### LAPORAN PROYEK AKHIR

### VISI KOMPUTER

# CAR PLATE RECOGNITION



Dikembangkan oleh:

Hamdi Ahmadi Muzakkiy **(5112100091)**

Muhammad Nadzeri Munawar **(5112100136)**

Brillian T. Nugraha  **(5112100172)**

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**

**SURABAYA**

**2015**

# PENDAHULUAN

# Latar Belakang

Sistem parkir modern yang terdapat di gedung perkantoran, rumah sakit, *mall*, dsb. menggunakan aplikasi komputer dalam penerapannya. Selama ini, sistem parkir modern tersebut menggunakan ketik nomor plat kendaraan pada saat kendaraan masuk dan keluar dari parkiran. Tetapi ada beberapa kendala dalam penerapannya seperti kesalahan pengetikan, waktu pengetikan yang lama, serta kesalahan operator lainnya.

Oleh karena itu, perlu adanya inovasi dalam sistem parkir modern tersebut. Salah satunya yaitu menggunakan foto dalam mengenali teks plat nomor kendaraan. Tujuannya yaitu agar menghindari kesalahan pengetikan, waktu pengetikan yang lama, serta kesalahan operator lainnya.

# Rumusan Masalah

Rumusan masalah yang diangkat dalam final project ini dipaparkan sebagai berikut :

1. Bagaimana membuat aplikasi yang mampu mengambil teks dari foto plat nomor kendaraan?
2. Bagaimana membuat aplikasi yang mampu menentukan warna dari citra plat nomor kendaraan?

# Batasan Masalah

Permasalahan yang akan dibahas dalam final project ini memiliki beberapa batasan antara lain :

1. Bahasa yang digunakan yaitu Python.
2. IDE yang digunakan yaitu PyCharm, Sublime Text 3, Matlab.
3. Citra masukan yaitu plat nomor kendaraan.
4. Hasil keluaran yaitu nomor plat dan warna plat kendaraan.

# Tujuan Pembuatan Final Project

Tujuan dari pembuatan final project ini antara lain :

1. Membuat aplikasi yang mampu mengidentifikasi karakter dari plat nomor kendaraan.
2. Membuat aplikasi yang mampu mengidentifikasi warna plat nomor kendaraan.

# Manfaat Final Project

Manfaat dari hasil pembuatan final project ini antara lain :

1. Menjadi salah satu solusi yang dapat diterapkan dalam sistem parkir otomatis.
2. Dapat mengurangi waktu dan pekerja parkir yang diperlukan.
3. Menghindari kesalahan operator dan waktu pengetikan nomor plat yang memakan waktu lama.

# METODOLOGI DAN IMPLEMENTASI

# Tinjauan Pustaka

Dalam final project ini, digunakan beberapa pustaka sebagai berikut :

## Bahasa Python

Python merupakan bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sisterm operasi.

## Color recognition

Metode pengenalan warna citra plat nomor kendaraan berfungsi untuk mengidentifikasi jenis kendaraan. Metode yang digunakan dapat menggunakan rata-rata warna dari elemen red, green, dan blue (RGB) dan mengklasifikasikan jenis plat kendaraan dengan memanfaatkan jarak antara nilai RGB dari data masukan dengan nilai dari database.

## Character recognition

Metode pengenalan karakter citra plat nomor kendaraan berfungsi untuk mengidentifikasi karakter pada plat nomor. Metode yang digunakan untuk dapat mengenali karakter yaitu menggunakan klasifikasi KNN.

## K-Nearest Neighbor (KNN)

Metode klasifikasi KNN merupakan klasifikasi yang memanfaatkan jarak antara data. Pada citra, hal ini dapat dimanfaatkan untuk menentukan jarak antara nilai *red*, *green*, dan *blue* (RGB) untuk digunakan dalam mengidentifikasi warna dominan dari citra tersebut. Selain itu metode klasifikasi ini dimanfaatkan juga untuk mengidentifikasi pengenalan karakter, diantaranya yaitu 26 huruf (A s/d Z) dan 10 angka (0 s/d 9).

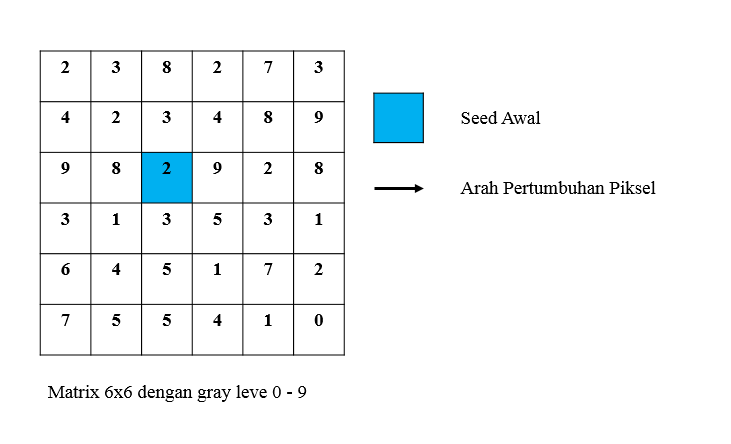
## Median Filter

Median filter adalah salah satu metode untuk *bluring*. Dilakukan dengan cara melakukan filter citra menggunakan suatu filter dan mengablil nilai tengah dari filter tersebut ( yang sudah diurutkan ).

