**Lucas Kanade**

foregroundDetector = vision.ForegroundDetector('NumGaussians', 5, ...  
 'NumTrainingFrames', 60);  
videoReader = vision.VideoFileReader('live6.mp4');  
  
frameRate = 29.98;  
scale = 1/260; % meter/pixel  
videoPlayer = vision.VideoPlayer('Position',[100 100 600 400]);  
oldPoints = [];  
  
for i = 1:182  
 frame = step(videoReader); % read the next video frame  
 foreground = step(foregroundDetector, frame);  
end  
figure; imshow(frame); title('Video Frame');  
figure; imshow(foreground); title('Foreground');  
se = strel('square', 3);  
filteredForeground = imopen(foreground, se);  
figure; imshow(filteredForeground); title('Clean Foreground');  
blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...  
 'AreaOutputPort', false, 'CentroidOutputPort', false, ...  
 'MinimumBlobArea', 182);  
bbox = step(blobAnalysis, filteredForeground);  
result = insertShape(frame, 'Rectangle', bbox, 'Color', 'blue');  
numCars = size(bbox, 1);  
result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...  
 'FontSize', 14);  
figure; imshow(result); title('Detected Cars');  
videoPlayer = vision.VideoPlayer('Name', 'Detected Cars');  
videoPlayer.Position(3:4) = [300,350]; % 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);  
  
 G = rgb2gray(frame);  
BW = G > 0.8;  
BW2 = bwareaopen(BW, 100);  
BW3 = imfill(BW2, 'holes');  
stats = regionprops('table',BW3,'Centroid');  
stats1 = regionprops('table', BW3,'Area');  
points = table2array(stats);  
oldPoints = table2array(stats1);  
  
 if ~isempty(oldPoints)  
 % Calculate velocity (pixels/frame)  
 vel\_pix = sqrt(sum((points-oldPoints).^2,2));  
 vel = vel\_pix \* frameRate \* scale; % pixels/frame \* frame/seconds \* meter/pixels  
 else  
 vel\_pix = 0;  
 vel = 0;  
 end  
  
   
 % 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', 'blue');  
  
% labels= cellstr(num2str(vel,'%2.2f'));  
 numCars = size(bbox, 1);  
 result = insertText(result, [10 10],numCars, 'BoxOpacity', 1, ...  
 'FontSize', 14);  
  
   
   
 %Visualize the velocity  
 videoFrameOut = insertObjectAnnotation(result, 'circle', ...  
 [points 10\*ones(size(points,1),1)], ...  
 cellstr(num2str(vel,'%0.2f')));  
 imshow(videoFrameOut);  
 step(videoPlayer, videoFrameOut);  
 oldPoints = points;   
 step(videoPlayer, result); % display the results   
end  
  
release(videoReader);% close the video file  
release(videoPlayer);

**Kalman Filter**

clear all;  
close all;  
inputvideo=vision.VideoFileReader('live5.mp4');  
vid1=vision.VideoPlayer;  
while~isDone(inputvideo)  
 frame1=step(inputvideo);  
 step(vid1,frame1);  
 pause(0.005);  
  
end  
imwrite(frame1,'referenceimage.jpg','jpg');  
release(inputvideo);  
release(vid1);  
referenceimage=imread('referenceimage.jpg');  
vid2=vision.VideoFileReader('live5.mp4');  
count=1;  
for i=2:121  
% disp(count)  
 frame=step(vid2);  
 frame2=((im2double(frame))-(im2double(referenceimage)));  
 frame1=im2bw(frame2,0.2);  
 [labelimage]=bwlabel(frame1);  
 stats=regionprops(labelimage,'basic');  
 BB=stats.BoundingBox;  
 X(i)=BB(1);  
 Y(i)=BB(2);  
 Dist=((X(i)-X(i-1))^2+(Y(i)-Y(i-1))^2)^(1/2)  
 Z(i)=Dist;  
% duration = inputvideo.FrameRate;  
M=median(Z);  
Speed=(M)\*680/(26);  
 disp('Speed');  
disp(Speed);  
 if(Dist>10&&Dist<20)  
% display('MEDIUM SPEED');  
 elseif(Dist<10)  
% display('SLOW SPEED');  
 else  
% display('FAST SPEED');  
 end  
 S=strel('disk',6);  
 frame3=imclose(frame1,S);  
 step(vid1,frame1);  
 pause(0.05);  
 count=count+1;  
end