eISSN : 2621-4970
pISSN : 2621-4962

Volume 7 | No 2
Juli | **2024**

Penerapan *Algoritma Huffman Coding* Dalam Menghemat Ruang Penyimpanan Data Multimedia File (Teks dan Gambar) Berbasis *Python*

Syahril Hasan[1], Saiful Do Abdullah[2], Arisandy Ambarita[3] , M Irfan Amirudin[4]
[1]Program Studi Komputerisasi Akuntansi, [2]Program Studi Teknik Infomatika,
[3,4]Program Studi Manajemen Informatika
[1,3,4]Politeknik Sains & Teknologi Wiratama, [2]Universitas Khairun
aloesz@yahoo.com

Abstrak

Kompresi adalah teknik yang digunakan untuk mengompres data agar sesuai dengan ukuran pada media yang digunakan. seperti backup data, transmisi data, dan keamanan data. Algoritma *Huffman* dalam kompresi teks dapat menghasilkan pengurangan ukuran file yang signifikan tanpa kehilangan informasi, struktur pohon Huffman dan pengkodean karakter memberikan wawasan yang mendalam tentang cara algoritma bekerja. Metode Pengembangan yang gunakan adalah Algoritma Huffman yang di mulai dengan Pengumpulan Frekuensi, Pembuatan *Tree Huffman*, Pembuatan Kode Huffman, Pembuatan Tabel Kompresi, dan Kompresi Data. System ini dibangun dengan menggunakan bahasa pemrogram Phyton yang di terjemahkan pada *Visual Studio Code* yang mempermudah dalam menganalisa logika perhitungannya sehingga menghasilkan sistem kompresi data file teks dan gambar yang dapat menghemat ruang data multimedia file teks dan gambar

Kata kunci: Data kompres teks dan gambar; *Huffman coding*; Aplikasi *python*

Abstract

Compression is a technique used to compress data to match the media size, such as backup Compression is a technique used to compress data to match the size of the media used, such as data backup, data transmission, and data security. Huffman's algorithms in text compression can result in significant file size reductions without losing information, Huffmann's tree structure and character encoding provide in-depth insight into how the algorithm works. The developmental methods used are the Huffman algorithm, which starts with Frequency Collections, Huffman Tree Compression, Huftman Code Composition, Compression Tables, and Data Compression. The system is built using the Phyton programming language translated into Visual Studio Code which makes it easy to analyze the logic of the calculation, resulting in a data compression system of text and image files that can save multimedia data space for text and picture files.

Keywords: *Text and image data compression; Huffman coding; Python Applications*

PENDAHULUAN

Meningkatnya kecanggihan teknologi ditambah dengan meningkatnya perangkat keras dan perangkat lunak, membuatnya lebih mudah untuk menyebarkan informasi dengan cepat ke seluruh dunia melalui Internet. parjito, dkk (2022), Ada beberapa bentuk informasi; dalam bentuk gambar, suara, atau video. Informasi yang diperoleh dengan mudah diakses melalui internet sebagai media komunikasi jenis baru. Namun, tidak semua informasi dapat diakses dengan mudah, Ada beberapa data yang memiliki ukuran besar dan juga dapat

menghambat proses transmisi dan menggunakan area penyimpanan yang sangat besar pada komputer. Untuk memastikan bahwa informasi atau data yang akan ditranskripsikan atau ditulis ulang secara akurat dapat dilakukan dengan cepat, Proses kompresi yang dapat mengurangi kesalahan transkripsi dan mempercepat transmisi data. Marpaung, dkk (2022)

Irwan Wardoyo dkk (2012) mengungkapkan bahwa kompresi digunakan untuk mengubah kumpulan data menjadi kode yang dapat digunakan untuk menghitung jumlah data yang harus dikirim dan diterima, sehingga lebih mudah untuk mengirimkan data. Kompresi data memiliki kemampuan untuk mengurangi waktu transmisi yang ada di medium gelombang lateral. Ada banyak algoritma kompresi data yang dapat digunakan dan berfungsi sesuai kebutuhan. Teknik algoritma kompresi data diantaranya adalah *Huffman Coding, Run-Length Encoding (RLE), Lempel-Ziv (LZ) Compression, Burrows-Wheeler Transform (BWT), Arithmetic Coding* dan masih banyak lagi.

