

## BAB II

### TINJAUAN PUSTAKA

#### 2.1 Penelitian Terkait

Beberapa penelitian terkait tentang segmentasi dengan algoritma *mean shift*, algoritma *normalized cut* dan metode *region merging* adalah sebagai berikut:

1. Penelitian yang dilakukan oleh Comaniciu *et al* [12] dengan judul “ *Mean Shift: A Robust Approach Toward Feature Space Analysis* ” pada tahun 2002. Penelitian ini memperkenalkan penggunaan algoritma mean shift untuk segmentasi citra dengan menggunakan pendekatan analisis ruang fitur (*feature space*) atau analisis ruang berdasarkan densitas warna pada citra serta penggunaan pergeseran titik tengah warna pada sebuah area. Hasil yang berhasil didapatkan adalah area dengan densitas warna yang hampir sama akan disatukan dan detail dari tepi obyek yang disegmentasi tetap terjaga.
2. Penelitian yang dilakukan oleh Li *et al* [15] dengan judul “ *The Segmentation in Textile Printing Image Based on Mean Shift* ” pada tahun 2009. Penelitian ini menggunakan pendekatan informasi warna (*color information*) pada sebuah citra berwarna yaitu dengan pemodelan *Luv color space* dan pendekatan spasial yang dirujuk dengan nilai  $x$ , dan  $y$ . Setiap piksel warna akan dikomputasi ke dalam 5 dimensi ruang fitur yaitu  $(x, y, L, u, v)$ . Hasil penelitian dilaporkan bahwa pemilihan parameter secara tepat akan meningkatkan akurasi sehingga dapat mensegmentasi citra tekstil dengan tingkat pencahayaan yang berbeda dengan lebih akurat.
3. Penelitian yang dilakukan oleh Nagau *et al* [16] dengan judul “ *Segmentation of color images of plants with a Markovian Mean Shift* ” pada tahun 2009. Peneliti mengusulkan penyederhanaan komputasi algoritma mean shift dengan menambahkan metode *markovian*. Dalam mensegmentasi citra, algoritma *mean shift* akan membandingkan antara sebuah piksel warna dengan warna pada *colormetric space* yang merupakan densitas tertinggi dari sebuah warna tertentu. Tiap piksel akan dibandingkan satu per satu. Apabila citra mempunyai ukuran 500 x 500 piksel, maka akan ada 250000 piksel yang harus dibandingkan padahal sering terjadi piksel yang berdekatan mempunyai densitas warna yang hampir sama. Metode

*markovian* digunakan untuk menyederhanakan komputasi *mean shift* karena piksel yang berdekatan akan dikomputasi terlebih dahulu sehingga akan mengurangi jumlah iterasi dari algoritma *mean shift* secara signifikan.

4. Penelitian yang dilakukan oleh Paris *et al* [17] dengan judul “*A Topological Approach to Hierarchical Segmentation using Mean Shift*” pada tahun 2010. Penelitian ini menggunakan pendekatan topologi dengan menggunakan *mean shift*. Hasil yang didapatkan adalah algoritma *mean shift* mampu mendeteksi tepi obyek yang rumit dan kabur dari sebuah citra sehingga batas-batas obyek terlihat dengan jelas.
5. Penelitian yang dilakukan oleh Cour *et al* [13] dengan judul “*Spectral Segmentation with Multiscale Graph Decomposition*” pada tahun 2005. Penelitian ini mengusulkan penggunaan algoritma *normalized cuts* untuk melakukan segmentasi citra dengan cara menghitung tepi area dari region yang saling bersinggungan. Hasil yang didapatkan adalah citra yang terbagi menjadi beberapa *region tersegmen*.
6. Penelitian yang dilakukan oleh Tao *et al* [14] dengan judul “*Color Image Segmentation Based on Mean Shift and Normalized Cuts*” pada tahun 2007. Penelitian ini menfokuskan perhatian kepada penggunaan algoritma *normalized cuts* dalam mensegmentasi citra berwarna dan didapatkan hasil bahwa *normalized cuts* mampu mensegmentasi citra dengan jumlah area *tersegmen* yang lebih moderat.
7. Penelitian yang dilakukan oleh Peng *et al* [11] dengan judul “*Automatic Image Segmentation by Dynamic Region Merging*” pada tahun 2009. Penelitian ini menggunakan metode *dynamic region merging* dengan menghitung kesamaan antara dua area yang *tersegmen* berdekatan. Area *tersegmen* dibuat dengan algoritma *watershed*. Algoritma *watershed* membagi citra menjadi banyak area kecil dengan luas yang sama. Area ini selanjutnya akan dikomputasi dengan metode *dynamic region merging*. Area yang berdekatan dan mempunyai kesamaan warna akan digabung. Hasil yang dilaporkan adalah berhasil didapatkan adalah obyek dapat dipisahkan dari *background* di sekelilingnya.
8. Penelitian yang dilakukan oleh Ning *et al* [5] dengan judul “*Interactive Image Segmentation by Maximal Similarity based Region Merging*” pada tahun 2010. Penelitian ini menggunakan perhitungan *maximal similarity* pada *region-region* *tersegmen* dengan menggunakan histogram warna dalam melakukan *region merging* atau penyatuan area *region* tersebut. Proses segmentasi dilakukan secara *semi-otomatis* artinya membutuhkan sedikit bantuan

*input* dari *user*. Input berupa *marker* penanda obyek dan *marker* penanda *background* pada citra. Proses *region merging* dilakukan dengan menggunakan bantuan penanda atau *marker* tersebut. Hasil yang didapatkan berupa obyek yang berhasil dipisahkan secara akurat dari *background* di sekelilingnya.

Pada penelitian ini diusulkan metode segmentasi *semi*-otomatis menggunakan *region merging maximal similarity* dengan input menggunakan citra tersegmen hasil segmentasi *low level* integrasi antara algoritma *mean shift* dan *normalized cuts* untuk memisahkan obyek pada *background* yang kompleks secara lebih akurat.

## **2.2 Landasan Teori**

### **2.2.1 Segmentasi Citra**

Segmentasi citra adalah proses yang penting dan hasil yang didapatkan dari proses ini digunakan di dalam banyak aplikasi pengolahan citra tingkat lanjut. Ghost *et al* [2] mengatakan bahwa segmentasi citra adalah membagi citra menjadi bagian-bagian yang homogen berdasarkan pola, warna atau tepi gambar. Segmentasi citra dapat berarti memisahkan obyek dari *background* sehingga obyek tersebut dapat digunakan untuk keperluan lain. Segmentasi membagi citra ke dalam bagian-bagian yang membentuknya dimana setiap bagian tergantung dari permasalahan yang akan dipecahkan. Proses segmentasi citra diperlukan untuk memproses sebuah obyek misalnya di dalam aplikasi pengenalan obyek, pengenalan tulisan, deteksi wajah dan lain sebagainya.

Berbagai metode telah banyak dikembangkan dalam kajian segmentasi citra. Berbagai fitur di dalam gambar misalnya tekstur, bentuk, tingkat intensitas keabuan dan warna merupakan komponen yang sering digunakan di dalam segmentasi citra [18]. Tetapi secara umum tidak ada metode yang berhasil untuk mensegmentasi semua jenis gambar. Setiap metode mempunyai kelebihan dan kekurangannya masing-masing.

Citra berwarna memiliki informasi yang lebih banyak dibandingkan dengan citra *grayscale* sehingga informasi ini dapat digunakan untuk mendapatkan kualitas hasil lebih baik dalam proses segmentasi. Tetapi citra berwarna juga meningkatkan kompleksitas komputasi.

Segmentasi citra berwarna yang kompleks merupakan salah satu tugas yang sulit dilakukan dalam pengolahan citra.

