeciwnika, zmniejszaniu go po zadaniu obrażeń przez gracza, zmianie elementów graficznych w zależności od ilości punktów życia przeciwnika, oraz za wywołanie animacji i dźwięków mających nastąpić po śmierci przeciwnika. Skrypt posiada także parametr związany z ilością obrażeń zadawanych podczas zderzenia z graczem i zarządza samą akcją kolizji, zabierając graczowi ustaloną ilość punktów życia.

   – **EnemyMovement**, odpowiadający za sposób poruszania się przeciwnika. W zależności od typu wroga, skrypt zarządza czy ma się on zbliżać do gracza, okrążać go, czy może stać w miejscu. Zawiera on parametry dotyczące szybkości przeciwnika, maksymalnej odległości od gracza, oraz dystansu od niego, na jakim część przeciwników ma się zatrzymać.

   – **EnemyShooter**, odpowiadający za kontrolę strzałów przeciwnika, ich szybkość, stworzenie pocisków oraz wywołanie efektów dźwiękowych strzału. Skrypt ten jest przypisywany wyłącznie do przeciwników typu strzelającego.

   – **EnemyBomber**, zarządzający logiką wybuchu przeciwnika. Kontroluje przede wszystkim zadanie graczowi obrażeń i odepchnięcie innych przeciwników, jeśli znajdują się w zadanej przez parametr odległości. Odpowiada także za uruchomienie animacji oraz dźwięku związanych z odliczaniem do wybuchu.

3. **Obiekty reprezentujące wzmocnienia, które gracz może zdobyć w trakcie rozgrywki**. Poza elementami graficznymi, w zależności od rodzaju ulepszenia, każdy z nich ma przypisany do siebie następujące skrypty:

   – **PowerUp**, odpowiadający za wywołanie efektu wzmacniającego gracza, oraz ciągły obrót obiektu wzmocnienia. Jest to skrypt abstrakcyjny, co oznacza, że nie jest przypisany bezpośrednio do elementu, a stanowi bazę dla skryptów mogących rozszerzać jego zachowanie. W tym przypadku rozszerzeniem jest funkcja odpowiedzialna za nałożenie efektu wzmacniającego na gracza.

   – **HealthPowerUp**, będący rozszerzeniem skryptu PowerUp. W ramach wzmocnienia skrypt przywracał pewną ilość punktów życia określoną przez parametr.

– **WeaponPowerUp**, będący rozszerzeniem skryptu PowerUp. Wzmocnienie w tym przypadku polegało na zmianie broni gracza na tę, która była przypisana do wzmocnienia w formie parametru.

Poza wyżej wymienionymi elementami, stanowiącymi część gry widoczną dla gracza, stworzone zostały obiekty będące częścią menu głównego, a także nieposiadające reprezentacji graficznej elementy zarządzające. Do każdego z nich został przypisany skrypt odpowiadający za kontrolowanie innych elementów rozgrywki:

– **MainMenu**, zawierający metody uruchamiające oraz zamykające grę. W ramach funkcji inicjującej rozgrywkę skrypt uruchamiał animacje wyświetlające ekran powitalny, a następnie rozpoczynał faktyczną rozgrywkę.

– **PowerUpGenerator**, odpowiadający za generowanie obiektów wzmocnień w trakcie gry. Kontrolował on tworzenie wybranego ulepszenia w losowo wybranej pozycji w określonej odległości od gracza. Ilość wzmocnień znajdujących się w trakcie gry jest ograniczona, aby użytkownik odczuł, że są to elementy wyjątkowe. Wartość ta, wraz z częstotliwością generowania wzmocnień i odległością od gracza, w jakiej obiekty mają być generowane, sterowane są przez parametry skryptu.

– **EnemySpawner**, wykorzystywany do zarządzania logiką odpowiadającą za generowanie przeciwników w świecie gry. Skrypt ten posiadając określoną w parametrze listę szablonów wrogich jednostek, które mają być generowane, co pewien czas tworzy każdego z przeciwników w formie fali. Każdy z nich pojawia się w losowo dobranych miejscach. Częstotliwość fal i odległość, w jakiej przeciwnicy są generowani od gracza, określane są poprzez parametry skryptu.

– **GameManager**, zarządzający zatrzymywaniem rozgrywki, oraz sterowaniem elementami po zakończeniu gry. Skrypt ten kontrolował flagi blokujące wszystkie inne obiekty w przypadku zatrzymania gry. Odpowiadał on także za wywołanie animacji wyświetlanych po śmierci gracza, zmianę sceny podczas wyjścia z gry, czy jej ponowne uruchomienie w przypadku chęci ponownego rozpoczęcia rozgrywki.

– **Progress**, będący miejscem sterującym postępami gracza. Przy jego pomocy gromadzone są punkty zdobyte w trakcie rozgrywki. Po zdobyciu punktów skrypt sprawdza, czy przekroczony został wynik wymagany do ukończenia poziomu. Jeżeli tak to ze sceny usuwane są wszystkie obiekty przeciwników i pociski, a następnie w zależności od tego, czy dany poziom był ostatni, uruchamia funkcje odpowiadające za zakończenie gry lub przejście do kolejnego poziomu. W przypadku końca rozgrywki, skrypt aktywuje animację wyświetlającą gratulacje dla gracza, a następnie oznacza rozgrywkę jako zakończoną przy pomocy flagi ze skryptu GameManager. W przypadku przejścia na kolejny poziom aplikowana jest nagroda w postaci zwiększonych punktów życia, a następnie ładowane są parametry nowego poziomu. Jednocześnie, niezależnie od tego, czy gracz ukończył poziom, aktualizowane są elementy interfejsu wyświetlającego ilość zdobytych punktów oraz sprawdzany jest warunek wymagany do uruchomienia trybu trudnego gry. Jeżeli został

on spełniony, to lista generowanych przeciwników zostaje zmieniona na trudniejszą wersję, a częstotliwość fal jest podwajana. W ramach parametrów skryptu należy wyróżnić warunki uruchomienia trybu trudnego i czas jego trwania, a także listę poziomów w grze. Ten ostatni parametr stanowi główny rdzeń rozgrywki, ponieważ zawiera w sobie szablon każdego z poziomów, na który składają się: listy generowanych przeciwników, zarówno w wersji klasycznej, jak i trudnej, częstotliwość fal wrogów, wynik wymagany do ukończenia poziomu, liczba punktów życia dodana graczowi po jego ukończeniu, a także flaga oznaczająca czy dany poziom jest ostatnim.

Wszystkie powyższe elementy stanowiły bazę dla podstawowej wersji gry. Parametry każdego z obiektów zostały dobrane tak, aby gracz odczuwał postęp, który ma doprowadzić go ostatecznie do końca gry. Struktura projektu została zachowana w formie dwóch scen, jednej odpowiedzialnej za menu gry, w drugiej natomiast odbywała się faktyczna rozgrywka. Tak jak to zaznaczono podczas opisywania skryptów, całość gry odbywa się w jednej scenie. Wynika to przede wszystkim z powtarzalności poziomów różniących się między sobą wyłącznie parametrami.

Ostatnim etapem tworzenia podstawowej wersji gry było dostosowanie sterowania do wykorzystywanego kontrolera Dualshock 4. Ponieważ Unity dla każdej z gier może mieć zdefiniowaną listę nazw odpowiadającym danym przyciskom lub elementom, które mają pewien zakres działania, poza przystosowaniem standardowych nazw do odpowiednich przycisków na kontrolerze, dodane zostały osie zawierające w sobie odczyty z prawego analoga kontrolera. Pozwoliło to na wykorzystanie go do kontrolowania kierunku, w którym miał się obracać statek.

6.3. Odczyt danych fizjologicznych i zmian emocji

Równolegle, w trakcie tworzenia implementacji podstawowej gry, stworzony został moduł umożliwiający pobieranie danych fizjologicznych i odczytów akcelerometru z urządzeń opisanych w rozdziale 4. Pobrane informacje wykorzystano do przygotowania prostego interfejsu umożliwiającego implementację mechanizmów oddziaływania na strukturę gry w zależności od otrzymanych danych i tym samym domknięcie pętli afektywnej.

