Detecting Cars Using Gaussian Mixture Models

[Try it in MATLAB](matlab:openExample('vision/videotrafficgmm'))

This example shows how to detect and count cars in a video sequence using foreground detector based on Gaussian mixture models (GMMs).

**Introduction**

Detecting and counting cars can be used to analyze traffic patterns. Detection is also a first step prior to performing more sophisticated tasks such as tracking or categorization of vehicles by their type.

This example shows how to use the foreground detector and blob analysis to detect and count cars in a video sequence. It assumes that the camera is stationary. The example focuses on detecting objects. To learn more about tracking objects, see the example titled [Motion-Based Multiple Object Tracking](https://in.mathworks.com/help/vision/examples/motion-based-multiple-object-tracking.html).

**Step 1 - Import Video and Initialize Foreground Detector**

Rather than immediately processing the entire video, the example starts by obtaining an initial video frame in which the moving objects are segmented from the background. This helps to gradually introduce the steps used to process the video.

The foreground detector requires a certain number of video frames in order to initialize the Gaussian mixture model. This example uses the first 50 frames to initialize three Gaussian modes in the mixture model.

foregroundDetector = vision.ForegroundDetector('NumGaussians', 3, ...

'NumTrainingFrames', 50);

videoReader = vision.VideoFileReader('visiontraffic.avi');

for i = 1:150

frame = step(videoReader); % read the next video frame

foreground = step(foregroundDetector, frame);

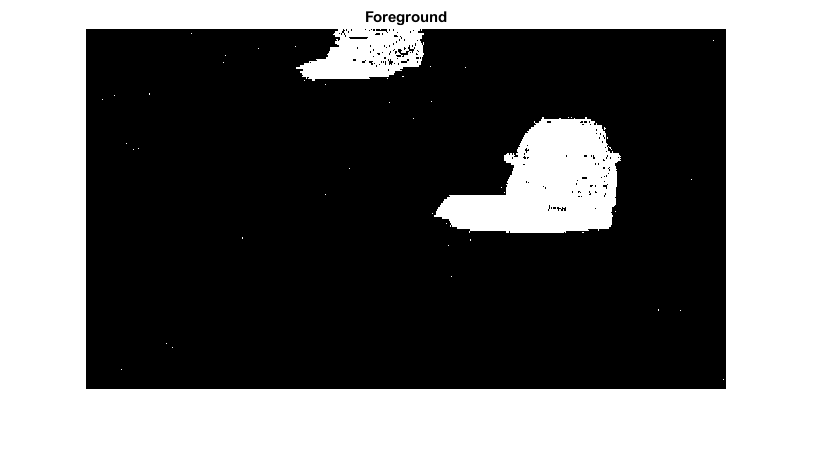
end

After the training, the detector begins to output more reliable segmentation results. The two figures below show one of the video frames and the foreground mask computed by the detector.

figure; imshow(frame); title('Video Frame');



figure; imshow(foreground); title('Foreground');



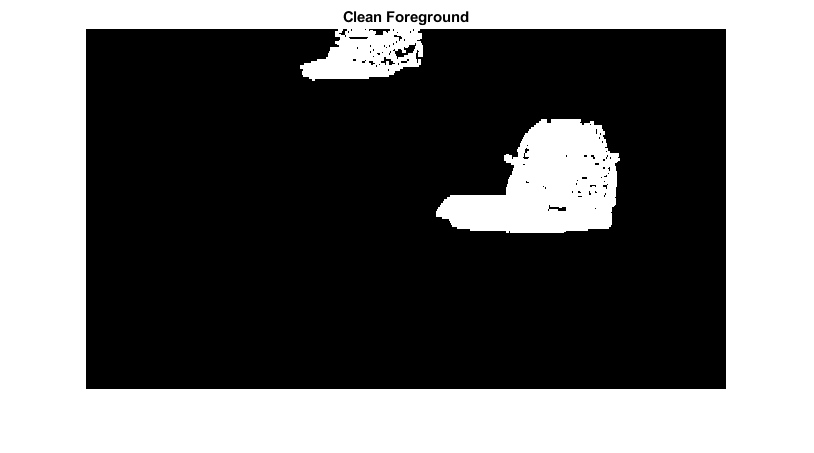
**Step 2 - Detect Cars in an Initial Video Frame**

The foreground segmentation process is not perfect and often includes undesirable noise. The example uses morphological opening to remove the noise and to fill gaps in the detected objects.

se = strel('square', 3);

filteredForeground = imopen(foreground, se);

figure; imshow(filteredForeground); title('Clean Foreground');



Next, we find bounding boxes of each connected component corresponding to a moving car by using vision.BlobAnalysis object. The object further filters the detected foreground by rejecting blobs which contain fewer than 150 pixels.

blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...

