Object Tracking In Video Sequence

I. Object Tracking in a Video Sequence using Optical Flow Vectors

IN this section we look at approach for object tracking in video sequence. We are given a video sequence.The Aim is to detect objects that are in motion in the video and successfully track these objects.

A. *Optical Flow Problem Statement and Assumptions*

Object tacking is achieved by detecting the objects that are in motion in the video sequence. We need to detect the motion of the object at time instant t and t+δt.

Assumptions
1.Motions flow between objects at time t and t+δt is small.
2.Displacement of image contents between two nearby instants is small and approximately constant
within a small neighbourhood of point under consideration.
3.Velocity of object within a small neighbourhood will remain constant . 4.Brightness constancy is maintained , if a section of image is displaced from (x,y) to (a,b) the brightness of the image will remain approximately the same.
the object cannot abruptly move from one location to another between successive frames.
5.No or minimum illumination changes in video sequence.
6.The objects are assumed to be rigid objects.

Optical Flow Equations in continuous domain
Each image is considered as 2D continuous array (x,y) with intensity I(x,y) corresponding to each pixel location and I(x,y,t) represents the intensity map of image at time t.

We need to find the motion of objects at time instants t and t+δt therefore we need to find the motion of the objects between image I(x,y,t) and I(x,y,t+δt)

A pixel at location (x,y,t) with intensity I(x,y,t) will have moved by $\delta x, \delta, y \delta t$ between the two image frames,Image constraint equation is given by
$I(x,y,t) = I(x+\delta x, y+\delta y, t+\delta t)$

Given a pixel location,pixel intensity of neighbouring pixels will not change abruptly.Usually there will be a gradual change in pixel intensities in real world images.We have made small motion flow assumption ,hence the change images at time instants t and t+k given by I(x,y,t) and I(x,y,t+k) will be small,and essentially the current pixel would have been replaced by some neighbourhood pixels whose pixel intensity values will be differ by a small amount.

We can represent can approximate this using a Taylor's series approximation due to the Brightness constancy assumption of brightness constancy and small displacement assumptions

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + [H.O.T.]$$

We approximate equation and discard the higher order terms. Substituting the image constraint equation we get below equation

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0$$

or

$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t}\frac{\delta t}{\delta t} = 0$$

which results in

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0$$

where V_x, V_y are the x and y components of the velocity or optical flow vectors of image $I(x,y,t)$ and $\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}$ and $\frac{\partial I}{\partial t}$ are the partial derivatives of the image at (x,y,t) in the corresponding directions.

$I_x V_x + I_y V_y = -I_t$

This is an equation in two unknowns and cannot be solved as such. To find the optical flow another set of equations is needed, given by some additional equation or constraint.

Lucas-Kanade method

We will attempt to solve the above equation using Lucas-Kanade method
The Lucas-Kanade method assumes that the displacement of the image contents between two nearby instants (frames) is small and approximately constant within a neighbourhood of the point p under consideration .It require the small displacement and brightness constancy assumptions

$$\begin{aligned}
&I_x(q_1)V_x + I_y(q_1)V_y = -I_t(q_1) \\
&I_x(q_2)V_x + I_y(q_2)V_y = -I_t(q_2) \\
&\vdots \\
&I_x(q_n)V_x + I_y(q_n)V_y = -I_t(q_n)
\end{aligned}$$

where $q_1, q_2, \ldots, q_n$ are the pixels inside the window, and $I_x(q_i), I_y(q_i), I_t(q_i)$ are the partial derivatives of the image I with respect to position x,y and time t, evaluated at the point q_i and at the current time.

B. *Calculation of parameters of Equation in Discrete Domain*

In the discrete domain the image is a pixel intensity map I(x,y,k) where k is frame numbers in the video sequence corresponding to time t.

We will consecutive frame for our analysis ie the frames at time t and t+δt correspond to consecutive frames in the image sequence.

Calcuation of Gradients
1.To compute the gradients in X and Y directions at each pixel locations we have used normalized sobel operator x and y directions.

2.Before applying the gradient operations we perform gaussian smoothing operation .Typically image is divide into sub frames and constraint equations are applied to individual frames and same effect can be achieved by performing smoothing operation and then proceeding with further steps

3.The gradient in time is simply the difference in the images between two consecutive frames I2-I1.

