Penerapan Algoritma Sequitur Untuk Pemampatan Citra

Dzaky Satrio Nugroho - 13522059[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13522059@std.stei.itb.ac.id, [2]dzakysatrio0122@gmail.com

Abstrak*—Dengan pesatnya perkembangan teknologi di era modern, kebutuhan akan penyimpanan dan transmisi data yang efisien semakin meningkat, khususnya untuk data visual seperti citra digital. Ukuran data citra yang relatif besar dibandingkan dengan jenis data lain, seperti teks, menjadikan proses penyimpanan dan pengirimannya kurang efisien apabila tidak dilakukan pengolahan lebih lanjut. Salah satu solusi untuk mengatasi permasalahan tersebut adalah dengan menerapkan metode pemampatan citra. Metode pemampatan citra secara umum terbagi menjadi dua pendekatan utama, yaitu lossy dan lossless. Pendekatan lossy mampu menghasilkan rasio pemampatan yang tinggi dengan mengorbankan sebagian informasi citra, sedangkan pendekatan lossless mempertahankan seluruh informasi citra sehingga citra hasil pemulihan identik dengan citra asli. Salah satu algoritma yang dapat digunakan dalam pemampatan lossless adalah algoritma Sequitur. Algoritma Sequitur bekerja dengan membangun struktur tata bahasa (grammar) secara dinamis berdasarkan prinsip *digram uniqueness* dan *rule utility*, sehingga pola atau urutan simbol yang berulang dapat direpresentasikan dalam bentuk aturan yang lebih ringkas. Pada pemampatan citra, nilai pixel citra direpresentasikan sebagai rangkaian simbol yang kemudian diproses oleh algoritma Sequitur untuk menemukan pola pengulangan. Semakin banyak pola berulang yang terdapat dalam citra, maka semakin efektif pemampatan yang dihasilkan. Oleh karena itu, kinerja algoritma Sequitur sangat dipengaruhi oleh karakteristik citra, khususnya tingkat redundansi dan variasi nilai pixel. Penerapan algoritma Sequitur diharapkan mampu menghasilkan pemampatan citra yang efisien tanpa kehilangan informasi, sehingga sesuai untuk kebutuhan penyimpanan dan transmisi data citra digital.

***Keywords*—Algoritma, Citra, Citra Digital, Pemampatan, Sequitur.**

I. Pendahuluan

Perkembangan teknologi informasi yang semakin pesat menyebabkan peningkatan signifikan dalam penggunaan dan pertukaran data digital, khususnya data visual berupa citra digital. Citra digital banyak digunakan pada berbagai bidang, seperti multimedia, sistem informasi, pengolahan citra, dan dokumentasi digital. Namun, ukuran data citra yang relatif besar dibandingkan dengan jenis data lain menjadi permasalahan tersendiri dalam hal penyimpanan dan transmisi, terutama pada sistem dengan keterbatasan kapasitas memori dan bandwidth.

Salah satu solusi untuk mengatasi permasalahan tersebut adalah dengan menerapkan teknik pemampatan citra digital. Pemampatan bertujuan untuk mengurangi ukuran data citra tanpa menghilangkan informasi penting yang terkandung di dalamnya. Berdasarkan sifatnya, pemampatan citra dibagi menjadi dua pendekatan utama, yaitu lossy dan lossless. Pendekatan lossless mempertahankan seluruh informasi citra asli sehingga citra hasil penirmampatan identik dengan citra sebelum pemampatan. Pendekatan ini sangat penting untuk aplikasi yang menuntut keakuratan tinggi, seperti citra medis dan citra ilmiah.

Algoritma Sequitur merupakan salah satu algoritma pemampatan lossless berbasis pembentukan tata bahasa yang memanfaatkan pola pengulangan dalam data. Dengan merepresentasikan citra digital sebagai rangkaian nilai pixel, algoritma Sequitur dapat mendeteksi redundansi dan membentuk aturan tata bahasa untuk menghasilkan representasi data yang lebih ringkas. Penelitian ini bertujuan untuk menerapkan algoritma Sequitur pada pemampatan citra digital serta menganalisis pengaruh karakteristik citra terhadap efektivitas hasil pemampatan.

