

Laporan Tugas Kecil 2 Strategi Algoritma

Faqih Muhammad Syuhada - 13523057

Muhammad Izzat Jundy - 13523092

1. Algoritma

Program ini menggunakan algoritma quadtree untuk mengompresi gambar. Quadtree adalah sebuah algoritma yang membagi sebuah blok menjadi 4 bagian yang lebih kecil terus-menerus hingga pada ukuran blok atau tingkat kesamaan warna setiap piksel tertentu, kemudian menyamakan warna seluruh piksel pada blok tersebut menjadi warna rata-rata pada blok tersebut.

Dalam program ini, algoritma quadtree diimplementasikan sebagai berikut.

a. Algoritma Quadtree

Urutan eksekusi algoritma quadtree yang diimplementasikan mengikuti algoritma *breadth first search*.

1. Membuat queue sebagai tree dengan setiap vertice-nya merupakan sebuah Node dengan atribut integer x, y, currentSize, dan depthOfVertice.
2. Selagi queue tidak kosong, dilakukan langkah 3 hingga 9.
3. Mendapatkan x, y, currentSize, dan depthOfVertice simpul terkini.
4. Mendapatkan rata-rata nilai dari masing-masing *Blue*, *Green*, dan *Red* pada blok tersebut.

5. Mendapatkan ukuran dari kandidat blok baru, yaitu 4 blok lebih kecil dari pembagian blok terkini.
6. Memeriksa apakah ukuran kandidat block baru masih lebih dari atau sama dengan *minimum block size*, dan apakah warna blok baru dianggap kurang seragam.
7. Jika keduanya iya, pembagian tersebut jadi dilakukan dan menge-push ke queue sebanyak 4 kali: untuk blok kiri atas, blok kanan atas, blok kiri bawah, dan blok kanan atas.
8. Jika salah satunya tidak, dilakukan penyeragaman warna setiap piksel pada blok terkini.
9. Pelepasan simpul terkini (melanjutkan ke simpul selanjutnya).

b. Algoritma Rata-rata Warna

1. Menghitung total nilai dari masing-masing *Blue*, *Green*, dan *Red* pada setiap piksel dalam blok.
2. Menghitung total piksel pada blok.
3. Mengembalikan total nilai *Blue*, *Green*, dan *Red* yang masing-masing telah dibagi dengan total piksel pada blok dalam bentuk tipe data BGR.

c. Algoritma Memeriksa Tingkat Keseragaman

1. Memeriksa apa metode pengukuran perbedaan warna yang digunakan, alias *error measurement method*.
 - a. Metode 1, *variance*. Implementasi sederhana dari persamaan menggunakan pengulangan.
 - b. Metode 2, *mean absolute deviation*. Implementasi sederhana dari persamaan menggunakan pengulangan.

- c. Metode 3, *max pixel difference*. Implementasi sederhana dari persamaan menggunakan pengulangan.
 - d. Metode 4, *entropy*. Implementasi sederhana dari persamaan menggunakan pengulangan. Untuk mendapatkan probabilitas level setiap warna, dilakukan perhitungan frekuensi setiap level warna terlebih dahulu.
2. Jika besar eror yang diperoleh lebih besar atau sama dengan *threshold* (dari parameter, di-*input* oleh pengguna), blok tersebut dianggap tidak cukup seragam.

2. Source Code

Program ini dibuat secara modular menggunakan bahasa pemrograman C++ dengan library OpenCV untuk memproses gambar. Struktur file-nya adalah sebagai berikut.

```
Tucil2_13523092/
├── bin/
├── doc/
└── src/
    ├── Function/
    │   ├── imagecompression.cpp
    │   └── imagecompression.hpp
    ├── IO/
    │   ├── input.cpp
    │   ├── input.hpp
    │   ├── output.cpp
    │   └── output.hpp
    └── main.cpp
└── test/
```

a. Main.cpp

```
#include "./IO/input.hpp"
#include "./IO/output.hpp"
#include "./Function/imagecompression.hpp"
#include <opencv2/opencv.hpp>
#include <opencv2/highgui.hpp>
#include <chrono>
using namespace std::chrono;

int main() {

    int cmd, errorMeasurementMethod, minBlockSize;
    double threshold, compressionPercentage;
    string addressOld, addressNew;
    cv::Mat image, imageCompressed;

    while(1){

        cmd = input_home();

        if(cmd == 0){
            goodbye();
            break;
        }

        addressOld = input_image_address_import();
        errorMeasurementMethod = input_error_measurement_method();
        threshold = input_threshold(errorMeasurementMethod);
        minBlockSize = input_minimum_block_size();
        compressionPercentage = input_compression_percentage();

        image = cv::imread(addressOld, cv::IMREAD_COLOR);
        // cv::imshow("Gambar", image);
        // cv::waitKey(0);

        auto start = high_resolution_clock::now();

        // BEGIN COMPRESSION
        imageCompressed = compress_image(image, addressOld,
errorMeasurementMethod, threshold, minBlockSize, compressionPercentage);
        // END COMPRESSION
    }
}
```

```

        auto stop = high_resolution_clock::now();
        auto duration = duration_cast<milliseconds>(stop - start);

        compression_succeed();

        addressNew = input_image_address_export();
        cv::imwrite(addressNew, imageCompressed);

        save_succeed();

        process_information(duration.count(), addressOld, addressNew);

        // cv::imshow("Gambar Terkompres", imageCompressed);
        // cv::waitKey(0);

    }

    return 0;
}

```

b. Function/imagecompression.hpp

```

#ifndef _IMAGECOMPRESSION_HPP
#define _IMAGECOMPRESSION_HPP

#include <bits/stdc++.h>
#include <opencv2/opencv.hpp>
#include <opencv2/highgui.hpp>
#include <fstream>
#include <filesystem>
using namespace std;

extern int depth, verticesCount;

struct Node{
    int x, y, currWidth, currHeight, depthOfVertice;
};

cv::Mat compress_image(cv::Mat image, string address, int
errorMeasurementMethod, double threshold, int minBlockSize, double
compressionPercentage);

