LAPORAN TUGAS BESAR III
IF2211 STRATEGI ALGORITMA
"Pemanfaatan Pattern Matching dalam Membangun Sistem Deteksi Individu Berbasis Biometrik Melalui Citra Sidik Jari"

Dosen:

Ir. Rila Mandala, M. Eng, Ph. D.

Monterico Adrian, S. T., M. T.

Kelompok Ireng Jeung Paris:

13522130 Justin Aditya P.P.

13522155 Axel Santadi W.

13522159 Rafif Ardhinto I.

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

SEMESTER II TAHUN 2023/2024

DAFTAR ISI

BAB I
DESKRIPSI TUGAS

Gambar 1. Ilustrasi fingerprint recognition pada deteksi berbasis biometrik.
Sumber: https://www.aratek.co/news/unlocking-the-secrets-of-fingerprint-recognition

Di era digital ini, keamanan data dan akses menjadi semakin penting. Perkembangan teknologi membuka peluang untuk berbagai metode identifikasi yang canggih dan praktis. Beberapa metode umum yang sering digunakan seperti kata sandi atau pin, namun memiliki kelemahan seperti mudah terlupakan atau dicuri. Oleh karena itu, biometrik menjadi alternatif metode akses keamanan yang semakin populer. Salah satu teknologi biometrik yang banyak digunakan adalah identifikasi sidik jari. Sidik jari setiap orang memiliki pola yang unik dan tidak dapat ditiru, sehingga cocok untuk digunakan sebagai identitas individu. Pattern matching merupakan teknik penting dalam sistem identifikasi sidik jari. Teknik ini digunakan untuk mencocokkan pola sidik jari yang ditangkap dengan pola sidik jari yang terdaftar di database. Algoritma pattern matching yang umum digunakan adalah Bozorth dan Boyer-Moore. Algoritma ini memungkinkan sistem untuk mengenali sidik jari dengan cepat dan akurat, bahkan jika sidik jari yang ditangkap tidak sempurna.

Dengan menggabungkan teknologi identifikasi sidik jari dan pattern matching, dimungkinkan untuk membangun sistem identifikasi biometrik yang aman, handal, dan

mudah digunakan. Sistem ini dapat diaplikasikan di berbagai bidang, seperti kontrol akses, absensi karyawan, dan verifikasi identitas dalam transaksi keuangan.

Di dalam Tugas Besar 3 ini, Anda diminta untuk mengimplementasikan sistem yang dapat melakukan identifikasi individu berbasis biometrik dengan menggunakan sidik jari. Metode yang akan digunakan untuk melakukan deteksi sidik jari adalah Boyer-Moore dan Knuth-Morris-Pratt. Selain itu, sistem ini akan dihubungkan dengan identitas sebuah individu melalui basis data sehingga harapannya terbentuk sebuah sistem yang dapat mengenali identitas seseorang secara lengkap hanya dengan menggunakan sidik jari.

BAB II

LANDASAN TEORI

2.1 Deskripsi singkat algoritma KMP, BM, dan REGEX

2.1.1 Algoritma Knuth-Morris-Pratt (KMP)

Algoritma Knuth-Morris-Pratt (KMP) adalah algoritma pencocokan pola yang efisien untuk menemukan satu atau beberapa kejadian dari sebuah "pola" (pattern) dalam "teks" (text). Algoritma ini memanfaatkan informasi dari pola untuk menghindari pencarian yang berlebihan. KMP terdiri dari dua tahap utama:

- **Preprocessing**: Membangun sebuah array yang dikenal sebagai "failure function" atau "partial match table" yang digunakan untuk menyimpan informasi tentang seberapa banyak pola yang cocok dengan dirinya sendiri. Fungsi ini membantu menentukan dari mana pencarian berikutnya harus dilanjutkan saat terjadi ketidakcocokan.
- **Searching**: Menggunakan tabel failure function untuk menggeser pola dengan cara yang paling efisien tanpa harus membandingkan ulang karakter yang sudah diketahui cocok.

KMP memiliki kompleksitas waktu O(n + m), dengan n adalah panjang teks dan m adalah panjang pola, menjadikannya sangat efisien untuk pencocokan pola.

