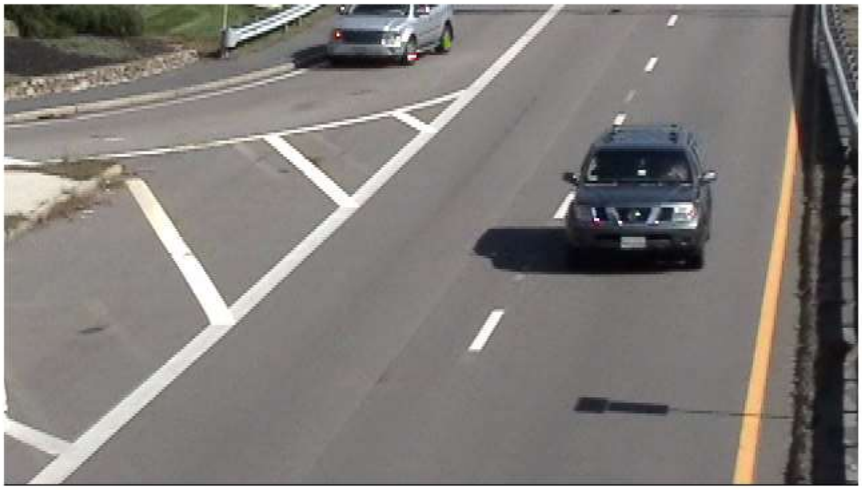
Contents

Initialization

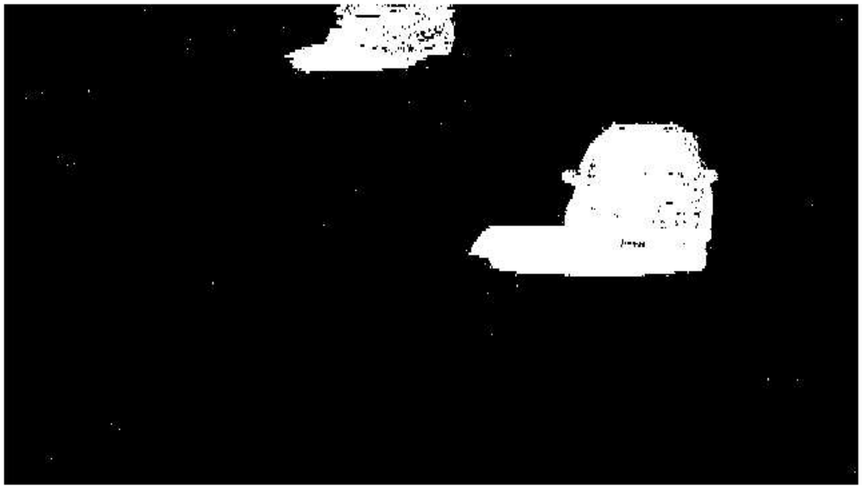
```
clc; clear; close all;
kalmanFilter = []; detectLocation = []; predictedCentroid = []; isTrackInitialized = false; counter = 0;
```

Foreground Detection

```
foregroundDetector = vision.ForegroundDetector('NumGaussians', 3, ...
    'NumTrainingFrames', 50);

videoReader = vision.VideoFileReader('visiontraffic.avi');
% videoReader = vision.VideoFileReader('Kalman_BO.mp4');
for i = 1:150
    frame = step(videoReader); % read the next video frame
    foreground = step(foregroundDetector, frame);
end

figure; imshow(frame); title('Video Frame');
figure; imshow(foreground); title('Foreground');
```



