Exercise 1 Report

	Total number of cars	Cars per minute
Traffic_Laramie_1.mp4	6	2
Traffic_Laramie_2.mp4	4	2

Traffic_Laramie_1.mp4 runs for 2 minutes and 58 seconds, which is equivalent to 2.97 mins. (rounded up)
Meaning there are approximately 2 cars per minute. (exact value is 2.020202)

Traffic_Laramie_2.mp4 runs for 1 minutes and 46 seconds, which is equivalent to 1.77 mins. (rounded up)
Meaning there are approximately 2 cars per minute. (exact value is 2.259887)

Background Subtraction is a common technique for detecting moving objects in a series of static camera frames. This is done by creating a foreground mask, which is a binary picture comprising the pixels from the scene's moving objects. Background subtraction computes the foreground mask by subtracting the current frame from a background model that contains the static component of the picture or, more broadly, everything that may be called background given the observed scene's features.

The Frame Difference technique is an algorithm for determining the motion of an object. We were able to distinguish an item moving in the environment using this algorithm. The approach's goal is to detect moving objects based on the difference between the current frame and the reference frame. To locate the moving item, this approach uses pixel-based difference.

The Mixture of Gaussians, abbreviated as MoG, is a combination of k Gaussian distribution models for each backdrop pixel, with k values between 3 and 5. The creator of this MoG thinks that various distributions indicate distinct background and foreground colours. The weight of each of the employed distributions on the model is proportional to the amount of time each colour spends on that pixel. As a result, a pixel is identified as a foreground pixel when the weight of the pixel distribution is low.

The OpenCV API cv2 has built a background subtraction algorithm based on MoG2 and KNN. cv2 and createBackgroundSubtractorMOG2() createBackgroundSubtractorKNN() to generate and construct the foreground mask.

```
#this is to initialise OpenCV - Background Subtractor for MOG2 and KNN
BS_MOG2 = cv2.createBackgroundSubtractorMOG2()
BS_KNN = cv2.createBackgroundSubtractorKNN()
```

The object, cv2.VideoCapture, is used to read the input video or picture sequence.

```
videoCap = cv2.VideoCapture('Traffic_Laramie_1.mp4')
```

Every frame is utilised to calculate the foreground mask as well as to update the backdrop. If you wish to adjust the learning rate used for updating the background model, you may do so by passing a parameter to the apply method.

```
#this is to extract the foreground mask
forgMask = BS_MOG2.apply(frame)
```

This will draw the bounding rectangles over the cars. For this case, it will ignore contours which are small (contour area to be less than 3500). If the contour area is larger than 3500, the bounding rectangle will be drawn.

```
for c in conts:

    #ignore the small-sized contours (this will not create a bounding rectangle if countour area is less than 3500)
    if cv2.contourArea(c) < 3500:
        continue
    #however, if contour area is more than 3500, bounding rectangle will be drawn
    x, y, w, h = cv2.boundingRect(c)
    if y > 305 and x > 0 and x < 2000:

        #draw the bounding rectangle for all contours
        cv2.rectangle(frame,(x,y), (x+w, y+h), (0,255,0), 2)
        #this is to find the centre coordinate of the bounding rectangle, displaying the circle
        xMid = int((x + (x+w)) / 2)
        yMid = int((y + (y+h)) / 2)
        cv2.circle(frame, (xMid, yMid), 2, (35,97,235), 5)
```

This will be the car counter. It will add by one if a car crosses the centre reference line.

```
#cars counter
if xMid > 294 and xMid < 302:
    cars += 1
```

This will display the car counter. We are now ready to display the current input frame as well as the results.

```
#show the foreground mask and original video
cv2.imshow('Foreground Mask', forgMask)
cv2.putText(frame, 'CARS COUNTER: {}'.format(cars), (50, 200), cv2.FONT_HERSHEY_DUPLEX, 3, (0,225,255), 3)
cv2.imshow('Original Video', frame)
```