II. Landasan Teori

A. Citra Digital

Citra digital merupakan representasi visual dari suatu objek atau pemandangan yang direkam dan disimpan dalam bentuk data diskrit. Secara matematis, citra digital dapat dipandang sebagai sebuah fungsi dua dimensi f(x,y), di mana x dan y menyatakan koordinat spasial, sedangkan nilai f(x,y) merepresentasikan intensitas cahaya atau tingkat keabuan pada titik tersebut. Berbeda dengan citra analog yang bersifat kontinu, citra digital memiliki nilai koordinat dan intensitas yang terkuantisasi, sehingga dapat disimpan, diproses, dan ditransmisikan menggunakan sistem komputer.

Berdasarkan jenis nilai intensitasnya, citra digital dapat diklasifikasikan menjadi beberapa jenis, antara lain citra biner, citra grayscale, dan citra berwarna. Citra biner hanya memiliki dua kemungkinan nilai pixel, umumnya hitam dan putih. Citra grayscale memiliki satu kanal

dengan tingkat keabuan tertentu, biasanya direpresentasikan dalam rentang 0 hingga 255. Sementara itu, citra berwarna umumnya direpresentasikan menggunakan beberapa kanal warna, seperti model RGB (*Red*, *Green*, *Blue*), di mana setiap pixel terdiri dari kombinasi intensitas pada masing-masing kanal warna.

Setiap citra digital tersusun atas elemen-elemen terkecil yang disebut pixel (*picture element*). Jumlah pixel dalam suatu citra ditentukan oleh resolusi citra, sedangkan kedalaman bit (bit depth) menentukan jumlah tingkat intensitas yang dapat direpresentasikan oleh setiap pixel. Semakin tinggi resolusi dan kedalaman bit suatu citra, maka semakin besar pula ukuran data yang dihasilkan. Hal ini menyebabkan citra digital sering kali membutuhkan ruang penyimpanan yang besar serta bandwidth yang tinggi dalam proses transmisi.

Dalam konteks pemampatan citra, citra digital umumnya direpresentasikan sebagai rangkaian nilai pixel yang disusun secara satu dimensi. Proses ini dikenal sebagai flattening atau linearization, yaitu pengubahan struktur citra dua dimensi menjadi urutan simbol satu dimensi tanpa mengubah nilai pixel. Representasi ini memungkinkan algoritma pemampatan, khususnya algoritma berbasis teks atau simbol seperti Sequitur, untuk mendeteksi pola dan redundansi yang muncul pada urutan nilai pixel. Tingkat redundansi inilah yang sangat mempengaruhi efektivitas algoritma pemampatan lossless terhadap citra digital.

B. Pemampatan Citra Digital

Pemampatan citra digital merupakan proses pengurangan ukuran data citra dengan cara menghilangkan redundansi yang terdapat di dalam representasi citra tersebut, tanpa atau dengan sedikit mengorbankan kualitas visual. Tujuan utama dari pemampatan citra adalah untuk menghemat ruang penyimpanan dan meningkatkan efisiensi transmisi data, terutama pada sistem yang memiliki keterbatasan kapasitas penyimpanan dan bandwidth. Proses pemampatan umumnya terdiri dari dua tahap utama, yaitu tahap pemampatan dan tahap penirmampatan.

Berdasarkan sifatnya, metode pemampatan citra digital dibedakan menjadi dua pendekatan utama, yaitu pemampatan lossless dan pemampatan lossy. Pemampatan lossless memungkinkan citra hasil penirmampatan identik dengan citra asli, sehingga tidak ada informasi yang hilang selama proses pemampatan. Metode ini sangat diperlukan pada aplikasi yang menuntut keakuratan tinggi, seperti citra medis, citra satelit, dan citra ilmiah. Sebaliknya, pemampatan lossy menghilangkan sebagian informasi citra yang dianggap tidak terlalu signifikan secara visual, sehingga mampu menghasilkan rasio pemampatan yang lebih tinggi, namun citra hasil penirmampatan tidak sepenuhnya sama dengan citra asli.

