

Dotykowy translator tekstu do Braille'a

(Text to tactile Braille output translator)

Agata Pokorska

Praca inżynierska

Promotor: dr Marek Materzok

Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Informatyki

5 luty 2026

Streszczenie

W tej pracy przedstawiam opis urządzenia i aplikacji, które w połączeniu umożliwiają zamianę dowolnego tekstu pozyskanego z obrazu na wyczuwalną dotykiem reprezentację kolejnych znaków w alfabecie Braille'a. Zapoznaję czytelnika z dostępnymi obecnie rozwiązaniami, ich zaletami i wadami. Porównuję z nimi swój pomysł i prezentuję szczegółową analizę rozwiązania i kosztów z nim związanych.

In this paper I introduce my idea for a refreshable Braille translator (display) device and an application that together enable reader to translate text from a photo into tactile Braille output. I also review currently available solutions, highlighting their pros and cons, and compare them with my own idea. In the end I present a detailed analysis of how it works and costs involved.

Spis treści

1. Wprowadzenie	7
1.1. Przedmiot pracy	7
1.2. Skąd taki pomysł	7
1.2.1. Co to jest alfabet Braille’a	7
1.2.2. Dostępne rozwiązania i problemy z nimi związane	7
1.2.3. Moja motywacja	8
2. Przegląd i analiza dostępnych rozwiązań	9
2.1. Rozwiązanie wykorzystujące zjawisko elektromagnetyzmu	9
2.2. Rozwiązania wykorzystujące zjawisko piezoelektryczności	10
2.3. Inne dostępne rozwiązania warte uwagi	10
3. Porównanie użytych w projekcie rozwiązań	13
3.1. Translator	13
3.2. Narzędzia użyte do stworzenia oprogramowania translatora	15
3.3. Narzędzia użyte do stworzenia aplikacji	15
3.4. Zalety i wady mojego rozwiązania	16
4. Instrukcja dla użytkowników	17
4.1. Jak używać translatora	17
4.2. Jak zainstalować aplikację	17
4.3. Jak korzystać z aplikacji	17
4.4. Przypadki użycia	18
4.4.1. Wykonanie zdjęcia za pomocą aplikacji	18
4.4.2. Przesłanie tekstu do translatora	18
4.4.3. Zakończenie połączenia z translatorem	18
5. Podsumowanie	19
5.1. Szacowany koszt wytworzenia prototypu	19

5.2. Finalny efekt	20
5.3. Potencjalne usprawnienia do projektu	20
Bibliografia	21

1. Wprowadzenie

1.1. Przedmiot pracy

Przedmiotem mojej pracy jest *translator*, czyli urządzenie odpowiadające za translację do Braille’a otrzymanego znaku tekstu i wyświetlenie jego Braille’owskiej reprezentacji, oraz aplikacja *TextToBraille* umożliwiająca interaktywne korzystanie z translatora.

Aplikacja zaprojektowana na smartfony z systemem Android, służy do wykonania zdjęcia konkretnego tekstu, który następnie jest przesyłany do translatora i na jego podstawie, generowane jest odwzorowanie znaków tekstu w alfabecie Braille’a, poprzez podbicie konkretnych pinów translatora. Mówiąc pin, mam na myśli jeden z sześciu punktów będący w stanie wypukłym lub płaskim.

Translator wraz z aplikacją dają osobom niewidomym możliwość odczytu *dowolnego* tekstu, który da się uchwycić aparatem smartfona.

1.2. Skąd taki pomysł

1.2.1. Co to jest alfabet Braille’a

Alfabet Braille’a to wariant alfabetu, który umożliwia osobom niewidomym odczytywanie tekstu, za pomocą zmysłu dotyku. Pojedynczy znak alfabetu Braille’a składa się z sześciu pinów, ułożonych w trzy wiersze po dwie kolumny. Każdy pin jest w stanie płaskim lub wypukłym. Wszystkich możliwych kombinacji ułożenia pinów jest sześćdziesiąt cztery. Odwzorowują one odpowiednio litery a-z, znaki interpunkcyjne, znak cyfry, znak wielkiej litery, nawiasy itp.

