

**Pengenalan Pola Beras dan Gabah Berdasarkan
Ciri Warna Menggunakan Matlab**
Amir F. Sofyan

Abstraksi

Adanya pekerjaan memilah gabah –biji beras yang belum dikupas kulitnya—dari kumpulan beras dapat mengurangi produktifitas petani. Sistem penglihatan komputer dapat mengambil alih peran petani pada pekerjaan ini. Tulisan berikut menjelaskan salah satu modul dalam sistem penglihatan komputer yaitu modul pengenalan pola untuk membedakan gabah dengan beras berdasarkan ekstraksi ciri warna.

Kata kunci: pengenalan, pola

Pendahuluan

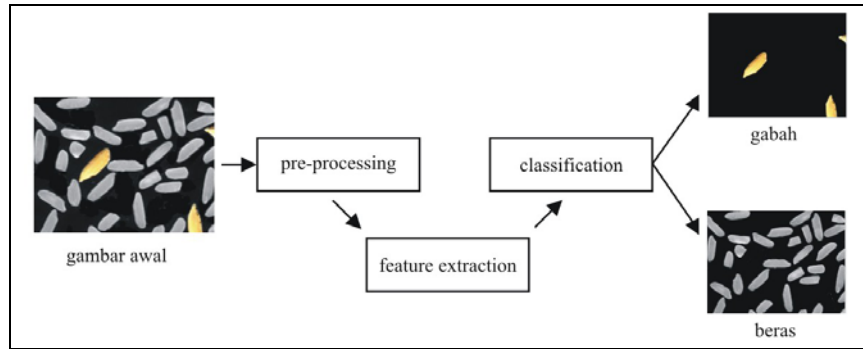
Secara umum terdapat beberapa ciri yang dimiliki oleh beras dan gabah, yaitu warna, luas, keliling, dan bentuk. Namun demikian pada ciri luas, keliling maupun bentuk terdapat kesamaan antara beras dan gabah, sehingga ketiga ciri ini tidak digunakan. Sehingga lebih tepat menggunakan ciri warna untuk membedakan beras dan gabah.

Tabel 1. Ekstraksi ciri beras dan gabah

Klasifikasi	Ekstraksi ciri			
	Warna	Luas	Keliling	Bentuk
Beras	Keputih-putihan	L	K	Lonjong
Gabah	Kekuning-kuningan	L	K	Lonjong
Latar belakang	Kehitam-hitaman	> L	> K	Kotak

Pembahasan

Tahapan dimulai dengan proses masukan yang dapat menggunakan piranti masukan seperti kamera digital. Selanjutnya dilakukan tahapan pre-processing untuk lebih menajamkan kualitas gambar. Kemudian dilakukan proses segmentasi menggunakan L*a*b* color yang hasilnya berupa gambar gabah yang telah terpisah dengan beras. Dan terakhir dilakukan proses perhitungan beras dan gabah.



Gambar 1. Gambaran umum proses pengenalan beras dan gabah

Berikut detail langkah-langkah proses pemilahan gambar gabah dari beras menggunakan Matlab dan Image Processing Toolbox. Langkah pertama membaca gambar awal. Di sini terlihat gambar awal yang telah dilakukan pengolahan sebelumnya berupa butiran beras dan gabah.

```
>> Beras = imread('beras01.tif');
>> imshow(Beras);
```



Gambar 2. Image Beras (a) dan image template (b)

Berikutnya adalah membaca gambar sample untuk membedakan warna gabah, beras dan latar belakang. Gambar template.tif berisi kombinasi dari tiga area gabah, beras dan latar belakang yang disatukan untuk memudahkan proses pencarian sample.

```
>> template = imread('template.tif');
>> imshow(template);
```

Selanjutnya menentukan area sample berupa area persegi pada masing-masing sample.

```
>> a=[1 70 70 1 ];
>> b=[1 1 161 161];
>> c=[75 140 140 75];
>> d=[1 1 161 161];
>> e=[145 210 210 145 ];
>> f=[1 1 161 161];
```

