Segmentasi Citra dengan Metode Ford-Fulkerson

Rika Dewi/13517147
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517147@stei.itb.ac.id

***Abstrak*—Pengolahan gambar digital turut berkembang Berkembangnya teknologi menciptakan sebuah tuntutan untuk menampilkan gambar digital dalam kualitas tinggi. Gambar kerkualitas tinggi memberikan data citra yang besar dan kompleks, sehingga tidaklah mudah untuk melakukan proses pengolahan terhadap gambar digital tersebut. Oleh sebab itu, dalam makalah ini penulis ingin menawarkan solusi untuk mengubah masukan gambar yang kompleks menjadi lebih sederhana melalui segmentasi citra. Segmentasi citra membagi gambar menurut klasifikasi tertentu sehingga peninjuan terhadap komponen gambar lebih mudah. Segmentasi citra ini menggunakan pendekatan BFS dan DFS pada metode Ford-Fulkerson.**

Keywords—Ford-Fulkerson; Segmentasi; DFS; BFS; Network Flow

I. PENDAHULUAN

Citra atau gambar adalah kombinasi antara titik, garis, bidang, dan warna untuk menciptakan imitasi dari objek tertentu. Gambar dapat bersifat analog dan digital. Gambar analog merupakan gambar yang dihasilkan melalui proses sketsa alat gambar ke suatu media di luar lingkungan media digital. Sedangkan gambar digital merupakan gambar yang dihasilkan dari pemrosesan oleh perangkat elektronik. Gambar digital sendiri terbagi dalam dua jenis yaitu bitmap dan vektor. Gambar bitmap tersusun dari struktur data yang mewakili susunan piksel warna dan gambar vektor tersusun atas sekumpulan garis dan kurva pada bidang tertentu.

Saat ini, kebutuhan akan gambar digital semakin meningkat pesat. Salah satu bidang ilmu komputer yang turut berkembang seiring kebutuhan ini yaitu pengolahan gambar digital. Secara umum, pengolahan gambar digital dapat diartikan sebagai proses memanipulasi gambar digital dengan menggunakan bantuan komputer untuk keperluan tertentu. Pengolahan gambar digital memiliki manfaat yang cukup luas dalam bidang sehari-hari, seperti dalam robotika, pengenalan pola, *computer vision*, dan penginderaan jarak jauh.

Dengan melakukan pengolahan gambar digital, data citra dapat menunjukkan suatu informasi tertentu. Pengenalan pola adalah cabang dari pengolahan data citra yang menitikberatkan pada penemuan pola pada data citra untuk mendapatkan informasi tersebut. Agar dapat mengeluarkan informasi yang terkandung di dalamnya, pola-pola itu diklasifikasi untuk mengenali objek di dalam gambar. Namun, bukanlah hal mudah untuk mengenali objek terlebih jika dihadapkan pada data citra yang besar dan sangat kompleks.

Berangkat dari masalah tersebut, penulis ingin melakukan pendekatan penyelesaian masalah dengan menggunakan salah satu teknik dalam strategi algoritma. Pendekatan yang akan digunakan untuk menyelesaikan masalah ini adalah metode DFS dan BFS pada metode Ford-Fulkerson. Penerapan BFS dan DFS ini akan diimplementasikan pada segmentasi citra.

Segmentasi citra merupakan teknik membagi suatu gambar berdasarkan kemiripan atribut. Tujuannya yaitu mengubah suatu gambar masukan yang kompleks menjadi gambar yang lebih sederhana, berdasarkan peninjauan terhadap komponen gambar. Dengan demikian, pengamat citra akan dimudahkan untuk melakukan analisis. Pada umumnya Dengan pendekatan algoritma ini, diharapkan diperoleh keluaran hasil segmentasi citra berupa gambar yang terdiri atas objek pengamatan (*foreground*) dan latar belakang (*background*).

II. DASAR TEORI

A. Bitmap

Salah satu cara komputer mengerti sebuah ilustrasi gambar yaitu dengan bitmap. Bitmap merepresentasikan sebuah gambar sebagai matriks dua dimensi dengan setiap sel pada matriksnya dinamakan piksel. Bitmap menyatakan besar kecilnya ukuran piksel pada gambar dan kuantisasi yang menyatakan jumlah warna pada gambar.

