Uniwersytet Rzeszowski

Wydział Nauk Ścisłych i Technicznych

Instytut Informatyki

BIOMETRYCZNE SYSTEMY ZABEZPIECZEŃ

INSTRUKCJA DO ĆWICZEŃ LABORATORYJNYCH

TREŚCI KSZTAŁCENIA: IDENTYFIKACJA DAKTYLOSKOPIJNA , ŚRODOWISKO MEGAMATCHER

Spis treści

Cele laboratorium

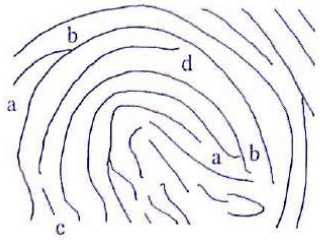
Celem laboratorium jest zapoznanie z technologiami identyfikacji daktyloskopijnej. W ramach zajęć omówiony będzie system Megamatcher do zbierania i porównywania odcisków palców. Zadania będą zawierać część doświadczalną, a także praktyczną, polegającą na implementacji prostej aplikacji do identyfikacji biometrycznej z użyciem środowiska Megamatcher SDK.

Wprowadzenie

Charakterystyka daktyloskopii

Daktyloskopia zajmuje się metodami ustalenia tożsamości (identyfikacją człowieka) na podstawie listewek skórnych (linii papilarnych) znajdujących się na wewnętrznych powierzchniach palców i dłoni. Widoczne odciski mogą być pozostawione przez substancję, które przyklejają się do palców takich jak: brud lub krew. Na opuszkach palców i dłoni znajduje się szereg grzbietów, bruzd, pętli, spiral i łuków, które tworzą odpowiednie wzory. Odciski zawierają również indywidualne cechy zwane – minucjami, które nie są widoczne gołym okiem.

Minucje są to szczególne cechy, które służą do porównywania i identyfikacji odcisków palca i ich zgodności. Minucje posiadają swoje cechy takie jak: zakończenia bruzd (a), rozgałęzienia bruzd (b), niezależna bruzda (c) (Rysunek 1).



Rysunek 1: Przykładowe minucje

Pozyskiwanie cech odcisku palca rozpoczyna się od sprawdzenia jakości obrazu wejściowego, obliczeniu kierunku przepływu bruzd, dopasowania parametrów filtrów w celu poprawienia obrazu i wykonania segmentacji bruzd. Następnie lokalizuje się cechy minucji.

Wyodrębnione cechy i informacje o ich rozmieszczeniu pozwalają na porównywanie odcisków palców, co jest niezbędne zarówno w kontekście identyfikacji osób, jak i weryfikacji tożsamości.

Rysunek 2: Zestawienie porównawcze dwóch pomiarów tego samego palca (analizę uzyskano za pomocą narzędzi Megamatcher SDK)