Dengan memahami dasar-dasar kompresi teks dan gambar, kita dapat mengimplementasikan solusi yang dapat meningkatkan efisiensi penggunaan data dalam berbagai konteks aplikasi. Dengan menggunakan bahasa pemrograman Python, kita dapat mengakses berbagai pustaka dan algoritma yang mendukung implementasi solusi kompresi data dengan cepat dan efisien. Proyek implementasi ini bertujuan untuk menyediakan alat yang efektif dalam mengelola dan mengoptimalkan penggunaan data dalam berbagai skenario aplikasi

Rumusan Masalah

Bagaimana Penerapan *Algoritma Huffman Coding* menggunakan bahasa pemrograman phyton dalam menghemat ukuran file (Teks dan Gambar)

Tujuan Penelitian

1. Untuk menerapkan *Algoritma Hufmman Coding* dengan menggunakan bahasa pemrograman *Phyton*
2. Untuk membantu pengguna melakukan Kompresi file teks dan gambar menggunakan *Algoritma Huffman Coding* berbasis *phyton*

Manfaat Penelitian

1. Membantu menghemat ruang penyimpanan pada multimedia text dan gambar.
2. Memberikan pengetahuan kepada pengguna tentang cara kompresi file teks dan gambar menggunakan *Algoritma Huffman Coding* berbasis *Phyton*

Tinjauan Pustaka

Penelitian yang dilakukan Ahmad Rafiqi Et al mengatakan bahwa Qanun mempunyai banyak pasal-pasal dan di dalam pasal terdapat ayat-ayat. Qanun berbentuk cetakan sehingga banyak menghabiskan kertas, File qanun berbentuk digital dengan ukuran file terlalu besar maka membutuhkan teknik untuk mengecilkan file. Dengan melakukan kompresi sehingga dapat menghemat kapasitas penyimpanan, Penilitian lain dilakukan Ahmad Fachriansyah, Muhammad Ali (2024), mengatakan bahwa *Algoritma Huffman* dalam pengkompresian file teks guna mengoptimalkan penggunaan ruang penyimpanan dan mempercepat

proses transfer data. *Algoritma Huffman* bekerja dengan prinsip memberikan representasi kode yang lebih pendek untuk karakter yang muncul lebih sering, sehingga menghasilkan file yang lebih kecil tanpa kehilangan informasi, menurut Hari purwanto Kompresi digunakan untuk berbagai keperluan antara lain: membackup data, transfer data dan salah satu bagian keamanan data, Algoritma Huffman merupakan algoritma kompresi lossless, yaitu Teknik kompresi yang tidak mengubah data aslinya. Hal tersebut yang menyebabkan algoritma ini banyak dipakai dalam proses kompresi. *Algoritma Huffman* bekerja dengan cara melakukan pengkodean dalam bentuk bit untuk mewakili data karakter

LANDASAN TEORI

Huffman Coding

Algoritma David A. Huffman yang dikembangkan pada tahun 1952 dikenal sebagai algoritma Huffman. Algorithma Huffman menggunakan prinsip pengkodean yang mirip dengan kode Morse, artinya setiap karakter (simbol) dikodekan menggunakan beberapa bit; Artinya, karakter yang sering muncul dikodekan menggunakan beberapa bit pendek, dan karakter yang jarang muncul dikodekan menggunakan sedikit lebih panjang (Karisma Mahesa dan Karpen, 2017). Menurut Prayoga (2018) Algoritma Huffman termasuk dalam kelas algoritma yang menggunakan metode statis, atau metode yang selalu menggunakan kode yang sama. Metode ini membutuhkan dua fase: fase pertama digunakan untuk menentukan kemungkinan bahwa setiap simbol akan muncul dan mengidentifikasi jenis codenya, dan fase kedua digunakan untuk mengubah input menjadi kumpulan kode yang akan diterjemahkan.