2.1.2 Algoritma Boyer-Moore (BM)

Algoritma Boyer-Moore (BM) adalah salah satu algoritma pencocokan pola yang paling efisien dan digunakan dalam praktik. Algoritma ini bekerja dari kanan ke kiri dalam pola dan menggunakan dua heuristik utama untuk mempercepat pencarian:

- **Bad Character Heuristic**: Jika terjadi ketidakcocokan, algoritma mencari karakter dalam teks yang tidak cocok dengan karakter dalam pola dan menggeser pola sedemikian rupa sehingga karakter yang salah tersebut berada di bawah karakter yang sesuai dalam pola.
- **Good Suffix Heuristic**: Jika bagian akhir dari pola cocok dengan bagian teks tetapi terjadi ketidakcocokan pada karakter sebelumnya, algoritma menggeser pola untuk menyelaraskan kemunculan berikutnya dari "suffix" yang baik dalam pola.

Boyer-Moore umumnya sangat cepat dalam praktik dan memiliki kompleksitas waktu rata-rata yang lebih baik daripada KMP.

2.1.3 Regular Expressions (REGEX)

Regular Expressions (REGEX) adalah serangkaian karakter yang mendefinisikan pola pencarian. REGEX sangat fleksibel dan dapat digunakan untuk pencocokan pola yang kompleks dalam teks. REGEX mendukung berbagai operator dan konstruk seperti:

- **Karakter Literal**: Karakter spesifik yang harus dicocokkan.
- **Metakarakter**: Karakter dengan makna khusus, seperti . (titik) untuk mencocokkan karakter apapun kecuali newline.
- **Kuantor**: Mengindikasikan berapa kali suatu elemen harus muncul (misalnya, * untuk nol atau lebih kali, + untuk satu atau lebih kali).
- **Grup dan Alternasi**: Mengelompokkan bagian dari pola dan mengizinkan pilihan alternatif (misalnya, (a|b) untuk mencocokkan a atau b).

REGEX digunakan secara luas dalam banyak bahasa pemrograman dan alat-alat untuk pencarian, penggantian, dan manipulasi teks.

2.2 Penjelasan teknik pengukuran persentase kemiripan.

Untuk mengukur persentase kemiripan antara dua citra sidik jari, kita akan menerapkan metode yang mengkombinasikan algoritma pattern matching (Knuth-Morris-Pratt dan Boyer-Moore) dan teknik perhitungan jarak (Hamming Distance, Levenshtein Distance, atau Longest Common Subsequence). Berikut adalah langkah-langkah yang diusulkan untuk mengukur persentase kemiripan:

1. **Ekstraksi dan Konversi Citra Sidik Jari**:
    - Citra sidik jari berukuran m x n piksel dipecah menjadi potongan-potongan kecil (misalnya, 30 x 30 piksel).
    - Setiap potongan diproses untuk menghasilkan representasi biner.
    - Data biner kemudian dikelompokkan setiap 8 bit untuk dikonversi menjadi karakter ASCII. Proses ini mengubah pola biner menjadi string yang lebih mudah diproses oleh algoritma pattern matching.
2. **Pattern Matching Menggunakan KMP dan BM**:
    - Dengan string ASCII yang dihasilkan, algoritma Knuth-Morris-Pratt (KMP) dan Boyer-Moore (BM) digunakan untuk mencocokkan potongan sidik jari masukan dengan basis data sidik jari.
    - Jika tidak ditemukan kecocokan sempurna (exact match), algoritma ini akan membantu menemukan potongan yang paling mirip.
3. **Perhitungan Tingkat Kemiripan**:
    - **Hamming Distance**: Mengukur jumlah posisi di mana karakter yang sesuai antara dua string berbeda. Nilai ini kemudian digunakan untuk menghitung persentase kemiripan.
    - **Levenshtein Distance**: Mengukur jumlah operasi (penyisipan, penghapusan, atau penggantian karakter) yang diperlukan untuk mengubah satu string menjadi string lain. Semakin kecil jaraknya, semakin tinggi kemiripannya.

    - **Longest Common Subsequence (LCS)**: Menghitung panjang subsekuens terpanjang yang muncul dalam urutan yang sama di kedua string. Semakin panjang LCS, semakin tinggi kemiripannya.
4. **Menghitung Persentase Kemiripan**:
    - **Hamming Distance**:

$$Persentase\ Kemiripan\ =\ (1\ -\ \frac{Hamming\ Distance}{Panjang\ String})\ x\ 100\%$$

5. **Penentuan Batas Kemiripan**:
    - Berdasarkan pengujian dan eksperimen, tentukan nilai batas persentase kemiripan yang dianggap cukup untuk mengidentifikasi sidik jari yang mirip. Misalnya, jika persentase kemiripan di atas 85% dianggap cukup untuk menyatakan dua sidik jari mirip, maka gunakan nilai ini sebagai ambang batas.
6. **Penanganan Kasus Ujung**:
    - Jika ukuran citra sidik jari tidak habis dibagi dengan ukuran piksel yang dipilih, potongan terakhir harus ditangani dengan mengisi sisa piksel atau dengan menggunakan teknik padding.