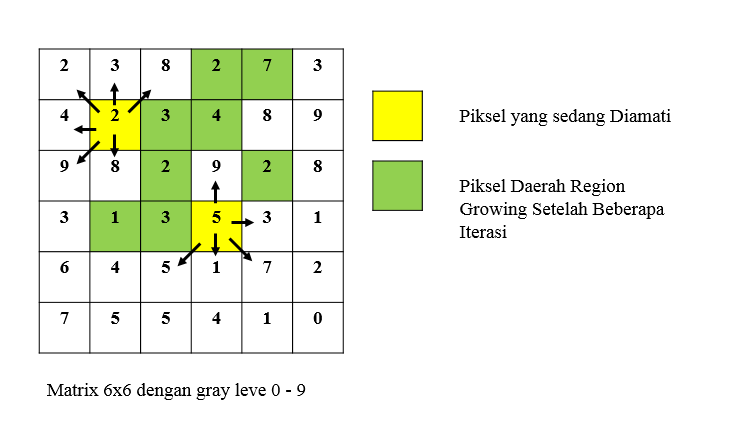
## Region Growing

Region growing digunakan untuk melakukan segmentasi citra. Citra yang akan disegmentasi dilakukan inisialisasi terlebih dahulu. Inisialisasai dilakukan dengan menentukan suatu titik (*seed*) dan dilakukan penumbuhan terhadap tetangga piksel. untuk metode ini, citra akan diubah kedalam citra keabuan. Dalam kasus *final project* ini, citra yang digunakan adalah citra binary.

Aturan penumbuhan *region* dilakukan dengan melakukan cek terhadap delapan tetangga piksel dari *seed*. Akan dicek tingkat homogen terhadap *seed*. Jika sama, maka tetangg piksel akan ditetapkan sebagai *seed* baru dan dilakukan penumbuhan menggunakan *seed* baru tersebut. Ilustrasi ditunjukkan pada Gambar 1 dan Gambar 2.



Gambar 1. Seed awal region growing



Gambar 2. Pengecekan terhadap tetangga piksel

## IDE PyCharm

Pycharm merupakan python editor yang bisa membaca dengan baik codebase python yang secara desain adalah weak-typing. Semua keyword yang diberikan sangat akurat.

# Ringkasan Isi Final Project

Pengenalan plat nomor kendaraan dilakukan dengan dua tahap, yaitu mendeteksi karakter pada plat nomor dan warna pada plat. Gambaran umum sistem bisa dilihap pada Gambar 3 dan Gambar 4

C:\Users\hamdiahmadi\Downloads\PlateRecognition.png

Gambar 3. Diagram alir pengenalan karakter

C:\Users\hamdiahmadi\Downloads\warna.png

Gambar 4. Diagram alir pengenalan warna plat

## Pengenalan Karakter Plat

Pengenalan karakter pada plat kendaraan dilakukan dengan dua tahap, yaitu *preprocessing* dan klasifikasi. *Preprocessing* dilakukan dengan tujuan mendapatkan karakter pada plat nomor. Setelah mendapatkan citra setiap karakter, dilakukan klasifikasi untuk menentukan karakter dari citra tersebut.

## Preprocessing

Citra yang akan dilakukan pengenalan karakter terlebih dahulu dilakukan *proprocessing* dengan tujuan mendapatkan karakter teks pada citra. Alur proses bisa dilihat pada Gambar 5.

C:\Users\hamdiahmadi\Downloads\proprocessing - visikom.png

Gambar 5. Diagram alir preprocessing

## Mengubah Citra Kedalam Citra Gray

Citra masukan akan diubah kedalam citra gray terlebih dahulu, citra masukkan dapat dilihat pada Gambar 6.



Gambar 6. Citra masukkan

Citra keluaran dari proses ini dapat dilihat pada Gambar 7.



Gambar 7. Citra Gray

## Denoising Menggunakan Median Filter

Citra masukan dari proses ini adalah citra gray yang sudah diproses pada tahap sebelu mnya. Denoising dilakukan dengan tujuan menghilangkan *nois* yang ada pada citra. Ilustrasi proses denoising dapat dilihat pada Gambar 8



Gambar 8. Citra blur

Citra keluaran dari proses ini akan digunakan pada proses selanjutnya yaitu mengubah citra kedalam citra binary.

