Segmentasi Citra Memanfaatkan Algoritma BFS dan DFS

Orvin Andika Ikhsan Abhista - 13523017
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: orvin.andika@gmail.com , 13523017@std.stei.itb.ac.id

***Abstrak*—Segmentasi citra merupakan salah satu metode paling fundamental dalam pemrosesan citra digital. Segmentasi citra memungkinkan pemisahan antara suatu objek dengan latar belakangnya atau dengan objek lain. Hal ini memungkinkan dilakukannya proses tingkat tinggi seperti pengenalan pola dan pelacakan. Salah satu metode paling efektif untuk segmentasi citra adalah metode CCL . Metode ini mengelompokkan piksel-piksel yang berhubungan menjadi satu objek. Metode ini memanfaatkan algoritma penelusuran graf seperti BFS dan DFS dalam pengelompokan piksel-pikselnya**

***Keywords*—Segmentasi citra, CCL, BFS, DFS**

I. Pendahuluan

Di era digital, kemampuan untuk mengekstrak suatu informasi dari data visual merupakan salah satu kemampuan terpenting dalam perkembangan teknologi. Teknologi-teknologi seperti navigasi robot, kendaraan otonom, hingga pemeriksaan medis membutuhkan kemampuan ini. Sebelum bisa mengekstrak informasi, diperlukan kemampuan untuk memungkinkan mesin bisa menginterpretasikan dunia melalui data visual.

Pengolahan citra digital merupakan salah satu cara agar mesin bisa menginterpretasikan dunia melalui data visual. Salah satu cara yang digunakan dalam pengolahan citra digital adalah segmentasi citra. Segmentasi citra adalah proses mempartisi sebuah citra menjadi beberapa wilayah atau objek yang berbeda. Segmentasi citra memungkinkan pemisahan antara objek dengan latar belakangnya atau objek dengan objek yang lain. Pemisahan ini memungkinkan dilakukannya proses-proses tingkat tinggi seperti pengenalan pola atau pelacakan.

Berbagai metode telah dikembangkan untuk melakukan segmentasi. Salah satunya adalah metode *Connected-Component Labeling* (CCL). Metode ini merupakan salah satu metode yang paling fundamental dan efektif, terutama untuk pemrosesan citra biner. Metode ini bekerja dengan mengelompokkan piksel-piksel yang terhubung menjadi sebuah objek yang sama. Proses pengelompokan ini memanfaarkan algoritma penelusuran graf seperti algoritma *Breadth First Search* dan *Depth First Search.*

Makalah ini ditulis untuk mengimplementasikan algoritma CCL dengan memanfaatkan algoritma BFS dan DFS. Makalah ini juga akan melakukan analisis komparatif untuk performa kedua algoritma, bukan hanya berfokus pada kebenaran implementasi pemisahan objek. Makalah ini akan membahas algoritma mana yang lebih unggul dalam mengimplementasikan metode CCL dan penyebab keunggulannya. Hipotesis yang akan diuji adalah sumber keunggulan sebenarnya bukan sepenuhnya berasal dari algoritma, tetapi ada faktor lain yang mempengaruhi.

II. Landasan Teori

A. Citra Digital

Citra digital adalah gambar dua dimensi yang dihasilkan melalui proses sampling dari analog dua dimensi yang kontinu [1]. Gambar analog dibagi menjadi N baris dan M kolom sehingga menjadi gambar diskrit. Citra digital juga merupakan jenis citra yang dapat diolah oleh komputer. Citra digital dapat diolah oleh komputer karena citra digital hanya disimpaan dalam bentuk angka di komputer. Angka-angka ini merepresentasikan intensitas warna pada masing-masung piksel.

Citra digital dibagi menjadi beberapa jenis. Citra berwarna dibagi menjadi matriks warna merah, biru, dan hijau. Sementara itu, citra *grayscale* adalah citra yang hanya memiliki tingkat keabu-abuan. Dalam kebanyakan pemrosesan citra digital, citra digital akan diubah menjadi citra *grayscale* terlebih dahulu untuk mempermudah pemrosesan.