Redundansi pada citra digital merupakan faktor utama yang dimanfaatkan dalam proses pemampatan. Secara umum, redundansi citra dapat diklasifikasikan menjadi redundansi spasial, redundansi statistik, dan redundansi perseptual. Redundansi spasial muncul akibat adanya korelasi antar pixel yang berdekatan, redundansi statistik disebabkan oleh distribusi nilai pixel yang tidak merata, sedangkan redundansi perseptual berkaitan dengan keterbatasan sistem penglihatan manusia dalam membedakan detail tertentu. Metode pemampatan lossless umumnya memanfaatkan redundansi spasial dan statistik, sedangkan metode lossy juga memanfaatkan redundansi perseptual.

Dalam implementasinya, pemampatan citra digital dapat dilakukan dengan berbagai pendekatan, seperti metode berbasis prediksi, transformasi, pengkodean entropi, dan pembentukan tata bahasa. Algoritma berbasis tata bahasa (grammar-based compression), seperti Sequitur, memanfaatkan pola pengulangan pada urutan simbol hasil representasi citra. Dengan merepresentasikan citra sebagai rangkaian nilai pixel satu dimensi, algoritma tersebut dapat membentuk aturan-aturan yang merepresentasikan pola berulang secara lebih ringkas. Efektivitas pemampatan sangat dipengaruhi oleh karakteristik citra, khususnya tingkat redundansi dan variasi nilai pixel yang terkandung di dalamnya.

C. Algoritma Sequitur

Algoritma Sequitur merupakan algoritma pembentukan tata bahasa (grammar-based compression) yang diperkenalkan oleh Nevill-Manning dan Witten. Algoritma ini bertujuan untuk merepresentasikan sebuah urutan simbol dalam bentuk tata bahasa bebas konteks (context-free grammar) yang ringkas dengan memanfaatkan pola pengulangan yang muncul di dalam data. Sequitur banyak digunakan dalam pemampatan data karena kemampuannya dalam mendeteksi dan mengeliminasi redundansi secara otomatis dan efisien.

Algoritma Sequitur bekerja berdasarkan dua prinsip utama, yaitu digram uniqueness dan rule utility. Prinsip digram uniqueness menyatakan bahwa setiap pasangan simbol berurutan (digram) dalam tata bahasa harus bersifat unik. Apabila sebuah digram yang sama muncul lebih dari satu kali, maka algoritma akan membuat sebuah aturan baru untuk menggantikan digram tersebut. Prinsip *rule utility* menyatakan bahwa setiap aturan yang dibentuk harus digunakan minimal dua kali; apabila sebuah aturan hanya digunakan satu kali, maka aturan tersebut akan dihapus dan digantikan kembali dengan isi aturannya.

Proses pembentukan tata bahasa pada algoritma Sequitur dilakukan secara inkremental. Simbol-simbol dari data masukan dibaca satu per satu dan ditambahkan ke dalam urutan utama (start rule). Setiap kali simbol baru ditambahkan, algoritma akan memeriksa keberadaan digram yang terbentuk. Jika digram tersebut sudah ada sebelumnya, maka akan dibuat aturan baru untuk menggantikan digram yang berulang tersebut. Selanjutnya, algoritma akan memastikan bahwa semua

aturan yang terbentuk memenuhi prinsip rule utility.

Dalam konteks pemampatan citra digital, nilai pixel citra direpresentasikan sebagai rangkaian simbol, baik dalam bentuk satu dimensi maupun hasil perataan (flattening) dari citra dua dimensi. Algoritma Sequitur kemudian memproses rangkaian simbol tersebut untuk menemukan pola pengulangan nilai pixel. Semakin tinggi tingkat redundansi pada citra, maka semakin banyak aturan yang dapat dibentuk, sehingga ukuran representasi data menjadi lebih kecil. Dengan sifatnya yang bersifat lossless, algoritma Sequitur mampu mempertahankan seluruh informasi citra asli, sehingga citra hasil penirmampatan identik dengan citra sebelum dilakukan pemampatan.