## Mengubah Citra Kedalam Citra Binary

Proses ini dilakukan untuk mendapatkan citra hitam putih yang nantinya digunakan untuk *region* *growing*. Citra hitam putih akan memudahkan dalam segmentasi citra. Berikut ilustrasi citra binary, bisa dilihat pada Gambar 9.



Gambar 9. Citra binary

Setelah melakukan proses ini, tahap terakhir dari *preprocessing* adalah mendapatkan gambar perkarakter menggunakan *region growing*.

## Region Growing

Proses ini dilakukan untuk karakter pada citra. Masukan dari proses ini adalah citra binary. Titik awal dari *seed* adalah indeks 0,0 dilakukan penumbuhan *region*. Nantinya setiap *region* yang ditemukan dan sudah selesai ditumbuhkan akan dicek apakah citra tersebut termasuk citra karakter atau tidak. Proses ini menggunakan *threshold* pada setiap karakter. Ilustrasi citra karakter dapat dilihat pada Gambar 10

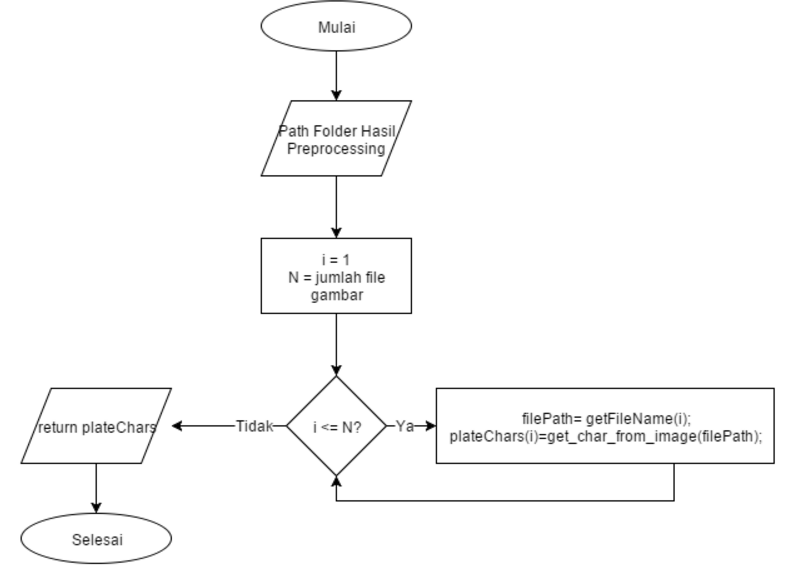
F:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_0.pngF:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_1.pngF:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_2.pngF:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_3.pngF:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_4.pngF:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_5.pngF:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_6.pngF:\KULIAH\AKADEMIK\semester 7\Visikom\plateRecognition\result\train2.png_7.png

Gambar 10. Citra Karakter

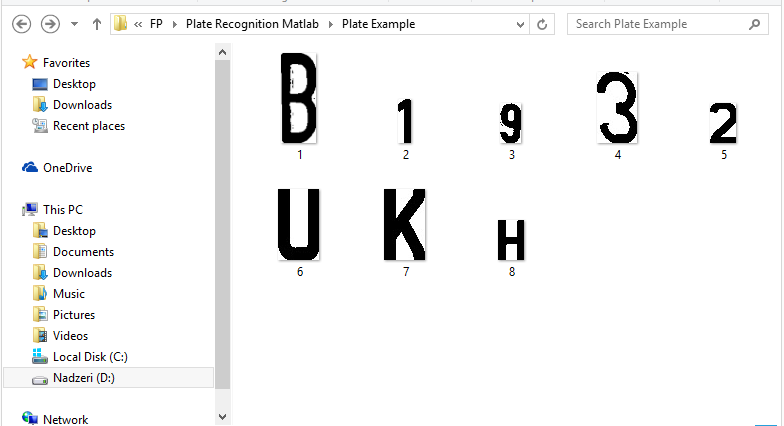
Selanjutnya, setiap karakter akan dilakukan pengenalan. Dilakukan pada proses klasifikasi

## Klasifikasi

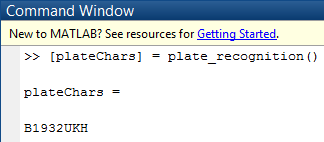
Citra yang akan dilakukan pengenalan karakter terlebih dahulu dilakukan *proprocessing* dengan tujuan mendapatkan masing-masing karakter teks pada citra. Setelah dilakukan *preprocessing*, citra tersebut diklasifikasi dengan tujuan mendapatkan text dari citra. Alur proses pengenalan karakter secara umum bisa dilihat pada Gambar 11. Dimana fungsi get\_char\_from\_image() akan dijelaskan pada bab 7.1.2.1. Adapun contoh citra input dapat dilihat pada Gambar 12. Sedangkan contoh output program dapat dilihat pada Gambar 13.