Menggunakan algoritma Huffman adalah metode yang digunakan untuk kompresi. yaitu teknik kompresi yang menggunakan pengkodean dalam bentuk bit untuk mengidentifikasi data karakter. Irwan Wardoyo, dkk (2012)

Cara kerja atau algoritma metode ini adalah sebagai berikut :

a. menjumlahkan dari banyaknya dan setiap jenis karakter yang ada pada sebuah file.
b. Analisis setiap jenis karakter dengan membandingkan jumlah elemennya dari yang paling sedikit hingga yang paling banyak.
c. Buat pohon biner berdasarkan urutan karakter dari angka yang lebih kecil ke angka yang lebih besar, dan tetapkan kode untuk setiap karakter.
d. Menganalisis data yang ada menggunakan kode bit berdasarkan pohon biner.
e. Menyimpan bit untuk kode bit terbesar, jenis karakter yang diurutkan dari frekuensi keluarnya terbesar ke terkecil beserta data yang sudah berubah menjadi kode bit seperti hasil kompresi.

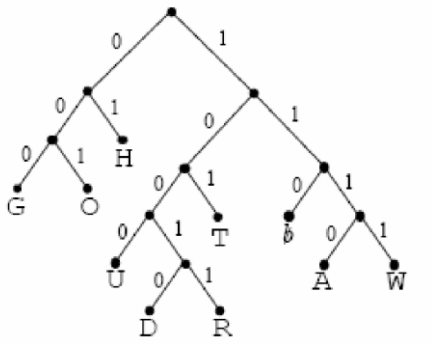
Pohon Huffman

Menurut Huffman, David A (Siti Zuhriyah, 2018)) Pohon Huffman adalah struktur data dengan metode pengkodean Huffman. yang digunakan untuk kompresi data. Pohon Huffman dirancang untuk memberikan representasi kode biner yang efisien untuk setiap simbol dalam set data berdasarkan frekuensinya.

Berikut adalah langkah-langkah umum dalam pembuatan pohon Huffman:

1. Hitung Frekuensi: Hitung frekuensi kemunculan setiap simbol dalam data yang akan dikompresi.
2. Buat Node: Buat node untuk setiap simbol, di mana setiap node memiliki frekuensi kemunculan sebagai nilai
3. Gabungkan Node: Gabungkan dua node dengan frekuensi terendah untuk membuat pohon baru. Frekuensi node baru adalah jumlah dari dua node yang digabung.
4. Ulangi: Ulangi langkah 3 hingga hanya ada satu node tersisa, yang akan menjadi akar pohon Huffman.
5. Assign Kode: Berikan representasi kode biner untuk setiap simbol berdasarkan jalur dari akar ke simpul itu. Misalnya, satu jalur bisa diwakili oleh kode 0 dan jalur lainnya oleh kode 1.

Pohon Huffman menghasilkan kode biner yang memiliki panjang berbeda untuk setiap simbol, dengan simbol yang lebih sering muncul memiliki kode yang lebih pendek. Hal ini memungkinkan penghematan ruang saat data dikompres, Fandi Susanto (2009), berikut gambar pohon *huffman*



Gambar.1 Pohon *Huffman* (Arya Tri Prabawa,2007-2008)

Aplikasi *Python*

Python adalah bahasa pemrograman yang populer akhir-akhir ini. Guido van Rossum menciptakan *Python*, pada tahun 1991. Python dapat digunakan untuk mengembangkan aplikasi web (server), membuat aplikasi serta pengembangan perangkat lunak (*software*), meningkatkan persamaan pada matematika, membuat sistem skrip, dan pemrograman mikrokontroler (*Micro-Python*). Beberapa fungsi dalam *Python* adalah dapat digunakan pada server untuk membuat aplikasi web, dapat dikombinasikan dengan perangkat lunak untuk membuat jadwal kerja, dapat berkomunikasi dengan sistem database, dan pengguna berbahasa *Python* dapat membaca dan memodifikasi file. Selain itu, *Python* dapat digunakan untuk membuat prototipe dengan cepat atau untuk mengintegrasikan perangkat lunak dan siap proses produksi..