Citra digital memiliki beberapa komponen utama. Komponen itu antara lain adalah kecerahan, kontur, kontras, dan warna. Kecerahan adalah tingkat terang atau gelapnya sebuah citra secara keseluruhan. Kontur adalah garis kurva yang menyeambungkan titik-titik kontinu di sepanjang batas atau tepi objek. Komponen inilah yang paling berguna untuk deteki objek. Kontras adalah tingkat perbedaan antara area paling gelap dan area paling terang di dalam citra. Citra dengan kontras tinggi artinya citra tersebut memiliki area hitam yang sangat pekat dan area putih yang sangat cerah. Warna adalah persepsi visual yang dihasilkan ketika mata manusia merespon spektrum cahaya yang berbeda-beda. Dalam citra digital, warna direpresentasikan menggunakan beberapa kanal. Model yang paling sering digunakan adalah model RGB. Model RGB menyatakan bahwa setiap warna merupakan kombinasi dari warna merah, hijau, dan biru. Ada juga modek HSV yang merepresentasikan warna berdasarkan hue (rona), saturation

(saturasi), dan value (nilai). Model inilah yang lebih intuitif bagi persepsi manusia

B. Pemrosesan Citra Digital

Pemrosesan citra digital adalah ilmu yang mempelajari hal-hal yang berkaitan dengan perbaikan kualitas suatu gambar, transformasi gambar, pemilihan ciri-ciri yang optimal, dan penyimpanan data. Perbaikan kualitas gambar dapat dilakukan dengan meningkatkan kontras, merngubah warna, dan restorasi citra. Transformasi gambar dilakukan dengan melakukan translasi, rotasi, atau mengubah skala gambar. Sementara itu, penyimpanan data mencakup reduksi, kompresi, dan transmisi data.

Pemrosesan citra dibagi menjadi beberapa tahap. Tahap pertama adalah *preprocessing.* Tahap ini bertujuan untuk meningkatkan kualitas citra. Tahap kedua adalah *enhancement.* Tahap ini bertujuan untuk meningkatkan fitur tertentu agar lebih mudah dilakukan analisis lebih lanjut. Tahap ketiga adalah *segmentation.* Tahap ini bertujuan untuk memisahkan objek dengan latar belakangnya dan dengan objek lain. Tahap ini bisa memanfaatkan metode CCL. Tahap keempat adalah *feature extraction.* Tahap ini bertujuan untuk mengekstrak informasi-informasi dari dalam gambar. Tahap terakhir adalah recognition. Tahap ini adalah tahap pengenalan pola atau objek yang ada di dalam gambar.

C. Segmentasi Citra

Segmentasi citra adalah tahapan untuk membagi citra R menjadi n subwilayah yang tidak tumpang tindih. Tujuan dari segmentasi citra adalah mengelompokkan piksel-piksel menjadi segmen-segmen yang memiliki properti visual yang mirip. Segmentasi merupakan salah satu tahapan fundamnetal agar bisa melakukan pemrosesan citra tingkat tinggi.

Ada beberapa metode untuk melakukan segmentasi. Metode pertama adalah segmentasi berbasis threshold. Metode ini memisahkan piksel berdasarkan intensitasnya relatif terhadap suatu nilai ambang batas. Metode kedua adalah segmentasi berbasis tepi. Segmentasi ini mendeteksi perubahan drastis pada intensitas untuk menemukan batas-batas objek. Metode ketiga adalah segmentasi berbasis wilayah. Metode ini akan mencoba membagi citra menjadi wilayah-wilayah yang homogen.

D. *Connected Component Labelling (CCL)*

CCL adalah salah satu algoritma yang sering digunakan dalam proses pendeteksian objek. Metode ini menggunakan teori hubungan antar piksel dalam sebuah citra. Piksel dianggap terhubung jika memiliki nilai yang sama atau perbedaannya kurang dari threshold yang ditentukan [2].

CCL bekerja dengan memberikan label pada objek yang berbeda. Algoritma ini akan memindai sebuah citra biner dan mengelompokkan piksel-piksel yang terhubung menjadi satu objek. Setiap objek yang teridentifikasi akan diberi sebuah label integer yang unik. Contohhnya, latar belakang akan diberi label 0, objek 1 akan diberi label 1, objek 2 akan diberi label 2, dan seterusnya.