Gambar 11. Diagram Alir Pengenalan Karakter Secara Umum



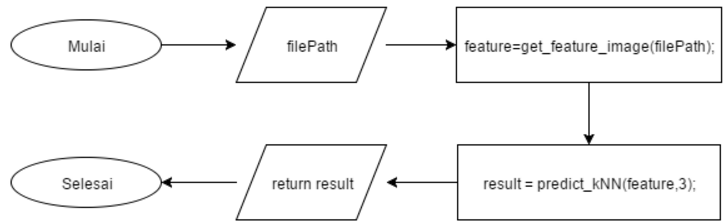
Gambar 12 Contoh Citra Input



Gambar 13 Contoh Output Program

## Mendapatkan Karakter Dari Citra

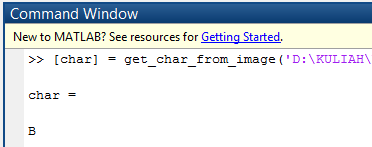
Langkah pertama, yaitu mendapatkan *path* dari file gambar. Setelah itu, memanggil fungsi get\_feature\_image(filePath) untuk mendapatkan fitur dari gambar yang diinputkan. Selanjutnya melakukan prediksi karakter menggunakan algoritma klasifikasi KNN. Alur proses mendapatkan karakter dari citra dapat dilihat pada gambar 14. Dimana fungsi get\_feture\_image() akan dijelaskan pada bab 7.1.2.2 sedangkan fungsi predict\_kNN() akan dijelaskan pada bab 7.1.2.3. Adapun contoh citra input dapat dilihat pada Gambar 15. Sedangkan contoh output program dapat dilihat pada Gambar 16.



Gambar 14. Alur Mendapatkan Karakter dari Citra

D:\KULIAH\Visikom\FP\Plate Recognition Matlab\Plate Example\1.png

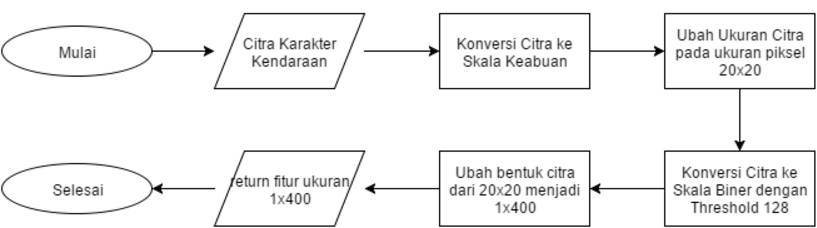
Gambar 15. Contoh Citra Input



Gambar 16 Contoh Output Program

## Mendapat Fitur Citra

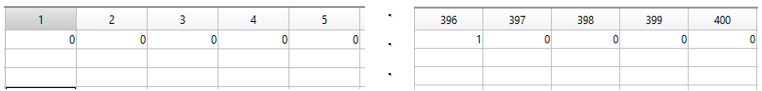
Langkah pertama, yaitu mendapatkan citra masing-masing karakter. Setelah itu, melakukan konversi citra ke skala keabuan. Selanjutnya ubah ukuran citra dengan ukuran piksel menjadi 20 x 20. Setelah itu konversi citra ke skala biner dengan threshold 128. Selanjutnya ubah bentuk citra dari semula berukuran 20 x 20 menjadi 1 x 400 (menjadi 1 baris data). Alur proses mendapatkan fitur dari citra dapat dilihat pada Gambar 17. Adapun contoh citra input dapat dilihat pada Gambar 18. Sedangkan contoh output program dapat dilihat pada Gambar 19.