```

```
#endif
```

c. Function/imagecompression.cpp

```
#include "imagecompression.hpp"

int depth, verticesCount;

// "PRIVATE"
cv::Vec3b average_color(const cv::Mat& image, int x, int y, int width,
int height){
    cv::Scalar sum = cv::sum(image(cv::Rect(x, y, width, height)));
    int totalPixels = width * height;
    return cv::Vec3b(sum[0] / totalPixels, sum[1] / totalPixels, sum[2]
/ totalPixels);
}

bool is_uniform(const cv::Mat& image, int x, int y, int width, int
height, int errorMeasurementMethod, double threshold, cv::Vec3b
averageColor){

    if(errorMeasurementMethod == 1){

        unsigned long long sumB = 0, sumG = 0, sumR = 0;
        for(int i = y; i < y + height; i++){
            for(int j = x; j < x + width; j++){
                cv::Vec3b pixel = image.at<cv::Vec3b>(i, j);
                sumB += pow((pixel[0] - averageColor[0]), 2);
                sumG += pow((pixel[1] - averageColor[1]), 2);
                sumR += pow((pixel[2] - averageColor[2]), 2);
            }
        }

        if((sumB + sumG + sumR) / (3*width*height) >= threshold){
            return false;
        }

    }else if(errorMeasurementMethod == 2){

        unsigned long long sumB = 0, sumG = 0, sumR = 0;
        for(int i = y; i < y + height; i++){
            for(int j = x; j < x + width; j++) {
```

```

        cv::Vec3b pixel = image.at<cv::Vec3b>(i, j);
        sumB += abs((int) pixel[0] - averageColor[0]);
        sumG += abs((int) pixel[1] - averageColor[1]);
        sumR += abs((int) pixel[2] - averageColor[2]);
    }
}

if((sumB + sumG + sumR) / (3 * width * height) >= threshold) {
    return false;
}

} else if(errorMeasurementMethod == 3) {

    int maxB = 0, maxG = 0, maxR = 0, minB = 255, minG = 255, minR = 255;
    for(int i = y; i < y + height; i++) {
        for(int j = x; j < x + width; j++) {
            cv::Vec3b pixel = image.at<cv::Vec3b>(i, j);
            if(maxB < pixel[0]) maxB = pixel[0];
            if(maxG < pixel[1]) maxG = pixel[1];
            if(maxR < pixel[2]) maxR = pixel[2];
            if(minB > pixel[0]) minB = pixel[0];
            if(minG > pixel[1]) minG = pixel[1];
            if(minR > pixel[2]) minR = pixel[2];
        }
    }

    if((maxB + maxG + maxR - minB - minG - minR) / 3 >= threshold) {
        return false;
    }
}

} else if(errorMeasurementMethod == 4) {
    int freq[256][3];
    for(int i = 0; i < 256; i++) {
        freq[i][0] = 0;
        freq[i][1] = 0;
        freq[i][2] = 0;
    }

    for(int i = y; i < y + height; i++) {
        for(int j = x; j < x + width; j++) {
            cv::Vec3b pixel = image.at<cv::Vec3b>(i, j);
            freq[pixel[0]][0] += 1;

```

```

        freq[pixel[1]][1] += 1;
        freq[pixel[2]][2] += 1;
    }
}

double hB = 0, hG = 0, hR = 0;
for(int i = y; i < y + height; i++){
    for(int j = x; j < x + width; j++) {
        cv::Vec3b pixel = image.at<cv::Vec3b>(i, j);
        double pB = (double) freq[pixel[0]][0]/(width*height);
        double pG = (double) freq[pixel[1]][1]/(width*height);
        double pR = (double) freq[pixel[2]][2]/(width*height);
        hB += (-1) * pB * log2(pB);
        hG += (-1) * pG * log2(pG);
        hR += (-1) * pR * log2(pR);
    }
}

if((hB + hG + hR) / 3 >= threshold) {
    return false;
}
}

return true;
}

void quadtree(cv::Mat& image, int width, int height, int
errorMeasurementMethod, double threshold, int minBlockSize){
queue<Node> q;
q.push({0, 0, width, height, 0}); // x, y, currWidth, currHeight,
depthOfVertice

while(!q.empty()) {
    Node temp = q.front();

    int x = temp.x;
    int y = temp.y;
    int currWidth = temp.currWidth;
    int currHeight = temp.currHeight;
    int depthOfVertice = temp.depthOfVertice;
    cv::Vec3b averageColor = average_color(image, x, y, currWidth,
currHeight);

    verticesCount++;
}
}

```

```

        if(depthOfVertice > depth) {
            depth = depthOfVertice;
        }

        int halfWidth = currWidth / 2;
        int halfHeight = currHeight / 2;
        if(halfWidth*halfHeight >= minBlockSize && !is_uniform(image, x,
y, currWidth, currHeight, errorMeasurementMethod, threshold,
averageColor)){
            q.push({x, y, halfWidth, halfHeight, depthOfVertice+1});
            q.push({x + halfWidth, y, currWidth - halfWidth, halfHeight,
depthOfVertice+1});
            q.push({x, y + halfHeight, halfWidth, currHeight -
halfHeight, depthOfVertice+1});
            q.push({x + halfWidth, y + halfHeight, currWidth -
halfWidth, currHeight - halfHeight, depthOfVertice+1});
        } else{
            for(int i = y; i < y + currHeight; i++){
                for(int j = x; j < x + currWidth; j++){
                    image.at<cv::Vec3b>(i, j) = averageColor;
                }
            }
        }
    }

    q.pop();
}

}

// "PUBLIC"
cv::Mat compress_image(cv::Mat image, string address, int
errorMeasurementMethod, double threshold, int minBlockSize, double
compressionPercentage){

    cout << "\n";
    cout << "Mengkompresi..." << endl;

    verticesCount = 0;
    depth = 0;

    if(compressionPercentage == 0){
        quadtree(image, image.cols, image.rows, errorMeasurementMethod,
threshold, minBlockSize);
    }
}