Hal yang paling penting dari CCL adalah mendefinisikan kapan dua piksel dianggap bertetangga. Karena dalam CCL hanya piksel yang bertetangga saja yang mungkin menjadi bagian objek yang sama. Untuk mendefinisikan ketetangaan piksel, bisa menggunakan aturan konektivitas.

E. Aturan Konektivitas

Aturan konektivitas adalah ketentuan yang mendefinisikan kapan dua piksel dianggap bertetangga. Aturan ini menjadi dasar pengelompokkan piksel dalam pembentukan objek. Secara umum, ada dua aturan yang paling sering digunakan dalam pemrosesan citra dua dimensi. Aturan itu adalah konektivitas 4 arah dan konektivitas 8 arah.

Konektivitas 4 arah adalah aturan yang menyatakan bahwa sebuah piksel terhubung dengan piksel-piksel yang berbagi sisi dengannya. Ini artinya sebuah piksel hanya terhubung dengan piksel dii kiri, kanan, atas, dan bawahnya. Contohnya, misal ada sebuah piksel dengan koordinat (x,y) maka piksel yang bertetangga dengan piksel tersebut adalah piksel di posisi (x+1,y), (x-1,y), (x,y+1), dan (x,y-1).

Konektivitas 8 arah adalah aturan yang menyatakan bahwa sebuah piksel terhubung dengan piksel-piksel yang berbagi sisi atau berbagi sudut dengan piksel tersebut. Ini artinya piksel akan bertetangga dengan piksel-piksel dari konektivitas 4 arah ditambah piksel yang bertetangga secara diagonal. Contohnya, misal ada sebuah piksel dengan koordinat (x,y) maka piksel yang bertetangga dengan piksel tersebut adalah piksel di koordinat (x+1,y), (x-1,y), (x,y+1), (x,y-1), (x-1,y-1), (x+1,y-1), (x-1,y+1), dan (x+1,y+1).

F. Penelusuran Graf

Penelusuran graf adalah salah satu topik dalam ilmu komputer yang membahas proses mengunjungi setiap simpul atau *vertex* dan setiap sisi atau *edge*. Tujuan penelusuran graf adalah memastikan bahwa seluruh simpul yang bisa dicapai dalam suatu graf setidaknya pernah dikunjungi satu kali, sehingga pemecahan masalah dalam bentuk graf bisa dilakukan dengan lebih mudah

Untuk melakukan penelusuran graf, terdapat dua algoritma klasik yang sudah sering digunakan dalam berbagai kasus. Algoritma tersebut adalah algoritma *Breadth First Search* dan algoritma *Depth First Search.* Kedua algoritma tersebut memiliki pendekatan penelusuran yang sangat berlawanan. Algoritma BFS mengunjungi semua simpul dalam level yang sama terlebih dahulu. Sementara itu, algoritma DFS menggunakan pendekatan mendalam terlebih dahulu. Keduanya memiliki kompleksitas waktu yang sama. Oleh karena itu, untuk memilih di antara kedua algoritma tersebut sangat bergantung kepada sifat masalah yang dihadapi. Algoritma BFS sangat cocok untuk penyelesaian masalah optimalitas jalur terpendek. Sementara itu, algoritma DFS cocok untuk seperti penyelesaian labirin.

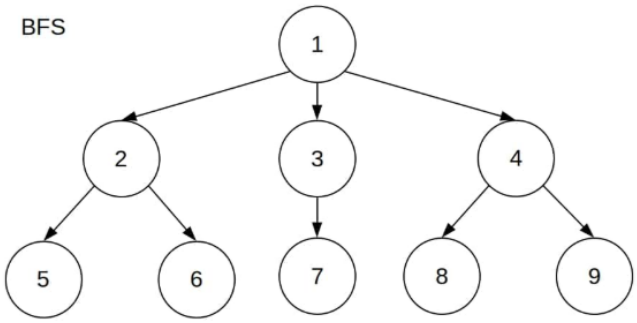
G. Algoritma BFS

Algoritma *Breadth First Search* merupakan sebuah algoritma penelusuran graf yang mengunjungi tiap *link* dari simpul tersebut terlebih dahulu [3]. Kemudian, untuk setiap *link* tersebut, ditelusuri simpul tetangganya yang belum

diperiksa, dan demikian seterusnya hingga ditemukan simpul tujuan.