Sebagai contoh sederhana, misalkan diberikan sebuah rangkaian simbol sebagai berikut:

$$a\ b\ a\ b\ a\ b$$

Langkah-langkah pembentukan tata bahasa menggunakan algoritma Sequitur adalah sebagai berikut:

1. Inisialisasi

   Urutan awal dimasukkan ke dalam aturan awal:

   $$S \rightarrow a\ b$$

2. Penambahan simbol

   $$S \rightarrow a\ b\ a$$
   $$S \rightarrow a\ b\ a\ b$$

   Pada tahap ini ditemukan digram “a b” yang muncul lebih dari satu kali.

3. Penerapan Digram *Uniqueness*

   Karena digram “a b” berulang, maka dibuat aturan baru:

   $$R1 \rightarrow a\ b$$

   Seluruh kemunculan a b digantikan dengan R1, sehingga:

   $$S \rightarrow R1\ R1$$

4. Penambahan simbol selanjutnya

   $$S \rightarrow R1\ R1\ a$$
   $$S \rightarrow R1\ R1\ a\ b$$

   Digram “a b” kembali muncul, sehingga digantikan dengan R1:

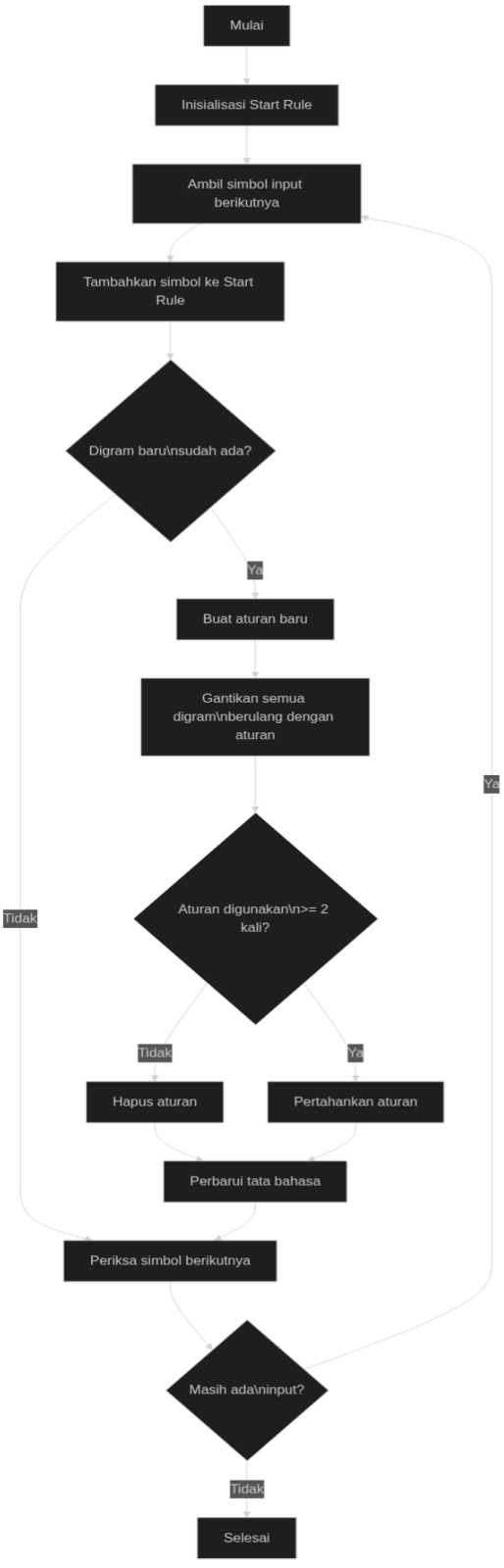
   $$S \rightarrow R1\ R1\ R1$$

5. Penerapan *Rule Utility*

   Aturan R1 digunakan lebih dari satu kali, sehingga aturan tetap dipertahankan.

Hasil akhir nya adalah :

$$S \rightarrow R1\ R1\ R1$$
$$R1 \rightarrow a\ b$$



Gambar 1. Alur Algoritma Pemampatan Sequitur

III. Implementasi Solusi

A. Algoritma Pemampatan Sequitur

Bertujuan untuk mengubah citra menjadi urutan simbol 1D dan menyusun “grammar” berisi aturan untuk pola berulang.

