**IMAGE ENCHANMENT**

**IMPLEMENTASI IMAGE ENCHANMENT PADA CITRA REKTORAT UNIVERSITAS ANDALAS**

**Dosen Pengampu: Afdal dinil hak , S. Kom, M. CS.**

****

**DISUSUN OLEH :**

**KELOMPOK 1**

**DEPARTEMEN INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS ANDALAS**

**2024**

**PENDAHULUAN**

Pengolahan citra digital merupakan cabang ilmu komputer yang berfokus pada manipulasi, analisis, dan interpretasi gambar dalam format digital. Teknik ini memungkinkan pemrosesan gambar untuk berbagai tujuan, mulai dari peningkatan kualitas hingga ekstraksi informasi yang berguna. Dalam dunia modern, pengolahan citra telah menjadi aspek penting dalam berbagai bidang, termasuk kedokteran, keamanan, industri, dan hiburan. Dengan kemajuan teknologi, teknik pengolahan citra semakin berkembang dan memungkinkan berbagai aplikasi yang lebih kompleks dan presisi.

Secara umum, pengolahan citra terbagi menjadi beberapa tahap utama, yaitu pre-processing, processing, dan post-processing. Pre-processing bertujuan untuk meningkatkan kualitas citra sebelum dianalisis lebih lanjut, seperti menghilangkan noise, meningkatkan kontras, atau mengubah format warna. Processing merupakan tahap utama yang melibatkan penerapan berbagai algoritma untuk mengekstraksi fitur atau mengidentifikasi objek tertentu dalam citra. Sementara itu, post-processing dilakukan setelah analisis utama, seperti dalam klasifikasi atau pengambilan keputusan berdasarkan hasil yang telah diperoleh.

Dalam dunia akademik dan industri, pengolahan citra digunakan dalam berbagai aplikasi praktis, seperti sistem pengenalan wajah, deteksi objek dalam citra satelit, serta analisis citra medis untuk mendeteksi penyakit. Penerapan dalam bidang kesehatan, misalnya, memungkinkan analisis lebih akurat terhadap citra hasil pemeriksaan medis seperti MRI atau CT scan. Dalam bidang industri, pengolahan citra digunakan untuk inspeksi kualitas produk secara otomatis, mendeteksi cacat, serta meningkatkan efisiensi produksi dengan bantuan sistem berbasis kecerdasan buatan.

Teknik pengolahan citra dapat dibedakan menjadi beberapa metode utama, termasuk pemrosesan berbasis spasial dan frekuensi. Pemrosesan berbasis spasial melibatkan manipulasi langsung terhadap piksel dalam citra untuk menghasilkan efek tertentu, seperti peningkatan kontras atau deteksi tepi. Sementara itu, pemrosesan berbasis frekuensi menggunakan transformasi matematika untuk menganalisis pola dalam citra, seperti dalam pemrosesan citra medis atau teknik pemampatan gambar untuk efisiensi penyimpanan dan transmisi data.

Perkembangan teknologi komputasi juga turut mendorong pengolahan citra ke tingkat yang lebih tinggi dengan integrasi kecerdasan buatan dan pembelajaran mesin. Algoritma berbasis jaringan saraf tiruan dan deep learning kini semakin banyak digunakan untuk meningkatkan akurasi dalam analisis citra. Penerapan model kecerdasan buatan ini telah membawa perubahan signifikan dalam berbagai bidang, seperti sistem pengawasan otomatis, pengenalan tulisan tangan, serta analisis pola perilaku manusia berdasarkan citra kamera pengawas.

Dengan semakin berkembangnya teknologi dan meningkatnya kebutuhan akan analisis citra yang lebih akurat dan efisien, pengolahan citra digital menjadi salah satu bidang yang terus mengalami inovasi. Berbagai penelitian terus dilakukan untuk mengembangkan metode baru yang lebih efektif dalam menangani tantangan yang muncul dalam pengolahan citra. Dengan demikian, pengolahan citra digital tidak hanya menjadi bidang yang menarik untuk dipelajari, tetapi juga memiliki dampak luas dalam berbagai sektor kehidupan manusia.