Algoritma BFS bersifat *First In First Out* (FIFO). Hal ini membuat BFS cocok diimplementasikan dengan struktur data *queue.* Simpul tetangga yang belum diperika akan dimasukkan ke dalam antrian simpul hidup. Sementara itu, simpul yang sudah diperiksa disebut dengan simpul ekspan.

Algoritma BFS memiliki beberapa sifat. Pertama, BFS dijamin akan menemukan solusi pada graf yang berhingga jika solusi itu ada. BFS juga dijamin menemukan jalur terpendek dalam hal jumlah sisi dari simpul awal ke simpul tujuan. Karena BFS mengunjugi setiap simpul dan setiap sisi tepat satu kali, kompleksitas waktunya adalah O(V+E) dengan V adalah jumlah simpul dan E adalah jumlah sisi. Untuk kompleksitas ruang, algoritma ini memiliki kompleksitas O(V) untuk kasus terburuk, yaitu saat semua simpul pada suatu level perlu disimpan dalam *queue.*

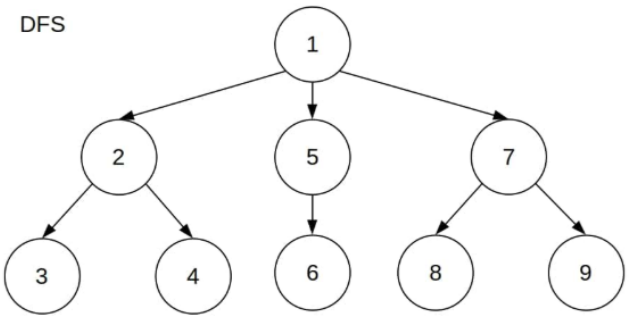


H. Algoritma Depth First Search

Algoritma *Depth First Search* adalah algoritma pencarian yang berjalan dengan mengunjungi anak akar pertama dari pohon pencarian yang dipilih dan terus berjalan lebih dalam hingga sampai di simpul tujuan atau simpul yang tidak memiliki anak [4]. Ketika sampai di simpul yabg tidak memiliki anak, algoritma akan melakukan *backtracking* ke simpul yang belum selesai ditelusuri.

Algoritma DFS bersifat *Last In First Out* (LIFO). Hal ini membuat DFS cocok diimplementasikan menggunakan struktur data *stack* karena simpul yang terakhir ditambahkan akan diproses pertama. Algoritma DFS juga bisa diimplementasikan secara rekursif.

Algoritma DFS memiliki beberapa sifat. Pertama, algoritma DFS dijamin menemukan solusi pada graf yang berhingga jika solusi itu ada. Akan tetapi, algoritma DFS tidak dijamin akan menemukan jalur terpendek. Jalur yang pertama ditemukan oleh algoritma ini bisa saja merupakan jalur panjang yang berliku. Sama seperti algoritma BFS, algoritma DFS juga mengunjungi setiap simpul dan setiap sisi tepat satu kali, kompleksitas waktunya adalah O(V+E) dengan V adalah jumlah simpul dan E adalah jumlah sisi. Untuk kompleksitas ruang, algoritma ini memiliki kompleksitas O(V) untuk kasus terburuk. Akan tetapi, pada praktiknya algoritma DFS biasanya menggunakan memori yang jauh lebih hemat jika dibandingkan dengan algoritma BFS.



III. IMPLEMENTASI

Implementasi program ini sepenuhnya dilakukan dalam bahasa pemrograman Python. Bahasa ini dipilih karena kematangan ekosistemnya untuk komputasi ilmiah dan tingginya keterbacaan kode programnya. Bahasa ini juga merupakan salah satu bahasa yang paling sederhana dalam proses penulisan programnya.