Gambar 17. Alur Mendapatkan Fitur Citra

D:\KULIAH\Visikom\FP\Plate Recognition Matlab\Plate Example\1.png

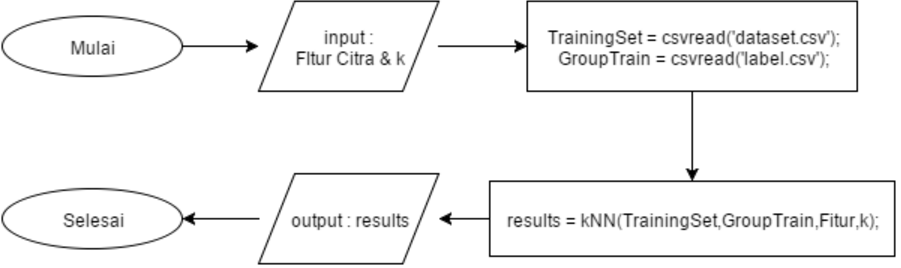
Gambar 18 Contoh Citra Input



Gambar 19 Contoh Fitur Citra

## Prediksi Citra menggunakan KNN

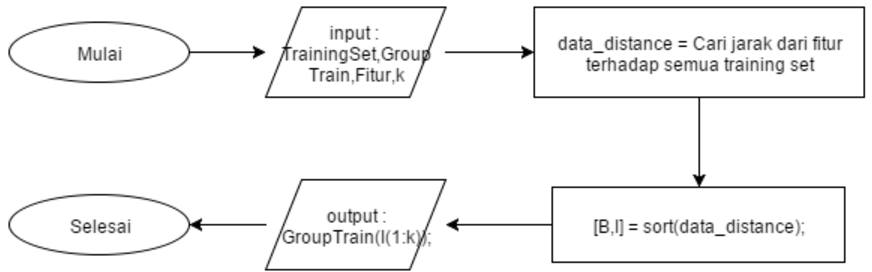
Langkah pertama, yaitu mendapatkan fitur citra beserta nilai k. Dalam kasus ini variabel k bernilai 3. Setelah itu ambil data training dan label yang sudah di-train sebelumnya. Selanjutnya yaitu memprediksi karakter menggunakan algoritma kNN. Adapun fungsi kNN akan dijelaskan pada bab 7.1.2.3. Alur proses prediksi citra menggunakan kNN dapat dilihat pada Gambar 20.



Gambar 20. Alur Prediksi Citra Menggunakan KNN

## Algoritma KNN

Langkah pertama, yaitu mendapatkan TrainingSet, GroupSet, fitur citra, dan nilai k. Dalam kasus ini variabel k bernilai 3. Setelah itu cari jakar dari fitur citra terhadap semua data TrainingSet. Selanjutnya yaitu melakukan pengurutan data berdasarkan jarak terkecil. Hasil dari algoritma ini yaitu data label berdasarkan k data dengan jarak terkecil. Alur proses algoritma kNN dapat dilihat pada Gambar 21.



Gambar 21. Algoritma KNN

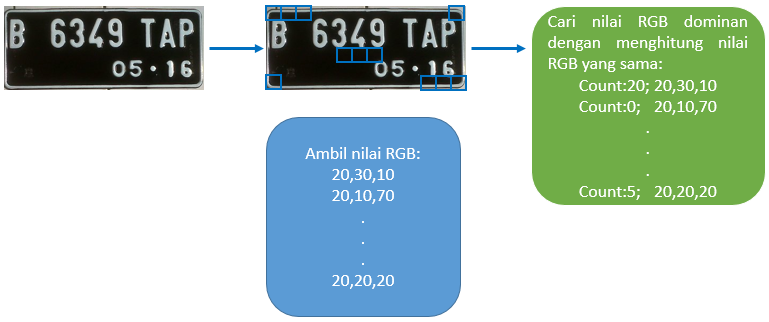
## Pengenalan Warna Citra

Proses pengenalan warna citra plat nomor kendaraan terdiri dari 2 bagian, pertama yaitu mendapatkan warna citra dominan (RGB) dan kedua klasifikasi warna citra plat nomor kendaraan.

## Mendapatkan Nilai RGB dari Citra Plat Nomor Kendaraan

Citra masukan yang ingin dikenali warna citranya serta citra *database* yang telah disiapkan diambil nilai RGB nya, kemudian dicari nilai RGB dengan nilai yang sama, dan dihitung jumlah nilai RGB yang sama untuk menentukan nilai RGB dominan di dalam citra masukan.

Ilustrasi mendapatkan warna citra dominan dari plat nomor dapat dilihat pada Gambar 22.



Gambar 22. Mencari warna dominan dari citra plat nomor kendaraan.

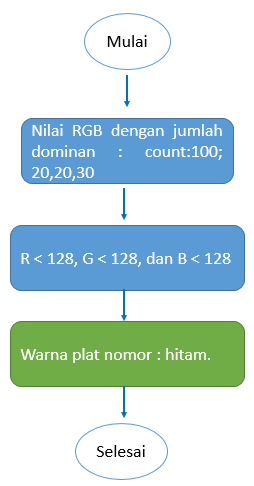
## Klasifikasi Warna Citra Plat Nomor Kendaraan

Citra masukan yang telah diambil nilai rata-rata RGB-nya kemudian dijadikan sebagai data untuk *testing* dan citra *database* dijadikan sebagai data *training* untuk dijadikan sebagai referensi pada proses klasifikasi.