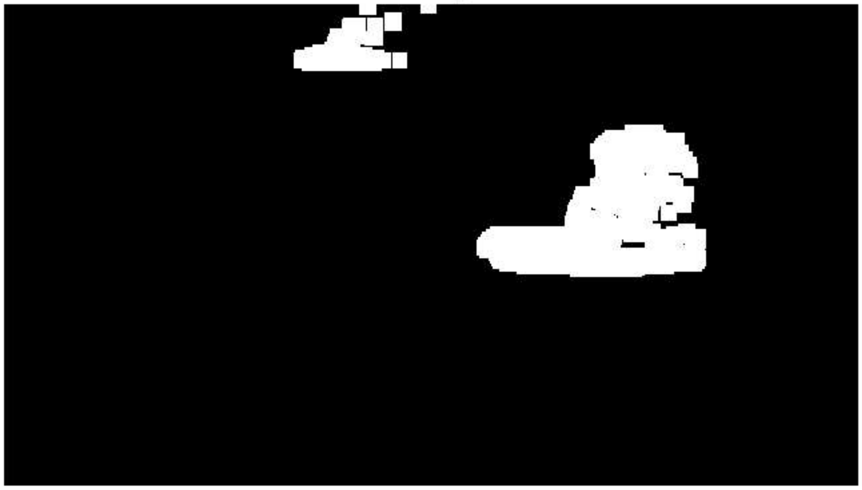
Video Frame



Foreground

Noise Removal from Foreground

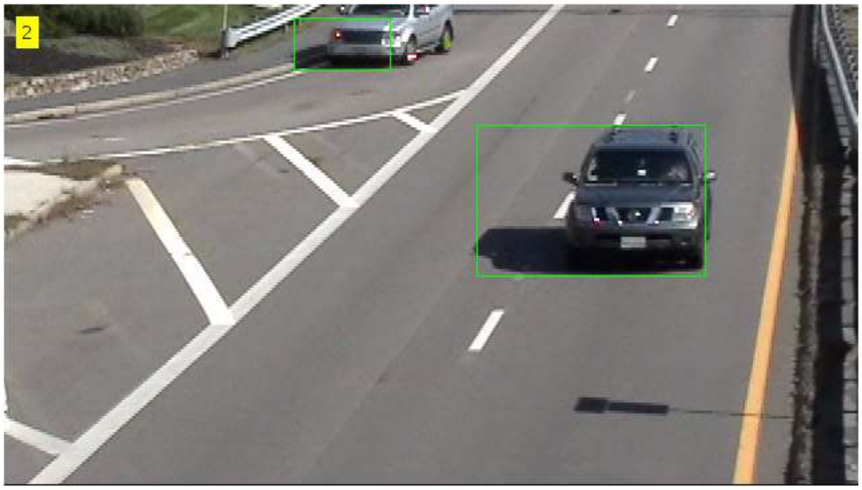
```
se = strel('square', 11);
filteredForeground = imopen(foreground, se);
figure; imshow(filteredForeground); title('Clean Foreground');
```



Clean Foreground

Blob Detection

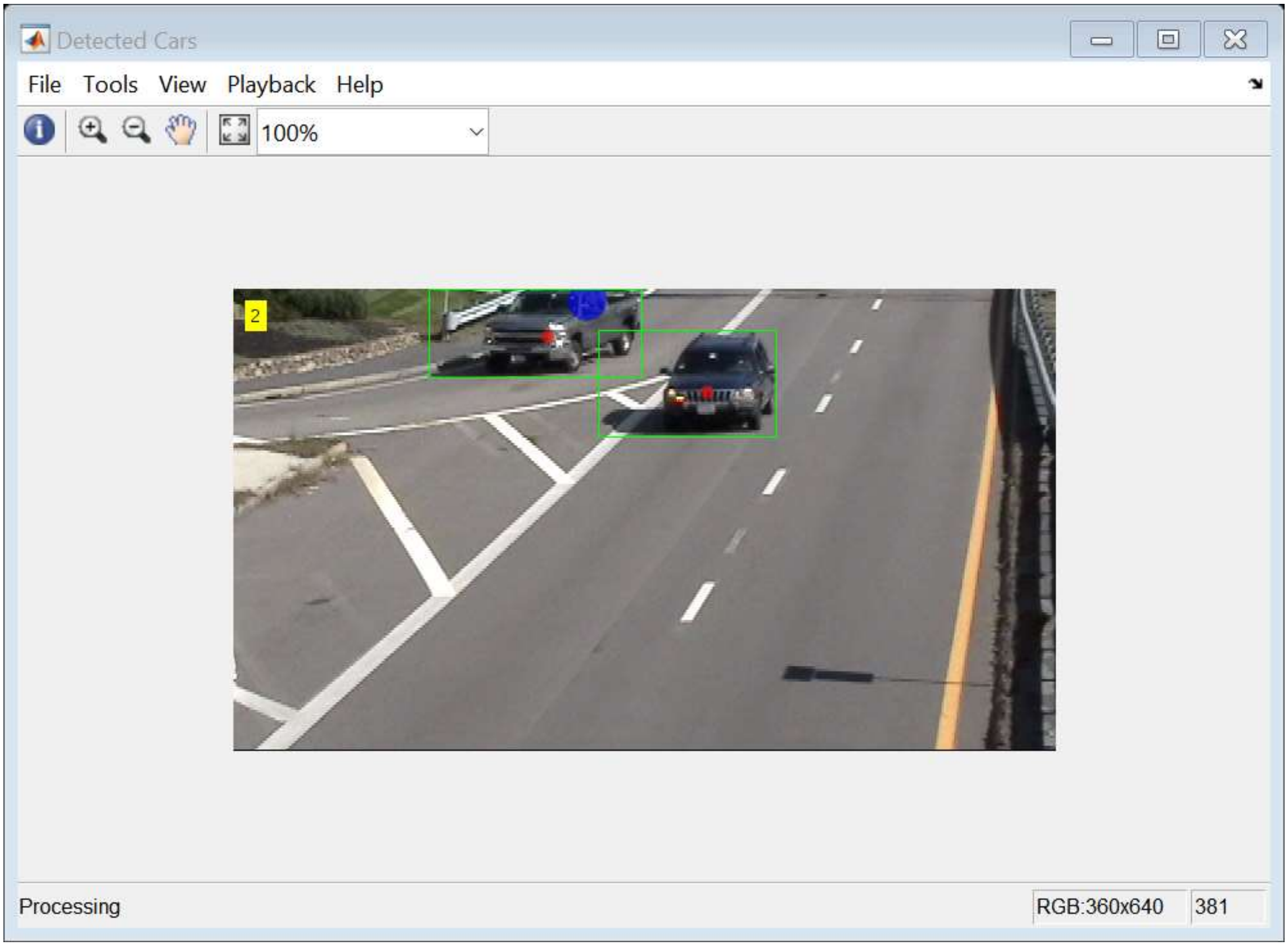
```
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', true, 'CentroidOutputPort', true, ...
    'MinimumBlobArea', 500);
[~, detectedLocation, bbox]  = step(blobAnalysis, filteredForeground);
result = insertShape(frame, 'Rectangle', bbox, 'Color', 'green');

numCars = size(bbox, 1);
result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...
    'FontSize', 14);
fig = figure;
set (fig, 'Units', 'normalized', 'Position', [0,0,1,1]);
imshow(result); title('Detected Cars');
```