Dalam implementasi program ini, program membutuhkan beberapa library. Pertama adalah library OpenCV. Library ini berfungsi untuk melakukan seluruh operasi yang berhubungan dengan pemrosesan citra, misalnya input/output, konversi ruang warna, dan operasi morfologi. Selain OpenCV, ada library Numpy yang digunakan untuk menyediakan struktur data array multidimensi berperforma tinggi. Ada juga library Matplotlib yang digunakan untuk visualiasi data dan hasil akhir. Penggunaan Matplotlib juga memungkinkan analisis komparatif secara visual.

Program ini dirancang agar bekerja secara terstruktur. Proses berawal dengan pemuatan citra dari *file* eksternal. Setelah citra berhasil dimuat, pemrosesan berlanjut ke tahap *preprocessing.* Tahap ini adalah tahap di mana input citra akan diubah ke format grayscale. Hal ini dilakukan untuk penyederhanaan analisis tiga kanal warna RGB menjadi satu kanal saja. Setelah diubah ke format grayscale, tahap selanjutnya adalah segmentasi biner menggunakan metode thresholding Otsu. Metode ini dipilih karena efektivitasnya dalam menentukan nilai ambang batas optimal secara otomatis berdasarkan histogram citra.

Metode Thresholding Otsu adalah pencarian nilai threshold secara otomatis yang sangat populer karena tidak perlunya intervensi manual dalam penggunaannya. Metode ini bekerja dengan mencoba semua kemungkinan nilai threshold dari 0 hingga 255 kemudian menghitung variansi intrakelasnya. Threshold yang dipilih adalah nilai yang meminimalkan variansi intrakelasnya. Metode Thresholding Otsu ini juga akan secara efektif menghasilkan citra biner muni yang seluruh pikselnya bernilai 1 untuk objek dan 0 untuk latar belakang. Untuk mengatasi citra dengan noise yang signifikan, dilakukan tahap pembersihan menggunakan operasi morfologi seperti menghilangkan bintik-bintik kecil dan mengisi lubang-lubang kecil.

Logika inti dari progran ini adalah implementasi dari algoritma Connected-Component Labelling atau CCL. Implementasi dari algoritma dirancang secara modular sehingga memisahkan logika penelusuran dengan fungsi kontrol utama. Pendekatan modular ini membuat perbandingan

yang dilakukan antara dua metode penelusuran bisa dilakukan secara adil.

Metode penelusuran pertama adalah *Breadth First Search.* Metode ini diiimplementasikan dengan struktur data deque. Deque adalah struktur data mirip queue tetapi bisa melakukan operasi baik pada awal maupun akhir antrian, sesuai namanya yaitu *double ended queue.* Struktur data ini memiliki operasi *enqueue* dan *dequeue* yang membuatnya sangat efisien dalam eksplorasi level per level secara sistematis.

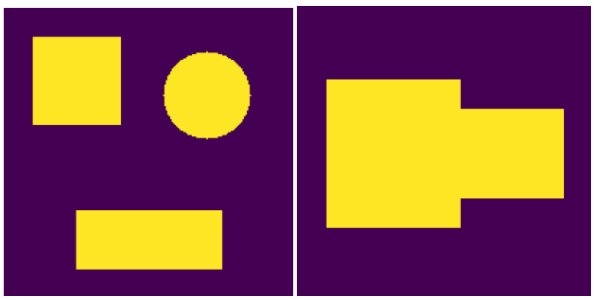
Metode penelusuran lain yang dijadikan pembanding adalah algoritma *Depth First Search.* Algoritma ini diimplementasikan menggunakan list standar Python yang sifatnya dibuat seperti *stack,* yaitu mengimplementasikan fungsi *push* dan *pop.* Algoritma ini juga diimplementasikan secara iteratif, bukan rekursif. Implementasi secara iteratif dipilih agar program ini tidak mengalami error saat citra memiliki komponen objek yang sangat besar atau topologi yang kompleks.

