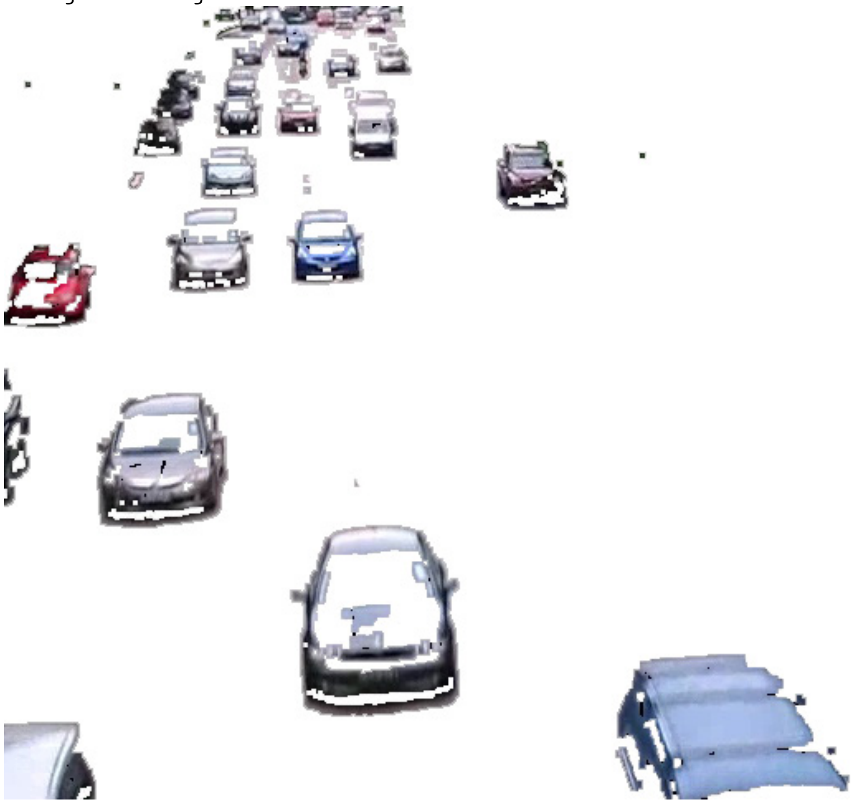
```
# CS4243 Lab 4
# ===================
# Name: Tay Yang Shun
# Matric: A0073063M
```

Background Image

Foreground Objects



```
import os
import cv2
import cv2.cv as cv
import numpy as np

VIDEO_FILE_NAME = 'traffic.mov'

cap = cv2.VideoCapture(VIDEO_FILE_NAME)

frame_width = cap.get(cv.CV_CAP_PROP_FRAME_WIDTH)
frame_height = cap.get(cv.CV_CAP_PROP_FRAME_HEIGHT)
fps = cap.get(cv.CV_CAP_PROP_FPS)
frame_count = cap.get(cv.CV_CAP_PROP_FRAME_COUNT)
print 'width:', frame_width
print 'height:', frame_height
print 'frames per second:', fps
print 'frame count:', frame_count

frame_width = int(frame_width)
frame_height = int(frame_height)
fps = int(fps)
frame_count = int(frame_count)

_, img = cap.read()
avgImg = np.float32(img)

for fr in range(1, frame_count):
  _, img = cap.read()
  alpha = 1 / float(fr + 1)
  cv2.accumulateWeighted(img, avgImg, alpha)
  normImg = cv2.convertScaleAbs(avgImg) # convert into uint8 image
  cv2.imshow('img', img)
  cv2.imshow('normImg', normImg)
  print 'fr = ', fr, 'alpha = ', alpha

cv2.waitKey(0)
cv2.destroyAllWindows()

cap = cv2.VideoCapture(VIDEO_FILE_NAME)
grAvgImg = cv2.cvtColor(normImg, cv2.COLOR_BGR2GRAY)

for fr in range(frame_count):
  _, img = cap.read()
  grImg = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
  diffImg = cv2.absdiff(grImg, grAvgImg)
  thresh, biImg = cv2.threshold(diffImg, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)
  fg = cv2.dilate(biImg, None, iterations=2)
  bgtemp = cv2.erode(biImg, None, iterations=3)
  thresh2, bg = cv2.threshold(bgtemp, 2, 255, cv2.THRESH_BINARY_INV)
  res = cv2.bitwise_and(img, img, fg, fg)
  cv2.imshow('foreground', res)
  cv2.waitKey(100)

fg = res

cv2.imshow('Binarized Image', biImg)
cv2.imwrite('Binarized Image.jpg', biImg)
cv2.waitKey(0)

cv2.imshow('Foreground Image', fg)
cv2.imwrite('Foreground Image.jpg', fg)
cv2.waitKey(0)

cv2.imshow('Background image', bg)
cv2.imwrite('Background Image.jpg', bg)
```

Step 4

To do the averaging of a video, we could have took the average of pixel values of all 285 frames in the video. However, this would take up a lot of memory space as we will need to store all 285 frames in the buffer. To save space, we iterate through all the frames and store the average image of frame 0 to fr in the variable avgImg. To process a new frame, we take a weighted average of the alphas of avgImg and the new img with the value of the weights set according to the number of frames. The variable alpha is defined by the formula 1/ float(fr+1). This is because as more frames are processed, its significance in contributing to the averaged image will be lower.

From OpenCV documentation, cv2. accumulateWeighted has the following function signature: cv2.**accumulateWeighted**(src, dst, alpha[, mask])

What accumulateWeighted does is to calculate the weighted sum of the input image src and the accumulator dst so that dst becomes a running average of a frame sequence according to this formula:

$$dst(x, y) = (1 - alpha) \cdot dst(x, y) + alpha \cdot src(x, y)$$

Step 5

A binary image, biImg is generated via a comparison between the grayscale image of the current frame and the grayscale image of avgImg in step 4. The result is a black/white image where the white areas indicate the presence of the foreground objects.

A dilation of the binary image is done to increase the coverage area of the foreground objects.

Lastly, a bitwise AND is applied on the current frame image, using the binary image as the mask. The foreground objects of the current frame image will hence be extracted out.