```

```

} else{
    double l = 0.0, r, mid;
    int i;
    if(errorMeasurementMethod == 1){
        r = 10000.0;
    } else if(errorMeasurementMethod == 2){
        r = 255.0;
    } else if(errorMeasurementMethod == 3){
        r = 765.0;
    } else if(errorMeasurementMethod == 4){
        r = 24.0;
    }

    ifstream originalImage(address, ios::binary | ios::ate);
    streamsize imageSize = originalImage.tellg();
    originalImage.close();

    cv::Mat temp = image.clone();
    cv::Mat optImage = temp.clone();
    double min = 1.0;
    i = 0;
    while(l < r && i < 20){

        temp = image.clone();
        mid = (l+r)/2.0;

        verticesCount = 0;
        depth = 0;
        quadtree(temp, temp.cols, temp.rows, errorMeasurementMethod,
mid, minBlockSize);

        cv::imwrite("tempImage.png", temp);
        ifstream tempImage("tempImage.png", ios::binary | ios::ate);
        streamsize tempSize = tempImage.tellg();
        tempImage.close();
        system("del tempImage.png");

        double tempCompressionRatio = (1.0 - (double) tempSize /
(double) imageSize);
        cout << "\rKompresi " << i+1 << " dengan threshold " << mid
<< ":" << tempCompressionRatio*100 << "%" << flush;
        if(tempCompressionRatio - compressionPercentage > -0.005 &&
tempCompressionRatio - compressionPercentage < 0.005){
    }
}

```

```

        break;
    } else if(tempCompressionRatio < compressionPercentage) {
        l = mid + 0.000001;
    } else{
        r = mid;
    }

    if(fabs(tempCompressionRatio - compressionPercentage) <
min) {
        min = fabs(tempCompressionRatio -
compressionPercentage);
        optImage = temp.clone();
    }

    i+=1;
}

if(i != 20) image = temp.clone();
else image = optImage.clone();
}

return image;
}

```

d. IO/input.hpp

```

#ifndef _INPUT_HPP
#define _INPUT_HPP

#include <bits/stdc++.h>
#include <fstream>
#include <filesystem>
using namespace std;
namespace fs = std::filesystem;

int input_home();
string input_image_address_import();
int input_error_measurement_method();
double input_threshold(int errorMeasurementMethod);
int input_minimum_block_size();

```

```
double input_compression_percentage();
string input_image_address_export();

#endif
```

e. IO/input.cpp

```
#include "input.hpp"
#include "output.hpp"

// "PRIVATE"
bool isValidPhoto(string s){
    string format = s.substr(s.find_last_of(".") + 1);

    vector<string> a = {"jpg", "jpeg", "png", "bmp", "tiff", "tif",
"webp", "gif", "jp2"};

    int i = 0;
    while(i < a.size()){
        if(format == a[i]) return true;
        i+=1;
    }

    return false;
}

// "PUBLIC"
int input_number(string s, int dari, int sampai){
    string cmd;

    while(1){
        cout << s << endl;
        cout << "> ";

        try{
            cin >> cmd;
            int n = stoi(cmd);

            if(n >= dari && n <= sampai){
                return n;
            }else{
                cout << endl;
                cout << "Input tidak sesuai." << endl;
            }
        }catch(...){
            cout << endl;
            cout << "Input tidak sesuai." << endl;
        }
    }
}
```

```
        }
    } catch(...){
        cout << endl;
        cout << "Input tidak sesuai." << endl;
    }
}

int input_home(){
    string cmd;

    while(1){
        set_home();

        try{
            cin >> cmd;
            int n = stoi(cmd);

            if(n >= 0 && n <= 1){
                return n;
            }else{
                cout << endl;
                cout << "Input tidak sesuai." << endl;
            }
        }catch(...){
            cout << endl;
            cout << "Input tidak sesuai." << endl;
        }
    }
}

string input_image_address_import(){
    string s;

    getchar();
    while(1){
        set_image_address();

        getline(cin, s);
        if(fs::exists(s) && isValidPhoto(s)) return s;
        else image_not_found();
    }
}
```

```
int input_error_measurement_method() {
    string cmd;

    while(1){
        set_error_measurement_method();

        try{
            cin >> cmd;
            int n = stoi(cmd);

            if(n >= 1 && n <= 4) {
                return n;
            }else{
                cout << endl;
                cout << "Input tidak sesuai." << endl;
            }
        }catch(...){
            cout << endl;
            cout << "Input tidak sesuai." << endl;
        }
    }
}

double input_threshold(int errorMeasurementMethod) {
    string cmd;

    while(1){
        set_threshold();

        try{
            cin >> cmd;
            double n = stod(cmd);

            if(errorMeasurementMethod == 1 && n >= 0){
                return n;
            }else if(errorMeasurementMethod == 2 && n >= 0 && n <= 255){
                return n;
            }else if(errorMeasurementMethod == 3 && n >= 0 && n <= 765){
                return n;
            }else if(errorMeasurementMethod == 4 && n >= 0 && n <= 24){
                return n;
            }else{

```

```
        cout << endl;
        cout << "Metode pengukuran eror yang dipilih tidak
men-support nilai threshold tersebut." << endl;
    }
} catch(...){
    cout << endl;
    cout << "Input tidak sesuai." << endl;
}
}

int input_minimum_block_size(){
string cmd;

while(1){
    set_minimum_block_size();

    try{
        cin >> cmd;
        int n = stoi(cmd);

        if(n >= 1){
            return n;
        }else{
            cout << endl;
            cout << "Input tidak sesuai." << endl;
        }
    } catch(...){
        cout << endl;
        cout << "Input tidak sesuai." << endl;
    }
}
}

double input_compression_percentage(){
string cmd;

while(1){
    set_compression_percentage();

    try{
        cin >> cmd;
        double n = stod(cmd);
    }
}
```

```

        if(n >= 0 && n <= 1) {
            return n;
        }else{
            cout << endl;
            cout << "Input tidak sesuai." << endl;
        }
    }catch(...){
        cout << endl;
        cout << "Input tidak sesuai." << endl;
    }
}

string input_image_address_export(){
    string s;

    getchar();
    while(1){
        set_image_address();

        getline(cin, s);
        if(isValidPhoto(s)) return s;
        else cout << "Tolong input alamat lengkap gambar baru dengan benar.\n";
    }
}

```

f. IO/output.hpp

```

#ifndef _OUTPUT_HPP
#define _OUTPUT_HPP

#include <bits/stdc++.h>
#include <opencv2/opencv.hpp>
#include <opencv2/imgproc.hpp>
#include <opencv2/highgui.hpp>
#include <fstream>
#include <filesystem>
using namespace std;

void set_home();
void set_image_address();

```

```

void set_error_measurement_method();
void set_threshold();
void set_minimum_block_size();
void set_compression_percentage();

void goodbye();
void image_not_found();
void compression_succeed();
void save_succeed();

void process_information(long long duration, string addressOld, string
addressNew);

#endif