Model warna yang digunakan pada umumnya yaitu RGB. Model warna RGB terdiri atas 3 jenis warna yaitu *Red, Green, dan Blue*. Jangkauan warna yang tersedia berkisar antara 0-255.

B. Segmentasi Citra

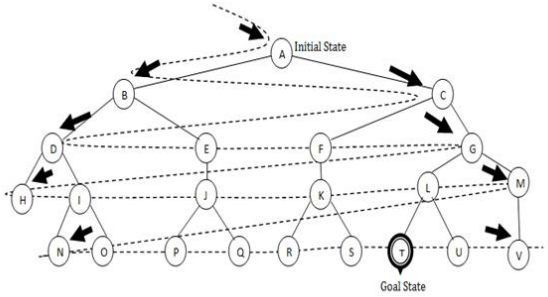
Segmentasi citra adalah membagi suatu citra menjadi wilayah-wilayah yang homogen berdasarkan kriteria keserupaan yang tertentu antara tingkat warna suatu piksel terhadap tingkat warna piksel-piksel tetangganya. Salah satu proses dalam segmentasi citra yaitu klasterisasi (*clustering*). Klaster (*cluster*) adalah kesatuan nilai-nilai dalam jarak tertentu pada kepadatan suatu daerah dibandingkan dengan kepadatan nilai-nilai daerah sekitarnya.

Suatu teknik segmentasi citra menggunakan metode klasterisasi adalah teknik pemetaan warna (*color mapping*). Masukkan citra akan dikelompokkan sesuai dengan kesamaan-kesamaan warna yang dimiliki.

C. Algoritma BFS

BFS adalah algoritma traversal dalam graf yang mengunjungi simpul secara sistematik secara melebar (*Breadth First Search*/BFS). Secara umum, skema algoritma BFS dapat dirumuskan sebagai berikut.

1. Traversal dimulai dari simpul v.
2. Kunjungi simpul v.
3. Kunjungi semua simpul yang bertetangga dengan simpul v terlebih dahulu.
4. Kunjungi simpul yang belum dikunjungi dan bertetangga dengan simpul-simpul yang tadi dikunjungi, demikian seterusnya.



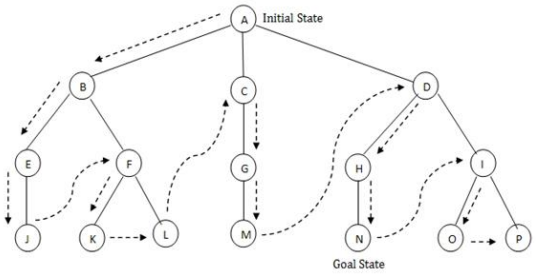
Gambar 1. Ilustrasi BFS

Sumber: http://aa-miracle.blogspot.com/2012/05/analisa-game-puzzle-slider-algoritma.html

D. Algoritma DFS

DFS adalah algoritma traversal dalam graf yang mengunjungi simpul secara sistematik secara mendalam (Depth First Search/BFS). Secara umum, skema algoritma BFS dapat dirumuskan sebagai berikut.

1. Traversal dimulai dari simpul v.
2. Kunjungi simpul v.
3. Kunjungi simpul w yang bertetangga dengan simpul v.
4. Ulangi DFS untuk simpul w.
5. Ketika mencapai simpul u sedemikian sehingga semua simpul yang bertetangga dengannya telah dikunjungi, pencarian runut balik (*backtrack*) ke simpul terakhir yang dikunjungi sebelumnya dan mempunyai simpul w yang belum dikunjungi.
6. Pencarian berakhir bila tidak ada lagi simpul yang belum dikunjungi yang dapat dicapai dari simpul yang telah dikunjungi.



Gambar 2. Ilustrasi DFS

Sumber: http://aa-miracle.blogspot.com/2012/05/analisa-game-puzzle-slider-algoritma.html

E. Network Flow Problem

Network Flow Problem adalah suatu masalah komputasi yang menerima input sebuah *flow network* (sebuah graf dengan kapasitas pada sisinya) dengan tujuan untuk membentuk sebuah *flow* nilai pada setiap sisi yang bergantung pada batasan-batasan tertentu.