Python adalah bahasa pemrograman yang dapat digunakan untuk tujuan umum, khususnya untuk membuat kode sumber yang mudah dibaca. Selain itu, *Python* memiliki library lengkap yang memungkinkan programmer untuk membuat aplikasi yang mutahir menggunakan kode sumber yang tampak sederhana. (Ljubomir Perkovic, 2012).

METODE PENELITIAN

Perancangan *Algoritma Huffman Coding* melibatkan beberapa langkah, termasuk pengumpulan frekuensi, pembuatan pohon Huffman, pembuatan kode Huffman, dan pembuatan tabel *Huffman.* Berikut adalah tahapan-tahapan umum dalam kompresi data dengan *Huffman Coding*, Sugara, Purboyo, dkk. (2018).

Pengumpulan Frekuensi:

a. Menghitung frekuensi kemunculan setiap simbol dalam data yang akan dikompresi.
b. Membuat daftar frekuensi simbol-simbol dan menyusunnya dalam urutan dari yang sudah sering dan yang tidak sering muncul.

Pembuatan Tree Huffman:

a. Membangun pohon Huffman dari daftar frekuensi yang telah dihasilkan.
b. Pohon ini dibangun dengan memilih dua simpul dengan frekuensi terendah, menggabungkannya menjadi satu simpul dengan frekuensi yang merupakan jumlah dari kedua simpul tersebut, dan menambahkan simpul hasil ke daftar frekuensi.
c. Proses ini diulangi hingga hanya tersisa satu simpul sebagai akar pohon Huffman.

Pembuatan Kode Huffman:

a. Menetapkan kode biner unik untuk setiap simbol berdasarkan jalur yang ditempuh dari akar pohon ke simpul yang mewakili simbol tersebut.
b. Simbol yang lebih sering muncul mendapatkan kode yang lebih pendek, sementara simbol yang lebih jarang mendapatkan kode yang lebih panjang.

Pembuatan Tabel Kompresi:

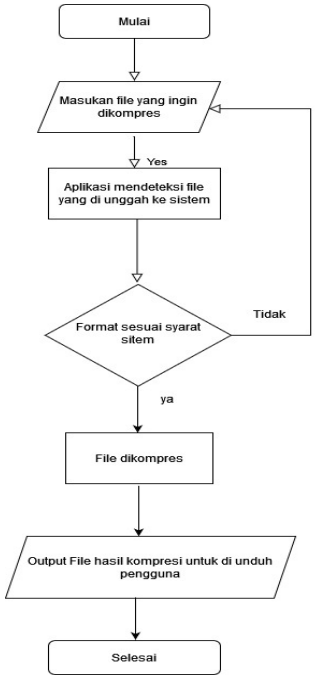
a. Membuat tabel yang memetakan setiap simbol ke representasi biner yang sesuai berdasarkan kode Huffman yang telah dibuat.
b. Tabel ini akan digunakan saat kompresi dan dekompresi untuk mengonversi simbol-simbol ke dan dari representasi biner Huffman.

Kompresi Data:

a. Mengonversi data asli ke representasi biner Huffman menggunakan tabel kompresi yang telah dibuat.
b. Menyusun bit-bit ini menjadi bentuk yang lebih padat, sesuai dengan kode Huffman.

Selama proses dekompresi, pohon Huffman yang sama digunakan untuk mengonversi kembali representasi biner ke simbol-simbol asli. Huffman Coding memberikan efisiensi dalam hal ukuran file terkompresi karena simbol-simbol yang lebih sering muncul diberikan representasi biner yang lebih pendek

Skema langkah penelitian merupakan awal dari tahap perencanaan dan analisis proses kompresi dibuat berdasarkan alur langkah-langkah penelitian. Langkah-langkah penelitian ini ditunjukkan pada Gambar berikut