```

g. IO/output.cpp

```

#include "output.hpp"
#include "../Function/imagecompression.hpp"

void set_home() {
    cout << "\n";
    cout << "Selamat datang!\n";
    cout << "Program ini dapat membantu mengkompresi gambar. Ingin
mencoba?\n";
    cout << "1. Iya\n";
    cout << "0. Keluar\n";
    cout << "> ";
}

void set_image_address(){
    cout << "\n";
    cout << "Masukkan alamat lengkap file: ";
}

void set_error_measurement_method(){
    cout << "\n";
    cout << "Silakan pilih metode pengukuran error berikut!\n";
    cout << "1. Variance\n";
    cout << "2. Mean Absolute Deviation\n";
    cout << "3. Max Pixel Difference\n";
    cout << "4. Entropy\n";
    cout << "> ";
}

```

```
}
```

```
void set_threshold(){
    cout << "\n";
    cout << "Silakan masukkan nilai ambang batas!\n";
    cout << "> ";
}
```

```
void set_minimum_block_size(){
    cout << "\n";
    cout << "Silakan masukkan nilai minimum dari ukuran blok!\n";
    cout << "> ";
}
```

```
void set_compression_percentage(){
    cout << "\n";
    cout << "Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)\n";
    cout << "> ";
}
```

```
void goodbye(){
    cout << "\n";
    cout << "Sampai jumpa!\n";
    cout << "Terima kasih telah menggunakan layanan kami.\n";
}
```

```
void image_not_found(){
    cout << "\n";
    cout << "Gambar tidak ditemukan.\n";
    cout << "Catatan: Kami hanya menerima file dengan format .jpg, .jpeg, .png, .bmp, .tiff, .tif, .webp, .gif, dan .jp2.\n";
}
```

```
void compression_succeed(){
    cout << "\n";
    cout << "Gambar berhasil dikompresi!\n";
    cout << "Di mana Anda ingin menyimpannya?\n";
}
```

```
void save_succeed(){
    cout << "\n";
    cout << "Gambar berhasil disimpan!\n";
}
```

```
}
```

```
void process_information(long long duration, string addressOld, string
addressNew) {
    cout << "\n";
    cout << "Waktu eksekusi kompresi\t: " << duration << " ms\n";
    ifstream oldImage(addressOld, ios::binary | ios::ate);
    streamsize oldSize = oldImage.tellg();
    oldImage.close();

    ifstream newImage(addressNew, ios::binary | ios::ate);
    streamsize newSize = newImage.tellg();
    newImage.close();

    cout << "Ukuran gambar sebelum\t: " << oldSize << " bytes\n";
    cout << "Ukuran gambar sesudah\t: " << newSize << " bytes\n";
    double compressionRatio = ((float) newSize) / ((float) oldSize);
    cout << "Persentase kompresi\t: " << (1.0 - compressionRatio) *
100.0 << "%\n";
    cout << "Kedalaman simpul\t: " << depth << endl;
    cout << "Banyak simpul pada pohon: " << verticesCount << endl;
}
```

3. Tangkapan Layar Input Output

a. Gambar 1

```
Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-nicole-avagliano-1132392-2749481.jpg

Silakan pilih metode pengukuran error berikut!
1. Variance
2. Mean Absolute Deviation
3. Max Pixel Difference
4. Entropy
> 4

Silakan masukkan nilai ambang batas!
> 20

Silakan masukkan nilai minimum dari ukuran blok!
> 20

Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)
> 0.8

Mengkompresi...
sh: 1: del: not found
Kompresi 1 dengan threshold 12: -450.972%sh: 1: del: not found
Kompresi 2 dengan threshold 18: -422.076%sh: 1: del: not found
Kompresi 3 dengan threshold 21: -411.355%sh: 1: del: not found
Kompresi 4 dengan threshold 22.5: -406.124%sh: 1: del: not found
Kompresi 5 dengan threshold 23.25: -403.603%sh: 1: del: not found
Kompresi 6 dengan threshold 23.625: -402.551%sh: 1: del: not found
Kompresi 7 dengan threshold 23.8125: -402.153%sh: 1: del: not found
Kompresi 8 dengan threshold 23.9063: -401.708%sh: 1: del: not found
Kompresi 9 dengan threshold 23.9531: -401.405%sh: 1: del: not found
Kompresi 10 dengan threshold 23.9766: -401.368%sh: 1: del: not found
Kompresi 11 dengan threshold 23.9883: -401.336%sh: 1: del: not found
Kompresi 12 dengan threshold 23.9941: -401.318%sh: 1: del: not found
Kompresi 13 dengan threshold 23.9971: -401.303%sh: 1: del: not found
Kompresi 14 dengan threshold 23.9985: -401.3%sh: 1: del: not found
Kompresi 15 dengan threshold 23.9993: -401.296%sh: 1: del: not found
Kompresi 16 dengan threshold 23.9996: -401.296%sh: 1: del: not found
Kompresi 17 dengan threshold 23.9998: -401.296%sh: 1: del: not found
Kompresi 18 dengan threshold 23.9999: -401.296%sh: 1: del: not found
Kompresi 19 dengan threshold 24: -401.296%sh: 1: del: not found
Kompresi 20 dengan threshold 24: -401.296%
Gambar berhasil dikompresi!
Di mana Anda ingin menyimpannya?

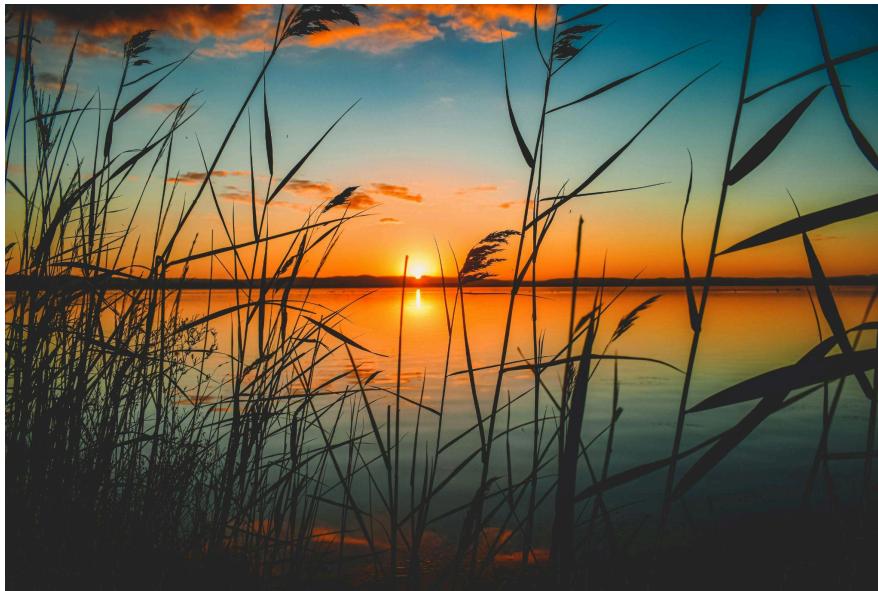
Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-nicole-avagliano.jpg

Gambar berhasil disimpan!

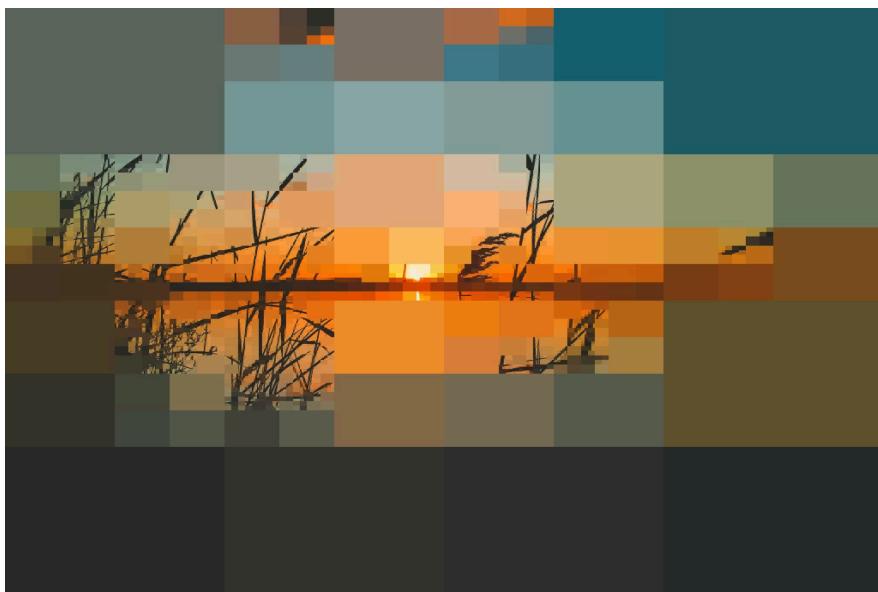
Waktu eksekusi kompresi : 363866 ms
Ukuran gambar sebelum   : 1463598 bytes
Ukuran gambar sesudah  : 2894786 bytes
Persentase kompresi     : -97.7856%
Kedalaman simpul       : 10
Banyak simpul pada pohon: 943417

Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
```