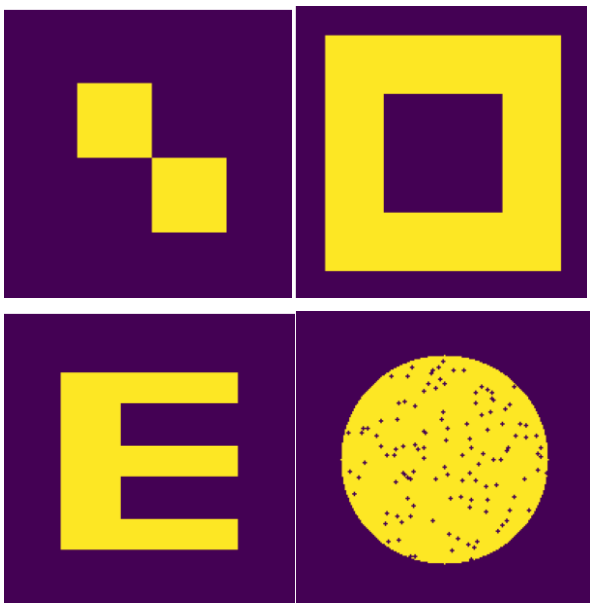
Algoritma yang diimplementasikan semuanya menggunakan aturan konektivitas 8 arah. Aturan ini memastikan bahwa piksel yang terhubung secara diagonal dapat diperlakukan sebagai suatu kesatuan objek.

Untuk menguji kebenaran algoritma, digunakan dua set data uji coba berbeda. Data uji coba ini terdiri dari set citra sintetis yang sengaja dibuat untuk menguji *edge case* program dan citra dari objek asli yang ada di dunia nyata. Set data pertama dibuat dengan memanfaatkan library OpenCv. Set data ini berfungsi untuk menguji ketepatan output untuk kasus-kasus seperti objek yang bersentuhan sisi, objek yang bersentuhan secara diagonal untuk validasi aturan konektivitas, objek yang berlubang untuk menguji penanganan topologi, serta bentuk-bentuk kompleks untuk menguji ketangguhan algoritma dalam navigasi struktur yang rumit. Set kedua adalah kumpulan citra dunia nyata yang lebih menantang karena ada variasi dalam pencahayaan dan tingkat noise. Penggunaan dua set berbeda ini bertujuan untuk memastikan algoritma tidak hanya valid pada data yang ideal, tetapi juga bisa menyelesaikan permasalahan asli yang relevan dengan dunia nyata.

IV. Hasil dan Pembahasan

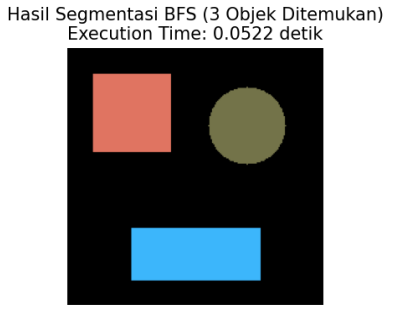
Pengujian diawali dengan set yang sengaja dibuat untuk menguji *edge cases.* Citra untuk pengujian dibuat dengan memanfaatkan library OpenCV. Citra ini dibuat secara modular melalui fungsi-fungsi berbeda untuk pembuatan satu citra. Berikut adalah citra-citra sintetis yang digunakan dalam pengujian.

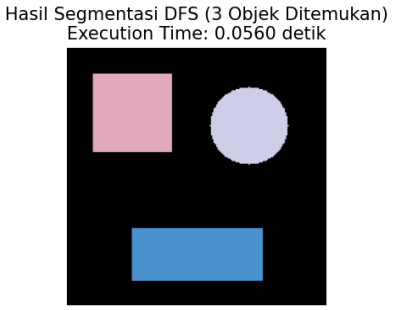




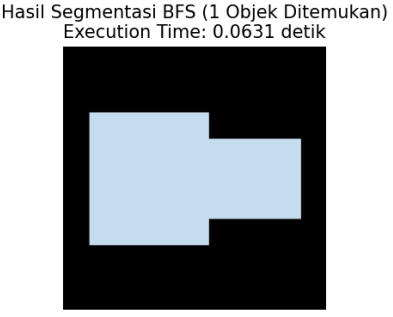
Berikut adalah hasil segmentasi objek citra sintetis menggunakan algoritma BFS dan DFS.