Proses klasifikasi warna citra plat nomor kendaraan ditentukan dengan memperhatikan nilai frekuensi RGB tertinggi, yang kemudian akan ditentukan jenis warnanya menggunakan threshold:

1. Warna hitam dengan threshold R < 128, G < 128, dan B < 128.
2. Warna kuning dengan threshold R ≥ 128, G ≥ 128, dan B < 128.
3. Warna putih dengan threshold R ≥ 128, G ≥ 128, dan B ≥ 128.
4. Warna biru dengan threshold R < 128, G < 128, dan B ≥ 128, dan
5. Warna merah dengan threshold R ≥ 128, G < 128, dan B < 128.

Proses klasifikasi warna citra plat nomor kendaraan diilustrasikan pada Gambar 12

.

Gambar 23. Proses klasifikasi warna citra plat nomor kendaraan.

## Mengkategorikan Warna Citra Plat Nomor Kendaraan

Hasil dari klasifikasi yaitu sebanyak k-terdekat dari rata-rata RGB, yang akan diambil indeksnya untuk dicari label yang sesuai dengan indeks hasil k-terdekat yang telah diklasifikasikan. Penentuan kategori warna citra didapatkan dengan mencari frekuensi nilai label terbanyak yang ada di dalam k-terdekat dari rata-rata RGB hasil klasifikasi.

# Jadwal Pengerjaan

Timeline pengerjaan adalah sebagai berikut :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Kegiatan** | **Minggu Ke -** | | | |
| **12** | **13** | **14** | **15** |
| 1 | Pembuatan Proposal |  |  |  |  |
| 2 | Studi Literatur |  |  |  |  |
| 3 | Pembuatan Aplikasi |  |  |  |  |

Tabel 1 Time line pengerjaan

# Implementasi

Proyek Akhir pendeteksian wajah ini diimplementasikan menggunakan python. Berikut implementasi yang dilakukan.

## Mengubah Citra Kedalam Citra Gray

Pada sub bab ini dijelaskan implementasi pengubahan citra kedalam citra gray.

|  |  |
| --- | --- |
| 1.  2. | def imageToGray(self,image):  return cv2.cvtColor(image,cv2.COLOR\_RGB2GRAY) |

Kode Sumber 1. Implementasi Tahap Mengubah Citra Kedalam Citra Gray

## Denoising Menggunakan Median Filter

Pada sub bab ini dijelaskan implementasi *denoising* pada citra masukkan sebelum masuk ketahap selanjutnya. Pada tahap *denoising* , dilakukan filter citra.

|  |  |
| --- | --- |
| 1.  2. | def medianBluring(self,image,filter\_size):  return cv2.medianBlur(image,filter\_size) |

Kode Sumber 2. Implementasi Denoising Menggunakan Median Filter

## Mengubah Citra Kedalam Citra Binary

Mengubah citra kedalam citra binary ditunjukkan pada Kode Sumber 3

|  |  |
| --- | --- |
| 1.  2. | def imageToBinary(self,image):  return cv2.threshold(image,127,255,cv2.THRESH\_BINARY\_INV) |

Kode Sumber 3. Implementasi Mengubah Citra Kedalam Citra Binary

## Region Growing

Dari citra binary, dilakukan *region growing* untuk medapatkan citra teks. Implementasi ditunjukkan pada Kode Sumber 4

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19.  20.  21.  22.  23.  24.  25.  26.  27.  28.  29.  30.  31.  32.  33.  34. | def localization(self,image):  background\_counter = dict()  background\_counter[0] = 0  background\_counter[255] = 0  is\_visit = copy.copy(image)\*0  region\_number = 0  region = []  for x in range(0,len(image[0])):  for y in range(0,len(image)):  background\_counter[image[y][x]]+=1  if is\_visit[y][x] == 0:  stack = []  region\_number+=1  stack.append([y,x])  is\_visit[y][x] = int(region\_number)  coordinate,is\_visit = self.growing(image,is\_visit,stack,region\_number)  img = image[coordinate[2]:coordinate[0],coordinate[3]:coordinate[1]]  if (coordinate[0]-coordinate[2])>= len(image)/3 and (coordinate[1]-coordinate[3])>=len(image[0])/50 and (coordinate[0]-coordinate[2])<= len(image)/1.5 and (coordinate[1]-coordinate[3])<=len(image[0])/1.5:  region.append(img)  if background\_counter[255] > background\_counter[0]:  result = region  else :  result = []  cntr = 0  for x in region:  cntr+=1  for row in range(0,len(x)):  for col in range(0,len(x[row])):  if x[row][col] == 255:  x[row][col] = 0  else :  x[row][col] = 255  result.append(x)  return result |

Kode Sumber 4. Implementasi Region Growing

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19. | def growing(self,image,is\_visit,stack,region\_number):  listsY = []  listsX = []  clock = Data.clockWise(self)  while len(stack)!=0:  coory,coorx = stack[0]  stack.pop(0)  listsY.append(coory)  listsX.append(coorx)  for x in clock:  try :  if coory+x[0] < 0 or coory+x[0] > len(is\_visit) or coorx+x[1] <0 or coorx+x[1] > len(is\_visit[0]):  pass  elif is\_visit[coory+x[0]][coorx+x[1]] == 0 and image[coory][coorx] == image[coory+x[0]][coorx+x[1]]:  is\_visit[coory+x[0]][coorx+x[1]] = region\_number  stack.append([coory+x[0],coorx+x[1]])  except :  pass  return [max(listsY),max(listsX),min(listsY),min(listsX)],is\_visit |

Kode Sumber 5. Implementasi Penumbuhan Seed

Hasil dari proses ini adalah citra plat nomor yang berhasil ditemukan selama proses *preprocess*.