Sebelum:



Sesudah



b. Gambar 2

```
Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 1

Masukkan alamat lengkap file: /mnt/d/LifeOS/Tucil2_13523057_13523092/test/bernadya-apamungkin.png

Silakan pilih metode pengukuran error berikut!
1. Variance
2. Mean Absolute Deviation
3. Max Pixel Difference
4. Entropy
> 1

Silakan masukkan nilai ambang batas!
> 20

Silakan masukkan nilai minimum dari ukuran blok!
> 1

Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)
> 0.4172

Mengkompresi...
sh: 1: del: not found
Kompresi 1 dengan threshold 5000: 98.1951%sh: 1: del: not found
Kompresi 2 dengan threshold 2500: 98.1951%sh: 1: del: not found
Kompresi 3 dengan threshold 1250: 86.2988%sh: 1: del: not found
Kompresi 4 dengan threshold 625: 80.6518%sh: 1: del: not found
Kompresi 5 dengan threshold 312.5: 69.4817%sh: 1: del: not found
Kompresi 6 dengan threshold 156.25: 56.1888%sh: 1: del: not found
Kompresi 7 dengan threshold 78.125: 41.4368%
Gambar berhasil dikompresi!
Di mana Anda ingin menyimpannya?

Masukkan alamat lengkap file: /mnt/d/LifeOS/Tucil2_13523057_13523092/test/bernadya-compressed.png

Gambar berhasil disimpan!

Waktu eksekusi kompresi : 3524 ms
Ukuran gambar sebelum   : 358860 bytes
Ukuran gambar sesudah  : 210160 bytes
Persentase kompresi     : 41.4368%
Kedalaman simpul       : 10
Banyak simpul pada pohon: 35489
```

Sebelum :



Bernadya

Sesudah :