Gambar 2. Diagram Alir Penelitian

Flowchart Diagram usulan sistem sebagai berikut

a. Mulai awal Pengguna masuk ke aplikasi

b. Pengguna mengunggah file asli yang akan dikompresi.
c. Sistem deteksi file yang akan di kompresi, jika file format sesuai maka file akan di lanjutkan dengan kompresi, Jika tidak, akan berakhir dengan selesai seperti pada simbol selesai
d. Sistem menghasilkan output file hasil kompresi, pengguna mengunduh file tersebut dan selesai

IMPLEMENTASI DAN PEMBAHASAN

Perhitungan Algoritma Huffman Coding

Pengkodean dengan huffman coding menggunakan panjang bit yang bervariasi dalam pengkodean sebuah karakter. Karakter dengan frekuensi kemunculan lebih besar memiliki panjang bit yang lebih pendek. Berlaku kebalikannya. Berikut Langkah-langkah Algoritma Huffman

1. Hitung statistik (frekuensi) jumlah kemunculan masing-masing simbol.
2. Simpan hasil informasi bobot masing-masing simbol.
3. Membangun pohon huffman berdasarkan larik bobot dari masing-masing simbol.
4. Konversi pohon huffman menjadi kode spesifik untuk tiap simbol.

Perhitungan manual Algoritma Huffman dapat di terapkan dalam program dalam penelitian ini menggunakan Bahasa pemograman python. Python mendukung program untuk dijalankan sebab termasuk kategori bahasa tingkat tinggi yang juga mempermudah manusia dalam analisa logika perhitungannya.

Proses dekompresi dilakukan dengan menggunakan tabel kode untuk mengembalikan data ke bentuk aslinya. Berikut Langkah-langkah perancangan yang sesuai dengan hasil perhitungan huffman coding dalam aplikasi python pada teks adalah sebagai berikut :

1. Penghitungan Frekuensi adalah Pada langkah ini, data yang ingin dikompresi diubah menjadi representasi frekuensi kemunculan setiap simbol. Dalam implementasi Python, digunakan *Counter*(data) untuk melakukan ini. Counter adalah sebuah objek yang membantu menghitung jumlah kemunculan setiap elemen dalam data. Misalnya, jika data adalah string " AABCABC", *Counter*(data) akan menghasilkan kamus yang berisi { 'A': 3, 'B': 2,'C': 2,}, yang menunjukkan bahwa huruf 'A' muncul tiga kali, 'B' munculdua kali, dan seterusnya.
2. Membangun Antrian Prioritas adalah Setelah frekuensi kemunculan setiap simbol diketahui, langkah selanjutnya adalah membangun antrian prioritas. Antrian prioritas digunakan untuk menyimpan node-node yang mewakili simbol-simbol dan frekuensinya dalam urutan prioritas. Dalam Python, antrian prioritas dapat diimplementasikan menggunakan modul heapq. Dalam kode, setiap node direpresentasikan oleh kelas Node, dan setelah itu dimasukkan ke dalam antrian prioritas menggunakan heapq.heapify.
3. Membangun Pohon Huffman adalah Dalam langkah ini, kita membangun pohon Huffman dari node-node yang ada dalam antrian prioritas. Algoritma Huffman Coding menggunakan pendekatan greedy, yaitu menggabungkan dua node dengan frekuensi terkecil pada setiap langkah. Terus menerus melakukan ini hingga hanya tersisa satu node dalam antrian prioritas, yang akan menjadi root dari

pohon Huffman.

4. Membuat Tabel Kode adalah Setelah pohon Huffman dibangun, selanjutnya membuat tabel kode. Tabel kode ini mengaitkan setiap simbol dengan kode binernya yang sesuai dalam pohon Huffman. Dalam menggunakan rekursi yaitu melewati pohon Huffman dari root ke setiap daun, sambil mengumpulkan kode biner yang sesuai untuk setiap simbol.
5. Kompresi dan Dekompresi adalah Setelah tabel kode dibuat, data asli bisa dikompresi dengan mengganti setiap simbol dengan kode binernya yang sesuai dari tabel kode. Ini menghasilkan data yang lebih kompak. Untuk dekompresi, dapat menggunakan tabel kode yang sama untuk mengonversi kembali kode biner ke simbol asli menggunakan pohon Huffman yang sama.

