EE 243 Advanced Computer Vision: Homework 1 Report

Tue-Cuong Dong-Si
tdongsi@ee.ucr.edu

1. Homework information

The chosen 100 frames are one in every two frames of the last 200 frames of the video *Day3_Scene28_Camera22.mpeg*. This dataset features two persons walking toward the camera and a third person walking by in the later section.

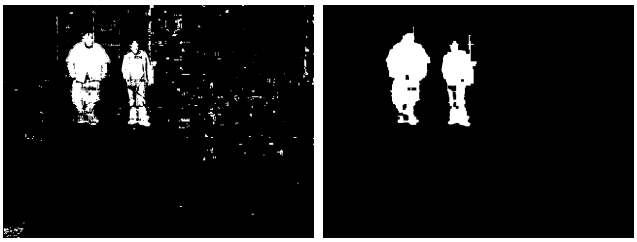
2. Image segmentation

In this section, the procedure to localize and segment walking people from a given frame is described.

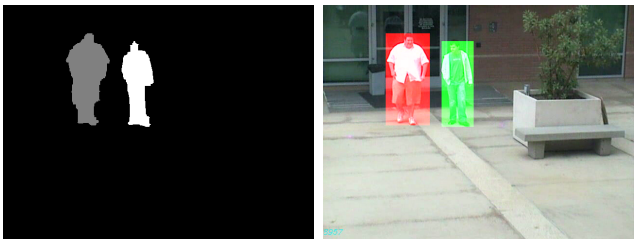
Since the camera pose is generally stable, we use background subtraction to detect image areas with walking people. The first 20 frames of the background are being averaged to obtain the background image. From the background image, any pixel in the current input image that has any of RGB or HSV color values that are different enough from the corresponding pixel in the background image are classified as foreground. From the obtained foreground, the image is morphologically dilated to remove the noises. Then, we perform connected component detection to find and isolate the significant components. Only components with certain large size, corresponding to walking persons, will be retained. The minimum component size is based on the minimum size of human that can appear in the scene, which is 2000 pixels in our method. The procedure of image segmentation is illustrated in Fig. 1.

In general, RGB color space is good enough to remove most of the background. However, it has a weakness against cast shadows when the differences in lighting between background and the current image can gives the impression of a foreground object, as shown in Fig. 2.a. To remove this weakness, we has to turn to HSV color space which is more robust against shadows. Since most changes in illumination is mostly reflected in the V channel, increasing the threshold value in this channel will remove false positives by the shadows. However, background subtraction in HSV color space is very noisy with many false negatives caused by Hue and Saturation channel, probably due to poor thresholding values. Therefore, we find the intersection (AND) of background subtraction results in both HSV and RGB color space and have a better result. The performance of the segmentation algorithm over the data set is shown in Fig. 3. In general, it gives satisfactory segmentation outputs if the persons do not appear close or overlapping in the images.

(a) Background image (b) Input image

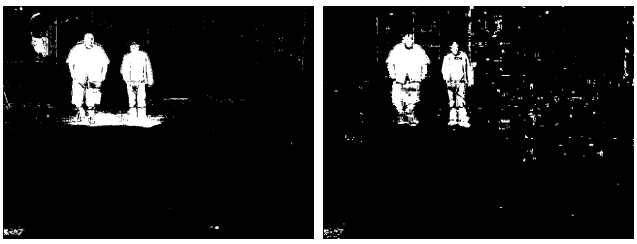


(c) Extracted foreground (d) Smoothed foreground



(e) Segmented image (f) Final output

Figure 1. Different steps of the segmentation algorithms.



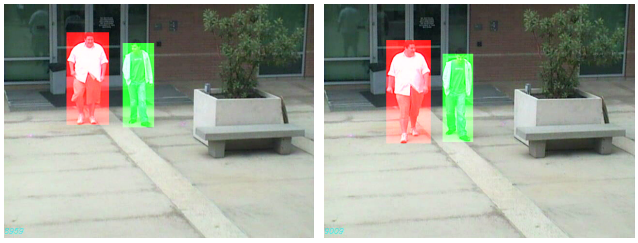
(a) Extracted foreground (RGB) (b) Extracted foreground (HSV)

Figure 2. Background subtraction in RGB and HSV color spaces, at frame 5.

3. Multi-target tracking using Kalman filter

In this section, we would use Kalman filter to find the tracks of the walking people in the scene with *partially occluded instances*. Mean-shift tracking is attempted but the results are generally unsatisfactory, probably because of poor initialization and changing target's size.

For each detected person, we would use a separate Kalman filter to independently track the positions (x, y) and velocities (u, v) of the bounding box's center in the image. In each Kalman filter, we will keep a 4×1



(a) Frame 5 (b) Frame 30

(c) Frame 55 (d) Frame 85

Figure 3. Segmentation results over some frames.

state vector, a 4×4 covariance matrix and a template of segmented silhouette (Fig. 1e) for data association. The measurements are the coordinates of the bounding box's center after a person is detected, segmented and matched to the corresponding tracked person. In data association, we store the latest silhouette template in the Kalman filter and shift to the predicted position in the propagation step. If a segmented component overlaps significantly with a projected template, then it is used as a new measurement for the corresponding filter. If a template is not detected in the frame, probably because of occlusion, the state and covariance matrix is just propagated and not corrected until the next available measurments. The system model and measurement model of each Kalman filter can be written as follows:

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ u_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ u_k \\ v_k \end{bmatrix} + \begin{bmatrix} w_x \\ w_y \\ w_u \\ w_v \end{bmatrix} \tag{1}$$

$$\mathbf{z}_{k+1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \mathbf{x}_{k+1} + \mathbf{n} \tag{2}$$

In our model, the boxes are assumed to move at a constant velocity, but statistically, we expect undetermined accelerations (w_u, w_v) occur with a Gaussian profile. In this formulation, the matrix $\mathbf{F}$ and $\mathbf{H}$ are the same, regardless of the bounding box's positions in the image. It is linear and time-invariant, which facilitates simple Kalman filter implementation. In addition, it also has the advantage that it does not require intrinsic and extrinsic parameters of the camera.

Figure 4. Tracking result using Kalman filter. Targets' tracks are denoted by red, green, blue stars (*).

For short video sequences, the approach is satisfactory, as shown in Fig. 4. However, the method is inherently *flawed* that prevents it from being robust or scalable to larger systems. Firstly, to effectively track people moving in a 3D world, their 3D positions and 3D velocity should be tracked as the state vector in Kalman filter. In limited simple scenarios, where people is walking relatively smoothly (no sitting, jumping, running, etc.), we can assume that their positions and velocities are moving in 2D plane. Even in those cases, the system model is not a linear function of their 2D positions and velocities in the world frame, as it is subject to camera-world relative pose and requires intrinsic and extrinsic camera parameters. Given no such information, the formulation in Eq. (1)-(2) is the *more feasible* approach to the tracking. In addition to being a greatly simplified model, in that formulation, the assumption that a box's acceleration independent of its position and velocity is not true in general. When a target is closer to the camera, its velocity changes are much larger and highly dependent on its current position and velocity. This contradict the Kalman filter assumption that the accelerations, modelled as noises, are independent of the states. Lastly, even when the target is far away, by modelling accelerations as noises, their covariance matrices need to be manually "tuned" to be representative of all motion in the video sequence.