Dengan menggunakan langkah-langkah di atas, sistem dapat mengukur dan menampilkan persentase kemiripan antara sidik jari yang menjadi masukan pengguna dengan sidik jari yang ada dalam basis data. Hasil ini membantu dalam proses identifikasi biometrik untuk menentukan individu yang memiliki sidik jari paling mirip.

2.3 Penjelasan singkat mengenai aplikasi desktop yang dibangun.

Aplikasi dibangun menggunakan Windows Forms App untuk GUI. Ketika dijalankan, aplikasi akan membuat instansi baru FingerprintService yang akan digunakan untuk memeriksa gambar input dari user.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1 Langkah-langkah Pemecahan Masalah

Untuk membangun sistem deteksi individu berbasis biometrik melalui citra sidik jari, langkah-langkah pemecahan masalah yang diambil adalah sebagai berikut:

1. **Pengumpulan dan Pra-pemrosesan Data**:
    - Mengumpulkan dataset citra sidik jari dari sumber yang tersedia.
    - Melakukan pra-pemrosesan pada citra untuk meningkatkan kualitas gambar, seperti normalisasi, penghapusan noise, dan peningkatan kontras.
    - Mengkonversi citra sidik jari menjadi representasi biner.
2. **Segmentasi dan Ekstraksi Fitur**:
    - Membagi citra sidik jari menjadi blok-blok kecil (misalnya, 30x30 piksel).
    - Mengkonversi setiap blok menjadi string biner.
    - Mengelompokkan setiap 8 bit biner menjadi karakter ASCII untuk representasi string yang lebih mudah diproses.
3. **Pencocokan Pola**:
    - Menggunakan algoritma Knuth-Morris-Pratt (KMP) dan Boyer-Moore (BM) untuk mencari pola string ASCII dari sidik jari masukan dalam basis data sidik jari.
    - Jika tidak ditemukan kecocokan sempurna, menghitung tingkat kemiripan menggunakan metode perhitungan jarak seperti Hamming Distance, Levenshtein Distance, atau Longest Common Subsequence (LCS).
4. **Evaluasi dan Penyesuaian Nilai Ambang**:
    - Melakukan pengujian dan eksperimen untuk menentukan nilai ambang batas persentase kemiripan yang dianggap cukup untuk mengidentifikasi dua sidik jari sebagai mirip.
    - Menggunakan hasil pengujian untuk mengatur nilai ambang dalam sistem.
5. **Penanganan Data Korup**:
    - Mengimplementasikan penanganan untuk kemungkinan data korup dalam basis data, terutama pada atribut nama yang bisa mengalami variasi bahasa alay.
    - Menggunakan Regular Expression (Regex) untuk mengonversi variasi bahasa alay kembali ke bentuk alfabetik yang bersesuaian.

3.2 Proses Penyelesaian Solusi dengan Algoritma KMP dan BM

3.2.1 Algoritma Knuth-Morris-Pratt (KMP)

Langkah-langkah Implementasi KMP:

1. **Preprocessing (Membangun Tabel Partial Match):**
    - Buat tabel partial match untuk pola yang akan dicari. Tabel ini menyimpan panjang prefix yang merupakan juga suffix untuk setiap posisi dalam pola.
    - Misalnya, untuk pola P, tabel pi dihitung sehingga pi[i] menunjukkan panjang prefix terpanjang dari P[0..i] yang juga merupakan suffix.
2. **Pencarian (Searching):**
    - Lakukan pencarian pola pada teks dengan menggunakan tabel partial match.
    - Jika terjadi ketidakcocokan, gunakan nilai dari tabel untuk menggeser pola ke posisi yang sesuai, tanpa perlu membandingkan karakter yang sudah diketahui cocok sebelumnya.
    - Proses ini dilakukan sampai seluruh teks telah dipindai atau pola ditemukan.

Contoh Implementasi:

- Misalkan pola adalah ABCDABD dan teks adalah ABC ABCDAB ABCDABCDABDE.
- Tabel partial match untuk pola ABCDABD adalah [0, 0, 0, 0, 1, 2, 0].
- Algoritma KMP menggunakan tabel ini untuk mencocokkan pola dengan teks secara efisien.