Pierwszym krokiem było przygotowanie skryptów umożliwiających odczyt sygnałów bezpośrednio z urządzeń. W przypadku BITalino (r)evolution kit skorzystano z rozwiązania dostępnego na stronie producenta [58]. Aby uniknąć ewentualnych problemów z kompatybilnością, spośród dostępnych interfejsów programistycznych na platformę Unity, wybrano ten dostosowany do wyższych wersji oprogramowania. Bazując na przykładach oferowanych w ramach interfejsu, stworzony został obiekt BITalino, który w pełni będzie odpowiadał za integrację z urządzeniem. Przypisano do niego następujące skrypty:

– **BitalinoSerialPort**, udostępniony jako część wykorzystanego interfejsu. Skrypt umożliwiał konfigurację parametrów związanych z nawiązaniem połączenia z płytką poprzez transmisję szeregową, zarówno przez Bluetooth, jak i kabel USB. W ramach parametrów możliwe było ustawienie między innymi limitów czasu oczekiwania na odczyt i zapis danych, jednak najistotniejsza była

możliwość wprowadzenia nazwy kanału, przy pomocy którego możliwa była komunikacja z urządzeniem.

– **BitalinoManager**, udostępniony jako część wykorzystanego interfejsu. Skrypt ten umożliwia przede wszystkim konfigurację listy wykorzystywanych kanałów i określenia, jakiego rodzaju sygnały są przypisane do każdego z nich. W przypadku przygotowywanego projektu ustalony został wyłącznie jeden kanał przyjmujący odczyty z elektromiogramu. Skrypt umożliwia także ustalenie częstotliwości wysyłania danych. Ze względu na chęć uzyskania jak najdokładniejszych pomiarów, zdecydowano się na ustawienie najwyższej możliwej wartości, czyli tysiąca próbek na sekundę.

– **BitalinoReader**, udostępniony jako część wykorzystanego interfejsu. Skrypt służy do zarządzania przetwarzaniem danych z płytki, tego, w jakiej postaci są one zwracane, oraz jak wielki jest bufor, w którym są przechowywane. Jego rozmiar ustawiono na sto ostatnich elementów. Wybór takiej wartości wynika przede wszystkim ze względu na chęć uzyskania jak najmniejszego opóźnienia od momentu otrzymania pomiarów bezpośrednio z urządzenia do czasu wykorzystania ich przetworzonej wersji. Ponieważ sygnał z elektromiografu ma służyć wykryciu zmian w napięciu mięśni, nie była tutaj wymagana wysoka ilość próbek.

– **SensorController**, przygotowany w ramach niniejszej pracy magisterskiej. Głównym zadaniem tego skryptu był odczyt w czasie rzeczywistym bufora danych, wyznaczanie średniej z wartości bezwzględnych pomiarów z elektromiografu, a następnie sprawdzenie, czy przez określony czas miało miejsce napięcie mięśni przedramienia. Użycie wartości bezwzględnej wynika z oscylacji występujących w sygnale pobranym z elektromiografu. Wykrycie napięcia zostało oparte na warunku:

$$|EMG|_{avg} \geq mul \cdot |EMG|_{base}$$

gdzie $|EMG|_{avg}$ jest aktualnie odczytaną średnią wartością, mul mnożnikiem ustawianym jako parametr umożliwiający regulację wykrycia napięcia mięśni, a $|EMG|_{base}$ średnim pomiarem bazowym. Ostatnia z wartości jest wyznaczana podczas fazy kalibracji odbywającej się po rozpoczęciu pomiarów. Przez ustaloną parametrem ilość sekund, w których trakcie użytkownik nie powinien napinać mięśni przedramienia, zbierane są odczyty z elektromiografu, a następnie na ich podstawie obliczana jest wartość służąca jako pomiar referencyjny, z którym porównywane są odczyty w trakcie rozgrywki.

Dodanie czasu, przez jaki miał być napięty mięsień oraz mnożnika w warunku miało na celu odrzucenie krótkich, losowych ruchów ręką, jakie mogły wystąpić w trakcie korzystania z kontrolera. Pozwoliło to także dać użytkownikowi świadomość kontroli nad napięciem mięśni, ponieważ musiał wykonać tę czynność przez określony czas.

Aby obiekty, mające reagować na napięcie mięśni, mogły w prosty sposób być o tym informowane, został wykorzystany mechanizm zdarzeń z języka C#. W ramach skryptu SensorController przygotowano delegata, będącego typem przechowującym referencję do metody i określającym jakie parametry i typ zwrotny powinna zawierać metoda, której sygnaturę określa delegat. Następnie dodane zostało zdarzenie o typie przygotowanego delegata służące jako miejsce rejestracji i wywołania metod, które mają obsłużyć daną sytuację. W przypadku skryptu SensorController jest ono wywoływane w momencie wykrycia napięcia mięśni przez określony czas. Tak przygotowany mechanizm pozwala na prawie całkowite uniezależnienie elementów związanych z pomiarami fizjologicznymi od skryptów będących częścią implementacji gry.

Kolejnym krokiem było przygotowanie skryptów do odczytu sygnałów z opaski Garmin HRM-Run i akcelerometru wbudowanego w kontroler Dualshock. Zostaną one następnie wykorzystane w skryptach komunikujących się z modelem opisanym rozdziale 5 i zwracających odczyty emocji odczuwanych przez użytkownika w trakcie rozgrywki.

Podobnie jak w przypadku odczytów z płytki BITalino (r)evolution kit, do pomiaru tętna wykorzystana została biblioteka do obsługi protokołu ANT+ oraz przykład znajdujące się na stronie producenta [59]. Przy ich pomocy przygotowano następujące skrypty umożliwiające pomiar tętna z opaski:

– **AntReader**, przygotowany na podstawie przykładu udostępnionego przez twórców biblioteki. Kod został dostosowany w taki sposób, aby można było uruchomić odczyt przy pomocy funkcji. Do najważniejszych parametrów skryptu należy przede wszystkim numer i typ urządzenia, z jakiego mają być odbierane dane. Ponieważ odczyty z czujnika przychodzą w formie ramki z danymi, do skryptu dodany został parser tworzący obiekt zawierający informacje na temat wartości tętna, ilości uderzeń od rozpoczęcia pomiarów, czas ostatniego odczytu, oraz wartość interwału pomiędzy ostatnimi dwoma uderzeniami w milisekundach. Tak przygotowany obiekt jest następnie wysyłany jako parametr zdarzenia, które także zostało dodane jako modyfikacja przykładu. Takie podejście wynika przede wszystkim z protokołu bazującego na przesyłaniu zdarzeń w momencie wystąpienia akcji.

– **HeartRateManager**, zarządzający danymi przesyłanymi przez skrypt AntReader. Jego głównym zadaniem jest przechowywanie obiektów z informacją na temat pracy serca w buforze, którego rozmiar jest zadany przez parametr. Skrypt ten umożliwia także wyizolowanie z bufora wartości tętna i interwałów pomiędzy uderzeniami w formie tablic.

W przypadku odczytów akcelerometru z kontrolera Dualshock 4 producent udostępnia oprogramowanie wbudowane w silnik Unity, jednak jest ono dostępne wyłącznie dla osób projektujących gry na konsolę PlayStation 4. W związku z tym wykorzystany został odpłatny dodatek Rewired, udostępniający obsługę wielu kontrolerów do gier, nie tylko na poziomie obsługi przycisków, ale również wbudowanych funkcjonalności takich jak kontrola wibracji, oświetlenia, czy odczytów z sensorów zawartych w kontrolerze, do których można zaliczyć akcelerometr oraz żyroskop. Do odbierania sygnałów z kontrolera i ich interpretacji przygotowane zostały następujące skrypty:

– **AccelerationReader**, odpowiadający za pomiar i wstępną filtrację danych z akcelerometru. Częstotliwość odczytu jest ustalana za pomocą parametru skryptu, a każda odebrana wartość jest filtrowana przy pomocy filtru Kalmana, aby pozbyć się ewentualnych szumów.