Pada tahap pemampatan , citra dikonversi menjadi array 1D (flatten) dengan metadata yang menyimpan shape, dtype, dan mode. Seluruh elemen urutan dimasukkan ke Rule 0 sebagai terminal ('T', nilai). Proses utama kemudian memindai digram (pasangan simbol berurutan) di semua aturan; setiap digram yang muncul lebih dari sekali dibentukkan aturan baru berisi dua simbol tersebut. Seluruh kemunculan digram yang sama kemudian diganti dengan referensi ('R', id_aturan_baru). Siklus ini diulang sampai tidak ada perubahan lagi atau mencapai batas iterasi untuk mencegah loop tak berhingga.

Fungsi find_and_replace_digrams() melakukan penghitungan semua digram lintas aturan, lalu membuat aturan baru untuk setiap digram yang terdeteksi berulang dan mengganti semua kemunculannya secara serentak (diproses mundur agar indeks aman). Pendekatan ini menyusun grammar yang memadatkan pola berulang secara hierarkis.

```
def compress(self, data):
    for symbol in data:
        self.rules[0].append(('T',
symbol))

    changed = True
    iteration = 0
    max_iterations = 1000

    while changed and iteration <
max_iterations:
        changed =
self.find_and_replace_digrams()
        iteration += 1

    return self.rules

def find_and_replace_digrams(self):
    digram_count = {}

    for rule_id, symbols in
self.rules.items():
        for i in range(len(symbols) -
1):
            digram = (symbols[i],
symbols[i+1])
            if digram not in
digram_count:
                digram_count[digram] =
[]

digram_count[digram].append((rule_id,
i))

    for digram, occurrences in
digram_count.items():
        if len(occurrences) > 1:
            new_rule_id =
self.rule_counter
            self.rule_counter += 1
            self.rules[new_rule_id] =
[digram[0], digram[1]]

            for rule_id, pos in
sorted(occurrences, reverse=True):
                if pos <
len(self.rules[rule_id]) - 1:
                    if
(self.rules[rule_id][pos] == digram[0]
and

self.rules[rule_id][pos + 1] ==
digram[1]):

self.rules[rule_id][pos:pos+2] =
[('R', new_rule_id)]

            return True

    return False
```

B. Algoritma Penirmampatan Sequitur

Bertujuan untuk mengembangkan grammar kembali menjadi urutan simbol asli dengan bantuan metadata citra.

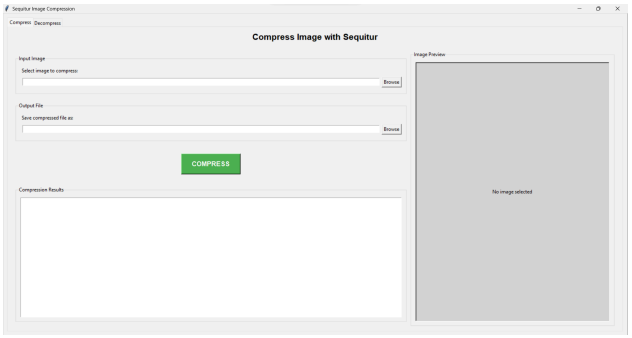
```
def decompress(self, grammar):
    if not grammar or 0 not in
grammar:
        return []

    result = []
    self.expand_rule(grammar, 0,
result, set())
    return result

def expand_rule(self, grammar,
rule_id, result, visited):
    if rule_id in visited:
        return

    if rule_id not in grammar:
        return

    visited.add(rule_id)

    for symbol_type, value in
grammar[rule_id]:
        if symbol_type == 'T':
            result.append(value)
        elif symbol_type == 'R':
            self.expand_rule(grammar,
value, result, visited.copy())
```

Pada penirmampatan , grammar yang dihasilkan pada tahap pemampatan diperluas kembali. Fungsi decompress() memulai dari Rule 0, sementara expand_rule() menelusuri simbol demi simbol: jika ('T', nilai) maka nilai langsung ditambahkan ke urutan hasil; jika ('R', id), maka aturan terkait diekspansi secara rekursif. Untuk *robustness*, terdapat deteksi siklus (melalui visited) agar rekursi tidak berputar jika grammar tidak valid.