3.2.2 Algoritma Boyer-Moore (BM)

Langkah-langkah Implementasi BM:

1. **Preprocessing (Membangun Tabel Bad Character dan Good Suffix):**
    - Buat tabel bad character yang menunjukkan berapa jauh pola dapat digeser ketika terjadi ketidakcocokan pada karakter tertentu.
    - Buat tabel good suffix yang menunjukkan berapa jauh pola dapat digeser ketika terdapat suffix yang cocok tetapi karakter sebelumnya tidak cocok.
2. **Pencarian (Searching):**
    - Lakukan pencarian pola pada teks dari kanan ke kiri.
    - Jika terjadi ketidakcocokan, gunakan nilai dari tabel bad character dan good suffix untuk menentukan seberapa jauh pola harus digeser.
    - Proses ini dilakukan sampai seluruh teks telah dipindai atau pola ditemukan.

Contoh Implementasi:

- Misalkan pola adalah EXAMPLE dan teks adalah HERE IS A SIMPLE EXAMPLE.
- Tabel bad character untuk pola EXAMPLE mungkin seperti berikut:
    - E: 0, X: 1, A: 2, M: 3, P: 4, L: 5.

- Tabel good suffix disesuaikan untuk mengoptimalkan pergeseran pola berdasarkan kecocokan suffix.
- Algoritma BM menggunakan tabel ini untuk mencocokkan pola dengan teks secara efisien.

Dengan mengimplementasikan algoritma KMP dan BM, sistem dapat melakukan pencocokan pola secara efektif dan efisien, baik untuk menemukan kecocokan sempurna maupun kecocokan parsial yang digunakan untuk menghitung persentase kemiripan antara sidik jari masukan dan basis data.

3.3 Fitur fungsional dan arsitektur aplikasi desktop yang dibangun.

3.3.1 Fitur Fungsional:

- **Input Citra Sidik Jari**: Pengguna dapat memasukkan citra sidik jari yang ingin dicari.
- **Pilih Algoritma**: Pengguna dapat memilih antara algoritma KMP atau BM untuk pencocokan.
- **Pencarian**: Aplikasi melakukan pencarian sidik jari yang paling mirip dalam basis data.
- **Hasil Pencarian**: Menampilkan sidik jari yang paling mirip beserta informasi biodata terkait.
- **Persentase Kemiripan**: Menunjukkan tingkat kemiripan dalam bentuk persentase.
- **Waktu Eksekusi**: Menampilkan informasi mengenai waktu yang dibutuhkan untuk proses pencarian.

3.3.2 Arsitektur Aplikasi Desktop:

- **Antarmuka Pengguna (UI)**: Dibangun dengan komponen GUI untuk input citra, pilihan algoritma, dan menampilkan hasil pencarian.
- **Logika Bisnis**: Mengimplementasikan algoritma KMP dan BM serta metode pengukuran kemiripan.
- **Basis Data**: Menggunakan SQL untuk menyimpan dan mengelola data sidik jari serta biodata pengguna.
- **Modul Pencocokan Sidik Jari**: Mengelola konversi citra sidik jari, pencocokan pola, dan pengukuran kemiripan.
- **Lapisan Data**: Menghubungkan aplikasi dengan basis data untuk mengambil dan menyimpan data.

3.4 Contoh ilustrasi kasus.

1. **Input**: Pengguna memasukkan citra sidik jari yang ingin dicocokkan.
2. **Proses**:
    - Citra sidik jari dikonversi menjadi representasi biner, kemudian ke ASCII.
    - Pengguna memilih algoritma KMP untuk pencocokan.

    - Algoritma KMP melakukan pencarian pola dalam string ASCII yang ada di basis data.
3. **Hasil**:
    - Aplikasi menampilkan sidik jari yang paling mirip beserta biodata yang sesuai.
    - Menunjukkan persentase kemiripan (misalnya, 92%) dan waktu eksekusi (misalnya, 1.5 detik).
    - Jika tidak ada kecocokan sempurna, aplikasi menampilkan sidik jari dengan kemiripan tertinggi di atas ambang batas yang ditentukan.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi teknis program