Matrix Form of Equations
These equations can be written in matrix form $Av = b$, where

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix}, \quad v = \begin{bmatrix} V_x \\ V_y \end{bmatrix}, \quad \text{and} b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix}$$

This system has more equations than unknowns and thus it is usually over-determined. The Lucas-Kanade method obtains a compromise solution by the least squares principle. Namely, it solves the 2X2 system

$A^T A v = A^T b$ or
v = $(A^T A)^{-1} A^T b$
where A^T is the transposed matrix transpose of matrix A. That is, it computes

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i) I_y(q_i) \\ \sum_i I_x(q_i) I_y(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i) I_t(q_i) \\ -\sum_i I_y(q_i) I_t(q_i) \end{bmatrix}$$

with the sums running from i=1 to n.
Solving the Matrix Equations and Conditions for Reliable Solution
Conditions for good solution
To solve the above equation we require the matrix $A^T A$ to be invertible.

We will first calculate the eigen values of the matrix

if Matrix is singular .The matrix must not have any 0 eigenvalue .
usually at the edge the matrix becomes singular and values in inverse matrix 1/0
at homogeneous regions the gradients have low values thus $A^T A$ has low values in these regions Eigenvalue of matrix should not be small.we will eliminate the eigenvalue below a specified threshold.

The approach would be ignore the eigenvalue that are small ie in homogeneous regions and consider eigenvalue that are high ie in the region about the edges

To solve the matrix equations

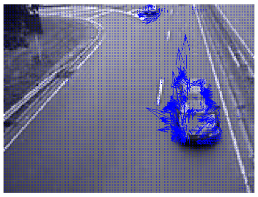
a . We calculate the Gxx,Gyy,Gxy represent the gradients in X and Y directions at each pixel locations
b. The approach would be to divide the image into number of sum images and solve the obtained matrix equation in each of these sub images.
But since we have used Gaussian smoothing on the image to average the pixel values in a neighbourhood.We can directly processed to application of the solving the optical flow matrix equations for each point in the image.
c.We would expect a region with constant texture to have similar low eigen values and we will ignore optical flow vectors at these positions corresponding to low eigen values.
d . $A^T A$ will have elements of Axx and Ayy along the diagonal elements and Axy at other positions of the matrix,$A^T A$ will be a 2X2 matrix ,thus determinant is given by Axx is sum of Gxx^2 and Ayy is the sum of Gyy^2 and Axy is the sum of Gxy of all pixel locations.
$detA = (Axx. * Ayy) - (Axy.^2)$; and its trace is given by trA=Axx+Ayy;

e . the characteristic polynomial is $\lambda^2 - \lambda(Axx + Ayy) - Axx * Ayy + Axy^2 = 0$
f . eigenvalue are given by $\frac{trA - \sqrt{trA^2 - 4 * detA}}{2}$
g . we take the inverse and solve the corresponding matrix equations
h. We will ignore the optical flow vectors corresponding to the eigen values that are too small
and remaining vectors are our optical flow matrix.The values of matrix with values 0 are regions where no motion has occurred and the regions with higher values are regions where motion is dominant between successive edges.

C. *Segmentation and Tracking of objects*

Now since optical flow vectors have been detected ie we know the region where motion has occured,The task is to segment and track the objects in motion.
Since optical flow vectors will be strong in the region where

Fig. 1. optical flow vectors



motion is more.It also indicated the region where object is present.

Hence given vectors we obtain the vector dd=vx+vy which contains only information that the particular position has optical flow vector present

we apply edge detection to this optical vector flow map.This should give us the edge of the object boundary present in the region where motion has occured.

Fig. 2. after applying edge detection

We apply morphological closing operations so that region within the boundary is filled as we obtain a blob which spans the actual object area

Fig. 3. After morphological closing operation

there may be many spurious optical flow vector due to improper solution of optical flow equations or noise hence we nee to only consider blob that are larger in size

hence again we use morphological operations and select blob objects which contains number of pixels than the specified threshold depending on the application .All blob which contains less number of pixels are eliminated.

Fig. 4. After selecting blob's of larger number of pixels

We determine the centroid and bounding box of the blob

we add the velocity vectors to the centroid bounding box which our estimate of the position of object in the next frame

Fig. 5. centroid of next frame and bounding box

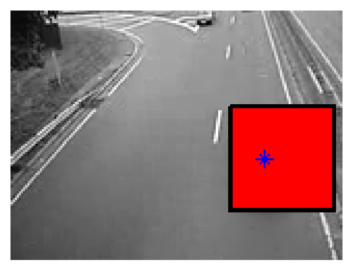


Fig. 6. centroid of next frame and bounding box