Bernadya

c. Gambar 3

```
Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-simon73-1183099.jpg

Silakan pilih metode pengukuran error berikut!
1. Variance
2. Mean Absolute Deviation
3. Max Pixel Difference
4. Entropy
> 2

Silakan masukkan nilai ambang batas!
> 20

Silakan masukkan nilai minimum dari ukuran blok!
> 20

Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)
> 0.3

Mengkompresi...
sh: 1: del: not found
Kompresi 1 dengan threshold 127.5: 96.8637%sh: 1: del: not found
Kompresi 2 dengan threshold 63.75: 96.8637%sh: 1: del: not found
Kompresi 3 dengan threshold 31.875: 96.8637%sh: 1: del: not found
Kompresi 4 dengan threshold 15.9375: 65.7236%sh: 1: del: not found
Kompresi 5 dengan threshold 7.96875: -36.4075%sh: 1: del: not found
Kompresi 6 dengan threshold 11.9531: 29.4721%sh: 1: del: not found
Kompresi 7 dengan threshold 13.9453: 48.8054%sh: 1: del: not found
Kompresi 8 dengan threshold 12.9492: 39.6307%sh: 1: del: not found
Kompresi 9 dengan threshold 12.4512: 39.6307%sh: 1: del: not found
Kompresi 10 dengan threshold 12.2021: 39.6307%sh: 1: del: not found
Kompresi 11 dengan threshold 12.0776: 39.6307%sh: 1: del: not found
Kompresi 12 dengan threshold 12.0154: 39.6307%sh: 1: del: not found
Kompresi 13 dengan threshold 11.9843: 29.4721%sh: 1: del: not found
Kompresi 14 dengan threshold 11.9998: -129.657%sh: 1: del: not found
Kompresi 15 dengan threshold 12.0076: 39.6307%sh: 1: del: not found
Kompresi 16 dengan threshold 12.0037: 39.6307%sh: 1: del: not found
Kompresi 17 dengan threshold 12.0018: 39.6307%sh: 1: del: not found
Kompresi 18 dengan threshold 12.0008: 39.6307%sh: 1: del: not found
Kompresi 19 dengan threshold 12.0003: 39.6307%sh: 1: del: not found
Kompresi 20 dengan threshold 12.0001: 39.6307%
Gambar berhasil dikompresi!
Di mana Anda ingin menyimpannya?

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-simon73-1183099.jpg

Gambar berhasil disimpan!

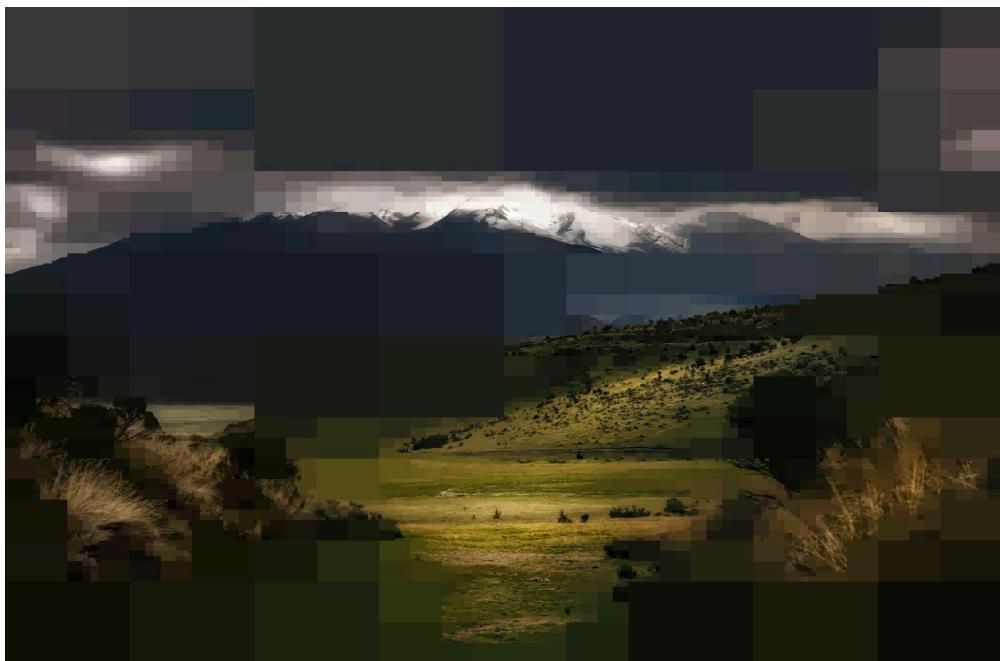
Waktu eksekusi kompresi : 94518 ms
Ukuran gambar sebelum : 2141048 bytes
Ukuran gambar sesudah : 2141048 bytes
Persentase kompresi : 0%
Kedalaman simpul : 10
Banyak simpul pada pohon: 100909

Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
```

Sebelum :



Sesudah :



d. Gambar 4

```
Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 1

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-pixabay-206359.jpg

Silakan pilih metode pengukuran error berikut!
1. Variance
2. Mean Absolute Deviation
3. Max Pixel Difference
4. Entropy
> 4

Silakan masukkan nilai ambang batas!
> 10

Silakan masukkan nilai minimum dari ukuran blok!
> 10

Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)
> 0.1

Mengkompresi...
sh: 1: del: not found
Kompresi 1 dengan threshold 12: -333.12%sh: 1: del: not found
Kompresi 2 dengan threshold 18: -265.048%sh: 1: del: not found
Kompresi 3 dengan threshold 21: -231.654%sh: 1: del: not found
Kompresi 4 dengan threshold 22.5: -216.456%sh: 1: del: not found
Kompresi 5 dengan threshold 23.25: -208.595%sh: 1: del: not found
Kompresi 6 dengan threshold 23.625: -204.932%sh: 1: del: not found
Kompresi 7 dengan threshold 23.8125: -203.295%sh: 1: del: not found
Kompresi 8 dengan threshold 23.9063: -202.121%sh: 1: del: not found
Kompresi 9 dengan threshold 23.9531: -201.93%sh: 1: del: not found
Kompresi 10 dengan threshold 23.9766: -201.649%sh: 1: del: not found
Kompresi 11 dengan threshold 23.9883: -201.53%sh: 1: del: not found
Kompresi 12 dengan threshold 23.9941: -201.493%sh: 1: del: not found
Kompresi 13 dengan threshold 23.9971: -201.477%sh: 1: del: not found
Kompresi 14 dengan threshold 23.9985: -201.466%sh: 1: del: not found
Kompresi 15 dengan threshold 23.9993: -201.466%sh: 1: del: not found
Kompresi 16 dengan threshold 23.9996: -201.465%sh: 1: del: not found
Kompresi 17 dengan threshold 23.9998: -201.465%sh: 1: del: not found
Kompresi 18 dengan threshold 23.9999: -201.465%sh: 1: del: not found
Kompresi 19 dengan threshold 24: -201.465%sh: 1: del: not found
Kompresi 20 dengan threshold 24: -201.461%
Gambar berhasil dikompresi!
Di mana Anda ingin menyimpannya?

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-pixabay.jpg

Gambar berhasil disimpan!
```

```
Waktu eksekusi kompresi : 47527 ms
Ukuran gambar sebelum   : 352318 bytes
Ukuran gambar sesudah  : 695176 bytes
Persentase kompresi     : -97.3149%
Kedalaman simpul       : 9
Banyak simpul pada pohon: 92341

Selamat datang!
Program ini dapat membantu mengompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 0

Sampai jumpa!
Terima kasih telah menggunakan layanan kami.
```

Sebelum :



