

---

## Table of Contents

.....	1
step 1: Image Processing .....	1
step 2: remove noise .....	2
step 3: licence plate detetion .....	3
step 4: character segmentation .....	5
step 5: OCR .....	9
Step 6: Check if Plate is Allowed to Enter after OCR .....	10

```
clc; clear; close all;
```

## step 1: Image Processing

```
originalImage = imread('mainImage.jpg');  
figure, imshow(originalImage);  
title('Original Image');  
if size(originalImage, 3) == 3  
    grayImage = rgb2gray(originalImage);  
else  
    grayImage = originalImage; % already grayscale  
end  
  
% use contrast stretching  
adjustedImage = imadjust(grayImage);  
figure, imshow(adjustedImage);  
title('Grayscale Adjusted Image');
```

Original Image





## step 2: remove noise

```
%-----  
% Method 1: Apply Gaussian Filter for smoothing  
%-----  
gaussian_filter = fspecial('gaussian', [5 5], 1); % [size] and sigma  
filtered_gaussian = imfilter(adjustedImage, gaussian_filter, 'replicate');  
  
%-----  
% Method 2: Apply Median Filter to remove salt & pepper noise  
%-----  
filtered_median = medfilt2(filtered_gaussian, [3 3]);  
  
%-----  
% Display Results  
%-----  
  
figure;  
imshow(filtered_gaussian);  
title('After Gaussian Filter');  
  
figure;  
imshow(filtered_median);  
title('After Median Filter');  
  
fprintf('Noise reduction and filtering completed \n');  
  
Noise reduction and filtering completed
```

---

After Gaussian Filter



After Median Filter



## step 3: licence plate detetion

```
% Load the license plate detector using the XML file
plateDetector = vision.CascadeObjectDetector('licensePlate.xml');

% Detect license plates in the image
bboxes = step(plateDetector, filtered_median);

% Check if any plate is detected
if ~isempty(bboxes)
    % Show the original image with detected plates
    figure, imshow(filtered_median);
    hold on;
    for i = 1:size(bboxes, 1)
        % Draw a green rectangle around each detected plate
        rectangle('Position', bboxes(i,:), 'EdgeColor', 'g', 'LineWidth', 2);
    end
    title('Detected License Plate');

    % Crop the first detected license plate from the image
```

---

```

bbox = bboxes(1,:); % Get the first detected bounding box

% Calculate the new bounding box by reducing 5% from each side
x = bbox(1);
y = bbox(2);
w = bbox(3);
h = bbox(4);

% Reduce 5% from each side
x = x + 0.05 * w; % Shift left by 5% of width
y = y + 0.05 * h; % Shift up by 5% of height
w = w - 0.1 * w; % Reduce the width by 10% (5% from each side)
h = h - 0.1 * h; % Reduce the height by 10% (5% from each side)

% Make sure the new coordinates are within the image boundaries
x = max(x, 1);
y = max(y, 1);
w = min(w, size(filtered_median, 2) - x);
h = min(h, size(filtered_median, 1) - y);

% Crop the image using the new bounding box
plateImage = imcrop(filtered_median, [x, y, w, h]);

% Show the cropped license plate in a new figure
figure, imshow(plateImage);
title('Cropped and Resized License Plate');
else
% If no plates are detected, display a message
disp('No plate detected. ');
return; % Stop execution if no plate detected
end

```

**Detected License Plate**



---

### Cropped and Resized License Plate



## step 4: character segmentation

```
% 1. Adaptive Thresholding
T = adaptthresh(plateImage, 0.79);
binary_plate = imbinarize(plateImage, T);
binary_plate = ~binary_plate;

% 2. Remove small noise
binary_plate = bwareaopen(binary_plate, 50);

% 3. Extract object properties
props = regionprops(binary_plate, 'BoundingBox', 'Centroid', 'Area');

% 4. Filter by size: exclude very small or very large objects
areas = [props.Area];
min_area = 80;
max_area = 2000; % To exclude large frames, e.g., the plate frame
valid_idx = find(areas >= min_area & areas <= max_area);
filtered_props = props(valid_idx);

% 5. Extract Y-centroid to find the baseline
centroids = cat(1, filtered_props.Centroid);
mean_y = mean(centroids(:,2));
tolerance = 25; % Allowable deviation for characters in the same line

% 6. Filter characters in the same line
line_idx = abs(centroids(:,2) - mean_y) < tolerance;
filtered_props = filtered_props(line_idx);
centroids = centroids(line_idx,:);

% 7. Sort from left to right
[~, sort_idx] = sort(centroids(:,1));
filtered_props = filtered_props(sort_idx);

% 8. Display bounding boxes with numbering
figure, imshow(plateImage);
```