Kemudian menyeleksi sample gabah, beras dan latar belakang dengan roipoly.

```
>> sample_regions(:,:,1) = roipoly(template,a,b);
>> sample_regions(:,:,2) = roipoly(template,c,d);
>> sample_regions(:,:,3) = roipoly(template,e,f);
```

Langkah berikutnya mengubah citra dari warna RGB menjadi L*a*b* menggunakan makecform dan applycform untuk keperluan proses segmentasi.

```
>> cform = makecform( 'srgb2lab' );
>> lab_Beras = applycform(Beras,cform);
>> imshow(lab_Beras)
```



Gambar 3. Image lab_Beras

Lalu menghitung nilai mean dari 'a*' dan 'b*' dari setiap daerah yang diekstrak dengan roipoly. Hasilnya digunakan sebagai color markers pada 'a*b*' space.

```
>> a = lab_Beras(:,:,2);
>> b = lab_Beras(:,:,3);
>> color_markers = repmat(0, [3, 2]);
```

```

>> for count = 1:2
>> color_markers(count,1) =
mean2(a(sample_regions(:,:,count)));
>> color_markers(count,2) =
mean2(b(sample_regions(:,:,count)));
>> end

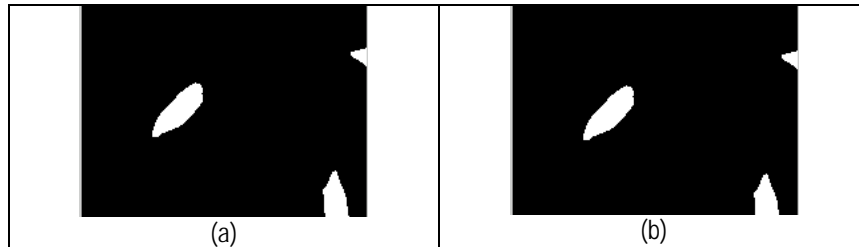
```

Berikutnya membuat klasifikasi tiap pixel menggunakan *nearest neighbor rule*.

```

>> color_labels = 0:2;
>> a = double(a);
>> b = double(b);
>> distance = repmat(0,[size(a), 3]);
>> for count = 1:3
>> distance(:,:,count) = ( (a -
color_markers(count,1)).^2 + (b -color_markers...
>> count,2)).^2 ).^0.5;
>> end
>> [value, label] = min(distance,[],3);
>> label = color_labels(label);
>> clear value distance;
>> imshow(label)

```



Gambar 4. Image label (a) dan rgb_label (b)