Sesudah :



e. Gambar 5

```
Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 1

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-katja-79053-592077.jpg

Silakan pilih metode pengukuran error berikut!
1. Variance
2. Mean Absolute Deviation
3. Max Pixel Difference
4. Entropy
> 3

Silakan masukkan nilai ambang batas!
> 10

Silakan masukkan nilai minimum dari ukuran blok!
> 10

Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)
> 0.8

Mengkompresi...
sh: 1: del: not found
Kompresi 1 dengan threshold 382.5: 96.3717%sh: 1: del: not found
Kompresi 2 dengan threshold 191.25: 88.6211%sh: 1: del: not found
Kompresi 3 dengan threshold 95.625: -39.8656%sh: 1: del: not found
Kompresi 4 dengan threshold 143.438: 48.6808%sh: 1: del: not found
Kompresi 5 dengan threshold 167.344: 81.0452%sh: 1: del: not found
Kompresi 6 dengan threshold 155.391: 75.6357%sh: 1: del: not found
Kompresi 7 dengan threshold 161.367: 78.3711%sh: 1: del: not found
Kompresi 8 dengan threshold 164.355: 79.5973%
Gambar berhasil dikompresi!
Di mana Anda ingin menyimpannya?

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-katja.jpg

Gambar berhasil disimpan!

Waktu eksekusi kompresi : 12089 ms
Ukuran gambar sebelum   : 1063313 bytes
Ukuran gambar sesudah  : 325784 bytes
Persentase kompresi    : 69.3614%
Kedalaman simpul      : 10
Banyak simpul pada pohon: 1389

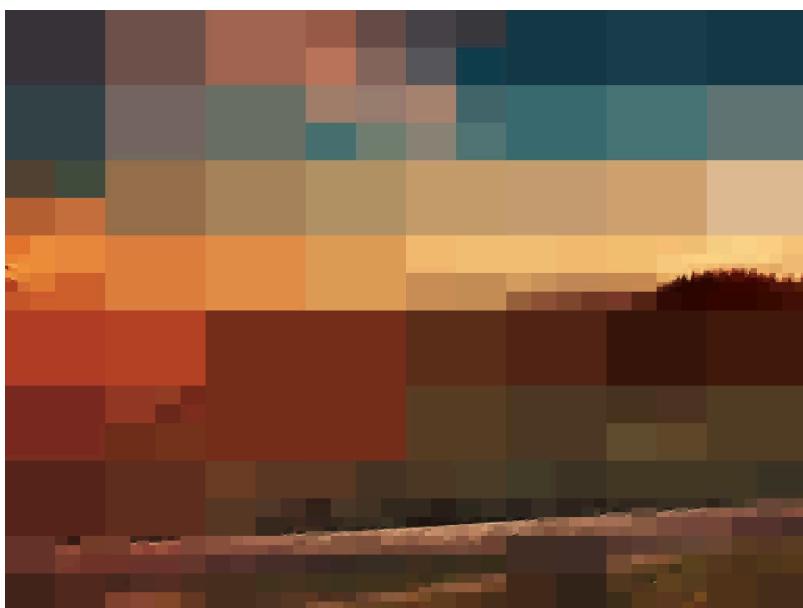
Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 0

Sampai jumpa!
Terima kasih telah menggunakan layanan kami.
```

Sebelum :



Sesudah :



f. Gambar 6

```
Selamat datang!
Program ini dapat membantu mengompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 1

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-freestockpro-2166559.jpg

Silakan pilih metode pengukuran error berikut!
1. Variance
2. Mean Absolute Deviation
3. Max Pixel Difference
4. Entropy
> 2

Silakan masukkan nilai ambang batas!
> 10

Silakan masukkan nilai minimum dari ukuran blok!
> 10

Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)
> 0.3

Mengompresi...
sh: 1: del: not found
Kompresi 1 dengan threshold 127.5: 95.6541%sh: 1: del: not found
Kompresi 2 dengan threshold 63.75: 94.6674%sh: 1: del: not found
Kompresi 3 dengan threshold 31.875: 75.5268%sh: 1: del: not found
Kompresi 4 dengan threshold 15.9375: -84.6043%sh: 1: del: not found
Kompresi 5 dengan threshold 23.9063: 36.3947%sh: 1: del: not found
Kompresi 6 dengan threshold 19.9219: -12.6923%sh: 1: del: not found
Kompresi 7 dengan threshold 21.9141: 15.3015%sh: 1: del: not found
Kompresi 8 dengan threshold 22.9102: 28.6304%sh: 1: del: not found
Kompresi 9 dengan threshold 23.4082: 36.3947%sh: 1: del: not found
Kompresi 10 dengan threshold 23.1592: 36.3947%sh: 1: del: not found
Kompresi 11 dengan threshold 23.0347: 36.3947%sh: 1: del: not found
Kompresi 12 dengan threshold 22.9724: 28.6304%sh: 1: del: not found
Kompresi 13 dengan threshold 23.0035: 36.3947%sh: 1: del: not found
Kompresi 14 dengan threshold 22.988: 28.6304%sh: 1: del: not found
Kompresi 15 dengan threshold 22.9958: 28.6304%sh: 1: del: not found
Kompresi 16 dengan threshold 22.9997: 28.6304%sh: 1: del: not found
Kompresi 17 dengan threshold 23.0016: 36.3947%sh: 1: del: not found
Kompresi 18 dengan threshold 23.0006: 36.3947%sh: 1: del: not found
Kompresi 19 dengan threshold 23.0001: 36.3947%sh: 1: del: not found
Kompresi 20 dengan threshold 22.9999: 28.6304%
Gambar berhasil dikompresi!
Di mana Anda ingin menyimpannya?

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-freestockpro.jpg

Gambar berhasil disimpan!
```

```
Waktu eksekusi kompresi : 67094 ms
Ukuran gambar sebelum   : 2083814 bytes
Ukuran gambar sesudah   : 1218932 bytes
Persentase kompresi      : 41.5048%
Kedalaman simpul        : 10
Banyak simpul pada pohon: 33233

Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 0

Sampai jumpa!
Terima kasih telah menggunakan layanan kami.
```

Sebelum:



Sesudah :



g. Gambar 7

```
Selamat datang!
Program ini dapat membantu mengkompresi gambar. Ingin mencoba?
1. Iya
0. Keluar
> 1

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/pexels-maxfrancis-2246476.jpg

Silakan pilih metode pengukuran error berikut!
1. Variance
2. Mean Absolute Deviation
3. Max Pixel Difference
4. Entropy
> 1

Silakan masukkan nilai ambang batas!
> 20

Silakan masukkan nilai minimum dari ukuran blok!
> 20

Silakan masukkan nilai compression dari 0 sampai 1.0 (0% - 100%)
> 0.5

Mengkompresi...
sh: 1: del: not found
Kompresi 1 dengan threshold 5000: 95.4367%sh: 1: del: not found
Kompresi 2 dengan threshold 2500: 91.9333%sh: 1: del: not found
Kompresi 3 dengan threshold 1250: 78.1797%sh: 1: del: not found
Kompresi 4 dengan threshold 625: 31.0179%sh: 1: del: not found
Kompresi 5 dengan threshold 937.5: 57.2223%sh: 1: del: not found
Kompresi 6 dengan threshold 781.25: 49.9571%
Gambar berhasil dikompresi!
Di mana Anda ingin menyimpannya?

Masukkan alamat lengkap file: /mnt/d/Downloads/Gambar/photo-1506104489822-562ca25152fe.jpg

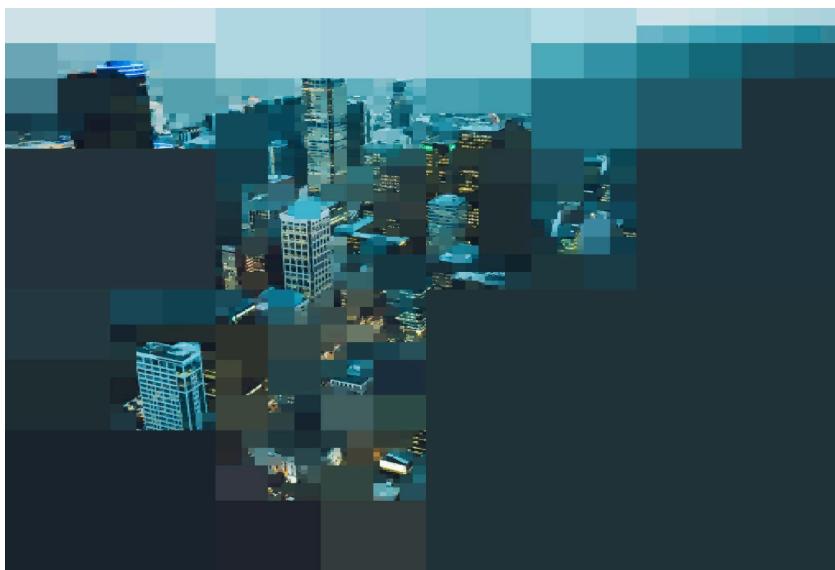
Gambar berhasil disimpan!

Waktu eksekusi kompresi : 40816 ms
Ukuran gambar sebelum   : 1556935 bytes
Ukuran gambar sesudah  : 1035324 bytes
Persentase kompresi     : 33.5024%
Kedalaman simpul       : 10
Banyak simpul pada pohon: 18173
```

Sebelum :



Sesudah :



4. Analisis Algoritma

Algoritma Quadtree yang diimplementasikan menerapkan *divide and conquer* dengan pembagi 4. Fungsi `average_color()` memiliki kompleksitas

waktu $O((currentSize)^2)$. Sementara fungsi `is_uniform()`, setiap metodenya memiliki kompleksitas waktu $O((currentSize)^2)$. Terdapat skenario terburuk berupa setiap piksel perlu dieksekusi. Dengan setiap level d perlu dieksekusi sebanyak 4^d kali. Deret geometri yang terbentuk adalah sebagai berikut.

$$T = 4^0 * 2 * size^2 / 2^{2^0} + 4^1 * 2 * size^2 / 2^{2^1} + \dots + 4^d * 2 * size^2 / 2^{2^d}$$

$$T = 2^{2^0} * 2 * size^2 / 2^{2^0} + 2^{2^1} * 2 * size^2 / 2^{2^1} + \dots + 2^{2^d} * 2 * size^2 / 2^{2^d}$$

$$T = 2 * size^2 + 2 * size^2 + \dots + 2 * size^2 \text{ (sebanyak } d+1\text{)}$$

$$T = (d+1) * 2 * size^2 = d * 2 * size^2 + 2 * size^2$$

Jika persamaan di atas dituliskan dalam notasi Big-O, diperoleh kompleksitas algoritma $O(d * size^2)$.

5. Penjelasan Implementasi Bonus

a. Target Persentase Kompresi

Pencarian *threshold* yang dapat menghasilkan hasil kompresi yang diinginkan dilakukan dengan menggunakan algoritma *binary search tree*. Pengujian dilakukan dari *threshold* dengan nilai tengah antara 0 dengan sebuah nilai sesuai dengan metode pengukuran eror yang dipilih. Lalu berjalan seperti algoritma *binary search tree* pada umumnya. Untuk menentukan apakah nilai persentase kompresi sudah cukup dekat dengan yang ditargetkan, selisih absolut dari nilai persentase kompresi simpul terkini dengan persentase kompresi yang ditargetkan dibandingkan dengan 0.005 (0.5%), apakah kurang dari atau tidak. Untuk memastikan program tidak berada dalam *infinite loop*, dibuat batas pengulangan sebanyak 20. Untuk setiap pengujian, image yang nilai persentase kompresinya paling dekat dengan target akan disimpan. Jika pengulangan berhenti karena persentase sudah cukup dekat, fungsi akan mengembalikan image terakhir yang diperiksa. Sementara jika pengulangan berhenti karena pengulangan sudah

terjadi sebanyak 20 kali, fungsi akan mengembalikan image yang persentase kompresinya paling mendekati target.

6. Lampiran

a. Pranala Repository

https://github.com/izzatjundy/Tucil2_13523092

b. Tabel Status Pengerjaan

Poin	Ya	Tidak
Program berhasil dikompilasi tanpa kesalahan	✓	
Program berhasil dijalankan	✓	
Program berhasil melakukan kompresi gambar sesuai parameter yang ditentukan	✓	
Mengimplementasi seluruh metode perhitungan eror wajib	✓	
[Bonus] Implementasi persentase kompresi sebagai parameter tambahan	✓	
[Bonus] Implementasi Structural Similarity Index (SSIM) sebagai metode pengukuran error		✓
[Bonus] Output berupa GIF visualisasi proses pembentukan Quadtree dalam kompresi gambar		✓
Program dan laporan dibuat sendiri	✓	