Beberapa teknik segmentasi dapat dipaparkan sebagai berikut [18]:

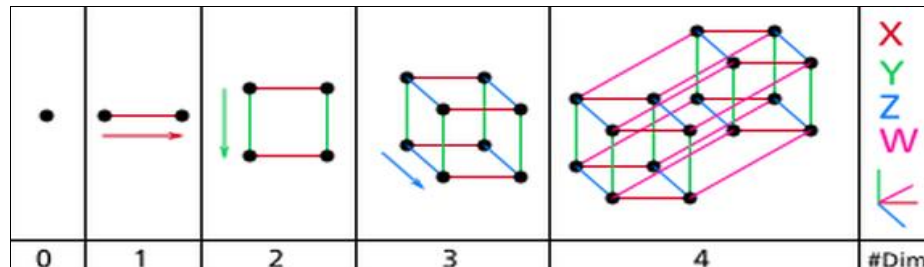
1. *Pixel based segmentation*: yaitu pendekatan segmentasi berbasis piksel atau titik piksel dan termasuk dalam pendekatan yang paling sederhana yang digunakan dalam proses segmentasi. Klasifikasi berbasis piksel dapat menggunakan *supervised* maupun *unsupervised classifier* yang tidak membutuhkan informasi awal tentang citra yang akan disegmentasi.
2. *Region based segmentation*: yaitu pendekatan segmentasi berbasis area. Metode ini mempunyai fokus perhatian kepada aspek penting dalam proses segmentasi. Segmentasi berbasis area membagi citra kepada area-area tersegmentasi yang identik dengan berdasarkan kepada satu set kriteria yang telah didefinisikan. Segmentasi dengan konsep perluasan region atau *region growing*, pemecahan region atau *region splitting* dan penyatuan area atau *region merging* adalah contoh segmentasi berbasis area.
3. *Edge based segmentation*: yaitu pendekatan segmentasi berbasis tepi dan biasanya digunakan untuk mensegmentasi obyek tanpa menggunakan skema pengembangan nilai yang kompleks.

Tujuan akhir dari segmentasi adalah menyederhanakan dan atau merubah representasi suatu citra ke dalam gambaran yang lebih mempunyai arti dan lebih mudah untuk dianalisa. Segmentasi citra secara khusus digunakan untuk melokalisasi objek atau batas, bisa berupa garis, kurva dan lain-lain dalam citra. Hasil dari segmentasi citra adalah sekumpulan wilayah yang melingkupi citra tersebut, atau sekumpulan kontur yang diekstrak dari citra. Tiap piksel dalam suatu wilayah mempunyai kesamaan karakteristik atau properti seperti warna (*color*), intensitas (*intensity*), dan tekstur (*texture*).

### **2.2.2 Algoritma Mean Shift**

Algoritma *mean shift* adalah teknik analisis *non-parametric feature space* atau analisis ruang fitur berbasis ruang non-parametrik [19]. Di dalam studi pengolahan citra dan pengenalan pola, *feature space* adalah ruang abstrak di dalam citra dimana setiap contoh pola digambarkan sebagai sebuah titik di dalam ruang dimensi  $n$  atau *n-dimensional space*. Besar dari dimensi

tersebut ditentukan oleh jumlah dari *feature* yang digunakan untuk mendeskripsikan pola. Pola yang sama dikelompokkan sehingga dapat dibuat sebuah estimasi kerapatan atau *density estimation* sehingga dapat ditemukan pola selanjutnya. Konsep ini biasanya digunakan dalam teknik klasifikasi. Ruang berdimensi  $n$  dapat digambarkan sebagai berikut:



Gambar 2.1 Dimensi  $n$  dari sebuah ruang

Di dalam studi statistik tiap data memiliki *standart deviasi* atau penyimpangan dari nilai yang telah diasumsikan di awal. Metode *parametrik* mengasumsikan bahwa tiap data akan memiliki *standart deviasi*. Contohnya adalah apabila kita mempunyai 100 buah data, dan dari 99 data yang diambil memiliki penyimpangan sebesar 2.55 maka data yang ke 100 juga akan tetap dihitung dengan mengkalkulasi berapa penyimpangannya. Sedangkan dengan metode non-parametrik, data yang ke 100 tersebut tidak perlu dihitung karena otomatis diasumsikan memiliki penyimpangan 2.55.

*Mean shift* adalah prosedur untuk menentukan *density function* atau fungsi kerapatan dari sebuah data [12]. Berdasarkan penamaannya, maka *function* pertama adalah *mean* artinya fungsi ini digunakan untuk menentukan *modes* atau data yang paling sering keluar atau mean dari sebuah kumpulan data. Sedangkan *function* kedua adalah *shift*, artinya sebuah pergerakan menuju area *modes* tersebut.

Di dalam algoritma *mean shift*, *density function* atau kerapatan sebuah data berhubungan sangat erat dengan analisa ruang fitur atau *feature space analysis* pada sebuah citra. Warna di dalam sebuah citra *RGB* akan dipetakan menjadi ruang warna  $L*u*v$  tiga dimensi.  $L$  adalah komponen *luminans* dan  $u$  serta  $v$  adalah komponen krominan. Komponen *luminans* dapat dianggap sebagai skala abu-abu dari sebuah citra dengan ruang warna *RGB*. Sedangkan nilai  $u$  dan  $v$  berisi informasi warna atau krominan.

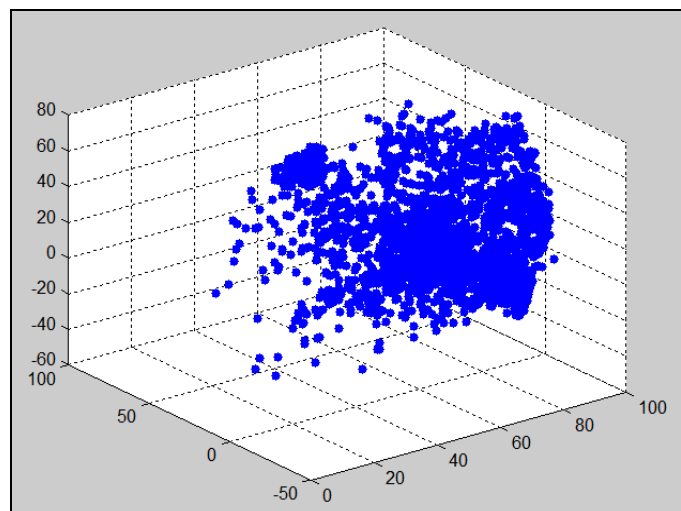
Ada transisi yang berhubungan dengan pengelompokan yang timbul dari warna yang dominan dan sebuah dekomposisi ruang akan terjadi. Ruang fitur yang tidak jelas strukturnya, misalnya pada gambar yang sangat kompleks dapat dianalisa dan dikelompokkan atau *clustering* dengan metode *non-parametrik* karena metode tersebut tidak memiliki asumsi yang melekat. Banyak metode clustering *non parametric* telah diteliti dan secara umum metode ini dibagi menjadi dua bagian besar, yaitu pengklasteran hirarkis dan estimasi kepekatan.

Pengklasteran hirarkis adalah memisahkan atau menggabungkan data berdasarkan ukuran terdekatnya. Sedangkan estimasi kepekatan adalah metode pengklasteran yang berbasis fungsi *kernel density estimation* (KDE) dari parameter yang telah direpresentasikan. Daerah-daerah yang padat dari ruang fitur dengan demikian berhubungan dengan sebuah *function* setempat yaitu menuju sebuah moda kepadatan yang tidak diketahui. Begitu lokasi moda tersebut ditentukan maka sebuah kluster yang berhubungan dengannya digambarkan berdasarkan pada struktur lokal dari ruang fitur.