– **AccelerationHandler**, odpowiadający za przechowywanie i analizę pomiarów odebranych przez skrypt AccelerationReader. Dane przechowywane są w dwóch listach: jednej przeznaczonej do fazy kalibracji oraz drugiej będącej buforem odczytów z określonej parametrem ilości czasu. Do wyznaczenia aktywności użytkownika wykorzystano zryw, czyli zmianę wartości przyspieszenia w czasie. Podczas fazy kalibracji po zebraniu danych z akcelerometru do przygotowanej listy obliczana była średnia wartość zrywu, która posłuży jako pomiar referencyjny. W ramach jednej z funkcji skrypt umożliwia pobranie aktualnej aktywności użytkownika w postaci jednego z trzech stanów: bardzo spokojny, spokojny i podekscytowany. Funkcja oblicza wartość zrywu z aktualnie zapisanych pomiarów, a następnie określa czy użytkownik jest podekscytowany na podstawie warunku:

$$jerk_{avg} \geq mul \cdot jerk_{base}$$

gdzie $jerk_{avg}$ jest aktualnie odczytaną wartością zrywu, mul mnożnikiem ustawianym jako parametr umożliwiający regulację poziomu granicznego dla ekscytacji, a $jerk_{base}$ jest pomiarem zrywu uzyskanym w trakcie kalibracji. Następnie, jeśli stan nie został określony jako ekscytacja, sprawdzana jest ostatnio odczytana wartość aktywności użytkownika. Jeżeli gracz był spokojny, a ostatni pomiar odbył się w określonym czasie, to skrypt uznawał gracza za bardzo spokojnego. W przeciwnym wypadku zwracany był stan spokojny.

Tak przygotowane skrypty do odczytu i wstępnej analizy danych z czujnika do pomiaru tętna i akcelerometru zostały następnie wykorzystane do ustalenia stanu emocjonalnego użytkownika. Umieszczono je w obiekcie EmotionManager, który skupia wszystkie skrypty odpowiedzialne za określenie emocji gracza. Aby powiązać odczytane dane z modelem uczenia maszynowego oraz wyznaczyć odczuwane przez użytkownika emocje, do obiektu EmotionManager dodane zostały następujące skrypty:

– **ClassifierApiManager**, odpowiadający za komunikację z serwerem zawierającym model rozpoznający emocje. Skrypt wykorzystując dane na temat tętna i interwałów pomiędzy kolejnymi uderzeniami znajdujące się w skrypcie HeartRateManager, wykonuje zapytanie HTTP do serwera i zapisuje zwracana klasę emocji oraz parametry, dla których dana wartość została zwrócona. Skrypt zawiera także flagę informującą, czy od momentu ostatniego odczytu emocji przez inne skrypty była ona aktualizowana.

– **EmotionManager**, zarządzający pozostałymi skryptami do wyznaczania stanu emocjonalnego, a także będący punktem wyjściowym dla przekazywania go do innych obiektów. Głównym zadaniem skryptu jest uruchomienie pobierania danych z opaski i akcelerometru, a następnie określenie emocji użytkownika na podstawie klasy zwróconej przez model rozpoznający emocję oraz stanu pobudzenia zwracanego przez skrypt AccelerationHandler. W momencie pobierania emocji

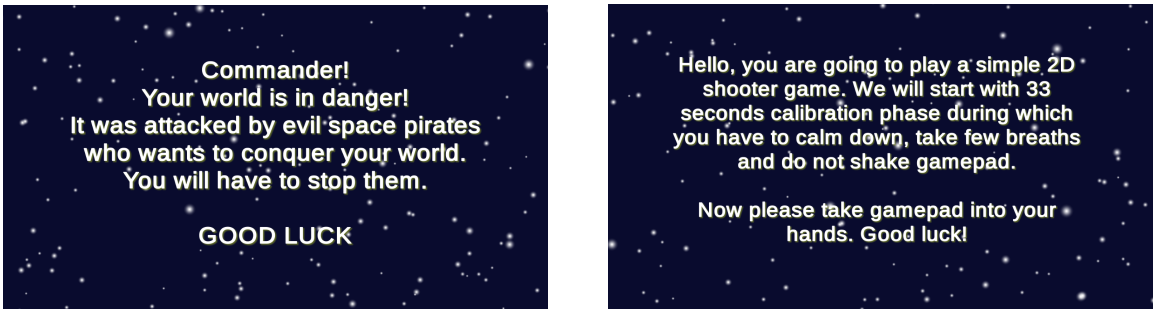
wywoływane jest zapytanie do serwera przy pomocy skryptu ClassifierApiManager o stan emocjonalny użytkownika na podstawie danych z opaski. Zwrócona klasa emocji jest konwertowana do obiektu zawierającego poziomy wartości *valence* i *arousal*, a następnie przy pomocy odczytanego z akcelerometru stanu pobudzenia, są one weryfikowane. Ponieważ pomiar *arousal* jest związany bezpośrednio z pobudzeniem gracza, to właśnie ta wartość była modyfikowana. Jeżeli stan zwracany przez odczyty z akcelerometru wskazywał na pobudzenie użytkownika, poziom *arousal* zostaje zwiększony, natomiast w przypadku bardzo spokojnej gry, jest on zmniejszany. Dzięki takiemu podejściu można łatwo weryfikować klasę zwracaną przez model, która nie zawsze mogła dokładnie odzwierciedlać stan emocjonalny gracza w danym momencie rozgrywki. Po odczytaniu stanu emocjonalnego następowało wywołanie zdarzenia o nowej emocji, które jako parametry przyjmowało wartość odczytaną w danym momencie oraz stan poprzedni. Wykorzystanie poprzednio odczytanego stanu miało na celu rozszerzenie kontekstu zmian emocji użytkownika i ich interpretację przez konkretne elementy gry. Do najważniejszych parametrów skryptu EmotionManager należą przede wszystkim czas poświęcony na kalibrację oraz parametr określający co ile ma zostać wykonane zapytanie o stan emocjonalny użytkownika.

Obiekty BITalino i EmotionManager zostały następnie połączone w hierarchii pod obiektem AffectiveManager. Można traktować go jako rdzeń przygotowywanego w ramach pracy magisterskiej interfejsu. Tak przygotowana hierarchia umożliwiała łatwe jego wprowadzenie do istniejącego już kodu gry. Do obiektu AffectiveManager został przypisany także skrypt o tej samej nazwie, którego głównymi zadaniami było uruchamianie oraz wyłączanie mechanizmów opisanych powyżej, a także rozpoczynanie fazy kalibracji dla pomiarów z akcelerometru oraz odczytów z elektromiografu. Dodany został także obiekt Logger odpowiadający za zapisywanie informacji o odczuwanych stanach emocjonalnych oraz momentach wykrycia napięcia mięśni. W przypadku odczytywania emocji zapisywane są poziomy *valence* i *arousal* po weryfikacji pomiarami z akcelerometru oraz wartości cech, dla których zostały one wyliczone. Pomiary te mogą posłużyć jako dodatkowe dane uczące mogące poprawić skuteczność modelu opisanego w rozdziale 5. W przypadku odczytów z elektromiografu zapisywane są momenty napięcia mięśni przedramienia wraz ze średnią wartością z elektromiografu. Każde zdarzenie jest przechowywane w pamięci wraz z czasem jego wystąpienia, a w momencie zakończenia rozgrywki są one zapisywane do pliku.

6.4. Domknięcie pętli afektywnej

Po stworzeniu opisanego w poprzedniej sekcji interfejsu kolejnym krokiem było domknięcie pętli afektywnej. Oznaczało to uzależnienie pewnych elementów gry od stanu emocjonalnego i reakcji użytkownika. Aby to osiągnąć, wymagana była modyfikacja podstawowej wersji gry i zmiana działania jej mechanik w zależności od wartości zwracanych przez skrypty EmotionManager i SensorController.

Aby umożliwić integrację gry z interfejsem, w głównym menu gry przygotowana została opcja, która decyduje o tym, czy mechaniki afektywne mają zostać uruchomione po rozpoczęciu gry. Następnie,



(a) Wprowadzenie w wersji podstawowej (b) Wprowadzenie w wersji afektywnej

Rys. 6.2. Ekran wprowadzenia dla obu wersji gier

w ramach skryptu MainMenu, zmodyfikowany został proces wyświetlania wprowadzenia. W przypadku wersji afektywnej wyświetlony zostaje tekst informujący użytkownika o przeprowadzanej na początku kalibracji interfejsu oraz zasadach, których powinien przestrzegać w trakcie jej trwania (rys. 6.2). Użytkownik proszony jest o uspokojenie się, wzięcie kilku głębokich oddechów oraz nieporuszanie kontrolerem w trakcie przeprowadzania kalibracji. W tym samym momencie uruchamiany zostaje odczyt danych z urządzeń pomiarowych, a po kilku sekundach rozpoczynana jest kalibracja trwająca przez określony w parametrach interfejsu czas. Wcześniejsze uruchomienie odczytu pozwoliło na odrzucenie pierwszych pomiarów z urządzeń, które mogły charakteryzować się dużym błędem ze względu na start sprzętu.