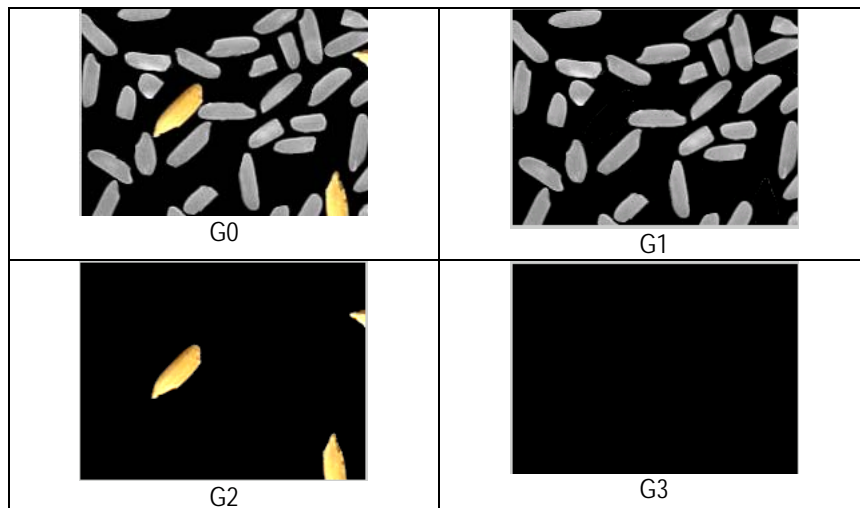
```

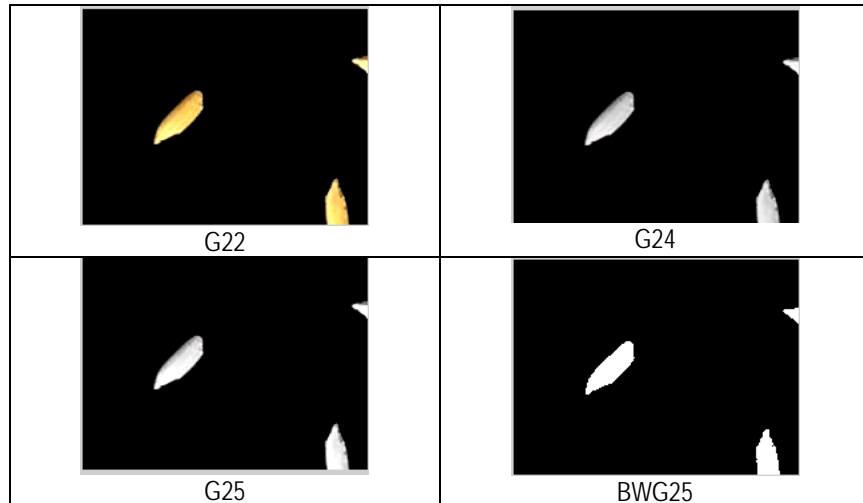
>> rgb_label = repmat(label,[1 1 3]);
>> imshow(rgb_label)
>> segmented_images = repmat(uint8(0),[size(Beras), 3]);
>> for count = 1:3
>> color = Beras;
>> color(rgb_label ~= color_labels(count)) = 0;
>> segmented_images(:,:,count) = color;
>> end

```

Menghitung jumlah gabah

```
>> G0 = Beras;  
>> imshow(G0);  
>> G1 = (segmented_images(:,:,1));  
>> imshow(G1);  
>> G2 = (segmented_images(:,:,2));  
>> imshow(G2);  
>> G3 = (segmented_images(:,:,3));  
>> imshow(G3);  
>> G22 = imsubtract(G2,G3);  
>> imshow(G22);  
>> G24=rgb2gray(G22);  
>> imshow(G24);  
>> G25 = imadjust(G24);  
>> imshow(G25);  
>> level = graythresh(G25);  
>> BWG25 = im2bw(G25,level);  
>> imshow(BWG25);
```





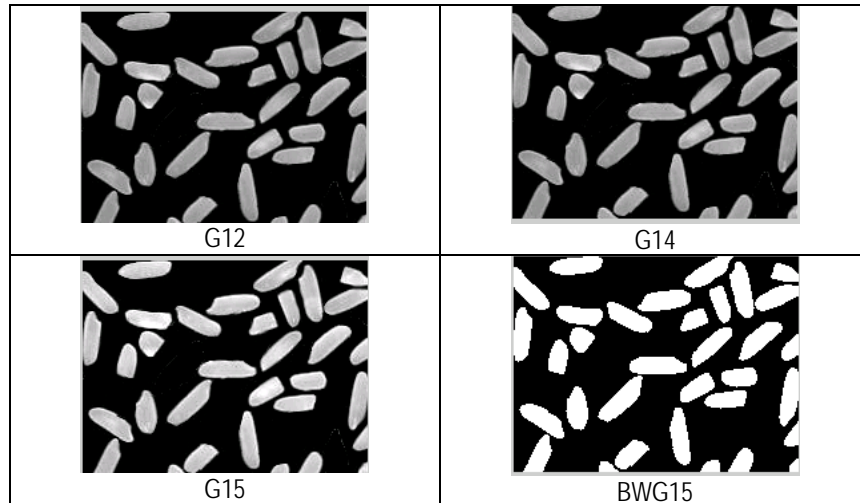
Gambar 5. Beberapa image hasil pemisahan gabah