1. **TUJUAN**

Tugas ini bertujuan untuk menerapkan dan membandingkan beberapa metode pengolahan citra, yaitu:

Segmentasi menggunakan thresholding

Pembuatan citra negatif

Peningkatan kontras dengan contrast stretching

Cropping citra menggunakan image subtraction

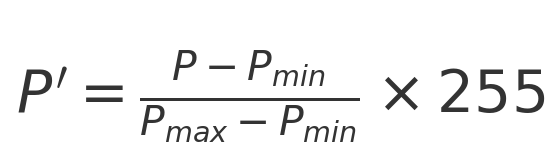
Normalisasi histogram citra

Membandingkan implementasi berbasis NumPy dan list manual

1. **METODE**

.

1. Membaca dan Mengonversi Citra
2. Gambar diunggah dan dibaca menggunakan PIL (Pillow).
3. Pada file pertama, gambar dikonversi ke array NumPy, yang memungkinkan pemrosesan lebih cepat dan efisien menggunakan operasi vektor.
4. Pada file kedua, gambar dikonversi ke skala abu-abu dan diubah menjadi list 2D, yang diproses secara manual menggunakan perulangan.
5. Segmentasi dengan Image Thresholding
6. Teknik ini digunakan untuk mengubah gambar menjadi hitam-putih dengan membandingkan nilai piksel terhadap ambang batas (threshold) tertentu.
7. Jika nilai piksel lebih besar dari 128, piksel diubah menjadi 255 (putih), sedangkan jika lebih kecil, piksel diubah menjadi 0 (hitam).
8. Pada implementasi pertama, fungsi NumPy digunakan untuk mempercepat pemrosesan.
9. Pada implementasi kedua, setiap piksel diproses secara manual menggunakan perulangan.
10. Transformasi Citra Negatif
11. Setiap nilai piksel dikonversi menggunakan rumus 255 - nilai piksel, sehingga warna terang menjadi gelap dan sebaliknya.
12. Pada file pertama, NumPy digunakan untuk mempercepat perhitungan seluruh piksel.
13. Pada file kedua, pengolahan dilakukan secara manual dengan membaca dan mengubah setiap nilai piksel satu per satu dalam list.
14. Peningkatan Kontras (Contrast Stretching)\*\*
15. Metode ini digunakan untuk memperlebar rentang nilai piksel agar gambar lebih kontras.
16. Langkah-langkah yang dilakukan:
17. Menentukan \*\*nilai minimum\*\* dan \*\*nilai maksimum\*\* piksel dalam gambar.
18. Menyesuaikan nilai setiap piksel dengan rumus:



1. Hasilnya adalah gambar dengan kontras yang lebih tinggi.
2. Pada file pertama, NumPy digunakan untuk menghitung nilai min dan max secara otomatis.
3. Pada file kedua, fungsi `get\_min\_max()` dibuat secara manual untuk mencari nilai minimum dan maksimum.
4. Cropping dengan Image Subtraction
5. Teknik ini mengurangi citra asli dengan hasil thresholding untuk menyorot bagian tertentu.
6. Perbedaan utama antara kedua implementasi adalah penggunaan NumPy dalam satu file dan operasi manual dalam file lainnya.
7. Normalisasi Histogram
8. Normalisasi histogram dilakukan untuk mendistribusikan nilai piksel agar tersebar lebih merata antara 0 hingga 255.
9. Implementasi pertama menggunakan \*\*NumPy\*\*, sementara implementasi kedua menghitung distribusi piksel secara manual dengan mencari nilai minimum dan maksimum.
10. **PENJELASAN KODE**

**Penjelasan Kode Pengolahan Citra**

Kode dalam implementasi ini berisi berbagai teknik pengolahan citra digital menggunakan bahasa pemrograman Python dengan pustaka **Pillow (PIL), NumPy, dan Matplotlib**. Berikut adalah penjelasan mengenai fungsionalitas yang ada dalam kode:

**1. Import Pustaka**

Kode pertama mengimpor pustaka yang digunakan dalam pengolahan citra, yaitu:

* **PIL (Pillow)** untuk membaca dan memproses gambar.
* **NumPy** untuk merepresentasikan citra sebagai array dan melakukan operasi matematis.
* **Matplotlib** untuk menampilkan hasil pengolahan citra.
* **Google Colab files** untuk mengunggah file gambar jika dijalankan dalam Google Colab.