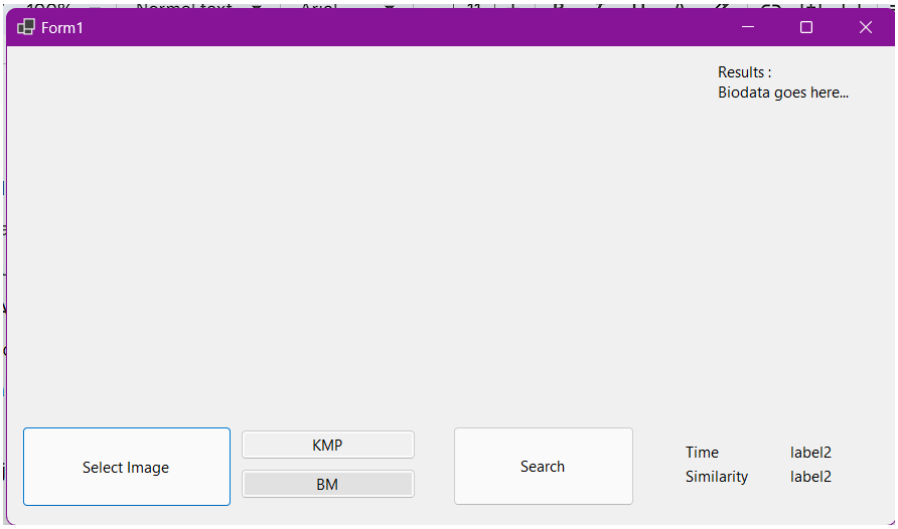
Program memiliki 3 projek utama: GUI, database, serta algorithm. GUI Memiliki file Program.cs yang merupakan main dan Form1.cs yang memegang seluruh kode GUI

Database terdiri dari fingerprintService.cs serta models.cs. fingerprintService.cs menyediakan interface untuk mempermudah query database dengan menyediakan class fingerprintService, kelas ini berisi metode-metode yang digunakan untuk mengakses database beberapa menggunakna code SQL. models.cs berisi pemodelan relasi basis data yang mana berisi tabel Biodata dan Sidikjari. Dalam database juga terdapat Program.cs yang berfungsi untuk mengisi data dalam database menggunakan faker. Setelah Program.cs dijalankan data akan terisi dalam blogging.db

Terahkir, Algoritma terdiri dari seluruh algoritma yang digunakan untuk tugas ini (KMP, BM, Levenshtein, Hamming, dan REGEX). Sesuai namanya, KMP.cs menyediakan string matching dengan algoritma Knuth–Morris–Pratt, dan BM.cs menyediakan string matching dengan algoritma Boyer-Moore. Sedangkan Levenshtein.cs dan Hamming.cs menyediakan *checker* untuk memeriksa seberapa mirip suatu string dengan string lainnya.

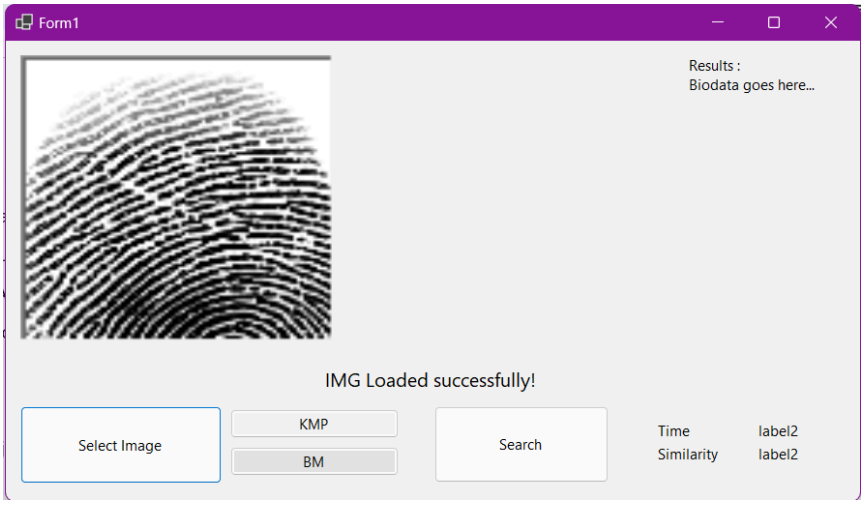
4.2 Penjelasan tata cara penggunaan program

Pengguna cukup membuka src > NewSidikJariGUI > bin > release > net8.0 dan menjalankan NewSidikJariGUI.exe