## Fungsi Utama Klasifikasi

Algoritma klasifikasi yang kami buat terdiri dari beberapa fungsi. Adapun fungsi utama klasifikasi berdasarkan perancangan yang telah dibuat pada bab 7.1.2 dapat dilihat pada kode Kode Sumber 6.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12. | function [plateChars] = plate\_recognition()  folderDatasetPath = '..\..\Image Preprocessing Python\result';  %folderDatasetPath = '..\Plate Example';  allFileList = dir(folderDatasetPath);  allFileName = { allFileList(3:end).name };    for i = 1:length(allFileName)  dataOneFile = allFileName{i};  filePath = strcat(folderDatasetPath,'\',dataOneFile);  plateChars(i) = get\_char\_from\_image(filePath);  end  end |

Kode Sumber 6 Fungsi Utama Klasifikasi

## Fungsi Mendapat Karakter dari Citra

Fungsi mendapat karakter dari citra berdasarkan perancangan yang telah dibuat pada bab 7.1.2.1 dapat dilihat pada Kode Sumber 7.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7. | function [char] = get\_char\_from\_image(pathFile)  label = '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ';    feature = get\_feature\_image(pathFile);  ind\_predict = predict\_kNN(feature,3);  char = label(ind\_predict);  end |

Kode Sumber 7 Fungsi Mendapat Karakter dari Citra

## Fungsi Mendapat Fitur dari Citra

Fungsi mendapat fitur dari citra berdasarkan perancangan yang telah dibuat pada bab 7.1.2.2 dapat dilihat pada Kode Sumber 8.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13. | function [featureImage] = get\_feature\_image(pathFile)  imageOriginal = imread(pathFile); %Open image file  if(strcmp(pathFile(end-2:end),'jpg'))  rgbImage = rgb2gray(imageOriginal); %Convert image to RGB  else  rgbImage = imageOriginal;  end  resizedImage = imresize(rgbImage,[20 20]); %Resize image to pixel 7 x 5  binaryImage = im2bw(resizedImage, 0.5); %Convert image to binary image  binaryImage = ~binaryImage;  featureImage = reshape(binaryImage,1,[]); %Reshape image to size 1 x N;  featureImage = double(featureImage); %Convert data type to double  end |

Kode Sumber 8 Fungsi Mendapat Fitur dari Citra

## Fungsi Prediksi Citra Menggunakan KNN

Fungsi prediksi citra menggunakan KNN berdasarkan perancangan yang telah dibuat pada bab 7.1.2.3 dapat dilihat pada kode Kode Sumber 9.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6. | function [results] = predict\_kNN(TestSet,k)  TrainingSet = csvread('dataset.csv');  GroupTrain = csvread('label.csv');    results = kNN(TrainingSet,GroupTrain,TestSet,k);  end |