1.2.2. Dostępne rozwiązania i problemy z nimi związane

Najbardziej powszechnym rozwiązaniem dla osób niewidomych pozostają audiobooki lub inne rozwiązania wykorzystujące technologię *text-to-speech*, co jednak okazuje się średnim kompromisem. Ludzie przyswajają informacje na *różne sposoby*. Niektórzy preferują wzrokowo, inni słuchowo, jeszcze inni muszą się poruszać czy poczuć dany obiekt, aby zostały stworzone odpowiednie połączenia neuronalne w procesie uczenia się. W tym [1] badaniu zasugerowano, Warto więc zadbać, aby osoby niewidome też miały zróżnicowane możliwości przyswajania informacji.

Oprócz audiobooków istnieją też książki drukowane Braillem za pomocą specjalnie stworzonego do tego mechanizmu [2], który powstał już w roku 1892. Koszt takich książek jest jednak horrendalnie wysoki (nawet czterokrotnie wyższy w porównaniu do ich drukowanych tradycyjnie odpowiedników). Katalog popularnych książek drukowanych Braillem znajduje się w [3].

Problemem obu wyżej wymienionych rozwiązań jest to, iż nie wszystkie interesujące pozycje mają swoje odzwierciedlenie w postaci audiobooka/książki drukowanej Braillem (choćby podręczniki). Pragnę zauważyć też, że zakup pojedynczych audiobooków wiąże się z wysokim kosztem. Istnieją aplikacje oferujące modele subskrybcyjne (np. Audible, EmpikGo czy Legimi) jednak mają one po pierwsze: bardzo okrojoną ofertę, a po drugie: zasób znajduje się na serwerze aplikacji, zatem nawet po zakupie, tak naprawdę nigdy nie należy do nas.

Alternatywą dla tychże rozwiązań, pozostają elektroniczne wyświetlacze znaków Braille’a, które też mają swoją wadę – są *niezwykle* drogie. Jako istotne dla tego faktu czynniki wyróżniam:

- Moduł do wyświetlania pojedynczego znaku Braille’a nie jest typowym komponentem elektronicznym.
- Ze względu na złożoność konstrukcji koszt wyświetlenia pojedynczego znaku jest wysoki.
- Popyt na tego typu produkty jest stosunkowo niski.

Pomimo tego, elektroniczne wyświetlacze mają jednak istotną przewagę – pozwalają na wyświetlenie dowolnego tekstu, do którego mamy już dostęp, bez dodatkowych kosztów. Biorąc to pod uwagę, myślę że znacząco minimalizuje to koszt pojedynczego użycia w porównaniu do dwóch poprzednich rozwiązań.

1.2.3. Moja motywacja

Chciałabym też wspomnieć o osobistym doświadczeniu, które dalej pozostaje motorem do rozwoju tego projektu. Będąc w bibliotece w swoim rodzinnym mieście, zapytałam o zbiór pozycji dostępnych dla osób niewidomych. Na całą bibliotekę, była tylko jedna książka [4] napisana Braillem. Przyglądając się książce dłużej i z bliska, zauważyłam że pędziesiąt znaków Braille’a było zwyczajnie wytartych od użytkowania. Ta książka miała dopiero cztery lata, a już była zużyta na tyle, żeby utrudnić czytanie. Uwzględniając do tego tempo degradacji książek drukowanych tradycyjnie, nie brzmi to korzystnie o książkach drukowanych Braillem.

Celem tej pracy jest przedstawienie mojego pomysłu na implementację możliwie prostego i niskobudżetowego elektronicznego wyświetlacza Braille’a, używając tanio dostępnych komponentów oraz darmowych narzędzi.

2. Przegląd i analiza dostępnych rozwiązań

Dostępne rozwiązania podzielę na trzy główne kategorie wg. zastosowanych przez nie technologii tj. elektromagnetyczne, piezoelektryczne oraz inne, czyli takie, które różnią się swoim podejściem od klasycznych urządzeń z dwóch pierwszych kategorii.