Kolejnym krokiem w integracji interfejsu było przygotowanie mechanik afektywnych, które wpływają na poszczególne elementy rozgrywki w zależności od odczuwanych przez użytkownika emocji lub napięcia mięśni przedramienia. W ramach skryptów odpowiadających za konkretne elementy rozgrywki przygotowane zostały funkcje, które następnie zarejestrowano jako obserwatorów wystąpienia nowej emocji lub napięcia mięśni. Aby ułatwić interpretację poziomów *valence* i *arousal* zwracanych w ramach obiektu reprezentującego odczuwaną emocję, stworzone zostały stałe, które przypisują każdą z par obu wartości do konkretnej emocji. Interpretacja każdej z kombinacji została przedstawiona w tabeli 6.1. W ramach zaimplementowanych mechanik afektywnych można wyróżnić:

1. **Aktywacja mocy specjalnej po wykryciu napięcia mięśni przedramienia**. Mechanika ta stanowi alternatywę dla uruchomienia tej mechaniki przy pomocy przycisku na kontrolerze. Uruchomienie jej bez wyłączania akcji z podstawowej wersji gry pozwala użytkownikowi wybrać sposób aktywacji, który jest dla niego wygodniejszy.

2. **Aktywacja mocy specjalnej w momencie, gdy gracz przez dłuższy czas odczuwa złość**. W odróżnieniu od poprzedniego punktu, w tym przypadku moc specjalna aktywowana jest niezależnie od jej dostępności w momencie, gdy poprzednia i aktualnie odczytana emocja to złość.

3. **Zmiana ilości punktów życia gracza na podstawie odczuwanych emocji**. Można wyróżnić tutaj następujące możliwości:

Tabela 6.1. Interpretacja kombinacji poziomów *valence* i *arousal* jako emocje.

Arousal / **Valence**	LOW	MEDIUM	HIGH
LOW	Smutek	Złość	Strach
MEDIUM	Zmęczenie	Emocja neutralna	Zaskoczenie
HIGH	Zrelaksowanie	Szczęście	Ekscytacja

  - Odebranie graczowi 40% punktów życia w przypadku odczuwania przez dłuższy czas emocji neutralnej, zmęczenia lub zrelaksowania. Taka reakcja miała na celu pobudzenie gracza do skupienia się na rozgrywce ze względu na nagłą trudną sytuację.
  - Odebranie graczowi 20% punktów życia w momencie odczuwania przez dłuższy czas szczęścia lub ekscytacji. Emocje te zostały wydzielone do lżejszej wersji negatywnego efektu, aby uniknąć nadmiernego zdenerwowania gracza, ale jednocześnie wzbudzić w nim potrzebę skupienia się na grze.
  - Przywrócenie 10% punktów życia w momencie odczuwania smutku lub strachu. Efekt ten miał posłużyć jako pomoc dla gracza i wywołanie przez to u niego pozytywnych emocji.

4. **Aktywacja dodatkowych fal przeciwników w momencie odczuwania emocji neutralnej, zrelaksowania, zmęczenia lub szczęścia**. W przypadku trzech pierwszych stanów emocjonalnych mechanika ta miała na celu postawienie graczowi wyzwania. Uruchomienie jej w momencie odczuwania szczęścia przez gracza miało na celu wywołanie u niego emocji przeciwstawnych takich jak strach czy złość.

5. **Aktywacja i dezaktywacja trybu trudnego gry w zależności od odczuwanej emocji**. Mechanika ta zastępowała jej podstawową wersję opartą na częstotliwości zdobywania punktów. W przypadku odczuwania przez użytkownika złości lub strachu była ona dezaktywowana, natomiast aktywacja następowała w momencie doświadczania przez gracza emocji neutralnej, zrelaksowania lub zmęczenia.

Implementacja interfejsu do rozpoznawania emocji i zachowań gracza umożliwiła stworzenie gry zawierającej pętlę afektywną. Aby zrealizować w pełni założenia programowania afektywnego, każda z mechanik zmieniała rozgrywkę w taki sposób, aby wzbudzić w użytkowniku inny rodzaj emocji od aktualnie odczuwanych. Przygotowany interfejs umożliwił łatwą implementację każdej z nich bez konieczności modyfikacji oryginalnego kodu gry dzięki rejestracji funkcji uruchamianych w momencie odczytania emocji.

7. Badania

Po przygotowaniu implementacji opisanej w poprzednim rozdziale przeprowadzona została ewaluacja rozwiązania, w ramach której badane osoby mogły wypróbować stworzoną grę w obu przygotowanych wersjach. Badania odbywały się przy pomocy komputera Lenovo Z-50-70 z zainstalowanym systemem Windows 10. Cechował się następującą konfiguracją sprzętową:

- Procesor Intel Core i7-4510U
- Karta graficzna NVIDIA GeForce 840M
- Pamięć RAM 16GB, DDR3
- Rozdzielczość ekranu 1920x1080

7.1. Procedura eksperymentu

1. Przedstawienie osobie badanej jak będzie wyglądać test. Wspomniano tutaj o zagraniu przez osobę badaną w dwie wersje gry. Omówiono także, jak będzie wyglądał montaż urządzeń w przypadku drugiej wersji gry.
2. Przedstawienie sterowania w pierwszej części gry.
3. Odpowiedzenie na ewentualne pytania osoby badanej.
4. Uruchomienie aplikacji w pierwszej wersji gry.
5. Pilnowanie czasu gry, wynoszącego od 5 do 10 minut.
6. Umożliwienie ewentualnej przerwy.
7. Pomoc osobie badanej w założeniu opaski do pomiaru tętna i elektrod na przedramieniu.
8. Upewnienie się, czy gra poprawnie wykrywa urządzenia. W razie problemów z połączeniem ponawiano próbę.
9. Uruchomienie aplikacji w wersji afektywnej.

10. Pilnowanie czasu gry, wynoszącego od 5 do 10 minut.
11. Wręczenie ankiety osobie badanej.
12. Odpowiedź na ewentualne pytania osoby badanej, zebranie dodatkowych uwag.

7.2. Uczestnicy

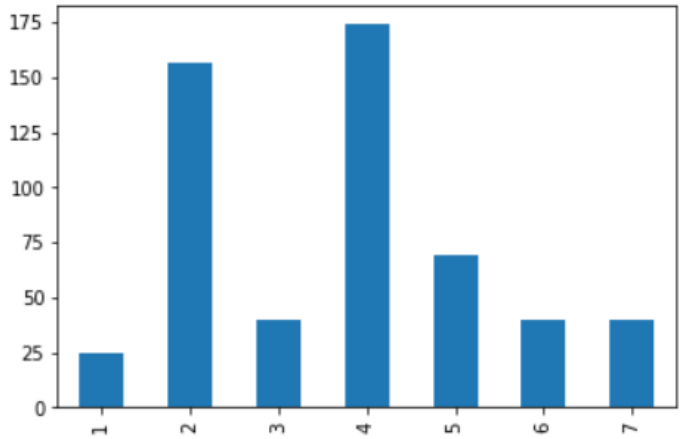
Badania zostały przeprowadzone na 7 osobach, gdzie tylko jedna z nich była kobietą. Wiek osób wahał się od 20 do 30 lat. Większość osób należała do grupy studentów, w dwóch przypadkach były to osoby pracujące w branży programistycznej. Wartym wspomnienia jest także fakt, że jedna z osób charakteryzowała się chorowaniem na nadciśnienie tętnicze, co mogło wpłynąć znacznie na jej wyniki badań i odczytywanych emocji.