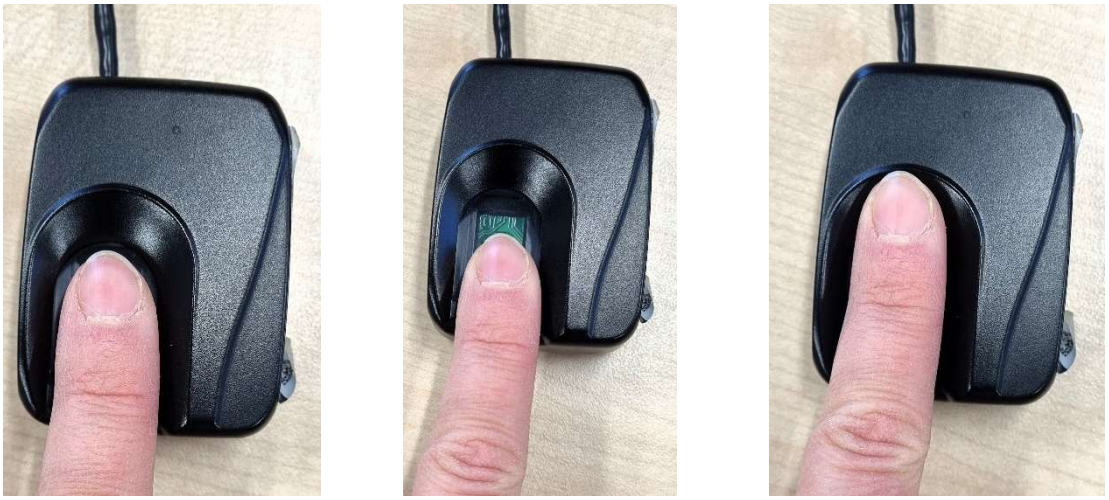
Skaner Futronic FS80H

W celu pobrania odcisków palców należy wykorzystać skaner Futronic FS80H (Rysunek 3). Jest to skaner optyczny obsługiwany przez program MegaMatcher, połączony z komputerem za pomocą kabla USB 2.0. Skanowanie odbywa się za pomocą przyłożenia palca do powierzchni płytki i przytrzymania przez kilka sekund do momentu pojawienia się skanowanego odcisku w programie.

Rysunek 3: Skaner Futronic FS80H

Wykonując pomiary należy zwrócić uwagę na kilka kwestii:

1. Każda osoba przystępująca do skanowanie powinna mieć czysty palec, ponieważ jednym z czynników wpływającym na złą jakość odcisku jest zanieczyszczenie palca, które może powodować niejednoznacznie wiarygodny i prawidłowy odcisk palca. Skaner używany do pobierania odcisków, także musi być wyczyszczony, aby nie pozostawały na nim odbicia linii papilarnych osób, które wcześniej brały udział w skanowaniu palca.
2. Nieprawidłowe umieszczenie palca na czujniku skanera może powodować zły obraz odcisku palca. Poniższe obrazy pokazują złe umieszczenie palca na skanerze (Rys. 12). Podczas kiedy palec zostanie źle umieszczony może zostać odrzucony, gdyż skaner nie wykryje go jako „żywy" palec, należy w takiej sytuacji zdjąć palec ze skanera i umieścić go jeszcze raz w prawidłowej pozycji.



(a) (b) (c)

Rysunek 4: Przykłady ilustrujące ułożenie palca na czujniku: umieszczenie prawidłowe (a), umieszczenia nieprawidłowe palec zbyt nisko (b) oraz palec zbyt wysoko(c)

Środowisko Megamatcher

MegaMatcher to zaawansowany zestaw narzędzi programistycznych (SDK) przeznaczony do tworzenia systemów identyfikacji biometrycznej na dużą skalę. Obsługuje różne modalności biometryczne, takie jak odciski palców, twarz, tęczówka, głos oraz odciski dłoni.

Przeglądanie i tworzenie własnych gotowych rozwiązań w oparciu o SDK

Pakiet megamatcher dysponuje bogatą dokumentacją SDK, tutorialami, które prezentują działanie na krótkich przykładach oraz przykładowe aplikacje z pełnym kodem źródłowym.

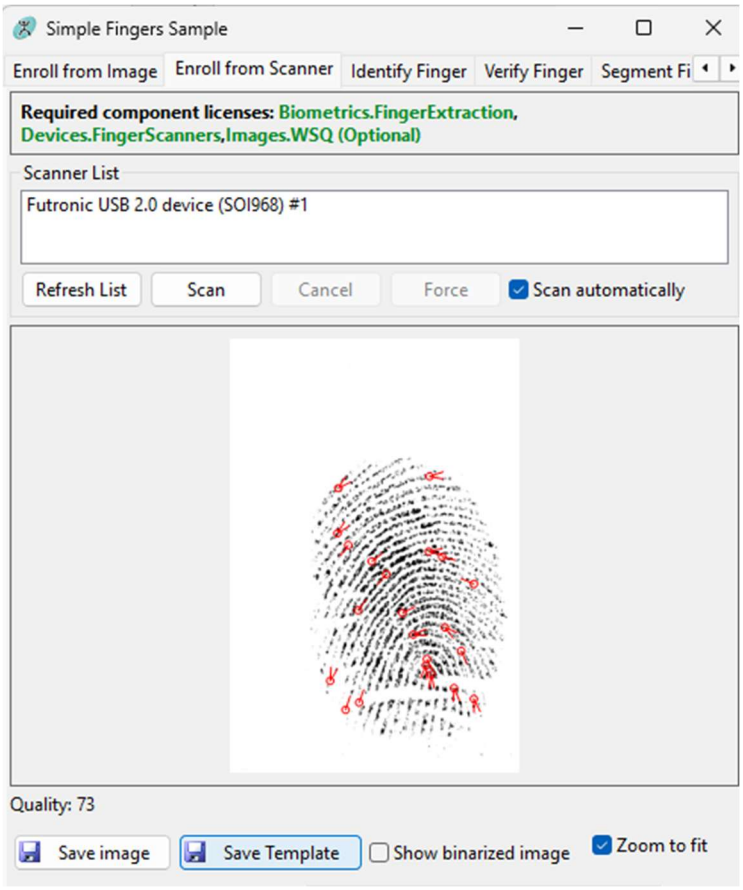
Aby móc uruchomić przykłady, należy tworzyć odpowiednią strukturę projektu. Bardzo ważna jest odpowiednia konfiguracji narzędzi budowania, integracją z natywnymi plikami dll zawierającymi niskopoziomiową logikę SDK, oraz systemem zarządzania licencjami. Dla uproszczenia można skorzystać z programu WorkspaceInitializer, który znajduje się na pulpicie stanowisk laboratoryjnych. Szczegółowa instrukcja korzystania z programu znajduje się w pliku *Inicjalizacja przestrzeni roboczej*.