2.1. Rozwiązanie wykorzystujące zjawisko elektromagnetyzmu

Przykładowe urządzenie wykorzystujące elektromagnetyzm wygląda następująco [5]. Upraszczając, elektromagnes przyciąga inny magnes lub pozostaje obojętny w zależności od obecności ładunku elektrycznego. We wspomnianym wcześniej przykładzie magnes został umieszczony w środku głowicy, która płynnym ruchem obrotowym (góra/dół) powoduje podbicie/spadek jednego z sześciu pinów znaku Braille’a. Działa to na podobnej zasadzie co ekran czytników E-ink [6] (np. Kindle)

Zalety i wady zastosowania elektromagnetyzmu

Powyższe rozwiązanie wygląda dość obiecująco. Z zalet, mogę wymienić otwartoźródłowość oraz to, że szacowany koszt produkcji takiego komponentu jest niski. Widzę w nim jednak kilka wad.

Oto cytat bezpośrednio z logów wymienionego wyżej projektu [7]: *"Having access to a printer directly made all the difference when iterating for 20-30 μ m tolerances. At these sizes, you can't really trust the CAD software anymore, and every assembly has to be empirically tested by printing it out."* Autor odnosi się tutaj do trudności w wytworzeniu działającego pojedynczego modułu, ze względu na jego bardzo mały rozmiar i twierdzi, że każdy komponent musi zostać złożony a następnie sprawdzony pod kątem poprawności działania. Inny problem jaki dostrzegam, to szybkie nagrzewanie się elektromagnesu [8]. Wymaga to kolejnych części do odprowadzania ciepła, co jeszcze bardziej komplikuje już i tak gęsto zwartą konstrukcję. Kolejny problem jaki widzę, związany jest z tarciem wywoływanym przez ruch głowicy, co może powodować stopniowe zużywanie się układu. Przy takich rozmiarach komponentów, wszelkie drobne różnice w strukturze, kurz, czy innego rodzaju zanieczyszczenia mogą powodować zakłócenia w prawidłowej pracy urządzenia.

2.2. Rozwiązania wykorzystujące zjawisko piezoelektryczności

Przechodząc do następnej kategorii rozwiązań, wykorzystujących piezoelektryki, należy wspomnieć na czym to polega. Jest to właściwość konkretnych materiałów (np. kwarcu), która sprawia, że w momencie oddziaływania siły na ten materiał, powstaje energia elektryczna. Czytniki Braille’a używają dokładnie *odwrotnej zasady*, czyli poprzez zastosowania odpowiedniego prądu, materiał zaczyna odkształcać się w mniej lub bardziej kontrolowany sposób. Na rynku dostępne są tzw. siłowniki piezoelektryczne [9] jak np. [10], który nawiasem mówiąc przez ostatnie 2 lata, podrożał o ponad 80 USD. Ewentualnie bardziej budżetowa opcja [11]. Nadal jednak koszt pojedynczych komponentów jest bardzo wysoki. Do wyświetlenia *pojedynczego* znaku Braille’a, potrzebujemy aż sześć tego rodzaju siłowników. W filmie [12] autor rozkłada na części popularny komercyjny wyświetlacz Braille Lite 2000 [13].

Zalety i wady zastosowania piezoelektryków

Główną wadą związaną z użyciem tej technologii w wyświetlaczach jest ich cena. Relatywnie tanią opcją używającą piezoelektryków jest wyświetlacz [14], który kosztuje zawrotne 800 USD. Inną wadą jest potrzeba prawidłowej konserwacji takiego urządzenia, ze względu na ilość otworów, w które może dostać się brud, co może zaburzyć mechanikę podbijania pinu, a nawet uszkodzić wyświetlacz.

Zaletą jest ich zgrabny wygląd i niezbyt duży rozmiar.

