Computer Assignment 6

Due Date:
04/05/2018

1 Background Subtraction

Write your own program for background subtraction and object tracking. Your program can contain the following steps. You can download the following test video (`Q1.avi`) from NYU Class course website.

- Use the first 10 frames of the video. Model the background from the sample video and apply background subtraction method to detect foreground pixels. You need to show the binary foreground image for each frame after background subtraction. You can use `cv2.BackgroundSubtractorMOG()`
- Based on the foreground map obtained for Frame 10 in previous step, generate a bounding box surrounding the object corresponding to the walking man. You need to show the image with the bounding box.
- Determine the location of the walking man in subsequent 10 frames, using the template matching method (exhaustive search), starting with the template in the bounding box of Frame 10. Specifically, once you have the tracked bounding box in frame t, you use the pattern in this bounding box as the template to determine its position in frame t+1. You need to show the tracking result for each frame by showing the frame image with the tracked bounding box. You should write your own program to perform template matching.

2 Camera motion detection

Write your own program to find global camera motion between two frames and detecting moving object. Your program can contain the following steps. You can download the testing video (`Q2.avi`) from NYU Class course website and use the first and fourth frame of the video.

- Find the optical flow between each frame pair. Use Harris detector to find feature points in first frame, and use Lucas-Kanade method to their corresponding positions in the second frame. You can use `cv2.cornerHarris()` for detecting features, and `cv2.calcOpticalFlowPyrLK()` for flow estimation. You need to show the feature points matching result between two frames as you did in programming assignment 3 question 3(c).
- Calculate the homography mapping between the two frames using the matching pair positions you found in (a). Use the least-squared method to calculate the homography. You are not allowed to use `cv2.findHomography()` or other build-in function. You need to show the original two images and image of second frame (frame 4) after homography transform.
- Detect the moving object (a binary image) by comparing frame 1 and mapped frame 4.

3 Sample code for Video with OpenCV

A useful link for getting started with video in OpenCV is here. Also you can refer to the following code for loading and playing a video.

```
import numpy as np
import cv2

src = 'Q1.avi'
# read video from file! set argument to 0 to capture from webcam
vid = cv2.VideoCapture(src)

width = vid.get(cv2.cv.CV_CAP_PROP_FRAME_WIDTH)
hight = vid.get(cv2.cv.CV_CAP_PROP_FRAME_HIGHT)
framerate = vid.get(cv2.cv.CV_CAP_PROP_FPS)
numframes =  vid.get(cv2.cv.CV_CAP_PROP_FRAME_COUNT)

count = 0

while count<numframes:
ret,frame = vid.read() #read a frame from video
cv2.imshow('video',frame)
cv2.waitKey(80)
count+=1

vid.release()
```