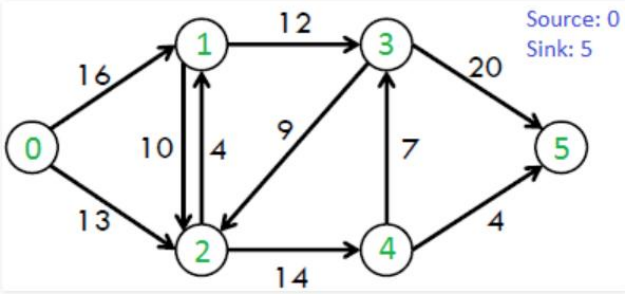
Sebuah *flow network* $G = (V, E)$ adalah sebuah graf berarah yang setiap sisinya $(u, v) \in E$ mempunyai nilai kapasitas non negatif $c(u, v) \geq 0$. Jika $(u, v) \notin E$ maka $c(u, v) = 0$. Terdapat dua simpul yang spesial yaitu source (s) dan sink (t). Diasumsikan bahwa untuk setiap simpul terdapat jalur dari source menuju ke sink.

Sebuah *flow* adalah fungsi dari pasangan simpul-simpul $f : V \times V \rightarrow R$ yang memenuhi tiga syarat yaitu:

1. *Capacity Constraint*
   Untuk semua $u, v \in V$, $f(u, v) \leq c(u, v)$, yang berarti tidak ada sisi yang melebihi kapasitas dari sisi tersebut.
2. *Skew Simmetry*
   Untuk semua $u, v \in V$, $f(u, v) = -f(v, u)$, yang berarti setiap *backward flow* adalah negatif *flow*.
3. *Flow Conservation*
   Untuk semua $u \in V - \{s, t\}$, harus dipenuhi persamaan
   $$\sum_{v \in V} f(u, v) = 0$$
   Hal ini berarti *flow* yang masuk harus sama dengan *flow* yang keluar untuk setiap simpul kecuali source dan sink.

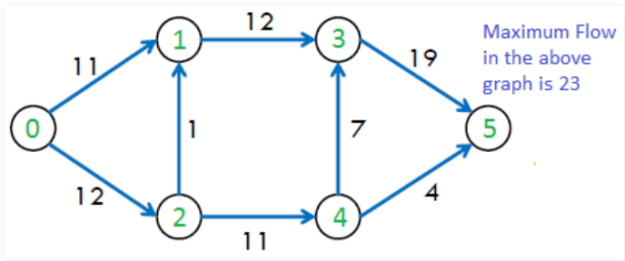
Persoalan yang muncul dari *Network Flow Problem* adalah untuk mencari *maximum flow* dengan tujuan memaksimalkan jumlah *flow* yang keluar dari source menuju ke simpul sink. Teorema *max-flow min-cut* merumuskan *maximum flow* sebagai *minimum cut* yaitu himpunan bagian dari simpul *flow network* yang meminimalkan total kapasitas pada himpunan bagian sisi.

Misal terdapat graf *flow network* sebagai berikut.



Gambar 3. Contoh Flow Network
Sumber: https://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/

Maka nilai maximum flow yang mungkin dari graf Gambar 3 di atas adalah 23. Hasilnya dapat dilihat pada Gambar 4.



Gambar 4. Contoh Maximum Flow
Sumber: https://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/

F. Metode Ford-Fulkerson

Metode Ford-Fulkerson adalah algortima greedy yang melakukan komputasi *maximum flow* dalam *flow network*. Sebuah jalur pada *flow network* dari s menuju t dengan *flow* yang dapat ditambahkan disebut *augmenting path*. Ford-Fulkerson lebih sering disebut sebagai metode dibandingkan dengan algoritma. Hal ini karena pendekatan untuk menermukan augmenting path tidak sepenuhnya disebutkan. Terdapat beberapa implementasi algoritma yang melakukan pendekatan terhadap metode Ford-Fulkerson ini. Secara umum, skema Metode Ford-Fulkerson adalah sebagai berikut.

1. Inisialisasi *flow* dengan nilai 0.
2. Selama masih terdapat *augmenting path* dari source menuju sink, tambahkan *path-flow* ini pada *flow*.
3. Jika sudah tidak mungkin ditambahkan *flow* pada *network*, maka hasil yang didapatkan adalah *maximum flow* yang mungkin dihasilkan (*maximum flow* local adalah *maximum flow* global).