Contoh dari sebuah ruang fitur berdasarkan kepekatan warna adalah sebagai berikut:



(a)



(b)

Gambar 2. 2 Contoh sebuah ruang fitur, (a) citra berwarna (b) ruang warna  $L^*u^*v$  yang berhubungan dengan titik data

Gambar di atas adalah sebuah *plotting 3D* dari sebuah citra dengan mode warna  $L^*u^*v$  dengan menggunakan MATLAB dan menggambarkan kepekatan warna dan hubungannya dengan titik

data yang membentuknya. Sebuah fitur analisis berbasis citra adalah sebuah paradigma baru untuk mencapai tujuan yang telah ditetapkan. Sebuah ruang fitur pemetaan *input* dapat diperoleh dari pengolahan *subset* kecil data pada sebuah waktu tertentu. Untuk setiap *subset*, sebuah representasi parametrik dari sebuah fitur adalah hal menarik yang diperoleh dan hasilnya dapat dipetakan kepada ke dalam sebuah ruang dimensi parameter. Setelah semua *input* diproses maka sebuah hubungan korespondensi pada sebuah *region* citra akan diperoleh yaitu sebuah kluster. Penghitungan dari kluster-kluster tersebut adalah penggambaran dari kelompok besar yang dianalisa.

Algoritma *mean shift* menggunakan perhitungan *kernel density estimation* (KDE). Fungsi ini untuk menghitung data yang paling sering keluar dalam sebuah area. Rumus KDE yang digunakan adalah sebagai berikut:

$$f(x) = \frac{1}{n} \sum_{i=1}^n K_H(x - x_i) \quad (2.1)$$

dimana:

$X$ : perulangan dari panjang  $n$

$K$ : *kernel* simetrikal

$H$ : parameter *smoothing*

Dalam hal segmentasi citra fungsi dari *kernel density estimation* akan menghitung densitas warna yang paling sering muncul dalam sebuah area tertentu. Pada sebuah luas area citra tertentu atau luas ruang fitur tertentu ditemukan bahwa warna yang paling sering keluar adalah warna merah, maka pada area tersebut semua warna akan diganti dengan warna merah. Inilah makna dari *shift* atau bergerak menuju nilai *mean* dalam hal ini adalah warna yang paling sering keluar yang ditemukan pada sebuah area citra. Begitu juga hal yang sama akan dilakukan terhadap area citra yang lain sehingga apabila hal ini secara berulang terhadap semua area citra maka hasil yang akan didapatkan adalah citra yang terbagi menjadi area-area atau *region* dengan warna yang homogen. Warna yang ada pada tiap *region* adalah representasi dari warna yang dominan pada area tersebut.

Contoh sebuah citra dengan ukuran 4x4 piksel seperti dibawah ini:

|    |    |    |    |
|----|----|----|----|
| 1  | 2  | 3  | 4  |
| 5  | 6  | 7  | 8  |
| 9  | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

Gambar 2. 3 Citra berukuran 4x4 piksel

Citra di atas apabila disegmentasi dengan algoritma mean shift dapat menghasilkan hasil segmentasi yang berbeda, tergantung dari berapa luas ruang fitur yang digunakan sebagai area pengukuran. Ruang fitur secara sederhana dapat dianggap sebagai berapa ukuran piksel minimal yang akan digunakan sebagai acuan pengukuran. Bila ada kerapatan warna yang jumlahnya dibawah ruag fitur minimal, warna tersebut akan diabaikan dan dianggap sebagai bagian dari warna yang dominan yang paling dekat. Misalnya apabila akan digunakan ruang fitur 2, 3, 5 maka akan menghasilkan citra segmentasi yang berbeda, yaitu:

|    |    |    |    |
|----|----|----|----|
| 1  | 2  | 3  | 4  |
| 5  | 6  | 7  | 8  |
| 9  | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

(a)

|    |    |    |    |
|----|----|----|----|
| 1  | 2  | 3  | 4  |
| 5  | 6  | 7  | 8  |
| 9  | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

(b)

|    |    |    |    |
|----|----|----|----|
| 1  | 2  | 3  | 4  |
| 5  | 6  | 7  | 8  |
| 9  | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

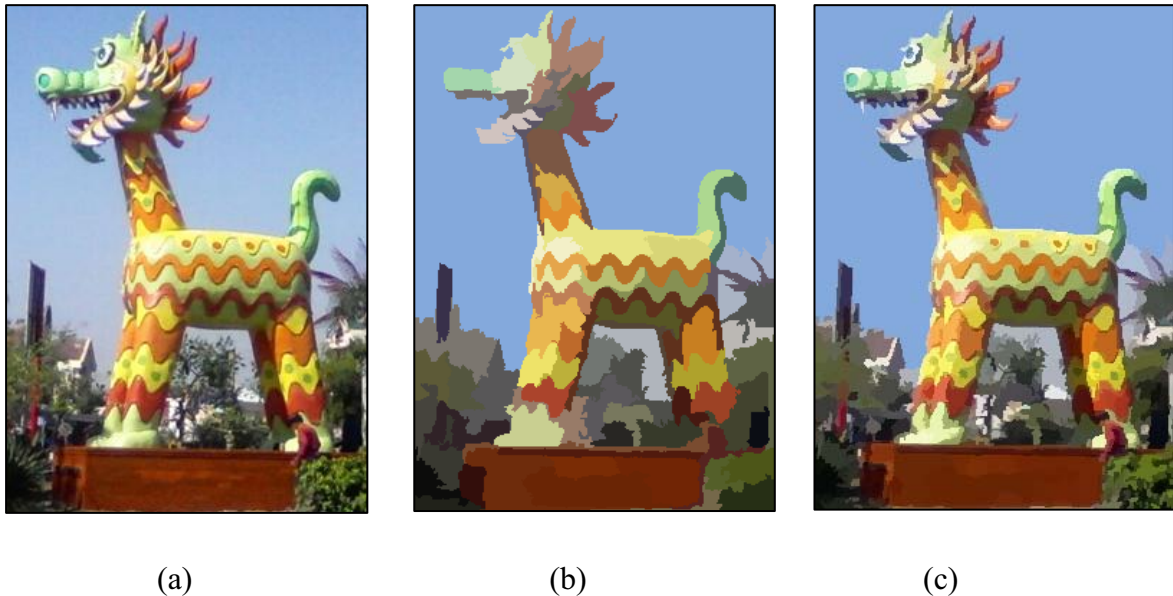
(c)

Gambar 2. 4 Segmentasi *mean shift* (a) ruang fitur 2 , (b) 3 dan (c) 5

Di dalam algoritma *mean shift*, perhitungan *density* atau kepekatan dalam sebuah area akan sangat terpengaruh kepada pada di bagian mana kepekatan tersebut diukur dan pada seberapa luas fitur ruang kepekatan tersebut diukur.



Contoh hasil segmentasi algoritma *mean shift* dengan basis perhitungan ruang fitur ruang yang berbeda adalah sebagai berikut:



Gambar 2. 5 Segmentasi citra dengan algoritma *mean shift*, (a) citra , (b),(c) hasil segmentasi dengan luas fitur ruang yang berbeda

Luas ruang fitur yang digunakan sebagai basis perhitungan *density* pada gambar (b) adalah besar. Hasil yang didapatkan adalah bahwa area warna yang mempunyai luas kecil ‘tertelan’ oleh warna di sekitarnya yang lebih dominan. Sedangkan pada gambar (c) dengan luas ruang fitur yang lebih kecil maka warna yang hilang tersebut berhasil dimunculkan. Dari contoh hasil segmentasi di atas dapat terlihat bahwa semakin kecil ruang fitur yang digunakan sebagai basis perhitungan pada algoritma *mean shift* maka hasil segmentasi akan semakin detail dalam memisahkan masing-masing warna. Tetapi hal ini juga akan meningkatkan kompleksitas dari proses yang akan dilakukan selanjutnya terhadap citra tersebut.

### 2.2.3 Algoritma *Normalized Cuts*

Algoritma *normalized cuts* diperkenalkan oleh Shi *et al* [20] dan merupakan pengembangan dari algoritma berbasis *graph*. Graph adalah salah satu pokok bahasan dalam matematika diskret yang telah lama dikenal dan banyak diaplikasikan pada berbagai bidang. Secara umum, *graph*  $G$  didefinisikan sebagai pasangan himpunan  $(V,E)$ , ditulis dengan notasi  $G=(V,E)$  yang dalam hal ini  $V$  adalah himpunan tidak kosong dari simpul-simpul (*nodes*) dan  $E$  adalah himpunan sisi/busur (*edges*) yang menghubungkan sepasang simpul. Umumnya *graph* digunakan untuk

memodelkan suatu masalah menjadi lebih mudah, yaitu dengan cara merepresentasikan obyek-obyek tersebut. *Graph* dapat dikelompokkan menjadi beberapa jenis tergantung dari sudut pandang pengelompokkannya. Pengelompokan *graph* dapat dipandang berdasarkan dari ada tidaknya sisi ganda, berdasarkan jumlah simpul, atau berdasarkan orientasi arah pada sisi *graph*.