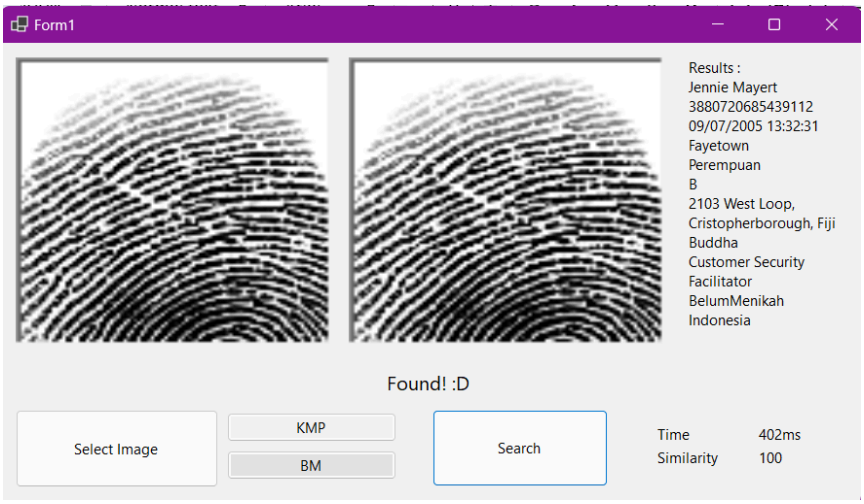
Tampilan aplikasi setelah dibuka

Pengguna kemudian memilih sidik jari yang ingin diperiksa menggunakan tombol “Select Image” dan memilih file gambar yang ingin diperiksa.



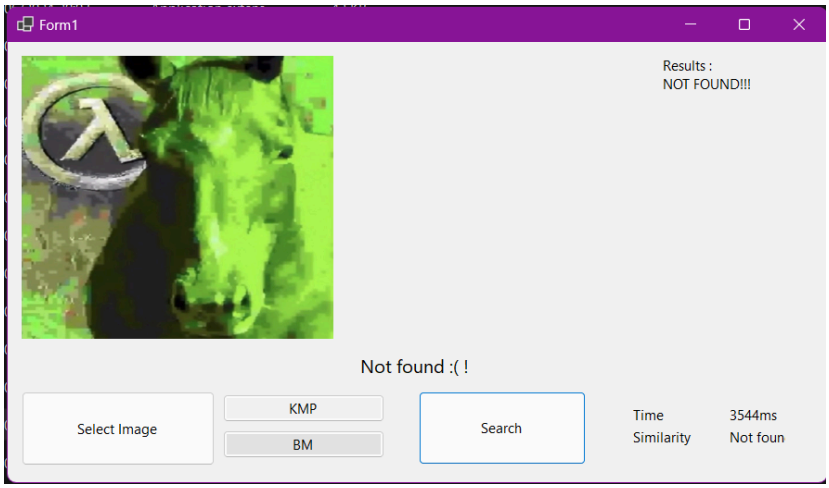
Setelah memilih gambar sidik jari

Pilih algoritma yang diinginkan (KMP atau BM, algoritma terpilih akan nampak lebih terang) dan tekan search.

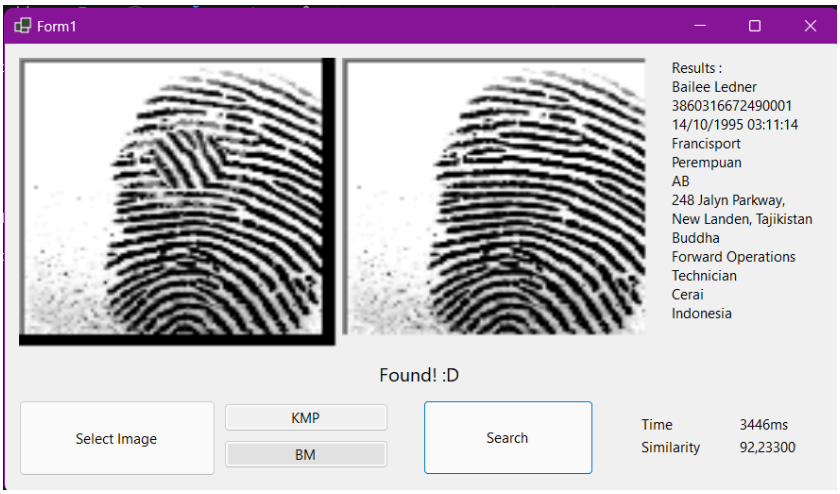


Sebagai catatan, program akan mencoba mencari terlebih dahulu menggunakan KMP atau BM (pilihan pengguna). Namun, jika tidak ditemukan sama sekali baru menggunakan algoritma Hemming distance untuk mencari foto terdekat.

