In [1]:
```
import numpy as np
import cv2
import os
from glob import glob
import matplotlib.pyplot as plt
```

In [2]:
```
# Convert Video to Picture (Frame by Frame)

cap = cv2.VideoCapture('stitch.avi')
i=0
br = 1
while cap.isOpened():
    ret,frame = cap.read()
    if not ret: break

    if i%br==0:
        name = "Images//{0:03}.jpg".format(i//br)
        #print(name)
        cv2.imwrite(str(name), frame)

    i+=1
print("Total number of Images = %d"%(i))
cap.release()
cv2.destroyAllWindows()
```

```
Total number of Images = 3600
```

In [3]:
```
# Feature Matching in the original Image

img_1 = cv2.imread('Images/1040.jpg')
img_1 = cv2.resize(img_1, (0,0), fx=0.75, fy=0.75)
img1_gray = cv2.cvtColor(img_1, cv2.COLOR_BGR2GRAY)

img_2 = cv2.imread('Images/1070.jpg')
img_2 = cv2.resize(img_2, (0,0), fx=0.75, fy=0.75)
img2_gray = cv2.cvtColor(img_2, cv2.COLOR_BGR2GRAY)

sift = cv2.xfeatures2d.SIFT_create()
kp1, des1 = sift.detectAndCompute(img1_gray,None)
kp2, des2 = sift.detectAndCompute(img2_gray,None)

print("Length of kp1 = %d"%(len(kp1)))
print("Length of kp2 = %d"%(len(kp2)))

cv2.imwrite('original_image_1_keypoints.jpg',cv2.drawKeypoints(img_1,kp1,None))
cv2.imwrite('original_image_2_keypoints.jpg',cv2.drawKeypoints(img_2,kp2,None))

match = cv2.BFMatcher()
matches = match.knnMatch(des1,des2,k=2)

good = []
for m,n in matches:
    if m.distance < 0.9*n.distance:
        good.append(m)
print("Number of Matching Points = %d"%(len(good)))


src = []
dst = []
good_tmp = []
for m in good:
    if m.queryIdx >= min( len(kp1),len(kp2)): continue
    a = kp1[m.queryIdx].pt
    b = kp2[m.queryIdx].pt
    dist = (abs(a[0]-b[0])**2 + abs(a[1]-b[1])**2)**0.5
    if dist>300:
        src.append(a)
        dst.append(b)
        good_tmp.append(m)

print("Number of src points = %d"%(len(src)))
src_pts = np.array(src).reshape(-1,1,2)
dst_pts = np.array(dst).reshape(-1,1,2)

draw_params = dict(matchColor = (0,255,0), # draw matches in green color
                   singlePointColor = None,
                   flags = 2)

img3 = cv2.drawMatches(img_1,kp1,img_2,kp2,good_tmp,None,**draw_params)
cv2.imwrite("original_image_drawMatches.jpg", img3)
```

```
Length of kp1 = 2882
Length of kp2 = 2506
Number of Matching Points = 794
Number of src points = 164
```

Out[3]: True

In [4]:
```
# Feature Matching in the Masked Image

img_1 = cv2.imread('Img/1000.jpg')
img_1 = cv2.resize(img_1, (0,0), fx=0.75, fy=0.75)
img1_gray = cv2.cvtColor(img_1, cv2.COLOR_BGR2GRAY)

img_2 = cv2.imread('Img/1050.jpg')
img_2 = cv2.resize(img_2, (0,0), fx=0.75, fy=0.75)
img2_gray = cv2.cvtColor(img_2, cv2.COLOR_BGR2GRAY)

sift = cv2.xfeatures2d.SIFT_create()
kp1, des1 = sift.detectAndCompute(img1_gray,None)
kp2, des2 = sift.detectAndCompute(img2_gray,None)

print("Length of kp1 = %d"%(len(kp1)))
print("Length of kp2 = %d"%(len(kp2)))

cv2.imwrite('masked_image_1_keypoints.jpg',cv2.drawKeypoints(img_1,kp1,None))
cv2.imwrite('masked_image_2_keypoints.jpg',cv2.drawKeypoints(img_2,kp2,None))

match = cv2.BFMatcher()
matches = match.knnMatch(des1,des2,k=2)

good = []
for m,n in matches:
    if m.distance < 0.8*n.distance:
        good.append(m)
print("Number of Matching Points = %d"%(len(good)))


src = []
dst = []
good_tmp = []
for m in good:
    if m.queryIdx >= min( len(kp1),len(kp2)): continue
    a = kp1[m.queryIdx].pt
    b = kp2[m.queryIdx].pt
    dist = (abs(a[0]-b[0]))**2 + abs(a[1]-b[1])**2)**0.5
    if dist>100:
        src.append(a)
        dst.append(b)
        good_tmp.append(m)

print("Number of src points = %d"%(len(src)))
src_pts = np.array(src).reshape(-1,1,2)
dst_pts = np.array(dst).reshape(-1,1,2)

draw_params = dict(matchColor = (0,255,0), # draw matches in green color
                   singlePointColor = None,
                   flags = 2)

img3 = cv2.drawMatches(img_1,kp1,img_2,kp2,good,None,**draw_params)
cv2.imwrite("masked_image_drawMatches.jpg", img3)
```

```
Length of kp1 = 802
Length of kp2 = 1284
Number of Matching Points = 104
Number of src points = 104
```