Ada beberapa sifat yang berkaitan dengan *graph*, yaitu :

1. Bertetangga (*adjacent*)

Dua buah simpul pada *graph* tak terarah  $G$  dikatakan bertetangga bila keduanya terhubung langsung pada sebuah sisi.

2. Bersisian (*incident*)

Untuk sembarang sisi  $e=(u,v)$  sisi  $e$  dikatakan bersisian dengan simpul  $u$  dan simpul  $v$ .

3. Derajat (*degree*)

Derajat suatu simpul pada *graph* tidak berarah adalah jumlah sisi yang bersisian pada simpul tersebut.

4. Lintasan (*path*)

Lintasan sepanjang  $n$  dari simpul awal  $v_0$  dan ke simpul tujuan  $v_n$  di dalam *graph*  $G$  adalah barisan berselang-seling simpul dan merupakan sisi-sisi yang berbentuk.

5. Sirkuit (*circuit*) atau siklus (*cycle*)

Lintasan yang berawal dan berakhir pada simpul yang sama disebut sirkuit atau siklus.

6. Terhubung (*connected*)

*Graph* tak berarah  $G$  disebut *graph* terhubung (*connected graph*) jika untuk setiap pasang simpul  $u$  dan  $v$  di dalam himpunan  $V$  terdapat lintasan dari  $u$  ke  $v$ .

7. Berbobot (*weighted graph*)

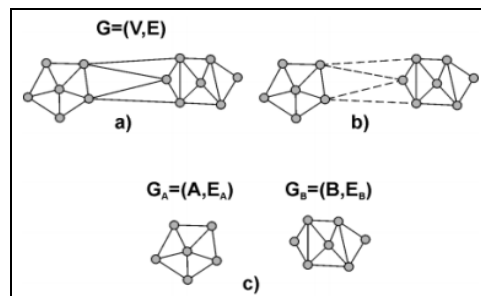
*Graph* berbobot adalah *graph* yang setiap sisinya diberikan nilai atau bobot.

### 2.2.3.1 Pemotongan *Graph*

Citra digital dapat dianggap sebagai *graph* yang berbobot  $G=(V, E)$  dimana pikselnya direpresentasikan oleh simpul-simpul  $v_i \in V$  dan setiap pasangan simpul yang berdekatan  $v_i$  dan  $v_j$  dihubungkan dengan sebuah sisi  $e_{ij}=\{v_i, v_j\} \in E$ . Masing-masing sisi memiliki bobot non-negatif  $w(i,j)$  yang menggambarkan kemiripan atau ketidakmiripan antara simpul insiden. Dengan memperhatikan representasi citra yang berbasis *graph*, segmentasi citra semata-mata merupakan penyekatan dari graf  $G=(V,E)$  menjadi dua rangkaian yang tidak berhubungan  $A$  dan

$B$  dimana  $A \cup B = V$  dan  $A \cap B = \emptyset$ . Penyekatan dilakukan menurut beberapa kriteria dan bertujuan untuk menghilangkan sisi-sisi yang menghubungkan *graph* kecil A dan B. Rangkaian yang terpisah ini hanya dengan membuang tepian-tepian yang menghubungkan kedua bagian tersebut. Tingkat ketidakmiripan diantara kedua bagian tadi dapat dihitung sebagai bobot keseluruhan dari tepian yang telah dibuang tadi. Dalam bahasa teori *graph*, hal ini disebut sebagai sebuah potongan.

Penggambarannya adalah sebagai berikut:



Gambar 2.6 Pemotongan *graph*, (a) *input graph*, (b) tepi yang akan dipotong, (c) hasil pemotongan [21]

Total bobot dari sisi yang harus dibuang selama penyekatan *graph* disebut *cut* (potongan) dan ditentukan dengan persamaan sebagai berikut:

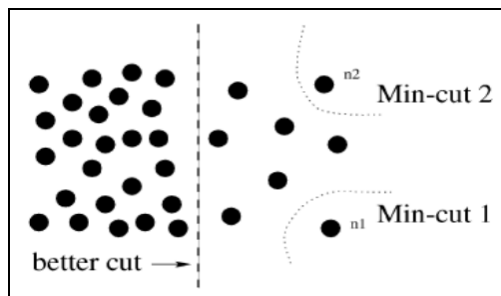
$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v) \quad (2.2)$$

Penyekatan ganda yang optimal dari sebuah *graph* adalah yang dapat meminimalkan nilai potongan ini. Meski ada sejumlah penyekatan seperti itu yang eksponensial, mencari potongan minimal dari sebuah grafis adalah sebuah masalah yang harus diteliti.

Wu dan Leahy [22] menyarankan sebuah metode pengklasteran yang berdasar pada kriteria potongan minimal ini. Secara khusus, mereka mencoba untuk menyekat sebuah *graph* menjadi *graph-graph* kecil karenanya potongan maksimal sepanjang kelompok-kelompok kecil tersebut diminimalkan. Masalahnya dapat diselesaikan dengan efisien dengan terus menerus mencari potongan minimal yang membelah segmen-segmen yang ada. Kriteria optimal global ini dapat digunakan untuk menghasilkan segmentasi yang bagus pada beberapa gambar.

Akan tetapi kriteria potongan minimal mendukung untuk pemotongan serangkaian kecil dari tangkai-tangkai yang terpisah dari *graph* tersebut. Hal ini tidaklah mengejutkan karena potongan naik seiring jumlah dari tepian yang melintang di kedua bagian yang tersekat. Dengan menganggap bobot tepian yang proporsional secara terbalik dibanding jarak antara kedua tangkai, kita dapat melihat potongan yang menyekat tangkai  $n_1$  atau  $n_2$  akan memiliki sebuah nilai yang kecil. Pada kenyataannya, potongan apapun yang menyekat tangkai-tangkai yang terpisah di bagian kanan akan memiliki potongan dengan yang lebih kecil dibanding potongan yang menyekat tangkai pada bagian kiri dan kanan.

Contoh dari sebuah potongan adalah sebagai berikut:



Gambar 2.7 Contoh sebuah potongan [22]

Untuk menghindari bias yang tidak alami untuk menyekat serangkaian titik yang kecil, sebuah pengukuran yang baru diasosiasikan diantara kedua kelompok tersebut. Perhitungan ini tidak menilai bobot secara keseluruhan pada kedua sekat, tetapi menghitung potongan secara pecahan dari keseluruhan koneksi tepi ke semua tangkai pada *graph* tersebut.

Pengukuran ini disebut dengan potongan yang dinormalkan atau *normalized cuts* dan dirumuskan sebagai berikut:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} \quad (2.3)$$

Dimana  $assoc(A, B)$  dan  $assoc(B, B)$  adalah bobot keseluruhan yang menghubungkan tangkai-tangkai di dalam A dan B, secara masing-masing. Hal ini adalah sebuah perhitungan yang tidak berbias yang mencerminkan bagaimana ketatnya tangkai-tangkai yang sedang terhubung secara ketat satu dengan lainnya. Sifat lain yang terhubung dari definisi asosiasi dan diasosiasi dari sebuah penyekat adalah bahwa keduanya terhubung secara alami:

$$\begin{aligned}
Ncut(A, B) &= \frac{cut(A, B)}{accoc(A, V)} + \frac{cut(A, B)}{accoc(B, V)} \\
&= \frac{cassoc(A, V) - assoc(A, A)}{accoc(A, V)} + \frac{assoc(B, V) - assoc(B, B)}{accoc(B, V)} \\
&= 2 - \left( \frac{assoc(A, A)}{assoc(A, V)} \right) + \left( \frac{assoc(B, B)}{assoc(B, V)} \right) \\
&= 2 - N \cdot assoc(A, B)
\end{aligned}$$

Dengan demikian kedua kriteria penyekat yang dicari dalam algoritma *normalized cuts* adalah meminimalkan disosiasi antar kelompok dan memaksimalkan asosiasi dalam kelompok tersebut dan hal ini sebenarnya adalah identik dan dapat dihasilkan secara simultan. *Normalized cuts* adalah teknik membagi dengan cara menghitung nilai cut pada *graph* yang dihasilkan dari citra dan membaginya dengan total *edge* pada *graph*.