Zadania do samodzielnego rozwiązania

Zadanie 1. Zbieranie danych daktyloskopijnych za pomocą Skanera Futronic

a) Stworzyć indywidualny folder dla swojego zespołu – będzie to baza szablonów odcisku palca.

b) Uruchomić program *SimpleFingersSample*, znajdujący się na pulpicie (Rysunek 5). Wybrać zakładkę *Enroll From Scanner*, upewnić się, że urządzenie jest na liście. Przycisk *Scan* rozpoczyna skanowanie.



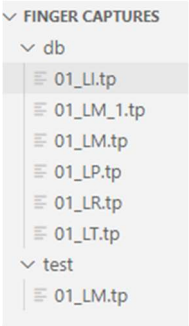
Rysunek 5: Program SimpleFingersSample umożliwiający m. in. zbieranie odcisków palców

c) Dla każdej osoby wykonać skany palców obu rąk. Po każdorazowym pomiarze zapisać plik szablonu w poniższym formacie, z rozszerzeniem .**tp**. Skorzystać z oznaczeń zaprezentowanych w Tabela 1. Wszystkie szablony niech znajdują się w folderze **db**. Stworzyć też folder **test** i wykonać dodatkowe, testowe skany (Rysunek 6).

<nr osoby>_<oznaczenie palca>.tp, Np.: 01_LT.tp, 01_RT.tp itd.

	Lewy	Prawy
Thumb (kciuk)	LT	RT
Index (wskazujący)	LI	RI
Middle (środkowy)	LM	RM
Ring (serdeczny)	LR	RR
Little (mały)	LL	RL

Tabela 1: Oznaczenia palców według standardu ANSI/NIST-ITL

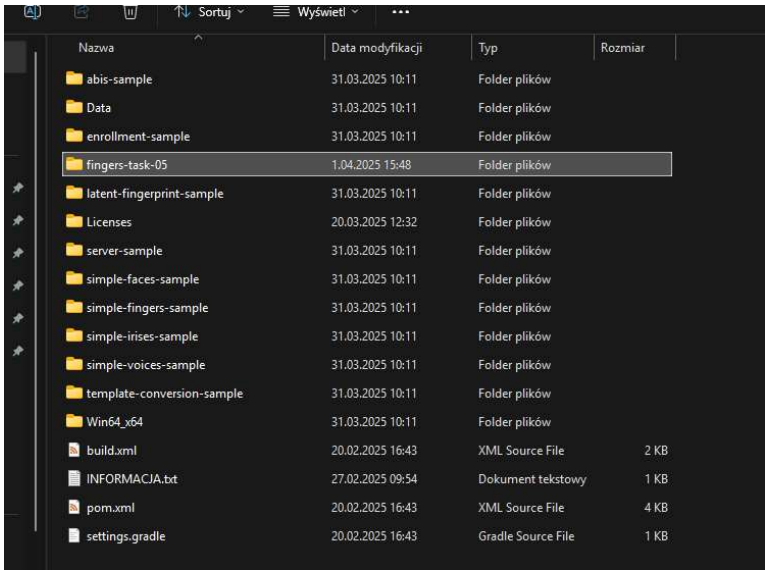


Rysunek 6: Fragment struktury bazy szablonów

Zadanie 2. Porównywanie szablonów w środowisku Megamatcher

Aby rozpocząć realizację zadania należy zainicjować nową przestrzeń roboczą. W tym celu należy skorzystać z programu WorkspaceInitializer, znajdującego się na pulpicie. Szczegółowa instrukcja znajduje się w pliku ***Inicjalizacja przestrzeni roboczej****.*