Berikut adalah pseudo-code dari metode Ford-Fulkerson.

```
FordFulkerson(G, s, t) {
    initialize flow f to 0;
    while (there exists an augmenting path p) {
        augment the flow along p;
    }
    output the final flow f;
}
```

G. Teorema Max-Flow Min-Cut

Teorema max-flow min-cut menyatakan bahwa dalam sebuah *network flow*, nilai *maximum flow* dari source menuju ke sink sama dengan bobot total dari himpunan sisi yang membentuk *minimum cut*. *Minimum cut* adalah total bobot terkecil dari himpunan sisi yang jika dihilangkan akan memutuskan hubungan source dan sink pada graf, sehingga graf akan terbagi menjadi dua bagian. Terdapat tiga syarat untuk menggunakan teorema max-flow min-cut yaitu sebagai berikut.

1. f adalah *maximum flow* dalam G.
2. Graf residual G_f tidak lagi memiliki *augmenting path*.
3. $|f| = c(S,T)$ untuk himpunan sisi *minimum cut* (S,T) pada G.

III. Pembatasan Masalah

Dalam penulisan makalah ini, penulis membatasi masalah yang akan dibahas, diselesaikan, dan diuji dalam beberapa batasan masalah. Batasan masalah ini diperlukan agar penyelesaian masalah dapat dilakukan dengan lebih mudah dan singkat. Selain itu, batasan masalah tersebut dapat menjadi hal-hal yang dapat ditingkatkan dalam riset-riset berkaitan di masa yang akan data. Adapun batasan masalah yang penulis ambil dalam penulis makalah ini adalah sebagai berikut.

A. Gambar Digital

Algoritma yang diimplementasikan dalam makalah ini memiliki kompeksitas waktu yang sebanding dengan kuadrat dari ukuran gambar masukan. Oleh karena itu, agar dapat berjalan relatif cepat, penulis membatasi agar gambar yang dimasukkan tidak melebihi ukuran 30x30 piksel.

B. Atribut yang Dibandingkan

Gambar digital yang akan diproses pada makalah ini adalah yang memiliki tipe bitmap, yaitu direpresentasikan dalam matriks RGB. Pada praktiknya, gambar digital memiliki variasi komponen nilai RGB (Red, Green, Blue). Untuk memperjelas perbedaan atribut antar piksel, penulis membatasi variasi komponen RGB hanya pada salah satu komponen yaitu Red. Matriks bitmap yang akan diproses memiliki nilai Green dan Blue yang sama untuk setiap piksel. Sehingga pada proses menentukan kesamaan antar piksel, penulis hanya akan membandingkan nilai dari komponen Red pada matriks bitmap

C. Foreground dan Background

Dalam penentuan piksel *foreground* dan *background*, penulis membatasi *foreground* sebagai piksel yang memiliki komponen Red mendekati 255 dan *background* sebagai piksel yang memiliki komponen Red mendekati 0. Hal ini untuk meminimalisir kesalahan dalam menentukan *background* dan *foreground*.

IV. Ide Penyelesaian Masalah

Dalam menyelesaikan masalah segmentasi gambar, terdapat beberapa ide penyelesaian yang penulis gunakan. Adapun ide dan tahapan pendekatan penyelesaian masalah tersebut sebagai berikut.

A. Konstruksi Graf Nework Flow

Pertama-pertama penulis melakukan tranformasi sebuah bitmap gambar menjadi graf matriks ketetanggaan. Sebuah piksel akan dipetakan pada graf secara *row-major order*. Terdapat dua tambahan simpul pada graf yang berfungsi sebagai source dan sink.

Terdapat dua tipe sisi pada graf yang dibangun. Sisi pertama disebut n-link, yang menghubungan setiap piksel dengan empat pixel tetangganya (atas, bawah, kiri, dan kanan). Sisi kedua disebut t-link yang menghubungkan simpul source atau sink dengan piksel.