Berdasarkan Langkah Langkah diatas berikut perancangan pemograman Huffman coding menggunakan Bahasa pemograman phyton adalah sebagai berikut

```
import heapq
from collections import defaultdict, Counter

class Node:
    def __init__(self, char, freq):
        self.char = char
        self.freq = freq
        self.left = None
        self.right = None

    def __lt__(self, other):
        return self.freq < other.freq

def build_huffman_tree(text):
    freq_map = Counter(text)
    heap = [Node(char, freq) for char, freq in freq_map.items()]
```

Gambar 3. Tampilan pemograman Huffman coding menggunakan pyhton (1)

```
def build_huffman_tree(text):
    heapq.heapify(heap)

    while len(heap) > 1:
        left = heapq.heappop(heap)
        right = heapq.heappop(heap)

        merged = Node(None, left.freq + right.freq)
        merged.left = left
        merged.right = right

        heapq.heappush(heap, merged)

    return heap[0]

def build_huffman_codes(node, prefix='', codes={}):
```

Gambar 4. Tampilan pemograman Huffman coding menggunakan pyhton (2)

```
def build_huffman_codes(node, prefix='', codes={}):
    if node is not None:
        if node.char is not None:
            codes[node.char] = prefix
        build_huffman_codes(node.left, prefix + '0', codes)
        build_huffman_codes(node.right, prefix + '1', codes)
    return codes

def encode_text(text, codes):
    encoded_text = ''.join([codes[char] for char in text])
    return encoded_text

def decode_text(encoded_text, tree):
    decoded_text = ''
    current_node = tree
```

Gambar 5. Tampilan pemograman Huffman coding menggunakan pyhton (3)

```
    text = "AABCABC"

    huffman_tree = build_huffman_tree(text)
    huffman_codes = build_huffman_codes(huffman_tree)

    print("Huffman Codes:")
    for char, code in huffman_codes.items():
        print(f"{char}: {code}")

    encoded_text = encode_text(text, huffman_codes)
    print("\nEncoded Text:")
    print(encoded_text)

    decoded_text = decode_text(encoded_text, huffman_tree)
    print("\nDecoded Text:")
```

Gambar 6. Tampilan pemograman Huffman coding menggunakan pyhton (4)

Setelah program perancangan dengan hasil perhitungan huffman coding dalam aplikasi phyton kemudian program di jalankan (*Running*) maka hasilnya dapat dilihat pada gambar

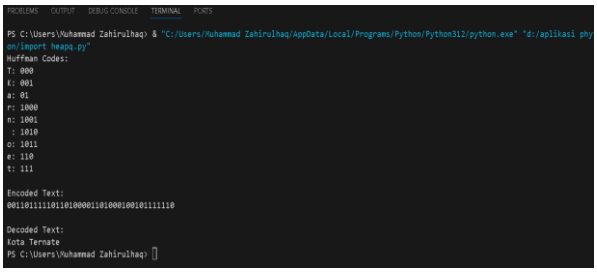
```
PS C:\Users\Muhammad Zahirulhaq> & "C:/Users/Muhammad Zahirulhaq/AppData/Local/Programs/Python/Python312/python.exe" "d:/aplikasi phyton/import heapq.py"
Huffman Codes:
A: 0
C: 10
B: 11

Encoded Text:
00111001110

Decoded Text:
AABCABC
PS C:\Users\Muhammad Zahirulhaq>
```

Gambar 7. Hasil Setelah di *Running* (1)

Pada gambar 7 menujukkan hasil Huffman coding dari string " **AABCABC**", yang menghasilkan Encoded Text: **00111001110**



Gambar 8. Hasil Setelah di *Running* (2)

Pada gambar 8 menujukkan hasil Huffman coding dari string " **Kota Ternate**", yang menghasilkan Encoded Text:**000001011000100110101011110111**