Out[4]: True

In [5]:
```
# Masking of Truck using optical flow

ls = []
for i in glob('Images/???.jpg'): ls.append(i)
for i in glob('Images/????.jpg'): ls.append(i)
print("The Total number of Images = %d"%(len(ls)))


st = 1000
end = 1200

frame1 = cv2.imread(ls[st])
prvs = cv2.cvtColor(frame1,cv2.COLOR_BGR2GRAY)
hsv = np.zeros_like(frame1)
hsv[...,1] = 255

cnt=0
for img_name in ls[st+1:end]:
    frame2 = cv2.imread(img_name)
    next = cv2.cvtColor(frame2,cv2.COLOR_BGR2GRAY)

    flow = cv2.calcOpticalFlowFarneback(prvs,next, None, 0.5, 3, 15, 3, 5, 1.2, 0)

    mag, ang = cv2.cartToPolar(flow[...,0], flow[...,1])
    hsv[...,0] = ang*180/np.pi/2
    hsv[...,2] = cv2.normalize(mag,None,0,255,cv2.NORM_MINMAX)
    rgb = cv2.cvtColor(hsv,cv2.COLOR_HSV2BGR)

    blur = cv2.GaussianBlur(hsv[:,:,2],(25,25),0)
    ret4,th4 = cv2.threshold(blur,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

    kernel = np.ones((15,15),np.uint8)
    dilation = cv2.dilate(th4,kernel,iterations = 1)

    cv2.imwrite('Masked_image_of_moving_vehicle/%d.jpg'%(cnt),dilation)
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break
    prvs = next
    cnt+=1

cap.release()
cv2.destroyAllWindows()
```

```
The Total number of Images = 3600
```

In [10]:
```
# Stitching Manually Masked Images

ls = []
for i in glob('Img/???.jpg'): ls.append(i)
for i in glob('Img/????.jpg'): ls.append(i)
print("Length of Manually Masked Images = %d"%(len(ls)))

images = []
for img_name in ls:
    img_1 = cv2.imread(img_name)
    images.append(img_1)
print("Total Number of Images = %d"%(len(images)))


# Try Stiching all possibe combinations of these images
l = len(images)
for i in range(l):
    for j in range(i+1,l):
        images = []
        for img_name in ls[i:j+1]:
            img_1 = cv2.imread(img_name)
            images.append(img_1)
        sticher = cv2.Stitcher_create()
        res,stiched_img = sticher.stitch(images)
        if res==0:
            print("Stitched Image Found from %d to %d "%(i,j))
            cv2.imwrite('Output/%d_%d.jpg'%(i,j),stiched_img)
            cv2.waitKey(0)
            cv2.destroyAllWindows()
```

```
Length of Manually Masked Images = 21
Total Number of Images = 21
Stitched Image Found from 0 to 1
Stitched Image Found from 0 to 2
Stitched Image Found from 0 to 3
Stitched Image Found from 0 to 4
Stitched Image Found from 0 to 5
Stitched Image Found from 0 to 6
Stitched Image Found from 0 to 11
Stitched Image Found from 0 to 13
Stitched Image Found from 0 to 15
Stitched Image Found from 1 to 2
Stitched Image Found from 1 to 3
Stitched Image Found from 1 to 4
Stitched Image Found from 1 to 5
Stitched Image Found from 1 to 6
Stitched Image Found from 1 to 7
Stitched Image Found from 1 to 10
Stitched Image Found from 1 to 14
Stitched Image Found from 2 to 3
Stitched Image Found from 2 to 4
Stitched Image Found from 2 to 5
Stitched Image Found from 2 to 6
Stitched Image Found from 2 to 7
Stitched Image Found from 2 to 9
Stitched Image Found from 2 to 10
Stitched Image Found from 2 to 11
Stitched Image Found from 2 to 15
Stitched Image Found from 3 to 4
Stitched Image Found from 3 to 5
Stitched Image Found from 3 to 6
Stitched Image Found from 3 to 10
Stitched Image Found from 3 to 11
Stitched Image Found from 3 to 12
Stitched Image Found from 3 to 14
Stitched Image Found from 4 to 5
Stitched Image Found from 4 to 6
Stitched Image Found from 4 to 7
Stitched Image Found from 4 to 8
Stitched Image Found from 4 to 9
Stitched Image Found from 4 to 10
Stitched Image Found from 4 to 12
Stitched Image Found from 4 to 13
Stitched Image Found from 4 to 14
Stitched Image Found from 5 to 6
Stitched Image Found from 5 to 7
Stitched Image Found from 5 to 8
Stitched Image Found from 5 to 9
Stitched Image Found from 5 to 10
Stitched Image Found from 5 to 11
Stitched Image Found from 5 to 14
Stitched Image Found from 6 to 7
Stitched Image Found from 6 to 8
Stitched Image Found from 6 to 9
Stitched Image Found from 6 to 10
Stitched Image Found from 6 to 11
Stitched Image Found from 6 to 12
Stitched Image Found from 7 to 8
Stitched Image Found from 7 to 9
Stitched Image Found from 7 to 10
Stitched Image Found from 7 to 11
Stitched Image Found from 7 to 12
```

In [12]:
```
# Function to remove the extra Black part of the images
def trim(frame):
    #crop top
    if not np.sum(frame[0]):
        return trim(frame[1:])
    #crop top
    if not np.sum(frame[-1]):
        return trim(frame[:-2])
    #crop top
    if not np.sum(frame[:,0]):
        return trim(frame[:,1:])
    #crop top
    if not np.sum(frame[:,-1]):
        return trim(frame[:,:-2])
    return frame
```

In [21]:
```
# Cleaning the Output Images
ls = glob('Output/*')
print("Number of Output Images = %d"%(len(ls)))
for img_name in ls:
    img = cv2.imread(img_name)
    img = cv2.resize(img,(1280,720))
    img = trim(img)
    name = img_name.split('\\')[-1]
    cv2.imwrite('Clean_output_images/%s'%(name), img)
```

```
Number of Output Images = 72
```

In [ ]:
```
```