2.3. Inne dostępne rozwiązania warte uwagi

Jednym z bardziej kompaktowych rozwiązań jest BrailleRing [15], który również pozwala na odczytywanie linii tekstu, jednak robi to w sposób zupełnie inny. Urządzenie to przyjmuje kształt pierścienia a wewnątrz ma wbudowane dwadzieścia modułów. Moduł składa się z sześciu pinów i służy do wyświetlenia pojedynczego znaku Braille’a. Wygląda i działa jak kostka Rubika, ale taka 3x2 a nie klasyczna 3x3. Sześć pinów w module zostało podzielone na trzy poziome grupy, każda po dwa piny (dwa górne, dwa środkowe i dwa dolne). Para pinów ma cztery możliwe stany: oba płaskie, jeden wypukły lub oba wypukłe. Każda taka kombinacja pary pinów jest odwzorowana na jednej ze ścianek „kostki”, która to ma możliwość obrotu o 90 stopni. Poprzez wybór odpowiedniej grupy i rotacje, możemy za pomocą jednego modułu przedstawić dowolny znak Braille’a.

Pomysł jest o tyle innowacyjny, że właśnie dzięki ułożeniu modułów w pierścień, w jego górnej połowie moduły najpierw są obracane do reprezentacji odpowiednich znaków, a następnie użytkownik poruszając palcem po wnętrzu dolnej połowy pierścienia, sam nadaje tempo czytania podsuwając sobie kolejne znaki linii. Taka architektura znacznie ogranicza liczbę potrzebnych siłowników do dziewięciu. Jeden siłownik jest w stanie obrócić jeden wiersz kostki, z czego do zmiany reprezentacji na inny znak potrzebne są maksymalnie trzy obroty dla każdego wiersza.

Jeszcze inny pomysł, wywodzi się z popularnego serwisu z otwartoźródłowymi projektami do druku 3D, Thingiverse [16]. Projekt ten wykorzystuje silniki do poruszania długą zębatkową przekładnią, która obsługuje trzy z sześciu pinów (jedną kolumnę) i pozycja tejże przekładni (czyli to jak bardzo jest przesunięta w lewo lub w prawo) odwzorowuje jedną z ośmiu kombinacji tych trzech pinów. Rozwiązanie różni się od pozostałych, gdyż działa w pełni mechanicznie. Potencjalne problemy jakie tu widzę, to fakt iż drukarka 3D ma swoje ograniczenia, jeśli chodzi o precyzję w tak małej skali oraz to, że w pełni mechaniczny moduł jest podatny na nadmierne tarcie części o siebie i wynikające z tego powodu uszkodzenia.

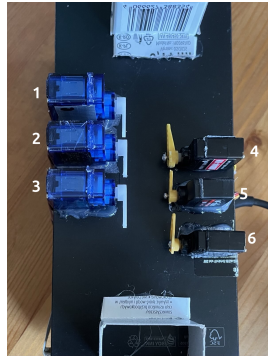
3. Porównanie użytych w projekcie rozwiązań

UWAGA: Serwo - potoczna nazwa na serwomechanizm, w tym przypadku wprawiający w ruch orczyk, którego stopień wychylenia decyduje o położeniu pinu.

3.1. Translator

Na urządzenie składa się zaledwie kilka komponentów:

- ESP32 WiFi+BT 4.2 WROOM-32 (mózg układu),
- trzy mikroserwa 3.3V oraz trzy mikroserwa 5V (to podbija piny),
- sześć walców o 20mm x 6mm wydrukowanych w 3D z plastiku (to są właśnie piny, one są bezpośrednio wyczuwalne palcami),
- uniwersalna płytką PCB 5cm x 7cm dwustronna (tu jest zamocowana cała elektronika),
- trzy konwertery poziomów logicznych napięć 2CH 3.3V/5V (gdyż ESP32 działa na 3.3V a serwa potrzebują wyższego napięcia),
- kabel USB do USB-C (potrzebny do zasilania układu),
- akumulator Li-ion Samsung 3.6V; 5000mAh; 25A,
- koszyk na akumulator,
- kondensator $1000\mu F$,
- sterownik ładowania baterii,
- konwerter step-up ustawiony na 5V.