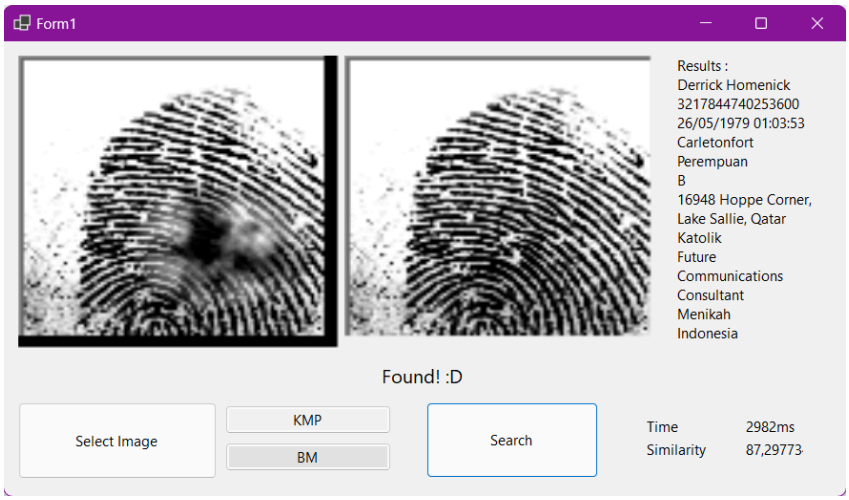
4.3 Hasil pengujian



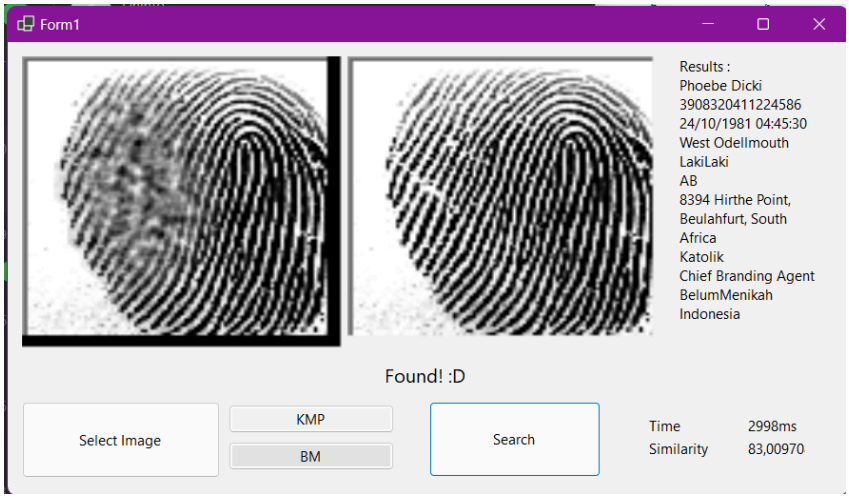
Contoh gambar yang pasti gagal



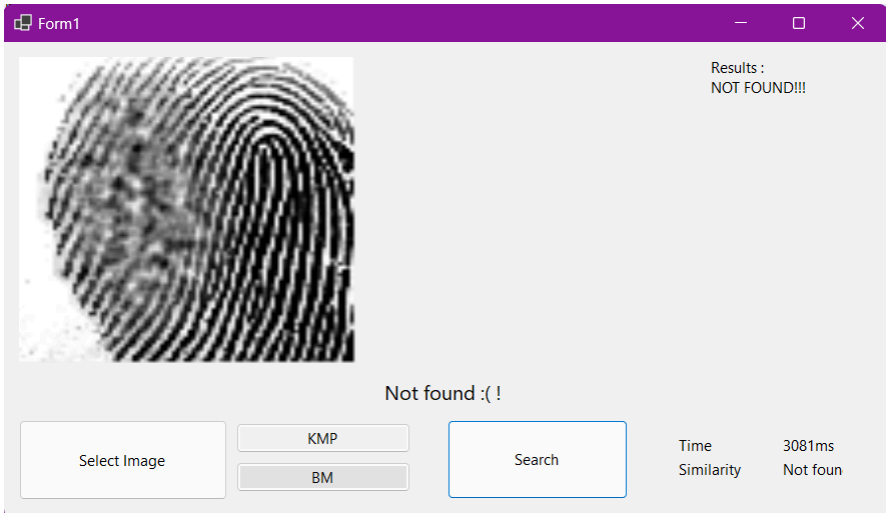
Testcase untuk gambar yang serupa tapi tidak sama



Testcase lebih susah



Testcase paling susah



Program gagal jika diberikan foto yang ukuran nya tidak sesuai spek (96 * 103)

4.4 Analisis hasil pengujian

Program berhasil mendeteksi/mengenali foto yang kabur/*blur* dengan cukup baik, hanya saja terdapat kekurangan dalam implementasi algoritma *hamming distance* dimana algoritma tidak berfungsi jika gambar yang diberikan ukuran nya tidak sesuai dengan seluruh gambar pada database (semua gambar pada database berukuran 96 * 103).