Langkah selanjutnya memberi label untuk menandai setiap butiran gabah yang ada. Dan hasilnya dapat digunakan untuk menghitung butiran .

```
>> [labeledGabah, jumlahGabah] = bwlabel(BWG25,4);
>> jumlahGabah
```

Serupa dengan pada proses yang dilakukan pada gambar gabah sebelumnya, langkah berikut bertujuan menghitung jumlah beras.

```
>> G12 = imsubtract(G1,G3);
>> imshow(G12);
>> G14=rgb2gray(G12);
>> imshow(G14);
>> G15 = imadjust(G14);
>> imshow(G15);
>> level = graythresh(G15);
>> BWG15 = im2bw(G15,level);
>> imshow(BWG15);
```



Gambar 6. Beberapa image hasil pemisahan beras

Selanjutnya memberi label untuk menandai setiap butiran beras yang ada.

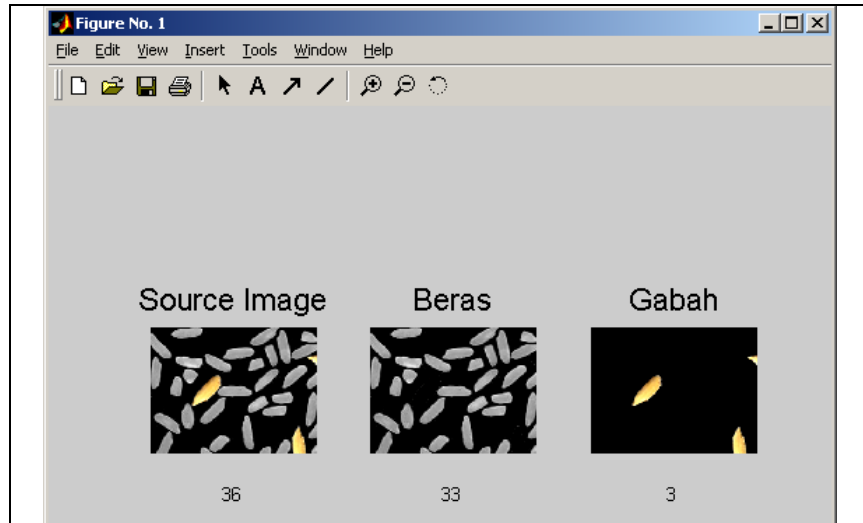
```
>> [labeledBeras, jumlahBeras] = bwlabel(BWG15,4);
>> jumlahBeras
```

Untu menghitung semua butiran beras yang ada, maka tinggal dijumlahkan antara jumlah beras dengan jumlah gabah.

```
>> jumlahCampur = (jumlahBeras + jumlahGabah);
```

Dan langkah terakhir menampilkan gambar beras dan gabah yang telah terpisah berikut hasil perhitungannya.

```
>> [X1]=G0;
>> [X2]=G1;
>> [X3]=G2;
>> subplot(1,3,1), imshow(X1), title('Source Image',
'FontSize',16), xlabel(jumlahCampur);
>> subplot(1,3,2), imshow(X2), title('Beras',
'FontSize',16), xlabel(jumlahBeras);
>> subplot(1,3,3), imshow(X3),
title('Gabah','FontSize',16), xlabel(jumlahGabah);
```



Gambar 7. Hasil akhir pemilahan beras dan gabah

Kesimpulan

Aplikasi pengenalan beras dan gabah ini dapat berjalan cukup baik. Namun demikian terdapat kekurangan pada ketelitian perhitungan karena segmentasi yang kurang sempurna. Akan lebih baik bila dikembangkan dengan memanfaatkan fasilitas Graphic User Interface (GUI) pada MATLAB, sehingga akan memudahkan pengguna dalam pemilihan sumber gambar.

Daftar Pustaka

On Line Help MATLAB 6.5.1