1. Fungsi Huffman_encode dan Huffman_decode, pada struktur codingan diatas adalah mengambil string atau text yang digunakan adalah " **AABCABC**", setelah itu menghitung karakter yang sama dan menyimpan jumlah karakter ke encode_text. Return atau mengembalikan string asli hasil kompresi.
2. Fungsi Huffman_decode, yaitu mengambil string pada encode_text (yang sudah di kompresi) sebagai input lalu mengakses karakter dan jumahnya dari encoded_text untuk menghasilkan string yang telah didekompresi, fungsi selanjutnya mengembalikan decode_text

Aplikasi pyhton pada gambar

Huffman coding adalah metode pengompresian yang digunakan untuk mengurangi ukuran data dengan menggantikan simbol-simbol yang sering muncul dengan kode biner yang lebih pendek dan simbol-simbol yang jarang muncul dengan kode biner yang lebih panjang. Berikut adalah langkah-langkah perancangan menerapkan Huffman coding pada gambar dengan Python:

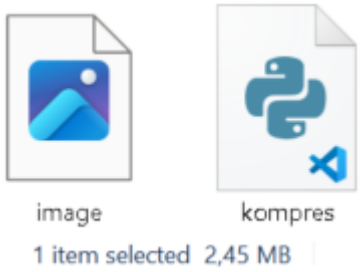
1. Baca gambar adalah Gunakan pustaka seperti OpenCV atau PIL untuk membaca gambar.
2. Hitung frekuensi kemunculan setiap nilai piksel adalah Hitung berapa kali setiap nilai piksel muncul dalam gambar. Ini akan memberi Anda informasi tentang probabilitas kemunculan setiap nilai piksel.
3. Bangun pohon Huffman adalah Gunakan frekuensi kemunculan untuk membangun pohon Huffman. Di sini, setiap nilai piksel akan menjadi node dalam pohon, dan frekuensi kemunculannya akan menjadi bobotnya.
4. Buat tabel kode adalah Lakukan transversal pada pohon Huffman untuk menghasilkan kode biner untuk setiap nilai piksel.
5. Kode gambar adalah Gunakan tabel kode untuk mengkodekan gambar. Gantikan setiap nilai piksel dengan kode binernya.
6. Simpan informasi adalah Simpan tabel kode dan kode gambar yang dihasilkan
7. Dekompresi (jika diperlukan) adalah Buat fungsi untuk mendekompresi gambar dari kode biner ke bentuk semula.

Berdasarkan Langkah Langkah diatas berikut perancangan pemograman Huffman coding menggunkan Bahasa pemograman python pada gambar atau foto ditunjukkan pada gambar 9 pemograman aplikasi pythonpada gambar

```
kompres.py X
kompres.py > ...
1  from PIL import Image
2  img = Image.open('image.jpg')
3
4  img.save('optimize-image.jpg', optimize_=True, Quality_=10)
```

Gambar 9. pemograman aplikasi python pada gambar/foto

foto yang dikompres adalah berukuran 2,45 MB dapad dilihat pada gambar 10 foto asli2,45 MB kemudian di jalankan untuk dikompres foto menjadi kecil



Gambar 10. foto asli 2,45 MB

Pada gambar 11 menunjukkan hasil setelah dijalnkan foto yang sudah dikompres berhasildan disimpan di file kompres foto

gambar 11. Hasil pemograman yang sudah dijalankan

setelah pemograman dijalankan maka hasil foto yang dikompres menjadi 529 KB

Gambar 12. foto dikompres 529 KB

Berdasarkan perancangan yang telah dilakukan dengan menggunakan aplikasi phyton dapat digunakan untuk Huffman coding pada representasi data teks dan gambar/foto

Fungsi *Huffman_encode* dan *Huffman_decode*, pada struktur codingan diatas adalah mengambil string atau text yang digunakan adalah " AABCABC", setelah itu menghitung karakter yang sama dan menyimpan jumlah karakter ke encode_text. Return atau mengembalikan string asli hasil kompresi. Sedangkan Fungsi Huffman_decode, yaitu mengambil string pada encode_text (yang sudah di kompresi) sebagai input lalu mengakses karakter dan jumahnya dari encoded_text untuk menghasilkan string yang telah didekompresi, fungsi selanjutnya mengembalikan decode_text