7.3. Analiza wyników

W ramach badań od uczestników zostały zebrane logi zawierające poziomy *valence* i *arousal* zwrócone przez przygotowany interfejs do rozpoznawania emocji z uwzględnieniem odczytów z akcelerometru. Jako dodatkowy atrybut przydatny do analizy zostały zapisane również wartości cech, dla których dana klasa emocji została określona. Dzięki temu zebrane dane mogą zostać wykorzystane jako uzupełnienie zbioru uczącego użytego do stworzenia modelu rozpoznawania stanów emocjonalnych opisanego w rozdziale 5. Innym zdarzeniem zapisanym w ramach logów były momenty wykrycia napięcia mięśni przez określony w grze czas. Każdy z uczestników po przeprowadzeniu badań wypełnił także kwestionariusz zawierający pytania na temat wygody użytkowania stanowiska pomiarowego, satysfakcji z zaimplementowanych mechanik afektywnych oraz opinii na temat gry. Wszystkie pytania znajdujące się w kwestionariuszu można znaleźć w dodatku A.

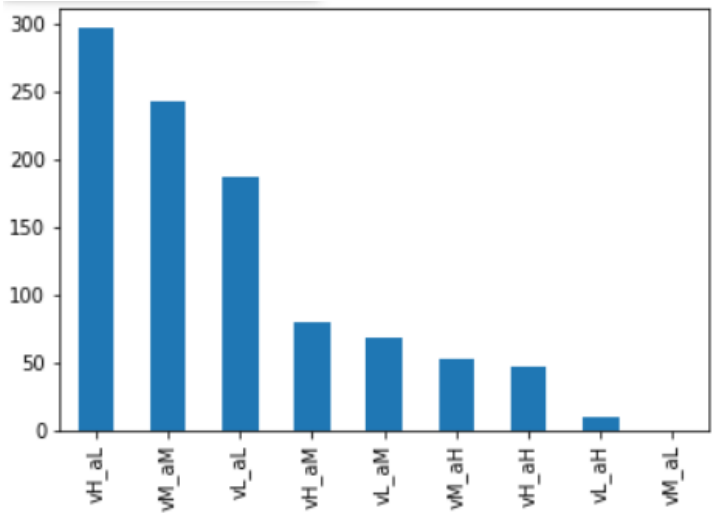
W trakcie wykonywania eksperymentów nie zauważono większych problemów z podłączeniem urządzeń do aplikacji. Dla sześciu osób eksperyment przebiegł w pełni poprawnie i nie wymagał powtórzenia. W przypadku jednej osoby napotkano problem w trakcie rozgrywki, polegający na użyciu mocy specjalnej w momentach, gdy badana osoba nie napinała mięśni przedramienia. Po dokładnym wglądzie okazało się, że w trakcie badania nastąpiło odklejenie się elektrody referencyjnej, przez co sygnał zwracany przez elektromiograf zawierał dużą ilość szumów wpływających na wyliczany w trakcie rozgrywki średni odczyt. Zwrócono na to uwagę podczas późniejszej analizy logów i zauważono, że w dwóch przypadkach wykrycie napięcia mięśni trwało przez dłuższy czas. Nie wpłynęło to jednak na samą rozgrywkę w trakcie eksperymentu, dlatego nie podjęto się powtórzenia badań. Rozkład ilości wykryć napięcia mięśni przedramienia przez osoby badane zostały przedstawione na rysunku 7.1.

W ramach analizy zebranych logów wydzielono zdarzenia zawierające odczyty emocji. Rysunek 7.2 przedstawia liczebność klas dla wszystkich zebranych pomiarów. Warto zauważyć, że najczęściej występującymi klasami są te, które interpretowane są kolejno jako zrelaksowanie, emocja neutralna oraz



Rys. 7.1. Ilość wykryć napięcia mięśni przedramienia dla każdej z badanych osób

smutek. Pierwsze dwie emocje łatwo można wytłumaczyć, ponieważ w trakcie rozgrywki gracze na początku nie znali jej wszystkich mechanik, natomiast po jakimś czasie od gry przyzwyczaili się do nich i czerpali z gry przyjemność. Dziwne wydaje się jednak tak częste występowanie smutku u graczy. Emocja ta wprawdzie mogła się pojawić podczas poznawania mechanik utrudniających rozgrywkę, jednak bardziej adekwatnymi emocjami mogłyby być tutaj złość lub strach. Powodem takiego odczytu mógł być także mechanizm odpowiadający za weryfikację emocji zwróconych przez model uczenia maszynowego przy pomocy odczytów z akcelerometru. W przypadku gdy gracz będzie okazywał emocje wyłącznie poprzez zmiany sygnałów fizjologicznych, natomiast kontroler będzie się starał trzymać nieruchomo, wartość *arousal* zostanie zmniejszona. Jeżeli emocją zwróconą przez model w tym przypadku będzie złość, która jest jedną z oczekiwanych reakcji na utrudnienie gry, to interfejs po weryfikacji zinterpretuję odczytaną emocję jako smutek.



Rys. 7.2. Liczebność poszczególnych klas zwróconych przez interfejs do rozpoznawania emocji

Tabela 7.1. Ocenienie poziomu trudności gry w obu jej wersjach

	Wersja pierwsza - bez pętli afektywnej	**Wersja druga - z pętlą afektywną**
Bardzo łatwa	0	0
Łatwa	0	0
Średnia	7	4
Trudna	0	3
Bardzo trudna	0	0

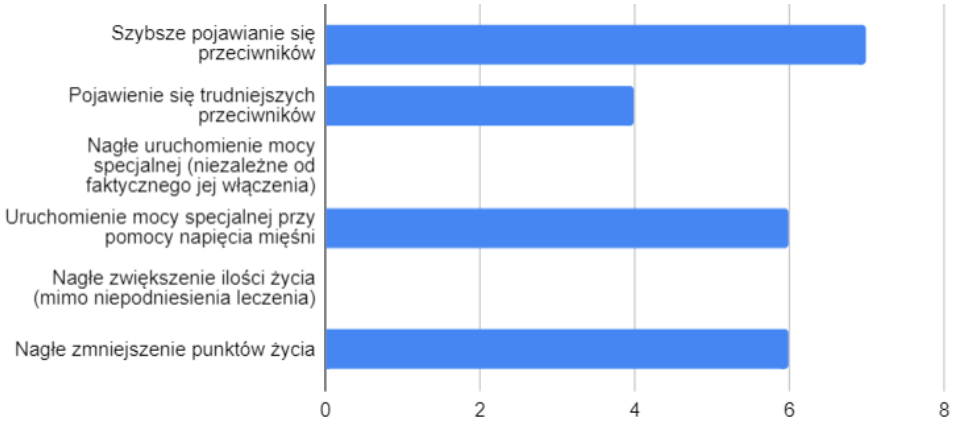
Tabela 7.2. Ocena wygody stanowiska pomiarowego

Ocena wygody stanowiska	
Niewygodne	0
Trochę niewygodne	0
Wygodne	5
Bardzo wygodne	2

Badani poproszeni o ocenę trudności gry (tab. 7.1), w przypadku wersji bez pętli afektywnej zgodnie określili ją jako rozgrywkę o średniej trudności. Co jest ciekawe, w przypadku wersji afektywnej trzy osoby określiły grę jako trudną, pozostali badani natomiast zostali przy takiej samej ocenie jak w wersji podstawowej. Podczas odpowiadania na pytania badanych, zauważyli oni, że w przypadku gry z pętlą afektywną pojawiało się znacznie więcej mechanik utrudniających grę i właśnie to mogło prowadzić do takiej oceny. W przypadku osób, które nie zauważyły różnicy w trudności, może to sugerować, że wprowadzenie pętli afektywnej nie wpłynęło znacząco na trudność rozgrywki.

W pytaniu o wygodę stanowiska wszystkie badane osoby odpowiedziały, że było ono wygodne (tab. 7.2). Taki odzew może sugerować, że wykorzystanie opaski do pomiaru tętna, BITalino (r)evolution kit oraz kontrolera Dualshock 4 jako platformy wykorzystanej w grach afektywnych jest właściwe. Tę hipotezę może potwierdzić także zgodność odpowiedzi badanych na pytanie, które z elementów stanowiska chcieliby zobaczyć w użyciu na komercyjnym rynku gier. Każda z osób wybrała wszystkie elementy stanowiska, co wskazuje na wysokie zainteresowanie przygotowaną platformą.