'AreaOutputPort', false, 'CentroidOutputPort', false, ...

'MinimumBlobArea', 150);

bbox = step(blobAnalysis, filteredForeground);

To highlight the detected cars, we draw green boxes around them.

result = insertShape(frame, 'Rectangle', bbox, 'Color', 'green');

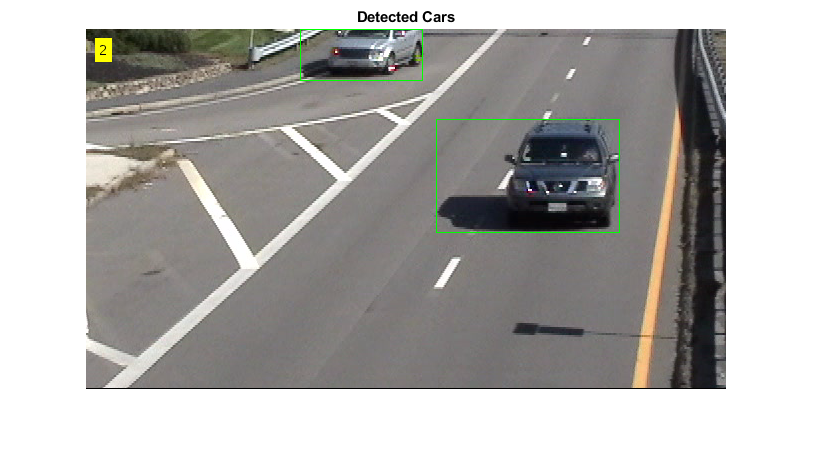
The number of bounding boxes corresponds to the number of cars found in the video frame. We display the number of found cars in the upper left corner of the processed video frame.

numCars = size(bbox, 1);

result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...

'FontSize', 14);

figure; imshow(result); title('Detected Cars');



**Step 3 - Process the Rest of Video Frames**

In the final step, we process the remaining video frames.

videoPlayer = vision.VideoPlayer('Name', 'Detected Cars');

videoPlayer.Position(3:4) = [650,400]; % window size: [width, height]

se = strel('square', 3); % morphological filter for noise removal

while ~isDone(videoReader)

frame = step(videoReader); % read the next video frame

% Detect the foreground in the current video frame

foreground = step(foregroundDetector, frame);

% Use morphological opening to remove noise in the foreground

filteredForeground = imopen(foreground, se);

% Detect the connected components with the specified minimum area, and

% compute their bounding boxes

bbox = step(blobAnalysis, filteredForeground);

% Draw bounding boxes around the detected cars

result = insertShape(frame, 'Rectangle', bbox, 'Color', 'green');

% Display the number of cars found in the video frame

numCars = size(bbox, 1);

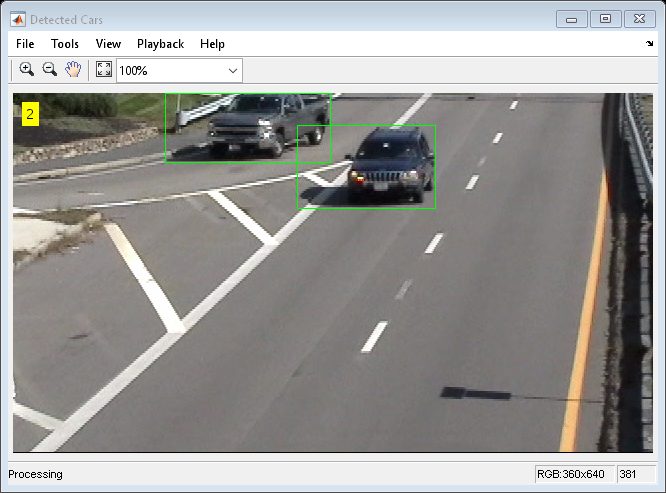
result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...

'FontSize', 14);

step(videoPlayer, result); % display the results

end

release(videoReader); % close the video file



The output video displays the bounding boxes around the cars. It also displays the number of cars in the upper left corner of the video.