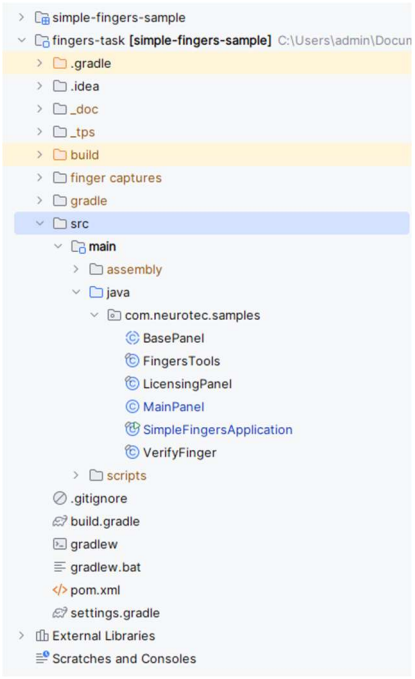
W przestrzeni roboczej umieścić folder z kodem początkowym, umieszczony w katalogu „Lab 07” na pulpicie.



Stworzyć prostą aplikację do porównywania szablonów odcisku palca. W tym celu uzupełnić klasę MainPanel w projekcie fingers-task. Do zrealizowania zadania może się przydać dokumentacja SDK, tutoriale oraz przykłady gotowych aplikacji.

Projekt został stworzony na bazie aplikacji demonstracyjnych pakietu Neurotec. W katalogu *com.neurotec.samples* znajduje się właściwy kod programu (Rysunek 7).

- Klasa **BasePanel –** klasa generyczna będąca modelem widoku. Po niej dziedziczy klasa MainPanel.
- Klasa **FingersTools** – klasa odpowiedzialna za pobieranie dostępu do komponentów biometrycznych – w tym przykładzie do licencji `Biometrics.FingerMatching`.
- Klasa **LicensingPanel** – wyświetla na górze okna aplikacji informacje o uzyskanych licencjach.
- Klasa **MainPanel** – główny widok aplikacji, który będzie modyfikowany w ramach zadania.
- Klasa **SimpleFingersApplication** – punkt wejściowy, inicjujący aplikację.



Rysunek 7: Struktura projektu

Wyjściowa postać aplikacji (Rysunek 8) pozwala na:

1. Wczytanie folderu bazy szablonów – są to pliki szablonów o rozszerzeniu .tp. o które będzie się opierał proces identyfikacji. Po wczytaniu wszystkie szablony pokazują się w tabeli.
2. Wczytanie szablonu do identyfikacji – jest to szablon, który będzie symulował nieznany odcisk. Po wczytaniu wyświetli się ścieżka do szablonu, oraz podgląd zawartości szablonu w prawym oknie.

Rysunek 8: Wyjściowa postać aplikacji

Należy uzupełnić kod następująco:

1. W wyniku kliknięcia przycisku Identyfikuj niech zostanie uruchomiona funkcja identyfikacyjna SDK. W tym celu należy uzupełnić funkcję identify. Pierwszym etapem jest inicjalizacja klienta NBiometricClient. Następnie tworzone jest nowa operacja UPDATE, polegająca na dodaniu do klienta osobników (NSubject) z bazy szablonów. Potem ustawiany jest próg dopasowania, sterujący minimalną wartość parametru score do uznania dwóch szablonów za zgodne.

```
updateFingersTools();

NBiometricClient client = FingersTools.getInstance().getClient();
client.clear();

NBiometricTask updateTask =
client.createTask(EnumSet.of(NBiometricOperation.UPDATE), null);
for (NSubject s : baseSubjects) {
    updateTask.getSubjects().add(s);
}

client.performTask(updateTask);

client.setMatchingThreshold(48);

NBiometricStatus status = client.identify(subjectToIdentify);
```

2. Po uzyskaniu wyników niech stworzona będzie mapa, przechowująca dla każdego osobnika (id to nazwa pliku, np. 01_LM.pt) wartość score dopasowania.