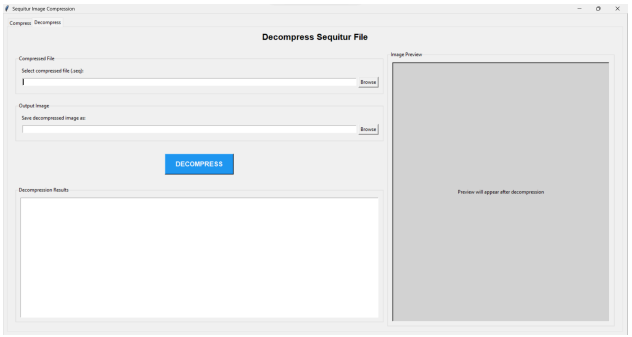
Setelah urutan lengkap dipulihkan, proses membentuk kembali citra dilakukan dengan menyusun ulang urutan 1D ke bentuk aslinya sesuai shape, mengembalikan dtype, dan mode untuk membangun objek citra.

IV. Hasil dan Analisis

Pada bab ini, akan dilakukan pengujian program beserta hasilnya menggunakan beberapa citra grayscale dan berwarna. Berikut adalah tangkapan layar antarmuka (GUI) dari program.



Gambar 2. Tampilan Antarmuka Pemampatan Citra

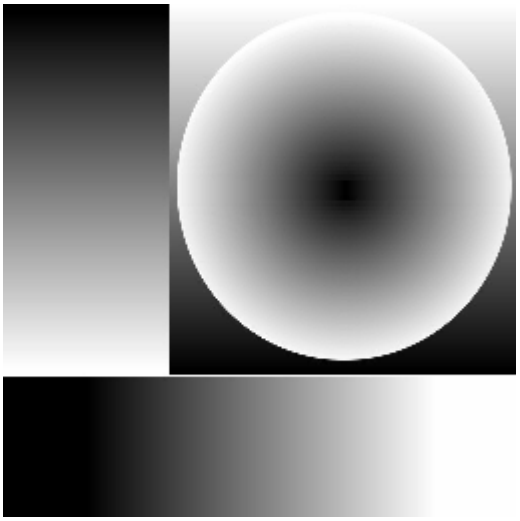


Gambar 3. Tampilan Antarmuka Penirmampatan Citra

Berikut adalah hasil uji dari program dengan beberapa citra:

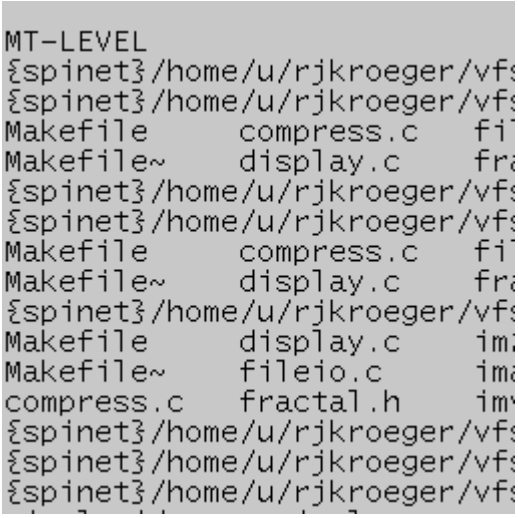
circles.bmp

Original file size: 66,614 bytes (65.05 KB)
Compressed file size: 16,527 bytes (16.14 KB)
Compression ratio: 24.81%
Space saved: 50,087 bytes (48.91 KB)



slope.bmp

Original file size: 66,614 bytes (65.05 KB)
Compressed file size: 299,651 bytes (292.63 KB)
Compression ratio: 449.83%
Space saved: -233,037 bytes (-227.58 KB)



text.bmp

Original file size: 66,614 bytes (65.05 KB)
Compressed file size: 40,990 bytes (40.03 KB)
Compression ratio: 61.53%
Space saved: 25,624 bytes (25.02 KB)

girl-warna.bmp

Original file size: 66,614 bytes (65.05 KB)
Compressed file size: 366,817 bytes (358.22 KB)
Compression ratio: 550.66%
Space saved: -300,203 bytes (-293.17 KB)