### 2.2.3.2 Pembobotan Sisi

Bobot  $w(i, j)$  diberikan pada sisi-sisi mencerminkan kemiripan antara *region-region* yang menyusun sebuah citra. Semakin besar bobot semakin tinggi kemiripannya antara *region-region* yang berhubungan dan sebaliknya. Untuk deskripsi kemiripan *region* ukuran digunakan perhitungan yang dilakukan Shi *et al* [20], yaitu:

$$w(i, j) = ae^{-\frac{\|F(i) - F(j)\|_2^2}{\alpha_1^2}} \quad (2.4)$$

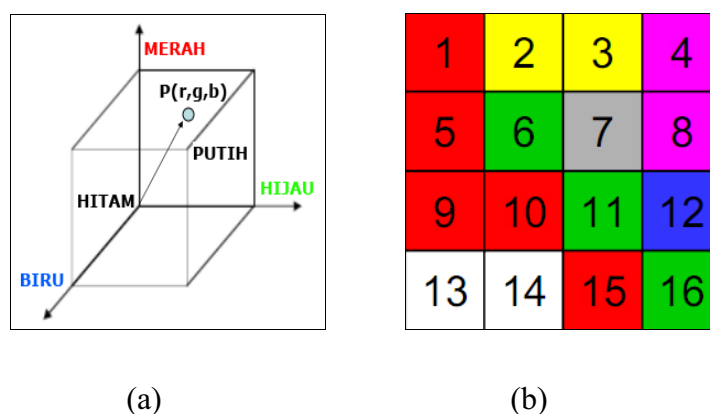
Dimana  $F(i)$  dan  $F(j)$  adalah vektor fitur yang menggambarkan *region-region* yang berhubungan dengan  $i$  dan  $j$ . Adapun langkah pertama yang dilakukan dalam membuat sebuah *graph*  $G=(V, E)$ , dengan mengambil masing-masing piksel sebagai sebuah simpul yang menghubungkan pasangan piksel dengan sebuah sisi. Bobot dari sisi tersebut harus mencerminkan kemungkinan bahwa kedua piksel tadi menjadi sebuah bagian dari satu obyek. Bobot sisi *graph* dapat diambil dengan menggunakan nilai kecerlangan (*brightness*) dari piksel dan lokasi spasial piksel.

Selanjutnya adalah menyelesaikan nilai  $(D - W)x = \lambda Dx$  untuk *eigen vektor* dengan *eigen value* yang terkecil. *Eigenvektor* memerlukan operasi  $O(n^3)$ , dimana  $n$  adalah jumlah dari simpul dalam *graph*. *Eigenvektor* dan *eigen value* adalah istilah yang digunakan dalam aljabar linier, persamaan diferensial linier, matriks linier dan transformasi linier. *Eigenvektor* dan *eigen value*

digunakan untuk mencari solusi umum dan solusi khusus, pengujian serta pembuktian linieritas dari suatu persamaan. Persamaan dari *eigenvektor* dan *eigen value* adalah  $Ax = A\lambda$  dimana  $A$  adalah matriks dari suatu persamaan,  $x$  adalah *eigenvektor* yang tidak boleh bernilai 0 dan  $\lambda$  adalah *eigen value* yang merupakan besaran skalar.

Setelah *eigenvektor* berhasil dihitung maka *graph* dapat disekat menggunakan *eigenvektor* terkecil selanjutnya. Dalam kasus ideal, *eigenvektor* hanya menggunakan dua nilai berbeda dan tanda-tanda dari nilai tersebut dapat memberi tahu dengan tepat bagaimana untuk menyekat *graph* tersebut. Meskipun demikian *eigenvektor* dapat menggunakan nilai-nilai yang tidak terputus sehingga perlu pemilihan titik pemotong untuk menyekatnya menjadi dua bagian. Setelah *graph* berhasil dipotong menjadi dua, lanjutkan perulangan pemotongan pada dua bagian yang tersekat tadi. Perulangan pemotongan akan berhenti ketika nilai  $Ncut$  telah melebihi batasan tertentu.

Setiap piksel warna yang dinyatakan dalam mode warna RGB (*red, green, blue*) atau merah, hijau dan biru sehingga merupakan gabungan dari nilai R, nilai G dan nilai B yang tidak bisa dipisahkan satu dengan lainnya. Hal ini dapat dituliskan sebagai  $P(r,g,b)$ . Misalnya nilai putih dinyatakan dengan nilai  $P(255,255,255)$ . Nilai hitam dinyatakan dengan  $P(0, 0, 0)$ . Nilai merah  $P(255, 0, 0)$ , hijau sebagai  $P(0, 255, 0)$  dan biru  $P(0,0, 255)$ . Sedangkan warna yang lain dinyatakan dengan kombinasi nilai dari nilai RGB tersebut, misalnya warna abu-abu dinyatakan sebagai  $P(128, 128, 128)$  artinya warna abu-abu merupakan gabungan dari 3 warna diatas dengan skala yang sama.



Gambar 2. 8 (a) mode warna RGB , (b) citra RGB berukuran 4x4 piksel

Kode warna dari citra di atas adalah sebagai berikut:

1. Merah : (255, 0, 0)
2. Kuning : (255, 255, 0)
3. Putih : (255, 255, 255)
4. Hijau : (0, 255, 0)
5. Biru : (0, 0, 255)
6. Abu-abu : (128, 128, 128)
7. Magenta : (255, 0, 255)

Pada segmentasi dengan algoritma *normalized cuts* setiap citra akan diubah terlebih dahulu ke dalam mode warna abu-abu atau *greyscale*. Sebuah citra dengan mode warna RGB dapat diubah menjadi citra *greyscale*. Citra *greyscale* dinyatakan dengan nilai 0 – 255, dengan 0 adalah representasi dari warna hitam, dan 255 adalah representasi warna putih. Nilai 1 – 254 adalah berada di antara warna hitam dan putih. Rumus yang populer digunakan untuk merubah citra mode warna RGB ke dalam citra *greyscale* adalah:

$$greyscale = 0.3 R + 0.5 G + 0.2 B \quad (2.5)$$

Sebagai contoh, warna magenta dinyatakan dalam mode warna RGB adalah (255,255,0) sehingga apabila diubah menjadi citra *greyscale* nilainya adalah  $(0.3 \times 255) + (0 \times 255) + (0.2 \times 255) = 128$ . Apabila citra RGB seperti pada gambar 2.8b diubah menjadi citra *greyscale* maka nilainya adalah sebagai berikut:

|                         |       |     |     |     |     |
|-------------------------|-------|-----|-----|-----|-----|
| Merah (255, 0, 0)       | = 76  | 76  | 204 | 204 | 128 |
| Kuning (255, 255, 0)    | = 204 | 76  | 128 | 128 | 128 |
| Putih (255, 255, 255)   | = 255 | 76  | 76  | 128 | 51  |
| Hijau (0, 255, 0)       | = 128 | 255 | 255 | 76  | 128 |
| Biru (0, 0, 255)        | = 51  |     |     |     |     |
| Abu-abu (128, 128, 128) | = 128 |     |     |     |     |
| Magenta (255, 255, 0)   | = 128 |     |     |     |     |

Gambar 2. 9 Citra *greyscale*

*Edge* yang menghubungkan piksel (simpul)  $p$  dan  $q$  pada *graph* memiliki bobot (*weight graph*) seperti ditunjukkan pada rumus berikut:

$$w((p, q)) = H - [f(p) - f(q)] \quad (2.6)$$

dimana:

$H$  : nilai intensitas keabuan tertinggi pada citra

$f(p)$  dan  $f(q)$  : nilai keabuan piksel  $p$  dan  $q$

Pada citra *greyscale* pada gambar 2.9, tiap nilai keabuan pada setiap titik piksel akan dibandingkan dengan simpul yang membentuk atau dengan piksel disekitarnya. Sebagai contoh pada piksel pada koordinat (1,1), yaitu yang bernilai intensitas keabuan 76 akan dibandingkan dengan titik di sebelah kanan dan disebelah bawahnya yang intensitas 204 dan 76, sehingga nilai bobotnya adalah:

$$\begin{aligned} W_{(1,1)} &= 255 - (76-204) + 255 - (76-76) \\ &= (255 - (-128) + 255) \\ &= 638 \end{aligned}$$

Sedangkan bobot pada koordinat (1,2) adalah:

$$\begin{aligned} W_{(1,2)} &= 255 - (204-76) + 255 - (204-128) + 255 - (204-204) \\ &= 127 + 179 + 255 \\ &= 561 \end{aligned}$$

Sedangkan bobot pada koordinat (1,3) adalah:

$$\begin{aligned} W_{(1,3)} &= 255 - (204-204) + 255 - (204-128) + 255 - (204-128) \\ &= 255 + 179 + 179 \\ &= 613 \end{aligned}$$

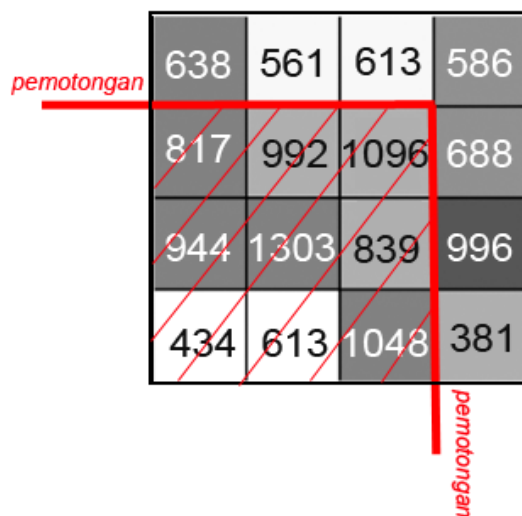


Keseluruhan bobot dari contoh citra pada gambar 2.9 adalah sebagai berikut:

|     |      |      |     |
|-----|------|------|-----|
| 638 | 561  | 613  | 586 |
| 817 | 992  | 1096 | 688 |
| 944 | 1303 | 839  | 996 |
| 434 | 613  | 1048 | 381 |

Gambar 2. 10 Bobot setiap piksel citra

Dari bobot citra seperti pada gambar 2.10, pemotongan dilakukan berdasarkan nilai bobot, yaitu bahwa simpul dengan bobot yang hampir sama akan dikelompokkan dalam potongan yang sama, sedangkan perbedaan antara dua potongan citra diambil dengan nilai bobot yang terbesar. Hasil pemotongannya adalah sebagai berikut:



|     |      |      |     |
|-----|------|------|-----|
| 638 | 561  | 613  | 586 |
| 817 | 992  | 1096 | 688 |
| 944 | 1303 | 839  | 996 |
| 434 | 613  | 1048 | 381 |

Gambar 2. 11 Pemotongan berdasarkan nilai bobot

Teknik *graph* yang digunakan untuk masalah segmentasi pada umumnya merepresentasikan masalah *graph*  $G=(V,E)$  dimana setiap simpul  $v_l \in V, l=1,2,3,...,n$  dianggap sebagai piksel-piksel dari citra dan sisi  $E$  merupakan pasangan-pasangan dari piksel-piksel yang bertetangga. Setiap

sisi yang menghubungkan piksel-piksel pada citra memiliki bobot (*weight*) yaitu nilai intensitas keabuan pada citra itu sendiri.

Proses segmentasi dalam pendekatan pada *graph* dilakukan melalui proses partisi *graph* yaitu dilakukan pengelompokan dengan cara mempartisi himpunan dari simpul-simpulnya. Segmentasi  $S$  merupakan partisi dari  $V$  ke dalam komponen-komponen  $C$  sehingga setiap  $C \in S$  bersesuaian dengan komponen yang terhubung dalam sebuah *graph*  $G'=(V,E')$  dimana  $E' \geq E$ . Dengan kata lain segmentasi yang dilakukan diinduksi oleh sebuah subset dari sisi  $E$ .

Hasil segmentasi citra dengan algoritma *normalized cuts* yang menerapkan teori *graph* adalah sebagai berikut:



Gambar 2.12 Hasil segmentasi *normalized cuts*

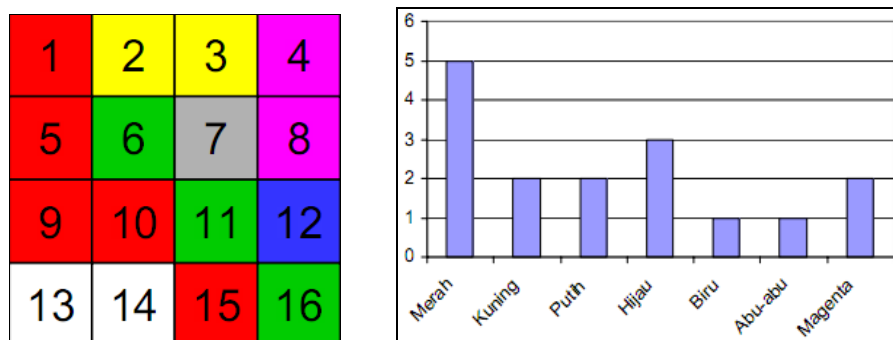
Ada beberapa cara dalam menentukan kualitas dari segmentasi, tetapi pada umumnya adalah elemen yang sama akan dikelompokkan dalam bidang yang sama dan elemen yang berbeda akan dikelompokkan dalam bidang yang lain. Ini berarti bahwa sisi di antara dua simpul di komponen yang sama harus mempunyai bobot yang rendah dan sisi diantara simpul-simpul yang berbeda komponen harus mempunyai bobot yang tinggi. Tujuan dari mempartisi citra ke dalam himpunan yang berbeda adalah untuk memperoleh struktur dari citra tersebut, untuk merepresentasikan citra sehingga menjadi lebih kompak, serta untuk mengetahui perbedaan struktur citra yang memiliki level tinggi maupun level rendah.

#### 2.2.4 Segmentasi Obyek dengan Metode *Region Merging*

Metode *region merging* adalah kebalikan dari proses membagi citra. *Region merging* bertujuan untuk menyatukan area yang ada pada citra menurut tujuan yang diinginkan. Dalam hal segmentasi obyek, tujuannya adalah memisahkan obyek dari *background* di sekeliling obyek. Segmentasi obyek dengan menggunakan metode *region merging* membutuhkan input berupa citra hasil segmentasi *low level* yaitu citra yang telah terbagi menjadi serangkaian *region* atau area. Contoh algoritma untuk melakukan segmentasi *low level* adalah *watershed*, *mean shift* dan *normalized cuts*. Algoritma *watershed*, *mean shift* dan *normalized cuts* akan membagi citra menjadi serangkaian area kecil atau *region* yang berjumlah banyak. Sebuah *region* atau area dapat didefinisikan ke dalam beberapa aspek, misalnya ke dalam aspek warna, tepi area, bentuk area maupun ukuran dari area tersebut. Untuk melakukan proses *region merging* maka pada area tersebut perlu ditambahkan sebuah *descriptor* untuk mendefinisikan aturan dari proses *region merging* yang diinginkan.