Ponieważ jednym z powodów wyboru BITalino (r)evolution kit była chęć sprawdzenia jak duży dyskomfort odczuwa użytkownik podczas używania elektrod przyklejanych do skóry, w ramach pytań o wygodę stanowiska pomiarowego wydzielone zostały dwa pytania dotyczące właśnie tego tematu. Badane osoby zapytane o to, czy elektrody i kable wpięte do urządzenia przeszkadzały w trakcie rozgrywki, w większości przypadków odpowiadały negatywnie. Jedynie jedna osoba nie potrafiła stwierdzić ich wpływu na komfort gry. W ramach drugiego pytania osoby badane zostały zapytane, czy do pomiaru pracy serca wolałyby skorzystać z wykorzystanego rozwiązania w postaci opaski z czujnikiem tętna, czy też wybrałyby urządzenie wykorzystujące elektrody. Wszyscy badani zgodnie wybrali opaskę jako lepsze



Rys. 7.3. Zauważenie mechanik afektywnych przez uczestników badań

rozwiązanie. Konfrontując ze sobą wyniki z obu pytań, można zasugerować, że wykorzystanie elektrod w przypadku zwykłych graczy mogłoby zostać zaakceptowane, aczkolwiek gdy będzie dostępne alternatywne rozwiązanie, gracze chętniej zdecydują się na nie. Ten fakt może wynikać między innymi ze skojarzenia elektrod z medycyną.

Badani zapytani o mechaniki afektywne zauważone w trakcie rozgrywki zdecydowanie wskazali te, które utrudniały rozgrywkę (rys. 7.3). Wybrana została także mechanika uruchamiania mocy specjalnej przy pomocy napięcia mięśni przedramienia. Co ciekawe, nie zostały zauważone mechaniki ułatwiające rozgrywkę, do których zaliczało się nagłe uruchomienie mocy specjalnej i zwiększenie ilości punktów życia. W trakcie rozmowy z badanymi stwierdzali oni, że ilość mechanik negatywnie wpływających na rozgrywkę i częstość ich występowania powodowało, że nie zauważyli oni elementów mających ułatwić grę w wersji afektywnej. Fakt ten mógł wynikać z warunków, jakie musiały zostać spełnione, aby uruchomić elementy utrudniające grę. Jak zostało to opisane w rozdziale 6, większość z nich była włączana w momencie wykrycia emocji takich jak zrelaksowanie czy emocja neutralna, które były najczęściej wykrywane.

Ponieważ część mechanik afektywnych pokrywała się z elementami rozgrywki w wersji podstawowej, badane osoby zostały zapytane o preferowaną wersję każdej z mechanik (tab. 7.3). W przypadku uruchomienia mocy specjalnej i aktywacji trybu trudnego gry, sześć osób zgodnie wybrała wersję afektywną. Opinie jednak były podzielone w przypadku mechaniki kontroli punktów życia, gdzie połowa osób wskazała wersję podstawową jako bardziej satysfakcjonującą. Wybór ten mógł być spowodowany przede wszystkim źle zaprojektowanymi mechanikami afektywnymi, które w przypadku zrelaksowania karały gracza poprzez odebranie mu punktów życia. Jedna z badanych osób nie zauważyła różnicy dla dwóch mechanik, natomiast wybrała wersję podstawową uruchamiania mocy specjalnej. Taki wybór mógł wyniknąć z powodu wadliwości elektrod, które w trakcie odczytów z elektromiografu mogły zostać naruszone, przez co średnia wartość nie była na tyle wysoka, aby aktywować mechanikę.

W ramach dodatkowych uwag badane osoby głównie zwróciły uwagę na pomiar przy pomocy elektrod, który określiły jako mało inwazyjny. Jedna osoba opisała gry wykorzystujące pętlę afektywną jako

Tabela 7.3. Wybór preferowanej wersji mechanik zaimplementowanych w obu wersjach gry

	wersja podstawowa	**wersja z pętlą afektywną**	**Bez różnicy**
Uruchomienie mocy specjalnej	1	6	0
Pojawianie się trudniejszych przeciwników (tryb trudny)	0	6	1
Kontrola punktów życia	3	3	1

ciekawe nowe wyzwanie wymagające skupienia się nie tylko na rozgrywce, ale również na swoich emocjach i reakcjach. Podczas rozmów badane osoby wskazywały elementy, które można poprawić w grze, takie jak wspomniane wcześniej mechaniki afektywne.

8. Podsumowanie

8.1. Wnioski

Niniejsza praca miała na celu zaprojektowanie i przygotowanie interfejsu składającego się ze zbioru platform sprzętowych umożliwiających pomiar sygnałów fizjologicznych i zachowań użytkownika oraz modułu programistycznego w wybranym środowisku do tworzenia gier, którego zadaniem jest określenie stanu emocjonalnego i zachowań użytkownika. Rozwiązanie zostało zrealizowane w kilku etapach.

Pracę rozpoczęto od przeglądu literatury o tematyce emocji i programowania afektywnego. Szczególną uwagę poświęcono modelom reprezentacji stanów emocjonalnych i pojęciu pętli afektywnej. Odwołano się także do aktualnie istniejących gier realizujących założenia programowania afektywnego oraz wykorzystujących sygnały fizjologiczne do wpływania na rozgrywkę.

Kolejnym krokiem była analiza dostępnych rozwiązań w zakresie platform sprzętowych do pomiaru sygnałów fizjologicznych, możliwych mechanizmów wnioskowania oraz narzędzi wykorzystywanych do budowy gier komputerowych. Opisane zostały wady i zalety wyboru poszczególnych urządzeń, działanie wybranych metod wnioskowania oraz możliwości kilku dostępnych darmowych silników do tworzenia gier. Podczas analizy rozwiązań sprzętowych zwrócono uwagę na wyższość platform nasobnych, które oferowały wysoki komfort użytkowania, jednocześnie pozwalając na dokładny odczyt sygnałów fizjologicznych. Wspomniano także o urządzeniach umożliwiające zbieranie danych pośrednich, takich jak odczyty z kontrolerów do gier, na podstawie których możliwe jest określenie stanu użytkownika.

Następnie przeprowadzone zostało przygotowanie części sprzętowej projektu. Określone zostały podstawowe założenia wymagane do spełnienia przez urządzenia wchodzące w skład interfejsu. Najważniejszym aspektem było uzyskanie jak najdokładniejszych pomiarów, nie powodując przy tym wzrostu dyskomfortu w czasie użytkowania. Dla każdego z urządzeń przedstawione zostały argumenty uzasadniające ich wybór. Zwrócono uwagę na dostępne sposoby komunikacji, a także sposób montażu każdej z platform sprzętowych.

Część programistyczną projektu rozpoczęto od przygotowania modelu uczenia maszynowego do rozpoznawania emocji. Wybrane zostały zbiory danych uczących, które następnie poddano wstępnemu przetworzeniu. Określono także cechy bazujące na pomiarach pracy serca oraz zbiór klas rozpoznawanych przez model stanowiących reprezentację poszczególnych stanów emocjonalnych. Następnie przetestowano modele oparte na kilku algorytmach uczenia maszynowego i wybrano ostatecznie klasyfikator ekstremalnych drzew losowych charakteryzujący się najwyższą skutecznością.

Ostatnią częścią przygotowaną w ramach niniejszej pracy była implementacja głównego elementu, którym był moduł programistyczny komunikujący się z wybranymi urządzeniami i określający stan emocjonalny użytkownika. Równocześnie została także stworzona gra komputerowa wykorzystująca interfejs do obsługi zmian mechanik rozgrywki w zależności od zachowań i emocji gracza.

Stworzona gra została następnie wykorzystana w celu ewaluacji zaimplementowanego rozwiązania. Głównymi kryteriami oceny były logi emocji określonych przez interfejs oraz kwestionariusze wypełnione przez uczestników badania po jego zakończeniu. Analizując wyniki dotyczące zauważenia zmian w rozgrywce w wersji gry zawierającej interfejs do określania stanu emocjonalnego gracza, wygody użytkowania części sprzętowej oraz satysfakcji osób badanych z rozgrywki można stwierdzić, że moduł stworzony w ramach niniejszej pracy magisterskiej realizuje główny cel projektu. Taka ocena może wskazywać także na to, że wprowadzenie mechanizmów opartych o emocje na rynek komercyjny gier komputerowych ma nadzieję spotkać się z pozytywnym odzewem branży oraz graczy.