Sedangkan Fungsi quantize_image (image, levels): Melakukan kuantisasi warna pada gambar, Fungsi compress_image(image): Mengompres gambar dengan menerapkan algoritma kompresi yang dipilih, pada Pustaka Python:

PIL (Python Imaging Library): Digunakan untuk memanipulasi gambar, termasuk membaca dan menyimpan gambar, dan numpy: Digunakan untuk manipulasi matriks dalam proses kuantisasi gambar

Ucapan Terima Kasih

Ucapan Terima Kasih Kepada Lembaga Layanan Pendidikan Tinggi Wilayah XII LLDIKTI telah memberikan dukungan Hibah Penelitian Dosen Pemula Tahun 2024

KESIMPULAN

1. Penerapan Algoritma Huffman Coding sistem kompresi data file teks dan gambar dibuat dengan bahasa pemrograman Python, Phyton dapat digunakan untuk pembuatan prototipe

dengan cepat, atau untuk pengembangan perangkat lunak siap produksi

2. Penerapan Algoritma Huffman Coding dapat menghemat ruang data multimedia file teks dan gambar

DAFTAR PUSTAKA

Ahmad Rafiqi, Nelly Astuti Hasibuan, Imam Saputra, *Perancangan Aplikasi Untuk Kompresi Ayat-Ayat Di Dalam Qanun Aceh Menerapkan Algoritma Huffman*, RESOLUSI : Rekayasa Teknik Informatika dan Informasi, Vol 1, No 6, Juli 2021 Hal 346-353

Ahmad Fachriansyah, Muhammad Ali (2024) *Penerapan Algoritma Huffman Pada Pengkompresian File Text*, VIRTUAL: Jurnal Teknik Informatika dan Komputer, Volume 1, No. 2, Mei 2024, Page 1−7, ISSN: 3047-7271 (Media Online), https://jurnal.vublish.id/index.php/jurtikom/index

Hari Purwanto (2015) *Penerapan Algoritma Huffman Pada Kompresi File Wave*, Jsi Jurnal Sistem Informasi, Vol 2 No 2 2015

Fischer, J., & Schindler, A. (2018). *A Guide to the JPEG Image Compression Standard*. Springer.

Irwan Wardoyo, Peri Kusdinar, Irvan Hasbi Taufik. (2012). *Kompresi Teks dengan Menggunakan Algoritma Huffman*. Jurnal Ilmiah, Vol. 5 No.2 (Fakultas Teknologi Informasi Institut Teknologi Sepuluh November (its) Surabaya), 3-6

Fandi Susanto, *Aplikasi Penggambar Pohon Biner Huffman Untuk Data Teks*, Algoritma Jurnal Ilmiah STMIK GI MDP Volume 5 Nomor 1, Maret 2009

Huffman, David A., 1952. A *Method for the Construction of Minimum-Redundancy Codes, Proceedings of I.R.E.* (September 1952)

Python Software Foundation. (2023). Python 3 Documentation. https://docs.python.org/3/

Pillow Documentation. (2023). https://pillow.readthedocs.io/en/stable/

Prayoga, Suryaningrum. (2018) "*Implementasi Algoritma Huffman dan Run Length Encoding Pada Aplikasi Kompresi Berbasis Web*" Vol, 4, no 2

Parjito, P. J., Rahmawati, O., & Ulum, F. (2022). *Rancang Bangun Aplikasi E-Agribisnis Untuk Meningkatkan Penjualan Hasil Tanaman Hortikultura*. Jurnal Informatika dan Rekayasa Perangkat Lunak, 3(3), 354-365.

Sayood, K. (2002). *Introduction to Data Compression. Morgan Kaufmann*.

Sugara, Purboyo, dkk. (2018). "*Implementasi dan Analisis Efektifitas Huffman Dan Discrete Cosine Transform Pada Berbagai Jenis Citra Digital*" Vol, 4, no 2

Witten, I. H., Moffat, A., & Bell, T. C. (1999). *Managing Gigabytes: Compressing and Indexing Documents and Images*.