Rysunek 3.1: Sześć serw podbijających piny o odpowiadających numerach

Przebieg komunikacji między użytkownikiem a aplikacją

- Następuje wybór tekstu do sfotografowania. Najlepiej taki kontrastujący z tłem np. książka.
- Po kliknięciu guzika „Photo” zostaje uruchomiony launcher aparatu.
- Model [17] AI przechwytuje tekst z wykonanego zdjęcia.
- Rozpoznany tekst zostaje wyświetlony w dedykowanym do tego polu aplikacji.
- Tekst ten może zostać wyświetlony na translatorze poprzez kliknięcie guzika „Send”.
- Tekst może być nadpisany, poprzez powtórzenie powyższych kroków.

Przebieg komunikacji od aplikacji, do translatora

- Tworzy się kanał komunikacji między ESP32 a aplikacją na smartfonie za pośrednictwem Bluetooth.
- Kolejne znaki tekstu są strumieniowo przesyłane do mikrokontrolera.
- Mikrokontroler zleca podbicie konkretnych pinów, poprzez zmianę pozycji odpowiednich serw.
- Stan serw jest podtrzymywany przez sekundę, a następnie wszystkie piny opadną.
- Po każdym wyświetleniu pojedynczego znaku następuje przerwa o długości 300 ms.

Oprogramowanie translatora zostało zaimplementowane w języku C++. Cała komunikacja translatora z aplikacją odbywa się za pomocą technologii Bluetooth 4.2.

Szczegóły implementacji znajdują się w moim repozytorium poświęconym rozwojowi tego projektu [18]

3.2. Narzędzia użyte do stworzenia oprogramowania translatora

- VS Code – bardzo wygodny edytor kodu, który wybrałam ze względu na wieloplatformowość i sporo dostępnych pluginów (m.in PlatformIO).
- PlatformIO – zintegrowane środowisko do kompilacji, wgrywania kodu źródłowego na ESP32.
- C++ – użyty ze względu na szeroką kompatybilność z interfejsem Arduino do programowania mikrokontrolerów.
- `Arduino.h` – używana do programowania ESP32 w prosty sposób.
- `BluetoothSerial.h` – używana do postawienia kanału i dalszej komunikacji mikrokontrolera z aplikacją.
- `ESP32Servo.h` – daje prosty interfejs do kontroli pozycji serwa.

3.3. Narzędzia użyte do stworzenia aplikacji

- Android Studio – najlepsze znane mi środowisko do wytwarzania aplikacji na Androida. Wybrałam też dlatego, że daje możliwość emulowania i podglądu interfejsu graficznego.
- Kotlin – powszechnie rekomendowany język programowania androidowych aplikacji, wspierany przez Android Studio.
- Google ML Kit Text recognition v2 – użyty do rozpoznawania tekstu ze zdjęcia. Główną zaletą jest to, że działa bez dostępu do internetu.

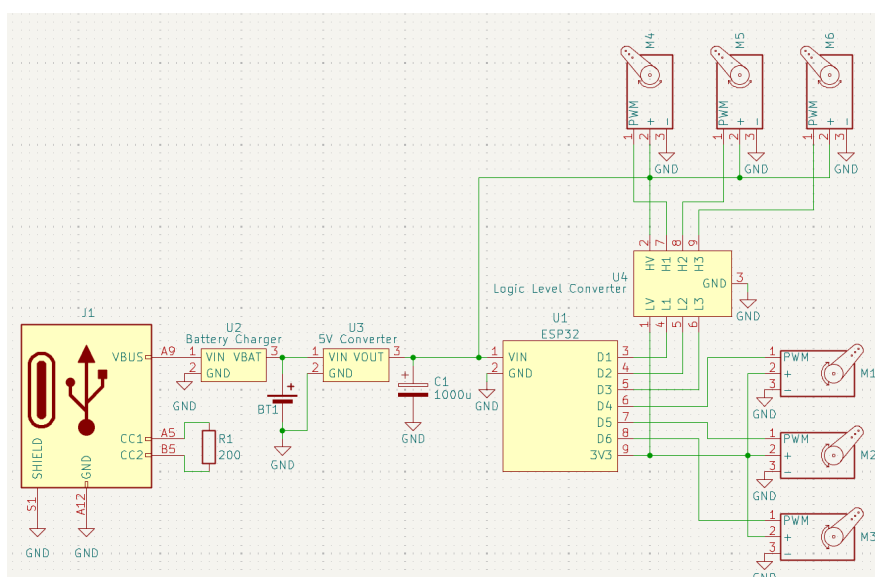
3.4. Zalety i wady mojego rozwiązania

Po omówieniu mojego rozwiązania, jak i tych z poprzedniego rozdziału, przedstawię jego zalety i wady. Zalety wyglądają następująco:

- Prostota konstrukcji – na pojedynczy pin przypada jeden serwomechanizm.
- Użycie gotowych, dostępnych na rynku tanich części elektronicznych.
- Relatywnie niski koszt produkcji w porównaniu do pozostałych rozwiązań.
- Piny są lekkie, plastikowe, ważą mniej niż 1 gram zatem serwo praktycznie nie traci energii przy podtrzymywaniu stanu pinu.
- Wszystkie piny są synchronizowane jednocześnie.
- Dedykowana aplikacja, łącząca się z translatorem przy użyciu Bluetooth (tak jak w przypadku wspomnianych wcześniej urządzeń).
- Ze względu na duże gabaryty translatora, jest on odporny na kurz i pył.

A teraz wady, które są dość istotne w odniesieniu do pomysłu na projekt:

- Na ten moment urządzenie wyświetla pojedyncze znaki tekstu po kolej.
- Znak jest wyświetlany statycznie – co jak wykazano w [19] nie jest optymalnym sposobem wyświetlania tekstu.
- Translator jest dość dużych rozmiarów jak na jeden znak.



Rysunek 3.2: Uproszony schemat elektroniczny translatora

4. Instrukcja dla użytkowników

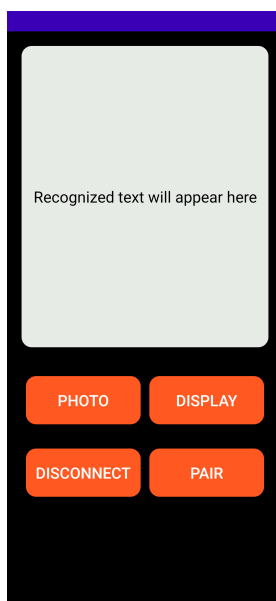
4.1. Jak używać translatora

Translator w pierwszej kolejności należy podłączyć do gniazda zasilania USB. Po podłączeniu translator jest w pełni gotowy, aby połączyć się ze smartfonem za pomocą aplikacji. *Aby translator działał poprawnie, nie należy stosować dużej siły podczas odczytywania pinów znaku Braille’a oraz odłączać go od źródła zasilania w trakcie działania*

4.2. Jak zainstalować aplikację

Aplikację można zainstalować pobierając plik *texttobracille.apk* z mojego repozytorium [18] a następnie instalując go na urządzeniu. Aby poprawnie zainstalować plik .apk na smartfonie z Androidem, należy wcześniej *zezwolić na instalowanie aplikacji z zewnętrznych źródeł*. Instrukcja jak wykonać to poprawnie znajduje się w artykule [20].

4.3. Jak korzystać z aplikacji



Rysunek 4.1: Zgodny z wytycznymi nt. dostępności interfejs aplikacji

Po otwarciu aplikacji *TextToBraille* naszym oczom ukazuje się biały obszar na zeskanowany tekst oraz kilka guzików o funkcjonalnościach opisanych poniżej:

Pair – Nawiązuje połączenie między aplikacją a translatorem. Następnie wyświetli się komunikat informujący o stanie połączenia (aktywne/nieaktywne)

Disconnect – Przerywa istniejące połączenie Bluetooth z translatorem

Photo – Uruchamia systemową aplikację do robienia zdjęć

Display – Zleca translatorowi wyświetlenie widocznego w białym obszarze tekstu. Jeśli nie ma tekstu do wyświetlenia, pokaże się informujący o tym komunikat.