8.2. Propozycje przyszłych prac

Choć stworzony interfejs spełnia wymagania określone w celach niniejszej pracy magisterskiej, możliwe jest rozszerzenie poszczególnych jego elementów. Można rozważyć dodanie nowych urządzeń, pozwalających na pobieranie innych sygnałów fizjologicznych, takich jak reakcja elektrodermalna, czy pomiary zwracane przez elektroencefalograf. Ponieważ struktura interfejsu jest modułowa, rozwiązanie to wymagałoby przygotowania skryptów do komunikacji i interpretacji nowych sygnałów.

Elementem wartym dopracowania jest także model rozpoznający emocje. Poprzez wprowadzenie nowych sygnałów możliwe byłoby przygotowanie zbioru o wyższej skuteczności. Jednocześnie rozwiązaniem wartym uwagi jest wykorzystanie sieci neuronowej zamiast klasyfikatorów. Możliwe, że taka zmiana podejścia podczas projektowania modelu do rozpoznawania emocji pozwoli uzyskać rezultaty znacznie wyższe od uzyskanych w ramach niniejszej pracy magisterskiej. Jednocześnie w ramach modelu mogłyby też zostać uwzględnione odczyty bazowe zbierane w trakcie fazy kalibracji. Pozwoliłoby to na porównanie sygnałów fizjologicznych zbieranych w czasie gry z wartościami spoczynkowymi.

Ponadto, przygotowany moduł zawiera także elementy wymagające poprawy. Poza wspomnianym wcześniej modelem do rozpoznawania emocji należałoby zabezpieczyć także moment weryfikacji zwracanych emocji przez odczyty z akcelerometru w taki sposób, aby nie obniżał nadmiernie wartości *arousal* w przypadku gdy użytkownik nie porusza kontrolerem w czasie gry.

A. Dodatek: Lista pytań i odpowiedzi z kwestionariusza po zakończeniu eksperymentu

1. **Oceń poziom trudności gry w pierwszej wersji gry:**
   (a) Bardzo łatwa
   (b) Łatwa
   (c) Średnia
   (d) Trudna
   (e) Bardzo trudna
2. **Oceń poziom trudności gry w drugiej wersji gry**:
   (a) Bardzo łatwa
   (b) Łatwa
   (c) Średnia
   (d) Trudna
   (e) Bardzo trudna
3. **Jak wygodne było stanowisko pomiarowe?**
   (a) Niewygodne
   (b) Trochę niewygodne
   (c) Wygodne
   (d) Bardzo wygodne
4. **Gdybyś miał/a możliwość zamiany opaski na klatce na 3 przyklejane elektrody do odczytu tętna, co byś wybrał/a?**
   (a) Elektrody
   (b) Opaska
   (c) Bez różnicy

5. **Czy przeszkadzały ci elektrody przyklejone do przedramienia i wpięte do nich kable?**

   (a) Nie

   (b) Raczej nie

   (c) Trudno powiedzieć

   (d) Raczej tak

   (e) Tak

6. **Które z poniższych elementów zauważyłeś w drugiej wersji gry?**

   (a) Szybsze pojawianie się przeciwników

   (b) Pojawienie się trudniejszych przeciwników

   (c) Nagłe uruchomienie mocy specjalnej (niezależne od faktycznego jej włączenia)

   (d) Uruchomienie mocy specjalnej przy pomocy napięcia mięśni

   (e) Nagłe zwiększenie ilości życia (mimo niepodniesienia leczenia)

   (f) Nagłe zmniejszenie punktów życia

7. **Który sposób działania gry był dla Ciebie bardziej przyjemny/satysfakcjonujący?**

   (a) Uruchomienie "Mocy specjalnejźa pomocą przycisku na padzie

   (b) Uruchomienie "Mocy specjalnejźa pomocą napięcia mięśni przedramienia

   (c) Bez różnicy

8. **Który sposób działania gry był dla Ciebie bardziej przyjemny/satysfakcjonujący?**

   (a) Uruchomienie "Mocy specjalnejźa pomocą przycisku na padzie

   (b) Uruchomienie "Mocy specjalnejźa pomocą napięcia mięśni przedramienia

   (c) Bez różnicy

9. **Który sposób działania gry był dla Ciebie bardziej przyjemny/satysfakcjonujący?**

   (a) Pojawienie się trudniejszych przeciwników zależne od szybkości zdobywania punktów

   (b) Pojawienie się trudniejszych przeciwników w zależności od tętna i Twojego zachowania podczas gry

   (c) Bez różnicy

10. **Który sposób działania gry był dla Ciebie bardziej przyjemny/satysfakcjonujący?**

    (a) Zmiana liczby punktów życia zależna wyłącznie od trafienia przeciwników i przedmiotów leczących

(b) Dodatkowa zmiana liczby punktów życia zależna od tętna i Twojego zachowania podczas gry

(c) Bez różnicy

11. **Czy gdyby wykorzystać taką platformę do pomiaru emocji w komercyjnych grach, byłbyś bardziej skłonny/a do grania w nie?**

(a) Nie

(b) Raczej nie

(c) Trudno powiedzieć

(d) Raczej tak

(e) Tak

12. **Które z elementów stanowiska chciałbyś zobaczyć w użyciu w komercyjnych grach do modyfikacji mechanik w nich zawartych?**

(a) Opaskę do pomiaru tętna (uzależnienie mechanik od tętna)

(b) Urządzenie do pomiaru ruchu mięśni (uzależnienie mechanik od ruchów mięśni)

(c) Opaska + odczyty z pada (uzależnienie mechanik od odczytanych emocji)

13. **Dodatkowe uwagi:**

Bibliografia

[1] Carlos A. Gomez-Uribe i Neil Hunt. „The Netflix Recommender System: Algorithms, Business Value, and Innovation". W: *ACM Trans. Manage. Inf. Syst.* 6.4 (grud. 2015), 13:1–13:19. ISSN: 2158-656X. DOI: *10.1145/2843948*.

[2] Oliver Amft. „How Wearable Computing Is Shaping Digital Health". W: *IEEE Pervasive Computing* 17.1 (2018), s. 92–98. ISSN: 1536-1268. DOI: *10.1109/MPRV.2018.011591067*.

[3] Murat Dikmen i Catherine Burns. „Trust in autonomous vehicles: The case of Tesla Autopilot and Summon". W: *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2017, s. 1093–1098. DOI: *10.1109/SMC.2017.8122757*.

[4] Sebastian Oberdörfer i Marc Erich Latoschik. „Develop your strengths by gaming: towards an inventory of gamificationable skills". W: *INFORMATIK 2013 – Informatik angepasst an Mensch, Organisation und Umwelt*. Red. Matthias Horbach. Bonn: Gesellschaft für Informatik e.V., 2013, s. 2346–2357.

[5] David Michael i Sandra Chen. „Serious Games: Games That Educate, Train, and Inform". W: (sty. 2006).

[6] Rosalind W. Picard. *Affective Computing*. Cambridge, MA, USA: MIT Press, 1997. ISBN: 0-262-16170-2.

[7] Gartner Inc. *Hype Cycle for Emerging Technologies, 2018*. Spraw. tech. 2018.

[8] Irene Kotsia, Stefanos Zafeiriou i Spiros Fotopoulos. „Affective Gaming: A Comprehensive Survey". W: *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2013, s. 663–670. DOI: *10.1109/CVPRW.2013.100*.

[9] Kristina Höök. „Affective Loop Experiences - What Are They?" W: czer. 2008, s. 1–12. DOI: *10.1007/978-3-540-68504-3_1*.

[10] Jianhua Tao i Tieniu Tan. „Affective Computing: A Review". W: paź. 2005, s. 981–995. DOI: *10.1007/11573548_125*.

[11] Ashley E. Coleman i John Snarey. „James-Lange Theory of Emotion". W: *Encyclopedia of Child Behavior and Development*. Red. Sam Goldstein i Jack A. Naglieri. Boston, MA: Springer US, 2011, s. 844–846. ISBN: 978-0-387-79061-9. DOI: *10.1007/978-0-387-79061-9_3146*.

[12] Walter B. Cannon. „The James-Lange Theory of Emotions: A Critical Examination and an Alternative Theory". W: *The American Journal of Psychology* 39.1/4 (1927), s. 106–124. ISSN: 00029556.