Nilai sisi n-link harus diperhitungkan sehingga mampu merepresentasikan kesamaan antar piksel. Nilai dari sisi ini akan semakin besar ketika kedua piksel mirip, dan semakin kecil ketika kedua piksel berbeda. Dalam menentukan nilai sisi n-link digunakan fungsi *boundary penalty*. Misal I_p adalah intensitas piksel pada simpul p. Untuk edge $(p, q) \in E$, *boundary penalty* $B(I_p, I_q)$ didefinisikan dengan persamaan berikut.

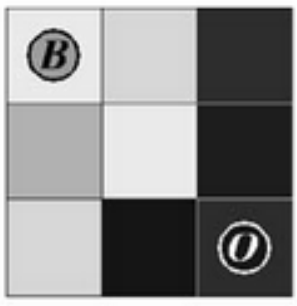
$$B(I_p, I_q) = 100 * \exp(\frac{-(I_p - I_q)^2}{2\sigma^2})$$

Nilai dari σ akan menjadi parameter yang nilainya dapat diubah-ubah untuk mencari hasil optimum dari pemetaan piksel. Pada makalah ini, penulis memilih menggunakan $\sigma = 30$.

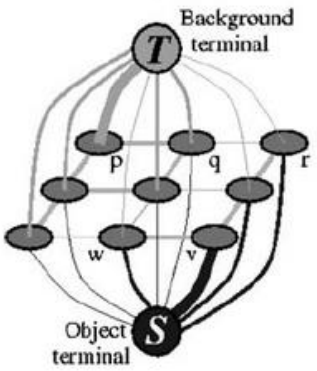
Untuk pembuatan sisi t-link, akan dipilih secara acak n piksel sebagai background dan n piksel sebagai foreground. Piksel-piksel ini akan disebut sebagai seed. Jumlah seed akan menjadi parameter yang nilainya dapat diubah-ubah untuk mencari hasil optimum pengklasifikasian foreground dan background. Pada makalah ini, penulis memilih menggunakan $n = 3$. Untuk setiap *background* seed, sebuah sisi ditambahkan dari source menuju *background* seed dengan kapasitas K. Nilai dari K adalah nilai maksimum dari *boundary penalty*, yang secara formal didefinisikan sebagai berikut.

$$K = \max(\{B(I_p, I_q) \mid (p, q) \in E\})$$

Dengan cara yang sama untuk setiap *foreground* seed, sebuah sisi ditambahkan menuju sink dari *foreground* seed dengan kapasitas K. Contoh hasil konstruksi graf network flow pada gambar bitmap Gambar 5(a) dapat dilihat pada Gambar 5(b).



(a)



(b)

Gambar 5. (a) Bitmap dengan seed, (b) *Graf Network Flow* yang terbentuk (perhatikan tebal dan tipis garis merepresentasikan bobot dari sisi tersebut).

Sumber: Boykov and Funka-Lea, 2006.

B. Augmenting Path

Algoritma Edmond-Karp adalah salah satu implementasi pencarian *augmenting path* pada metode Ford-Fulkerson menggunakan graf residual. Graf Residual adalah graf yang menunjukkan pertambahan *flow* yang mungkin. Setiap sisi pada graf residual memiliki nilai yang disebut kapasitas residual. Kapasitas residual didapat dari kapasitas dari sisi pada graf dikurangi *flow* saat ini. Kapasitas residual menunjukkan kapasitas saat ini pada sisi bersangkutan.

Pertama-tama graf residual $G_f = (V, E_f)$ didefinisikan sebagai graf yang sama dengan hasil konstruksi graf *network flow*, namun dengan kapasitas $c_f(u, v) = c(u, v) - f(u, v)$ dengan mula-mula tidak ada *flow*. Jika terdapat jalur dari source menuju sink pada graf residual, maka akan mungkin untuk menambah *flow* dengan jalur ini. Jalur inilah yang disebut *augmenting path*. Penambahan *flow* dari jalur ini akan memiliki nilai minimum dari kapasitas residual sisi-sisi yang dilalui sepanjang jalur. Pencarian ini akan dilakukan terus hingga tidak ada *augmenting path* yang ditemukan. Dengan demikian dapat disimpulkan *flow* saat ini adalah max-flow. Berikut adalah pseudo-code dari pencarian *augmenting path*.

```
while there is a path from s to t in the residual network:
    flow = min([residual capacity of every edge along this path])
    for every edge (u,v) in this path:
        residual capacity of (u,v) -= flow
        residual capacity of (v,u) += flow
```