4.4. Przypadki użycia

4.4.1. Wykonanie zdjęcia za pomocą aplikacji

Aktor: Osoba niewidoma posiadająca smartfona z Androidem i zainstalowaną aplikacją *TextToBraille*.

Przypadek użycia: Osoba naciska guzik *Photo* co otwiera systemową aplikację do robienia zdjęć. Następnie wykonuje zdjęcie tekstu, który chce przeczytać. Osoba potwierdza chęć przetworzenia tego zdjęcia, poprzez kliknięcie *akceptuj*. Zdjęcie zostaje przetworzone na tekst, który zostaje wyświetlony w aplikacji.

4.4.2. Przesłanie tekstu do translatora

Aktor: Osoba niewidoma posiadająca już tekst w aplikacji *TextToBraille*.

Przypadek użycia: Osoba naciska guzik *Connect*, co inicjuje połączenie z translatorem. Następnie klika guzik *Send*, po czym tekst z aplikacji zostaje przesłany do translatora. Translator natychmiast zaczyna wyświetlać litery tekstu w alfabecie Braille’a.

4.4.3. Zakończenie połączenia z translatorem

Aktor: Osoba niewidoma z uruchomioną aplikacją *TextToBraille* i połączonym po Bluetooth translatorem.

Przypadek użycia: Osoba naciska guzik *Disconnect*, a to powoduje zakończenie połączenia po stronie smartfona. Do chwili ponownego nawiązania połączenia, nie ma możliwości przesłania zeskanowanego tekstu.

5. Podsumowanie

5.1. Szacowany koszt wytworzenia prototypu

W poniższym wyliczeniu skupię się na cenie każdego komponentu za sztukę, następnie pomnożę razy ich użytą ilość. Podana cena komponentu jest najniższą spośród znalezionych ze stron popularnych sprzedawców internetowych tj. Botland, Kamami, Nettigo, Allegro oraz TME. Koszt dostawy nie został uwzględniony. Wiele z tych części jest już gotowymi modułami, składającymi się z mniejszych części elektronicznych już scalonych na płytkach.

- ESP32 WiFi+BT 4.2 WROOM-32 – **22.49 PLN**.
- Trzy mikroserwa 3.3V – **14.90 PLN x 3**.
- Trzy mikroserwa 5V – **17.90 PLN x 3**.
- Sześć walców o 20mm x 6mm – **0.70 PLN**.
- Uniwersalna płytka PCB – **3 PLN**.
- Trzy konwertery poziomów logicznych napięć – **2.50 PLN x 3**.
- Kabel USB-C – **3.47 PLN**.
- Akumulator Li-ion – **13.23 PLN**.
- Koszyk na akumulator – **2.89 PLN**.
- Kondensator $1000\mu F$ – **0.48 PLN**.
- Sterownik ładowania baterii – **2.28 PLN**.
- Konwerter step-up – **3.89 PLN**.

Sumarycznie: **158.33 PLN** a więc zdecydowanie mniej koszt opisanego wcześniej komercyjnego czytnika za 800 USD

5.2. Finalny efekt

Jako efekt końcowy swojej pracy, uzyskałam w pełni działające urządzenie pozwalające na translację dowolnego tekstu do postaci Braille’owskiej, oraz spełniającą swoje zadanie, dedykowaną aplikację. Translator jak i mechanizm podbijania pinów zostały skonstruowane w możliwie najprostszy sposób z tanich, łatwodostępnych komponentów elektronicznych. Aplikacja oraz kod mikrokontrolera zostały stworzone za pomocą darmowych narzędzi deweloperskich. Biorąc pod uwagę cenę części elektronicznych jak i materiałów użytych do złożenia urządzenia, ostateczna cena wynosi około 160 PLN.