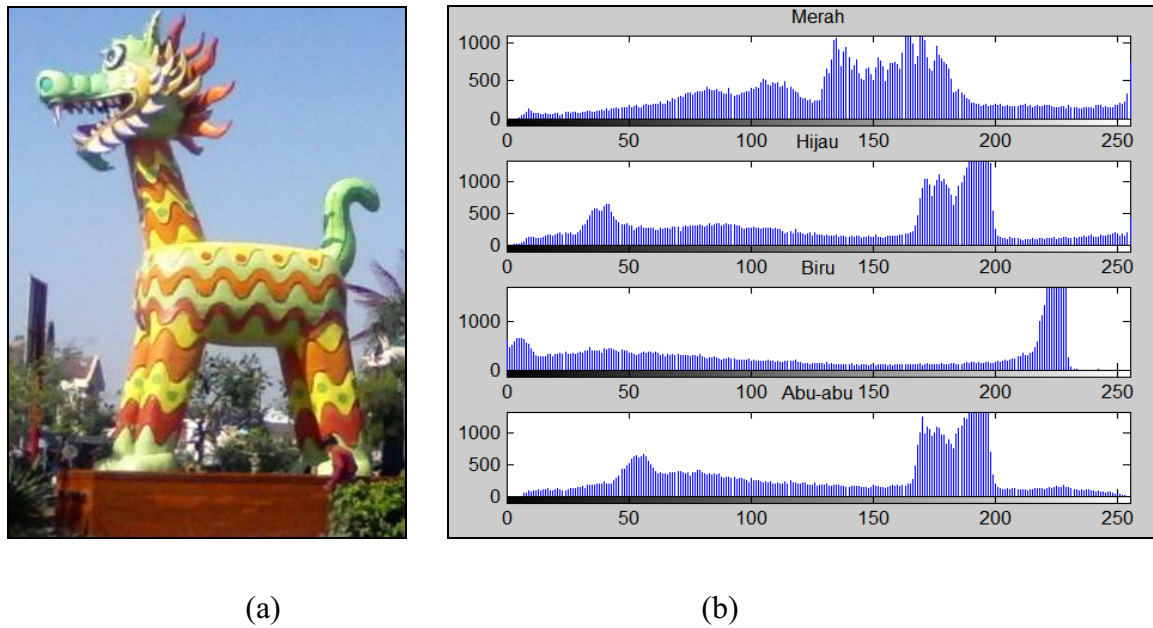
Warna adalah deskriptor yang paling efektif dalam analisa *feature space* [23] dan banyak digunakan dalam kajian *image segmentation*, *patern recognition* dan *object tracking* [5]. Di dalam konteks segmentasi berbasis *region merging*, *histogram* warna lebih handal untuk digunakan dibandingkan dengan deskriptor yang lain. Hal ini karena inisialisasi ukuran segmen kadang-kadang sangat kecil dan sangat banyak mengandung variasi ukuran dan bentuk.

Histogram warna adalah distribusi warna atau jumlah piksel dari keseluruhan warna. Histogram warna ke  $k$  adalah jumlah piksel yang mempunyai warna  $k$ . Contoh pada citra berukuran 4x4 piksel di bawah ini:



Gambar 2. 13. Histogram warna

Sebuah citra berwarna RGB akan mempunyai histogram untuk masing-masing komponen mulai dari nilai 0 – 255. Contoh citra dan histogram dimiliki citra tersebut adalah:



Gambar 2.14 Histogram sebuah citra, (a) citra RGB, (b) histogram citra

Dalam kaitan dengan proses *region merging* maka *histogram* sebuah region akan ditandai dengan nama *HistR*. Permasalahan berikutnya adalah bagaimana menyatukan *region* menggunakan histogram warna yang telah didapatkan sehingga obyek akan berhasil disegmentasi. Untuk itu diperlukan sebuah pengukuran similaritas di antara 2 *region*. Ada beberapa pengukuran similaritas yang dikenal yaitu *euclidean distance*, *bhattacharyya coefficient*, dan *log-likelihoodratio statistic* [29]. Koefisien *bhattacharyya* adalah perkiraan pengukuran jumlah tumpang tindih antara dua sampel statistik. Koefisien *bhattacharyya* dapat digunakan untuk menentukan kedekatan relatif dari kedua sampel yang dipertimbangkan.

Pada penelitian ini akan digunakan koefisien *bhattacharyya* untuk mengukur similaritas antara region  $R$  dan region  $Q$ . Persamaannya adalah sebagai berikut:

$$\rho(R, Q) = \sum_{u=1}^{4096} \sqrt{\text{Hist}_R^u \cdot \text{Hist}_Q^u} \quad (2.7)$$

dimana  $HistR$  dan  $HistQ$  adalah histogram awal dari area  $R$  dan  $Q$  dan nilai  $u$  merupakan elemen ke- $u$  dari histogram tersebut.  $\rho$  merupakan nilai dari koefisien *bhattacharyya* adalah ukuran perbedaan antara interpretasi geometris sederhana.

Di bawah ini adalah nilai *cosinus* di antara 2 sudut dari dua unit vektor:

$$\left( \sqrt{Hist_R^1}, \dots, \sqrt{Hist_R^{4096}} \right) \text{ and } \left( \sqrt{Hist_{RQ}^1}, \dots, \sqrt{Hist_Q^{4096}} \right) \quad (2.8)$$

dimana nilai tertinggi dari koefisien *bhattacharyya* antara region  $R$  dan  $Q$  adalah nilai persamaan tertinggi diantara keduanya.

### **2.3 Metode Segmentasi Obyek *Semi*-Otomatis Dengan Metode *Region Merging Maximal Similarity* Berbasis Algoritma *Mean Shift* dan *Normalized Cuts***

Salah satu metode dalam memisahkan obyek dengan *background* adalah metode *region merging*. Metode segmentasi secara *semi*-otomatis dengan teknik *region merging* dan perhitungan *maximal similarity* adalah metode yang tepat untuk melakukan segmentasi obyek. Proses *region merging* ini dibantu dengan *marker* atau penanda yang dibuat oleh *user*. Proses *region merging* membutuhkan *input* berupa citra hasil segmentasi *low level* yaitu citra yang terbagi menjadi area atau *region-region* yang kecil. Hipotesis yang muncul dari *computing approach* dan teori yang telah dipaparkan diatas adalah bahwa segmentasi obyek yang akurat memerlukan deteksi tepi obyek secara akurat sehingga dapat dibedakan antara bagian obyek dengan bagian *background* pada citra. Algoritma *mean shift* yang bekerja berdasarkan *kernel density estimation* (KDE) atau pengelompokan berdasarkan densitas warna pada sebuah citra mampu mendeteksi tepi obyek pada citra secara akurat. Hal ini karena *mean shift* akan membagi citra menjadi *region-region* kecil berdasarkan area warna yang dominan pada sebuah area citra. *Region-region* kecil yang saling berhubungan akan membentuk tepi obyek dari sebuah citra, tetapi hal ini juga menimbulkan sebuah masalah baru yaitu fenomena *over segmentation* yang artinya jumlah *region* yang dibuat oleh algoritma *mean shift* menjadi sangat banyak.

Algoritma *normalized cuts* diharapkan akan mampu mengatasi permasalahan *over segmentation* pada algoritma *mean shift* karena citra akan dipotong atau di *cut* berdasarkan bobot dari area. Hal ini karena algoritma *normalized cuts* bekerja berdasarkan pembobotan area pada *graph* yang ada

pada citra. Pembobotan area ditentukan oleh *user* dan tidak bergantung sepenuhnya kepada area warna dari citra. Hasil yang diharapkan dari kombinasi algoritma *mean shift* dan *normalized cuts* adalah citra yang terbagi menjadi sejumlah *region* yang tidak terlalu banyak dan hasil segmentasi mampu mengenali tepi obyek secara akurat.

Selanjutnya hasil segmentasi *mean shift* dan *normalized cuts* tadi akan diberikan operasi deteksi tepi *prewit* dan operasi *invers*. Hal ini bertujuan untuk membuat garis batas *region* terlihat jelas dan menghilangkan seluruh warna menjadi putih kecuali warna dari deteksi tepi yang tetap berwarna hitam. Langkah selanjutnya adalah menggabungkan antara hasil deteksi tepi dan operasi *invers* tersebut dengan citra asli sehingga dihasilkan citra asli plus garis tepi dari *region* tersegmen.