Algoritma Edmond-Karp menggunakan BFS untuk mencari path yang mungkin pada graf residual dalam setiap iterasinya. BFS akan melakukan proses traversal untuk menemukan path dan menghitung *flow* yang mungkin dengan mencari nilai kapasitas residual minimum di sepanjang path. Hasil ini kemudian ditambahkan pada keseluruhan *flow* dengan melakukan perubahan terhadap kapasitas residual pada graf residual. Pada setiap sisi yang dilalui path, kita mengurangi kapasitas residualnya dengan *path flow* sedangkan untuk sisi kebalikannya kita menambahkan kapasitas residualnya dengan *path flow*.

C. *Cutting Graph*

Dengan teorema max-flow min-cut, nilai dari *maximum flow* juga adalah nilai dari *minimum-cut*. S mengandung simpul yang dapat dijangkau dari sink pada graf residual. Oleh karena itu, akan dilakukan DFS dari sink menggunakan graf residual untuk menentukan sisi yang akan dihilangkan. Pada program yang dibuat penulis, sisi yang dihilangkan ini ditandai dengan warna hitam pada simpul yang menghubungkannya.

D. *Analisis Kompleksitas*

Akan ditinjau kompleksitas waktu algoritma Edmond-Karp. Untuk mencari sebuah *augmenting path* kompleksitasnya $O(E)$ dan panjang dari path adalah $O(V)$. Pada setiap iterasinya minimal ditemukan sebuah *augmenting path* sehingga untuk mencari semua *augmenting path* dibutuhkan komplesitas $O(VE)$. Iterasi *while-loop* memiliki komplesitas $O(E)$, sehingga total komplesitas waktu untuk algoritma Edmond-Karp adalah $O(VE^2)$.

V. Pengujian dan Hasil Uji

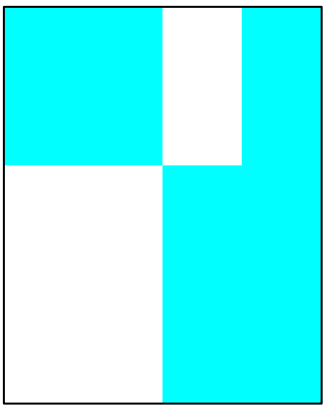
Pengujian dilakukan terhadap algoritma yang telah diimplementasikan dalam Bahasa pemrograman C++. Source code dari program dapat dilihat melalui tautan yang penulis lampirkan di bagian Apendiks. Data input dimasukkan secara *hard-coded* pada program. Data input berisi array yang merepresentasikan bitmap pada gambar digital.

```
pix[]={0,255,255,   0,255,255,   255,255,255, 0,255,255,
       0,255,255,   0,255,255,   255,255,255, 0,255,255,
       255,255,255, 255,255,255, 0,255,255,   0,255,255,
       255,255,255, 255,255,255, 0,255,255,   0,255,255,
       255,255,255, 255,255,255, 0,255,255,   0,255,255};
```

Gambar 6. Data Uji

Sumber: Arsip Penulis

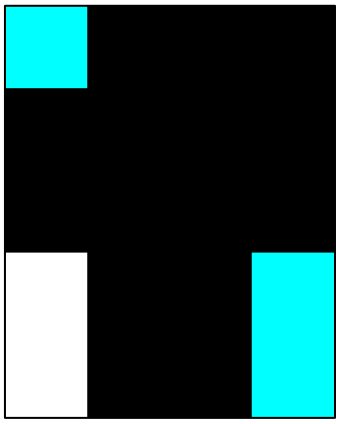
Data uji diambil dari Gambar 5 agar mudah diperiksa kebenaran hasil algoritmanya. Data uji adalah gambar dengan ukuran 4x6 pixel dengan setiap tuple tiga integer pada array data uji merepresentasikan nilai RGB dari pixel yang dipetakan secara *row-major ordering*. Berikut adalah tampak bitmap dari data uji.



Gambar 7. Bitmap Data Uji

Sumber: Arsip Penulis

Hasil dari eksekusi program dengan masukan data uji diperoleh output sebagai berikut.