---

```

title('Final Filtered Bounding Boxes');
hold on;
for i = 1:length(filtered_props)
    bbox = filtered_props(i).BoundingBox;
    rectangle('Position', bbox, 'EdgeColor', 'b', 'LineWidth', 2);
    text(bbox(1), bbox(2)-10, sprintf('%d', i), 'Color', 'yellow',
'FontSize', 10);
end
hold off;

% 10. Crop and save each character with expanded bounding box
for i = 1:length(filtered_props)
    bbox = filtered_props(i).BoundingBox;
    expand = 5;
    x1 = max(bbox(1) - expand, 1);
    y1 = max(bbox(2) - expand, 1);
    x2 = min(bbox(1) + bbox(3) + expand, size(binary_plate,2));
    y2 = min(bbox(2) + bbox(4) + expand, size(binary_plate,1));
    width = x2 - x1;
    height = y2 - y1;

    char_img = imcrop(binary_plate, [x1, y1, width, height]);
    char_img = imresize(char_img, [50 30]);

    % Confirmation display
    figure, imshow(char_img);
    title(['Character #' num2str(i)]);
end

disp('Characters have been segmented and saved with the image name prefix ');

Characters have been segmented and saved with the image name prefix

```

---



---

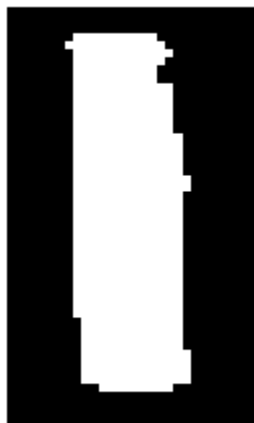
Character #1



Character #2



Character #3



---

Character #4



Character #5



Character #6





---

Character #7



## step 5: OCR

```
% Detect edges using Prewitt operator
edges = edge(filtered_median, 'prewitt');

% Find regions (connected components) and their bounding boxes
regions = regionprops(edges, 'BoundingBox', 'Area');

% Initialize detected plate text to empty string
detectedPlateText = '';

% Initialize maxTextLength to zero
maxTextLength = 0;

for i = 1:numel(regions)
    bbox = regions(i).BoundingBox;

    % Filtering Step
    width = bbox(3);
    height = bbox(4);
    aspect_ratio = width / height;

    % Keep regions with reasonable size and shape
    if width > 80 && height > 20 && aspect_ratio > 2 && aspect_ratio < 6
        croppedRegion = imcrop(filtered_median, bbox);
        ocrResult = ocr(croppedRegion);

        detectedText = strtrim(ocrResult.Text);
        detectedText = regexp(detectedText, '[^A-Za-z0-9 ]', '');

        if ~isempty(detectedText) && length(detectedText) > maxTextLength
            maxTextLength = length(detectedText);
            detectedPlateText = detectedText;
        end
    end
end
end
```

---

```
% Print the detected plate text
if ~isempty(detectedPlateText)
    disp(['Detected Plate Text: ', detectedPlateText]);
else
    disp('No valid plate text detected. ');
    return; % Stop execution if no plate text detected
end
```

*Detected Plate Text: CZI7 KOD*

## Step 6: Check if Plate is Allowed to Enter after OCR

```
% Define the Database of allowed plates
allowedPlates = {...
    'DL6S BYX', ...
    'EA0202AA', ...
    '8117 MP7', ...
    'DJ53096', ...
    'MAT1234', ...
    'ENG2025', ...
    'TEST432', ...
    'CZI7 KOD'
};

% Initialize entry permission flag
isAllowed = false;

% Compare detected plate with allowed plates database
for i = 1:length(allowedPlates)
    if strcmpi(detectedPlateText, allowedPlates{i})
        disp([' Access Granted: Car with Plate ', allowedPlates{i}, ' is
allowed to enter.']);
        isAllowed = true;
        break;
    end
end

if ~isAllowed
    disp(' Access Denied: Car with Plate not authorized to enter. ');
end
```

*Access Granted: Car with Plate CZI7 KOD is allowed to enter.*

*Published with MATLAB® R2024b*