Langkah selanjutnya adalah dilakukan proses *region merging* terhadap *region-region* yang telah terbentuk pada citra. *User* memberikan penanda berupa garis atau *stroke* yang menandai bagian obyek dan bagian *background* dari citra. Penanda inilah yang merupakan bentuk interaksi dari *user* dan menjadi ciri dari metode segmentasi *semi-otomatis*. Penanda obyek yang diberikan tidak harus presisi mengikuti bentuk dari obyek yang akan disegmentasi tetapi cukup hanya memberikan sebuah tanda bahwa area tersebut adalah area obyek. Demikian juga pada penanda *background* tidak harus detail memberikan tanda kepada keseluruhan *background*, cukup sebagian saja.

Proses *region merging* dibagi menjadi dua tahap dan akan diawali dari bagian *background* sampai selesai dan selanjutnya baru memulai proses untuk *region merging* pada bagian obyek. *Region* yang bersinggungan dengan garis penanda *background* akan disatukan terlebih dahulu dengan menggunakan perhitungan *maximal similarity*. Perhitungan ini menggunakan *histogram* warna pada masing-masing *region*. *Histogram* yang hampir sama mempunyai arti bahwa *region* tersebut berasal dari bagian yang sama, dalam hal ini adalah bagian *background* dan selanjutnya dua *region* tersebut akan dijadikan menjadi satu. Langkah selanjutnya akan membandingkan dengan *region* di sekitarnya. Bila *histogramnya* juga hampir sama maka *region* tersebut juga akan disatukan areanya tetapi hal ini tidak berlaku bagi *region* yang diberi area obyek. Setelah selesai seluruh area *background* disatukan maka barulah *region* yang ditandai sebagai obyek akan dikomputasi dengan proses yang sama seperti pada bagian *background*. Setelah proses ini dilakukan maka akan dihasilkan hanya dua *region* utama, yaitu *region background* dan *region*

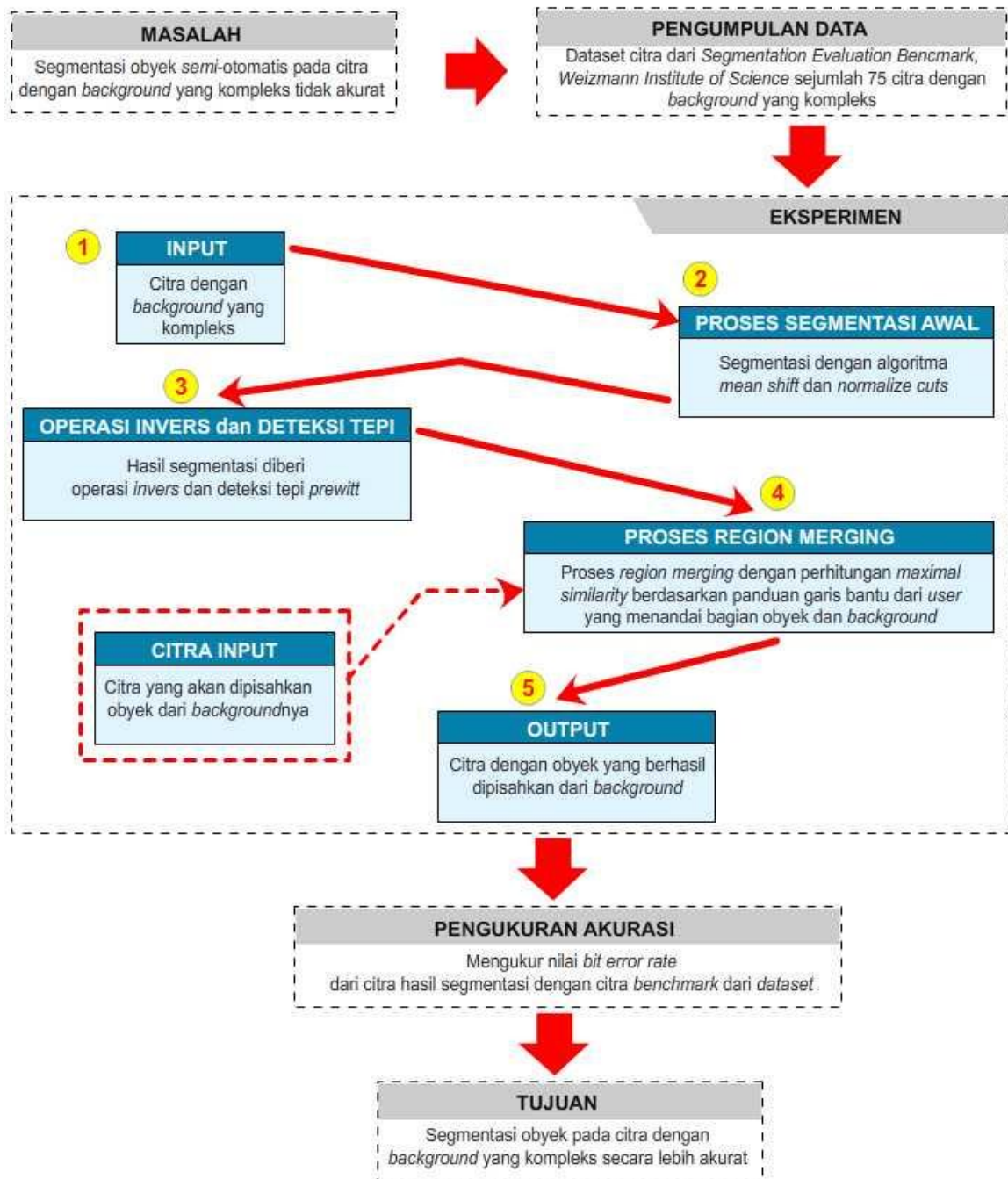
obyek. Tahap selanjutnya adalah mengambil mengekstraksi hanya *region* obyek dan proses segmentasi obyek selesai.

## 2.4 Kerangka Pemikiran

Kerangka pemikiran dalam penelitian ini digambarkan pada gambar 9. Permasalahan yang diambil adalah bahwa segmentasi obyek pada citra dengan *background* yang kompleks tidak akurat. Pendekatan komputasi atau *computing approach* yang diambil untuk menjawab permasalahan diatas adalah segmentasi obyek secara *semi*-otomatis dengan *region merging maximal similarity* menggunakan integrasi algoritma *mean shift* dan *normalized cuts*. Perkakas atau *tools* yang akan digunakan untuk melaksanakan eksperimen adalah MATLAB R2010a, sedangkan dataset citra yang diambil untuk eksperimen berasal dari dataset publik yang ada pada laman *Segmentation Evaluation Benchmark* pada *Weizmann Institute of Science*.

Untuk mengukur akurasi hasil segmentasi digunakan pengukuran *bit error rate* dengan menggunakan citra *benckmark* yang telah tersedia pada dataset. Pengukuran hasil akurasi dilakukan dengan membandingkan hasil pemotongan obyek yang berhasil dilakukan oleh metode yang diusulkan dalam penelitian ini dan dibandingkan dengan potongan obyek pada citra *benchmark*. Karena yang dibandingkan adalah potongan maka warna citra hasil dan citra *benchmark* akan diabaikan. Sebelumnya kedua citra yang dibandingkan akan diubah kedalam mode citra *biner*, atau citra hitam putih. Bagian obyek akan diubah menjadi berwarna putih sedangkan bagian *background* akan diubah menjadi berwarna hitam. Pengukuran *bit error rate* dilakukan dengan menjumlahkan keseluruhan titik piksel kedua citra yang mempunyai nilai berbeda, dan hasilnya dibagi dengan dimensi piksel dari citra. Hasil yang diharapkan adalah segmentasi obyek pada citra dengan *background* yang kompleks secara lebih akurat.

Kerangka pemikiran pada penelitian ini adalah sebagai berikut:



Gambar 2.15 Kerangka pemikiran