Gambar 8. Hasil Uji

Sumber: Arsip Penulis

Berdasarkan hasil Uji pada Gambar 8, implementasi algoritma menggunakan metode Ford-Fulkerson memberikan hasil uji berupa segmentasi gambar menjadi foreground dan background dengan dibatasi oleh garis hitam.

VI. Kesimpulan dan Saran

Berdasarkan tinjauan teori, hasil uji coba dan analisis dari implementasi algoritma DFS dan BFS pada metode Ford-Fulkerson, penulis menarik beberapa kesimpulan:

1. Permasalahan segmentasi citra dapat diselesaikan dengan algoritma DFS dan BFS pada metode Ford-Fulkerson. Hasil penyelesaiannya sesuai dengan harapan.
2. Eksekusi algoritma DFS dan BFS pada metode Ford-Fulkerson memiliki kompleksitas waktu yang besar yaitu $O(VE^2)$.
3. Penyelesaian menggunakan algoritma DFS dan BFS pada metode Ford-Fulkerson menurunkan kompleksitas dari gambar digital dengan membagi gambar menjadi komponen yang lebih kecil yaitu *foreground* dan *background*.

Pada penulisan makalah ini, terdapat beberapa saran untuk penulis maupun pembaca. Saran-saran tersebut antara lain:

1. Pada makalah ini terdapat beberapa batasan masalah. Dengan adanya batasan masalah ini diharapkan dapat membuka peluang untuk penelitian lebih lanjut terhadap topik dan metode ini dengan pembatasan masalah yang dikurangi.
2. Pada program yang penulis implementasikan, terdapat dua buah variabel bebas yaitu sigma dan jumlah seed yang belum sempat dieksplorasi sebagai hasil pengujian terhadap algoritma. Dengan adanya variabel bebas ini diharapkan dapat dilakukan penelitian lebih lanjut terkait nilai optimal dari kedua variabel bebas ini.

3. Terdapat algoritma lain untuk mencari *augmenting path* dengan metode Ford-Fulkerson. Kompleksitas waktu algoritma DFS dan BFS yang besar dapat menjadi bahan pertimbangan untuk mencoba algoritma lain dalam mencari *augmenting path* pada metode Ford-Fulkerson.

VII. Apendiks

Implementasi program yang dibuat penulis beserta hasil percobaan dapat diakses pada GitHub melalui tautan https://github.com/Rikadewi/image-segmentation. Program ditulis dengan bahasa C++ sehingga diperlukan compiler GNU C++ compiler untuk mengeksekusi program.

VIII. Ucapan Terima Kasih

Pertama penulis mengucapkan syukur kepada Tuhan Yang Maha Esa, sehingga penulis dapat menyelesaikan penulisan makalah strategi algoritma ini. Setelah itu penulis mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T., Ibu Dr. Masayu Leylia Khodra, dan Ibu Dr.Nur Ulfa Maulidevi, S.T. sebagai dosen mata kuliah strategi algoritma di program studi Teknik Informatika Institut Teknologi Bandung yang telah memberikan bimbingan dan dukungan kepada penulis beserta ilmu-ilmu bermanfaat yang telah diajarkan. Penulis juga berterima kasih kepada orang tua, keluarha, dan teman-teman yang telah memberikan dukungan, motivasi, dan bantuan kepada penulis selama pengerjaan makalah ini.

IX. Referensi

[1] 2009. Munir, Rinaldi. *Diktat Kuliah IF2211 Strategi Algoritma*. Bandung: Institut Teknologi Bandung.

[2] 2004. Munir, Rinaldi. *Pengolahan Citra Digital dengan Pendekatan Algoritmik.* Bandung: Informatika.

[3] 2017. Hidayatullah, Priyanto. *Pengolahan Citra Digital: Teori dan Aplikasi Nyata Plus*. Bandung: Informatika.

[4] 2009. Cormen, dkk. *Introduction to Algorithms*. London: The MIT Press.

[5] 2013. Mount, David M. *Design and Analysis of Computer Algorithms*. College Park: University of Maryland.

[6] https://julie-jiang.github.io/image-segmentation/ diakses pada 25 Mei 2019.

[7] https://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/ diakses pada 25 Mei 2019.

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, 26 April 2019

Rika Dewi/13517147