[13] Jesse Prinz. „Emotions Embodied". W: *Thinking About Feeling: Contemporary Philosophers on Emotions*. Red. Robert Solomon. Oxford University Press, 2004.

[14] Gerald W. Hohmann. „Some effects of spinal cord lesions on experienced emotional feelings." W: *Psychophysiology* 3 2 (1966), s. 143–56.

[15] SREEJA P S i Mahalakshmi G S. „Emotion Models: A Review". W: *International Journal of Control Theory and Applications* 10 (sty. 2017), s. 651–657.

[16] Keith Oatley i P. N. Johnson-laird. „Towards A Cognitive Theory of Emotions". W: *Cognition and Emotion* 1 (mar. 1987), s. 29–50. DOI: *10.1080/02699938708408362*.

[17] Paul Ekman i Wallace V. Friesen. „Constants across cultures in the face and emotion." W: *Journal of personality and social psychology* 17 2 (1971), s. 124–9.

[18] James Russell i Lisa Barrett. „Core affect, prototypical emotional episodes, and other things called emotion: Dissecting the elephant". W: *Journal of personality and social psychology* 76 (czer. 1999), s. 805–19. DOI: *10.1037//0022-3514.76.5.805*.

[19] David Watson i Auke Tellegen. „Toward a consensual structure of mood." W: *Psychological Bulletin* 98.2 (1985), s. 219–235. DOI: *10.1037/0033-2909.98.2.219*.

[20] James Russell. „A Circumplex Model of Affect". W: *Journal of Personality and Social Psychology* 39 (grud. 1980), s. 1161–1178. DOI: *10.1037/h0077714*.

[21] James Russell, Anna Weiss i G. Mendelsohn. „Affect Grid: A Single-Item Scale of Pleasure and Arousal". W: *Journal of Personality and Social Psychology* 57 (wrz. 1989), s. 493–502. DOI: *10.1037/0022-3514.57.3.493*.

[22] Sander Koelstra i in. „DEAP: A Database for Emotion Analysis Using Physiological Signals". W: *IEEE Transactions on Affective Computing* 3 (grud. 2011), s. 18–31. DOI: *10.1109/T-AFFC.2011.15*.

[23] Juan Miranda i in. „AMIGOS: A dataset for Mood, personality and affect research on Individuals and GrOupS". W: *IEEE Transactions on Affective Computing* PP (lut. 2017). DOI: *10.1109/TAFFC.2018.2884461*.

[24] Grzegorz Nalepa i in. „Analysis and Use of the Emotional Context with Wearable Devices for Games and Intelligent Assistants". W: *Sensors* 19 (maj 2019), s. 2509. DOI: *10.3390/s19112509*.

[25] Joris H. Janssen, Egon L. van den Broek i Joyce H. D. M. Westerink. „Tune in to your emotions: a robust personalized affective music player". W: *User Modeling and User-Adapted Interaction* 22.3 (2012), s. 255–279. DOI: *10.1007/s11257-011-9107-7*.

[26] Katarzyna Chrzanowska. *Mechanizm rekomendacji muzyki bazujący na wybranych charakterystykach emocjonalnych*. Praca dyplomowa inżynierska. Kraków, 2018.

[27] Charlene Jennett i in. „Measuring and Defining the Experience of the Immersion in Games". W: *International Journal of Human-Computer Studies* 66 (wrz. 2008), s. 641–661. DOI: *10.1016/j.ijhcs.2008.04.004*.

[28] Eva Hudlicka. „Affective computing for game design". W: *4th International North-American Conference on Intelligent Games and Simulation, Game-On 'NA 2008* (sty. 2008), s. 5–12.

[29] Chara Papoutsi i Athanasios Drigas. „Games for Empathy for Social Impact". W: *International Journal of Engineering Pedagogy (iJEP)* 6 (list. 2016), s. 36. DOI: *10.3991/ijep.v6i4.6064*.

[30] *The Walking Dead: Season Two*. Dostęp: 2018-08-20. Telltale Games, 2013.

[31] Grzegorz Nalepa i in. „Affective Design Patterns in Computer Games. Scrollrunner Case Study". W: wrz. 2017, s. 345–352. DOI: *10.15439/2017F192*.

[32] Paweł Jemioło, Barbara Giżycka i Grzegorz Nalepa. „Prototypes of Arcade Games Enabling Affective Interaction". W: maj 2019, s. 553–563. ISBN: 978-3-030-20914-8. DOI: *10.1007/978-3-030-20915-5_49*.

[33] Ashley Reed. *Horror game Nevermind uses biometrics to sense your fear. https://www.gamesradar.com/horror-game-nevermind-uses-biometrics-sense-your-fear*. Dostęp: 2018-08-20. GamesRadar.

[34] Red Meat Games.

[35] *Neurobit Optima User Manual*. Neurobit. 2010.

[36] Krzysztof Kutt i in. „Towards the Development of Sensor Platform for Processing Physiological Data from Wearable Sensors". W: czer. 2018, s. 168–178. ISBN: 978-3-319-91261-5. DOI: *10.1007/978-3-319-91262-2_16*.

[37] *Xiaomi Mi Band 3 Manual*. Xiaomi. 2018.

[38] *Microsoft Band Fact Sheet*. Microsoft. 2016.

[39] *Empatica E4 Manual*. Empatica. 2014.

[40] Robert Wang i in. „Accuracy of Wrist-Worn Heart Rate Monitors". W: *JAMA Cardiology* 2 (paź. 2016). DOI: *10.1001/jamacardio.2016.3340*.

[41] *Polar H10 Manual*. Polar. 2019.

[42] *Garmin-HRM Manual*. Garmin. 2015.

[43] *BITalino (r)evolution kit Documentation*. BITalino. 2018.

[44] Jonathan Sykes i Simon Brown. „Affective gaming: measuring emotion through the gamepad." W: sty. 2003, s. 732–733. DOI: *10.1145/765891.765957*.

[45] W: (2013). Dostęp: 2019-09-01.

[46] Peter Flach. *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012, s. 215,333. DOI: *10.1017/CBO9780511973000*.

[47] Philipp Moll i in. „A Network Traffic and Player Movement Model to Improve Networking for Competitive Online Games". W: czer. 2018, s. 1–6. DOI: *10.1109/NetGames.2018.8463390*.

[48] *Godot Engine latest documentation*. Godot. 2019.

[49] *Unity User Manual (2019.2)*. Unity. 2019.

[50] *Kerbal Your Enthusiasm. Kerbal Space Program by Squad*. *https://unity3d.com/showcase/case-stories/squad-kerbal-space-program*. Dostęp: 42019-09-01. Unity, wrz. 2013.

[51] *Unreal Engine 4 Documentation*. Epic Games. 2019.

[52] *Unreal Development Kit Documentation*. Epic Games. 2012.

[53] Mojtaba Khomami Abadi i in. „DECAF: MEG-based Multimodal Database for Decoding Affective Physiological Responses"". W: *IEEE Transactions on Affective Computing* PP (sty. 2015), s. 1. DOI: *10.1109/TAFFC.2015.2392932*.

[54] Ramanathan Subramanian i in. „ASCERTAIN: Emotion and Personality Recognition using Commercial Sensors". W: *IEEE Transactions on Affective Computing* PP (list. 2016), s. 1–1. DOI: *10.1109/TAFFC.2016.2625250*.

[55] Aleksandr Fedotov. „Selection of Parameters of Bandpass Filtering of the ECG Signal for Heart Rhythm Monitoring Systems". W: *Biomedical Engineering* 50 (wrz. 2016). DOI: *10.1007/s10527-016-9600-8*.

[56] Peter Welch. „The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms". W: *IEEE Transactions on Audio and Electroacoustics* 15.2 (1967), s. 70–73. DOI: *10.1109/TAU.1967.1161901*.

[57] Fredric Shaffer i Jp Ginsberg. „An Overview of Heart Rate Variability Metrics and Norms". W: *Frontiers in Public Health* 5 (wrz. 2017), s. 258. DOI: *10.3389/fpubh.2017.00258*.

[58] *BITalino (r)evolution kit APIs*. BITalino. 2019.

[59] *Android ANT+ SDK*. *https://www.thisisant.com/resources/android-ant-sdk/*. Dostęp: 2018-08-20.