BAB V
PENUTUP

5.1 Kesimpulan

Dalam Tugas Besar ini, kami berhasil mengembangkan sebuah aplikasi pencari sidik jari menggunakan pattern matching yang didasari algoritma Knuth–Morris–Pratt (KMP), Boyer-Moore (BM), dan juga Regular Expression (REGEX). Proses pencocokan pola yang efisien dan penggunaan teknik pengukuran kemiripan seperti Hamming Distance, Levenshtein Distance, atau Longest Common Subsequence (LCS) memungkinkan identifikasi individu dengan tingkat akurasi yang tinggi.

Kami dapat menyimpulkan beberapa poin penting dari Tugas Besar ini:

Pemahaman Masalah: Kami memulai Tugas Besar dengan memahami masalah yang dihadapi, yaitu pencarian sidik jari yang paling mirip dengan sidik jari acuan. Melalui analisis masalah, kami dapat menentukan pendekatan yang tepat dalam memecahkan masalah tersebut.

Penggunaan Algoritma: Kami memilih algoritma Knuth–Morris–Pratt (KMP), dan Boyer-Moore (BM) dikarenakan keduanya cocok digunakan untuk string matching, dan kami memilih menggunakan Regular Expression (REGEX) dikarenakan kewaspadaan kami akan data yang bisa saja *corrupt* pada database kami. Dengan adanya Regular Expression (REGEX), maka hal tersebut dapat teratasi.

Kemampuan Finder: *Fingerprint Finder* yang kami bangun dapat mengembalikan biodata lengkap sesuai dengan pemilik sidik jari acuan, dan berlaku untuk kesepuluh jari.

Dengan demikian, tugas besar ini tidak hanya memberikan pengalaman dalam mengembangkan Finder untuk fingerprint, tetapi juga mengasah kemampuan dalam menerapkan konsep *String Matching*, dan juga pembuatan aplikasi.

5.2 Saran

Untuk pengembangan *Fingerprint Finder* lebih lanjut, kami memiliki beberapa saran yaitu:

1. **Penggunaan Dataset Lebih Besar**: Untuk menguji keandalan dan kinerja sistem, disarankan untuk menggunakan dataset yang lebih besar dan lebih bervariasi.
2. **Pengoptimalan Algoritma**: Meskipun KMP dan BM sudah efisien, penggunaan kombinasi dengan algoritma lain atau optimasi lebih lanjut dapat meningkatkan kinerja.
3. **Implementasi Real-time**: Mengembangkan versi real-time dari aplikasi untuk keperluan yang membutuhkan respon cepat, seperti di bandara atau kantor pemerintahan.
4. **Integrasi Keamanan Tambahan**: Menambahkan lapisan keamanan tambahan untuk melindungi data biometrik dari akses yang tidak sah.
5. **Pengembangan Mobile App**: Selain aplikasi desktop, pengembangan aplikasi mobile bisa menjadi langkah berikutnya untuk memperluas jangkauan penggunaan.

5.3 Refleksi

Tugas besar ini telah memberikan kami pelajaran tentang pentingnya aplikasi ilmu komputasi dalam menyelesaikan masalah yang nyata. Kami memahami bahwa tidak ada satu solusi yang sempurna. Oleh karena itu, eksplorasi dan inovasi terus-menerus diperlukan. Kerja sama tim, komunikasi yang efektif, dan pengelolaan waktu yang baik telah menjadi kunci dalam menyelesaikan tugas besar ini. Kami juga belajar menghadapi kegagalan sebagai bagian dari proses pembelajaran dan menggunakan kegagalan tersebut sebagai langkah untuk maju lebih jauh.

LAMPIRAN

6.1 GitHub Repository (Latest Release)

https://github.com/AxelSantadi/Tubes3_Ireng-Jeung-Paris

DAFTAR PUSTAKA

Munir, R. (2024). *Pencocokan string*. Retrieved from Homepage Rinaldi Munir: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

Munir, R. (2024). *Pencocokan string dengan Regular Expression.* Retrieved from Homepage Rinaldi Munir: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2022-2023/String-Matching-dengan-Regex-2019.pdf

Ruiz, A. (2024). SOCOFing: A fingerprint database for testing and evaluating fingerprint recognition algorithms. Kaggle: https://www.kaggle.com/datasets/ruizgara/socofing