5.3. Potencjalne usprawnienia do projektu

- Stworzenie odpowiednika aplikacji na IOS.
- Modyfikacja rozmiaru i działania konstrukcji.
- Otwartoźródłowy model do rozpoznawania tekstu zamiast od Google.
- Zastosowanie bardziej zaawansowanego mechanizmu dźwigni, aby zmniejszyć odstęp między pinami

Bibliografia

- [1] Daniel Goldreich and Ingrid M. Kanics. “Tactile Acuity is Enhanced in Blindness”. In: *Journal of Neuroscience* 23.8 (2003), pp. 3439–3445. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.23-08-03439.2003. eprint: <https://www.jneurosci.org/content/23/8/3439.full.pdf>. URL: <https://www.jneurosci.org/content/23/8/3439>.
- [2] Wikipedia contributors. *Perkins Braille* — *Wikipedia, The Free Encyclopedia*. [Online; accessed 7-January-2026]. 2025. URL: https://en.wikipedia.org/w/index.php?title=Perkins_Braille&oldid=1321563538.
- [3] *Catalog of bestselling braille-printed books*. <http://www.braillebookstore.com/Adult-Bestsellers>.
- [4] *Czarna książka kolorów - Menena Cottin*. https://wydawnictwo-widnokrag.pl/ksiazki/czarna_ksiazka_kolorow/. 2012.
- [5] *Refreshable Braille Display with electromagnets usage*. <https://hackaday.io/project/191181-electromechanical-refreshable-braille-module>. 2023.
- [6] *Jak działa e-papier i co warto o nim wiedzieć*. <https://forbot.pl/blog/e-papier-jak-dziala-i-co-warto-o-nim-wiedziec-id53934>.
- [7] *12th time is the charm - A project log for Electromechanical Refreshable Braille Module*. <https://hackaday.io/project/191181-electromechanical-refreshable-braille-module/log/224233-12th-time-is-the-charm>. 2023.
- [8] *Why do electromagnets generate overheating during using - causes and solutions*. <https://www.dr-solenoid.com/news/why-do-electromagnets-generate-over-heating-during-using-causes-and-solutions/>. 2024.
- [9] Frédéric Giraud and Christophe Giraud-Audine. “Chapter One - Introduction”. In: *Piezoelectric Actuators: Vector Control Method*. Ed. by Frédéric Giraud and Christophe Giraud-Audine. Butterworth-Heinemann, 2019, pp. 1–42. ISBN: 978-0-12-814186-1. DOI: <https://doi.org/10.1016/B978-0-12-814186-1.00005-3>. URL: <https://www.sciencedirect.com/science/article/pii/B9780128141861000053>.
- [10] *Piezoelectric bending transducer expensive solution*. <https://piezo.com/products/piezoelectric-bending-transducer-s234-h5fr-1803xb>.
- [11] *Piezoelectric bending transducer cheaper solution*. <https://www.thorlabs.com/item/PB4VB2W>.
- [12] *Braillelite2000 device disassembly demonstration*. https://www.youtube.com/watch?v=guZ2rFT_FKQ. 2020.

- [13] *BrailleLite2000*. <https://www.eyeway.org.in/?q=braille-lite-2000>.
- [14] *Cheap commercial refreshable braille display*. <https://www.orbitresearch.com/product/orbit-reader-20/>.
- [15] Michael Trembl et al. In: *Current Directions in Biomedical Engineering* 6.2 (2020), p. 20202004. DOI: doi : 10 . 1515 / cdbme - 2020 - 2004. URL: <https://doi.org/10.1515/cdbme-2020-2004>.
- [16] *Fully mechanical concept for refreshable braille display module*. <https://www.thingiverse.com/thing:90144>. 2013.
- [17] *Google ML Kit Text recognition v2*. <https://developers.google.com/ml-kit/vision/text-recognition/v2>.
- [18] *My TextToBraille project repository*. <https://github.com/gagata7/TextToBraille>. 2025.
- [19] Joerg Fricke and Helmut Baehring. “Displaying laterally moving tactile information”. In: *Computers for Handicapped Persons*. Ed. by Wolfgang L. Zagler, Geoffrey Busby, and Roland R. Wagner. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 461–468. ISBN: 978-3-540-48989-4.
- [20] *Guide to Installing APK Files on Android Devices*. <https://www.lifewire.com/install-apk-on-android-4177185>. 2025.