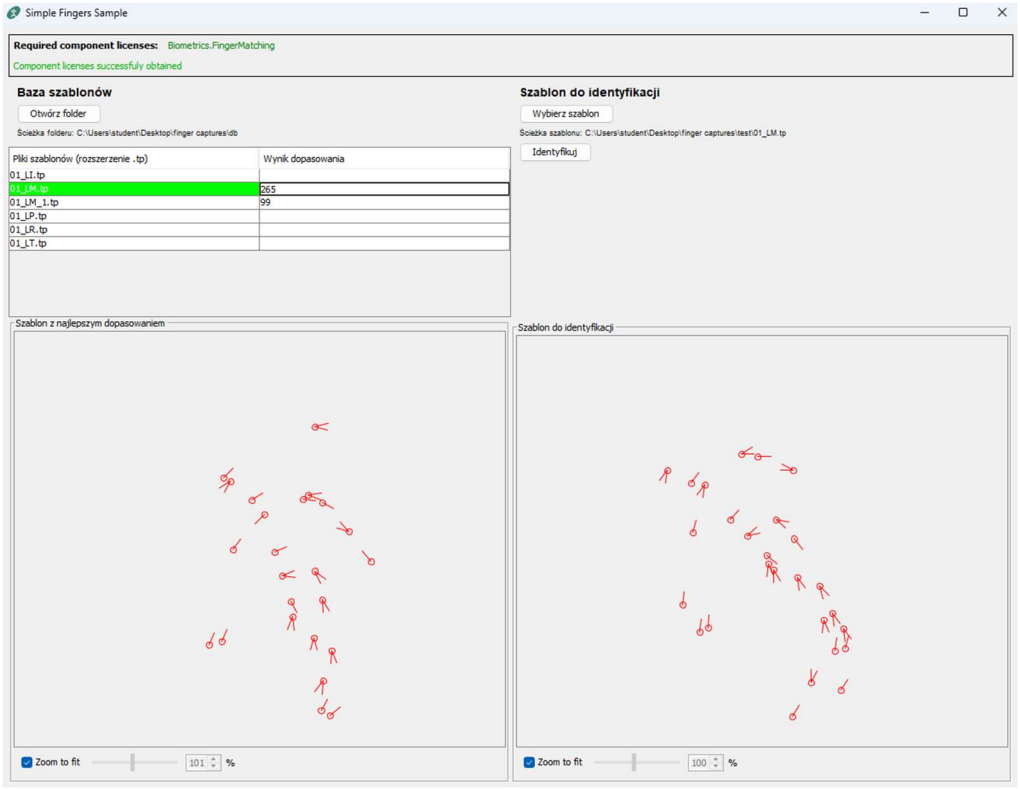
```
HashMap<String, Integer> resultsPerSubject = new HashMap<>();
if (status == NBiometricStatus.OK) {
    for (int i = 0; i < subjectToIdentify.getMatchingResults().size(); i++) {
        NMatchingResult result = subjectToIdentify.getMatchingResults().get(i);
        System.out.format("Matched with ID: '%s' with score %d\n",
result.getId(), result.getScore());
        resultsPerSubject.put(result.getId(), result.getScore());
    }
} else if (status == NBiometricStatus.MATCH_NOT_FOUND) {
    System.out.format("Match not found");
} else {
    System.out.format("Identification failed. Status: %s.\n", status);
}
```

3. W oparciu o wyznaczone dane, można zaktualizować widok tabelki wyników. W tym celu należy iterować po każdym wczytanym osobniku i pobierać jego score, po czym aktualizować dane w tabelce. Jednocześnie można wyznaczać indeks osobnika o najwyższym dopasowaniu do podświetlenia w tabeli i wyświetlenia podglądu szablonu (Rysunek 9).

```
int maxIndex = -1;
int maxValue = Integer.MIN_VALUE;
for (int i = 0; i < baseSubjects.size(); i++) {
    NSubject subject = baseSubjects.get(i);
    Integer result = resultsPerSubject.get(subject.getId());
    tableModel.setValueAt(result, i, 1);

    if (result != null && result > maxValue) {
        maxValue = result;
        maxIndex = i;
    }
}

setFingerView(baseSubjects.get(maxIndex), fingerViewLeft);
highlight(maxIndex);
```



Rysunek 9: Finalna postać aplikacji

4. Należy przetestować rezultat działania programu dla różnych wartości matchingThreshold.