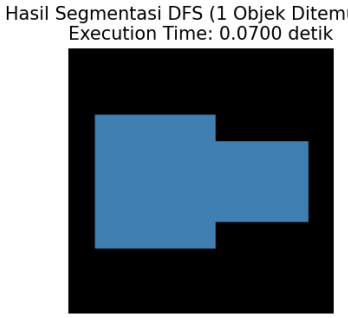
Hasil segmentasi gambar 1



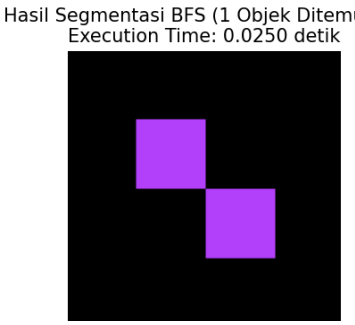


Hasil segmentasi gambar 2

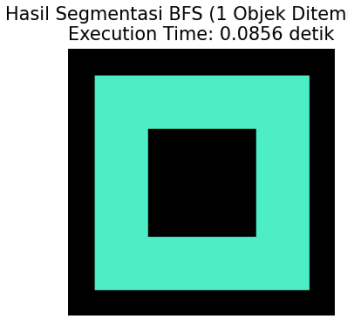




Hasil segmentasi gambar 3

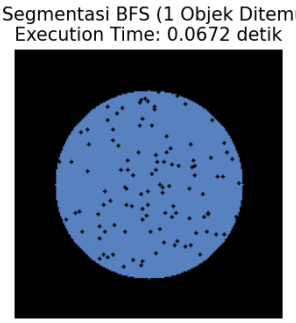


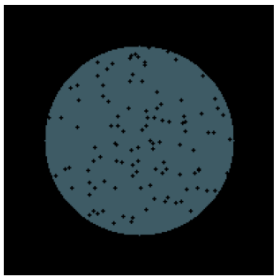
Hasil segmentasi gambar 4



Hasil segmentasi gambar 5

Hasil segmentasi gambar 6



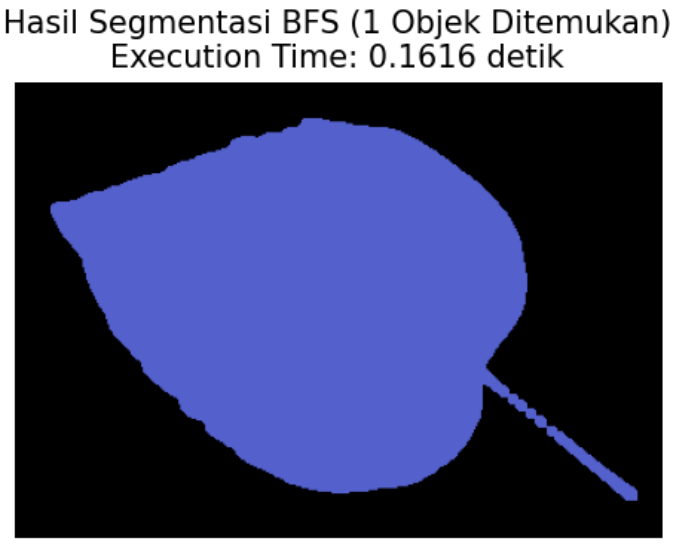


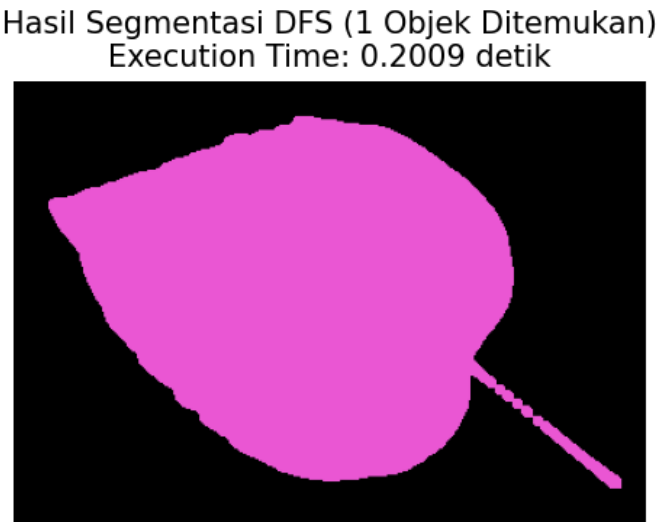
Hasil percobaan menunjukkan bahwa algoritma segmentasi objek telah bekerja dengan baik untuk citra sintetis. Algoritma ini tetap bisa mendeteksi objek sebagai satu kesatuan dalam beberapa situasi spesifik seperti saat objek bersentuhan dengan objek lain, objek memiliki lubang, objek memiliki bentuk rumit, dan objek memiliki banyak noise atau bintik-bintik kecil.