Detected Cars

Detect Object for Entire Video

```
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
    [~, detectedLocation, bbox]  = step(blobAnalysis, filteredForeground);

    detectSize = size(detectedLocation,1);
    for i = 1 : 1 : size(detectedLocation,1)
        isObjectDetected = size(detectedLocation, 1) > 0;
        % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
        % Applying Kalman Filter
        if ~isTrackInitialized
            if isObjectDetected
                kalmanFilter = configureKalmanFilter('ConstantAcceleration',...
                    detectedLocation(i,:), [1 1 1]*1e5, [25, 10, 10], 25);
                isTrackInitialized = true;
            end
            label = ''; circle_track = zeros(0,3); circle_predict = zeros(0,3);
        else
            predictedLocation = predict(kalmanFilter);
            circle_predict = [predictedLocation, 15];

            if isObjectDetected
                predictedCentroid = predict(kalmanFilter);
%                 trackedLocation(i,:) = correct(kalmanFilter, detectedLocation(i,:));
                predictedCentroid = int32(predictedCentroid) - bbox(3:4) / 2;
                circle_predict = [predictedCentroid, 15];
%                 bbox = [predictedCentroid, bbox(3:4)];
                label = 'Corrected';
            else
%                 trackedLocation(i,:) = predict(kalmanFilter);
                label = 'Predicted';
            end
%                         circle_track(i,:) = [trackedLocation(i,:), 5];
            %             result = insertShape(frame, 'FilledCircle', [trackedLocation(i,:), 5], ...
            %             'LineWidth',5, 'Color','red');
            result = insertShape(frame, 'FilledCircle', circle_predict, ...
                    'LineWidth',5, 'Color','blue');

        end
        % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%         result = insertShape(frame, 'FilledCircle', circle_predict, ...
%                     'LineWidth',5, 'Color','blue');
%                 result = insertShape(result, 'FilledCircle', circle_predict, ...
%                     'LineWidth',5, 'Color','blue');

    end
    %     tmp = ones(size(pobox,1))*35;
    % Draw bounding boxes around the detected cars

    result = insertShape(result, 'Rectangle', bbox, 'Color', 'green');
    for i = 1 : 1 : detectSize
        result = insertShape(result, 'FilledCircle', [detectedLocation(i,:) 5], ...
            'LineWidth',5, 'Color','red');
        %     plot(pobox(i,1),pobox(i,2),'r*')
    end
    % 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
```



Published with MATLAB® R2016a