Berdasarkan hasil percobaan yang telah dilakukan, penerapan algoritma Sequitur pada berbagai citra digital menunjukkan bahwa sebagian besar citra mengalami peningkatan ukuran data setelah proses pemampatan. Hal ini terjadi karena citra-citra uji memiliki variasi nilai pixel yang relatif tinggi, sehingga pola pengulangan yang dapat dimanfaatkan oleh algoritma Sequitur terbatas. Akibatnya, pembentukan aturan tata bahasa yang dihasilkan tidak cukup efektif untuk mengurangi ukuran representasi data, bahkan justru menambah *overhead* berupa penyimpanan aturan dan simbol tambahan.

Sebaliknya, hasil pemampatan yang lebih kecil dibandingkan ukuran citra asli hanya diperoleh pada citra dengan karakteristik tertentu, yaitu citra yang memiliki banyak pixel bertetangga dengan nilai yang sama atau hampir sama. Citra dengan warna *solid* atau area homogen yang luas menghasilkan urutan nilai pixel dengan tingkat redundansi tinggi. Pada kondisi tersebut, algoritma Sequitur mampu mendeteksi pola pengulangan secara signifikan dan membentuk aturan tata bahasa yang efektif, sehingga ukuran data hasil pemampatan menjadi lebih kecil.

Hasil analisis ini menunjukkan bahwa kinerja algoritma Sequitur sangat bergantung pada karakteristik citra yang diproses. Algoritma ini lebih sesuai untuk citra dengan tingkat redundansi tinggi, seperti citra berwarna solid atau citra dengan sedikit variasi intensitas. Sebaliknya, untuk citra dengan detail kompleks dan variasi pixel yang tinggi, algoritma Sequitur kurang efektif sebagai metode pemampatan citra. Temuan ini menegaskan bahwa pemilihan algoritma pemampatan citra harus mempertimbangkan karakteristik data agar diperoleh hasil pemampatan yang optimal.

V. Kesimpulan dan Saran

Berdasarkan hasil percobaan dan analisis yang telah dilakukan, dapat disimpulkan bahwa algoritma Sequitur belum tentu efektif untuk semua jenis citra digital. Sebagian besar citra uji mengalami peningkatan ukuran data setelah proses pemampatan, terutama pada citra dengan variasi nilai pixel yang tinggi. Pemampatan yang lebih efektif hanya diperoleh pada citra dengan tingkat redundansi tinggi, seperti citra dengan warna solid atau area homogen yang luas, di mana banyak pixel bertetangga memiliki nilai yang sama.

Sebagai saran untuk penelitian selanjutnya, algoritma Sequitur dapat dikombinasikan dengan metode praproses, seperti kuantisasi atau pengelompokan nilai pixel, untuk meningkatkan tingkat redundansi pada citra. Selain itu, penerapan algoritma ini dapat dikaji lebih lanjut pada jenis citra tertentu atau dibandingkan dengan algoritma pemampatan lossless lain guna memperoleh gambaran kinerja yang lebih komprehensif.

Ucapan Terima Kasih

Penulis mengucapkan syukur kepada Tuhan Yang Maha Esa atas rahmatNya yang membuat penulis menyelesaikan makalah ini. Penulis juga mengucapkan terimakasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T. selaku dosen pengajar mata kuliah pemrosesan citra digital tahun ajaran 2025/2026 yang telah membawakan ajaran materi dan bimbingan selama satu semester ini.

Pranala Github

https://github.com/Kizaaaa/sequitur-image-compression

Referensi

[1] Gonzalez, R. C. & Woods, R. E., Digital Image Processing Fourth Edition, Pearson, 2009.

[2] A. McAndrew, A Computational Introduction to Digital Image Processing (2nd ed.), Melbourne: CRC Press, 2016.

[3] E. Buulolo, Sequitur Algorithm For Particular Character Compression Or Words Always Returned, International Journal of Informatics and Computer Science (The IJICS), 2017.

[4] R. Munir, Pemampatan Citra, Bandung: Program Studi Teknik Informatika, 2025.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 24 Desember 2025

Dzaky Satrio Nugroho - 13522059