Percobaan dilanjutkan dengan menggunakan citra dari objek asli yang berada di dunia nyata. Berikut adalah citra yang digunakan.

Berikut adalah hasil segmentasi dari objek asli menggunakan algoritma BFS dan DFS.

Hasil segmentasi gambar 1





Hasil segmentasi gambar 2

Hasil percobaan menunjukkan bahwa algoritma juga berhasil melakukan segmentasi pada objek asli baik dengan algoritma BFS maupun algoritma DFS. Algoritma juga tetap bisa melakukan segmentasi walaupun terdapat noise pada gambar aslinya.

Kinerja algoritma BFS selalu lebih unggul dalam percobaan kali ini. Secara teori, dalam konteks ini seharusnya algoritma BFS dan DFS memiiliki kompleksitas yang sama yaitu O(V+E), dengan V adalah jumlah piksel dan E adalah jumlah koneksi. Keunggulan performa BFS dalam percobaan ini tidak disebabkan oleh jumlah langkah algoritmanya. Penyebab keunggulannya adalah cara operasi algoritma BFS yang lebih cocok untuk arsitektur komputer modern. BFS akan mengakses memori secara berurutan, karena itulah akan lebih sering terjadi cache hit jika dibandingkan dengan DFS yang melompat-lompat. Oleh karena itu, algoritma BFS memiliiki performa yang lebih tinggi untuk pemrosesan citra seperti pada percobaan ini.

V. KESIMPULAN

Berdasarkan percobaan yang telah dilakukan dapat disimpulkan bahwa makalah ini telah berhasil mengimplementasikan segmentasi objek baik dengan algoritma BFS maupun algoritma DFS. Meskipun memiliki kompleksitas algoritma yang sama, algoritma BFS memiliki performa yang lebih baik selama percobaan. Keunggulan performa ini disebabkan cara kerjanya yang lebih cocok dengan arsitektur komputer modern karena akan lebih banyak menghasilkan cache hit dibandingkan algoritma DFS.

ACKNOWLEDGMENT *(Heading 5)*

Penulis mengucapkan terima kasih kepada:

1. Tuhan Yang Maha Esa
2. Dosen pengampu mata kuliah Strategi Algoritma
3. Teman-teman penulis
4. Pihak-pihak lain

yang telah membantu penulis menyelesaikan penulisan makalah ini.

REFERENCES

[1] N. Z. Munantri, H. Sofyan, and M. Y. F., "Aplikasi Pengolahan Citra Digital untuk Identifikasi Umur Pohon," *TELEMATIKA*, vol. 16, no. 2, pp. 97–104, Okt. 2019.

[2] J Lestari, F., Putra, E. D. . ., Sonita, A. . ., & Darnita, Y. . . (2023). Human Object Counter Tracking Using Connected Component Labelling On Digital Image Processing. Jurnal Komputer, Informasi Dan Teknologi, 3(2), 251–258. https://doi.org/10.53697/jkomitek.v3i2.1218.

[3] Yuwono, Doddy T. "Perbandingan Algoritma Breadth First Search Dan Depth First Search Sebagai Focused Crawler." Annual Research Seminar: Computer Science and Information and Communications Technology 2016, Palembang, Indonesia, 2016. Sriwijaya University, 2016.

[4] O. Pribadi, "Maze Generator Dengan Menggunakan Algoritma Depth-First-Search", JTM, vol. 4, no. 1, hlm. 1–5, Jul 2015.R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[5] R. Munir, 'Homepage Rinaldi Munir'. https://informatika.stei.itb.ac.id/~rinaldi.munir/. [Diakses 21 Juni 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, 21 Juni 2025

Orvin Andika - 13523017