Kode Sumber 9 Fungsi Prediksi Citra Menggunakan KNN

## Fungsi Algoritma KNN

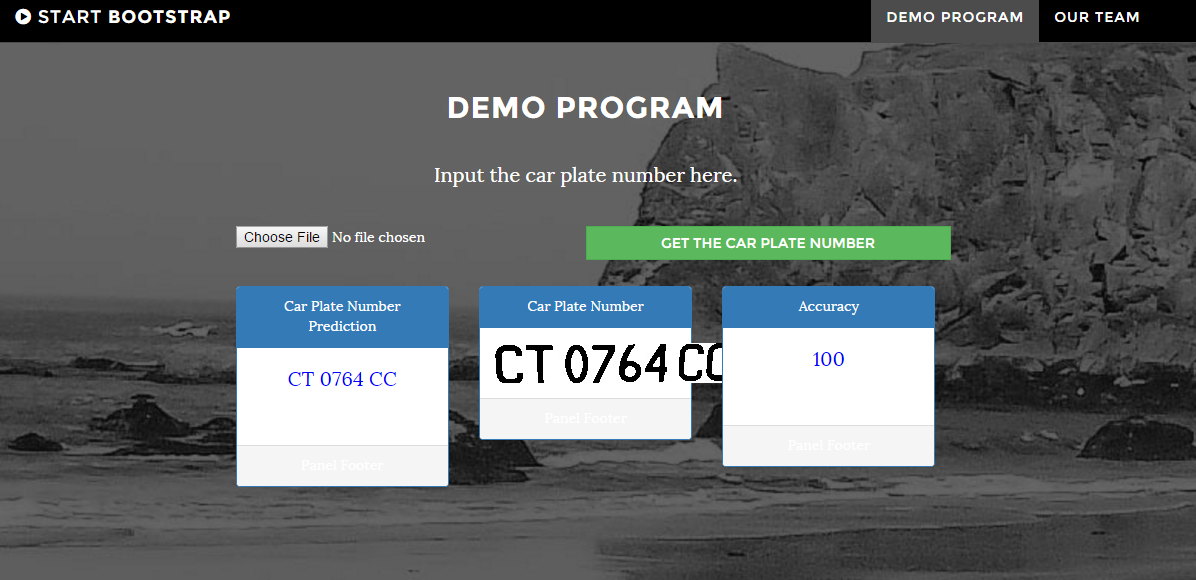
Fungsi prediksi citra menggunakan KNN berdasarkan perancangan yang telah dibuat pada bab 7.1.2.3 dapat dilihat pada Kode Sumber 10.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19. | function [results] = kNN(TrainingSet,GroupTrain,TestSet,k)  [L,L1] = size(TestSet);  for i = 1:L  [L2,L3] = size(TrainingSet);  data\_distance = zeros(L2);  for j = 1:L2  p = TrainingSet(j,:);  q = TestSet(i,:);    data\_distance (j) = sum(abs(p-q));  end    [B,I] = sort(data\_distance);    nearest = GroupTrain(I(1:k));  results(i)=mode(nearest);  end  end |

Kode Sumber 10 Fungsi Algoritma KNN

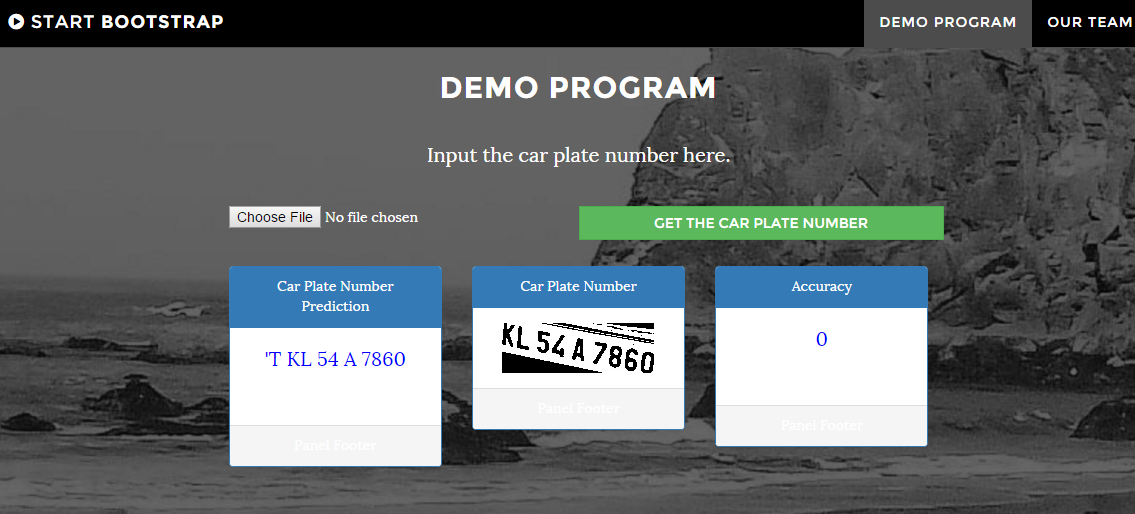
# Result

Hasil pengenalan karakter teks dari citra yang telah diproses menghasilkan hasil sebagai berikut.



Gambar 10.1 Hasil Character Recognition.

Pada gambar 9.1, hasil akurasi mencapai 100% dikarenakan label dari plat nomor yang telah dimasukkan sama dengan prediksi plat nomor kendaraan yang diproses oleh sistem.



Gambar 10.2 Hasil akurasi pengenalan plat nomor yang tidak pas.

Pada gambar 9.1, hasil akurasi mencapai 0% dikarenakan hasil prediksi plat nomor tercampur dengan prediksi karakter lain, sehingga hasil proses penghitungan akurasi menjadi tidak akurat ketika dimasukkan ke dalam sistem.

# Kesimpulan dan Saran

Berdasarkan implementasi dan hasil dari *character recognition*, didapatkan kesimpulan:

* 1. Metode Region Growing dapat melakukan segmentasi terhadap karakter pada citra plat nomor

Saran :

1. ..