**2. Fungsi Utama dalam Pengolahan Citra**

Berikut beberapa fungsi yang digunakan untuk memproses citra:

* **read\_image(image\_path)** → Membaca gambar dalam mode warna.
* **image\_to\_array(image)** → Mengubah gambar menjadi array NumPy untuk diproses lebih lanjut.
* **image\_thresholding(image\_array, threshold=128)** → Melakukan segmentasi gambar menggunakan teknik thresholding. Jika nilai piksel lebih besar dari ambang batas (128 secara default), maka menjadi putih (255), jika tidak maka menjadi hitam (0).
* **image\_negative(image\_array)** → Menghasilkan citra negatif dengan mengurangi setiap piksel dari 255.
* **contrast\_stretching(image\_array)** → Meningkatkan kontras gambar dengan teknik **contrast stretching**, yaitu meregangkan rentang nilai piksel dari minimum hingga maksimum ke rentang [0, 255].
* **image\_cropping(image\_array, threshold=128)** → Melakukan cropping citra berdasarkan **image subtraction**, yaitu mengurangi nilai piksel dengan hasil thresholding.
* **histogram\_normalization(image\_array)** → Melakukan normalisasi histogram, yaitu menyebarkan intensitas piksel secara lebih merata agar gambar lebih jelas.

**3. Pengunggahan dan Pemrosesan Gambar**

* Gambar diunggah secara manual dengan files.upload(), kemudian nama file gambar diambil secara otomatis.
* Setelah gambar dibaca dan dikonversi menjadi array, berbagai teknik pengolahan diterapkan:
  + **Thresholding** untuk segmentasi citra.
  + **Negatif citra** untuk membalik warna gambar.
  + **Stretching kontras** untuk meningkatkan perbedaan antara bagian terang dan gelap.
  + **Cropping** untuk menghilangkan bagian tertentu dari gambar.
  + **Normalisasi histogram** untuk menyebarkan distribusi piksel lebih merata.

**4. Visualisasi Hasil Pengolahan**

Setiap hasil pemrosesan citra ditampilkan menggunakan fungsi plot\_images(), yang membandingkan gambar asli dengan hasil pemrosesan. Gambar hasil thresholding, negatif, peningkatan kontras, cropping, dan normalisasi histogram ditampilkan berdampingan dengan gambar asli untuk mempermudah analisis

1. **HASIL ANALISA**
2. Gambar asli dibandingkan dengan hasil pemrosesan menggunakan fungsi plot.
3. Setiap metode menghasilkan efek yang berbeda sesuai dengan tujuan pemrosesan.
4. Perbandingan Implementasi:
5. NumPy lebih cepat karena operasi dilakukan dalam bentuk vektor dan matriks.
6. Implementasi manual lebih lambat karena dilakukan dalam bentuk perulangan, tetapi memberikan pemahaman yang lebih dalam tentang cara kerja algoritma.
7. Contrast Stretching dan Histogram Normalization lebih akurat menggunakan NumPy, karena NumPy memiliki fungsi yang telah dioptimalkan.
8. Metode manual lebih fleksibel untuk dimodifikasi, terutama jika ingin mengembangkan algoritma sendiri tanpa bergantung pada pustaka eksternal.
9. **KESIMPULAN**

Dari tugas ini, dapat disimpulkan bahwa berbagai teknik pengolahan citra seperti thresholding, transformasi negatif, contrast stretching, image subtraction, dan normalisasi histogram dapat digunakan untuk berbagai keperluan analisis dan manipulasi gambar. Implementasi menggunakan NumPy memungkinkan pemrosesan yang cepat dan efisien, sedangkan implementasi manual dengan list 2D memberikan pemahaman lebih mendalam tentang proses pengolahan citra secara mendasar.

Keunggulan NumPy:

1. Proses lebih cepat karena berbasis operasi vektor.
2. Memiliki fungsi bawaan yang mengoptimalkan pengolahan citra.
3. Mudah digunakan dan lebih sedikit kode.

Keunggulan Implementasi Manual:

1. Lebih fleksibel untuk modifikasi algoritma.
2. Memudahkan pemahaman terhadap proses transformasi citra.
3. Berguna dalam kondisi di mana pustaka eksternal tidak tersedia.

Dengan memahami kedua pendekatan ini, kita dapat memilih metode yang paling sesuai tergantung pada kebutuhan aplikasi pengolahan citra yang dihadapi.