Alexandria Univeristy
Faculty of Engineering
Computer and communications
Engineering Department

CCE: Computer Vision
Assigned: Dec 26, 2022
Due: Jan 23, 2022

Assignment 3 - Part 2

Tracking Objects in Videos

1 Overview

One incredibly important aspect of human and animal vision is the ability to follow objects and people in our view. Whether it is a tiger chasing its prey, or you trying to catch a basketball, tracking is so integral to our everyday lives that we forget how much we rely on it. In this assignment, you will be implementing an algorithm that will track an object in a video.

You will implement the Lucas-Kanade tracker, where two video sequences are provided: a car on a road, and a helicopter approaching a runway link. To initialize the tracker you need to define a template by drawing a bounding box around the object to be tracked in the first frame of the video. For each of the subsequent frames the tracker will update an affine transform that warps the current frame so that the template in the first frame is aligned with the warped current frame.

2 Preliminaries

An image transformation or warp is an operation that acts on pixel coordinates and maps pixel values from one place to another in an image. Translation, rotation and scaling are all examples of warps. We will use the symbol W to denote warps. A warp function W has a set of parameters p associated with it and maps a pixel with coordinates $x = [u, v]^T$ to $x = [u', v']^T$ with $x' = W(x; p)$

An affine transform is a warp that can include any combination of translation, anisotropic scaling and rotations. An affine warp can be parametrized in terms of 6 parameters $p = [p1, p2, p3, p4, p5, p6]^T$. One of the convenient things about an affine transformation is that it is linear; its action on a point with coordinates $x = [uv]^T$ can be described as a matrix operation:

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = W(p) \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Where W(p) is a 3 x 3 matrix such that:

$$W(p) = \begin{bmatrix} 1 + p_1 & p_3 & p_5 \\ p_2 & 1 + p_4 & p_6 \\ 0 & 0 & 1 \end{bmatrix}$$

Note that for convenience when we want to refer to the warp as a function we will use W(x; p) and when we want to refer to the matrix for an affine warp we will use W(p). Table 1 contains a summary of the variables used in this assignment.

Good Luck!

Dr. Marwan Torki
Eng. Menna Ghanem

Table 1: Summary of Variables

Symbol	Vector/Matrix Size	Description
u	1×1	Image horizontal coordinate
v	1×1	Image vertical coordinate
$\mathbf{x}$	2×1 or 1×1	pixel coordinates: (u, v) or unrolled
$\mathbf{I}$	$m \times 1$	Image unrolled into a vector (m pixels)
$\mathbf{T}$	$m \times 1$	Template unrolled into a vector (m pixels)
$\mathbf{W(p)}$	3×3	Affine warp matrix
$\mathbf{p}$	6×1	parameters of affine warp
$\frac{\partial \mathbf{I}}{\partial u}$	$m \times 1$	partial derivative of image wrt u
$\frac{\partial \mathbf{I}}{\partial v}$	$m \times 1$	partial derivative of image wrt v
$\frac{\partial \mathbf{T}}{\partial u}$	$m \times 1$	partial derivative of template wrt u
$\frac{\partial \mathbf{T}}{\partial v}$	$m \times 1$	partial derivative of template wrt v
$\nabla \mathbf{I}$	$m \times 2$	image gradient $\nabla \mathbf{I}(\mathbf{x}) = \left[\frac{\partial \mathbf{I}(\mathbf{x})}{\partial u} \frac{\partial \mathbf{I}(\mathbf{x})}{\partial v}\right]$
$\nabla \mathbf{T}$	$m \times 2$	image gradient $\nabla \mathbf{T}(\mathbf{x}) = \left[\frac{\partial \mathbf{T}(\mathbf{x})}{\partial u} \frac{\partial \mathbf{T}(\mathbf{x})}{\partial v}\right]$
$\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$	2×6	Jacobian of affine warp wrt its parameters
$\mathbf{J}$	$m \times 6$	Jacobian of error function L wrt $\mathbf{p}$
$\mathbf{H}$	6×6	Pseudo Hessian of L wrt $\mathbf{p}$

3 Lucas-Kanade: Forward Additive Alignment

A Lucas Kanade tracker maintains a warp W(x; p) which aligns a sequence of images It to a template T. We denote pixel locations by x, so I(x) is the pixel value at location x in image I. For the purposes of this derivation, I and T are treated as column vectors (think of them as unrolled image matrices). W(x; p) is the point obtained by warping x with a transform that has parameters p. W can be any transformation that is continuous in its parameters p. Examples of valid warp classes for W include translations (2 parameters), affine transforms (6 parameters) and full projective transforms (8 parameters). The Lucas Kanade tracker minimizes the pixel-wise sum of square difference between the warped image I(W(x; p)) and the template T.
In order to align an image or patch to a reference template, we seek to find the parameter vector p that minimizes L, where:

$$L = \sum_x [T(x) - I(W(x;p))]^2$$

In general this is a difficult non-linear optimization, but if we assume we already have a close estimate p of the correct warp, then we can assume that a small linear change Δ p is enough to get the best alignment. This is the forward additive form of the warp. The objective can then be written as:

$$L = \sum_x [T(x) - I(W(x; p + \Delta p))]^2$$

Expanding this to the first order with Taylor Series gives us:

$$L \sim \sum_x [T(x) - I(W(x; p)) - \Delta I(x) \frac{\mathrm{d}W}{\mathrm{d}p} \Delta p]^2$$

Where $\Delta I(x) = [\frac{\mathrm{d}I(x)}{\mathrm{d}u}, \frac{\mathrm{d}I(x)}{\mathrm{d}v}]$, which is the vector containing the horizontal and vertical gradient at pixel location x. Rearranging the Taylor expansion, it can be rewritten as a typical least squares approximation:$\Delta p^* = argmin_{\Delta p} ||A\Delta p - b||^2$

$$\Delta p^* = argmin_{\Delta p} \sum_x [\Delta I(x) \frac{\mathrm{d}W}{\mathrm{d}p} \Delta p - \{T(x) - I(W(x; p))\}]^2$$

This can be solved with $\Delta p^* = (A^T A)^{-1} A^T b$ where:

$$(A^T A) = H = \sum_x [\Delta I(x) \frac{\mathrm{d}W}{\mathrm{d}p}]^T [\Delta I(x) \frac{\mathrm{d}W}{\mathrm{d}p}]$$

$$A = \sum_x [\Delta I(x) \frac{\mathrm{d}W}{\mathrm{d}p}]$$

$$b = T(x) - I(W(x; p))]$$

Once Δ p is computed, the best estimate warp can be updated p = p + Δp, and the whole procedure can be repeated again, stopping when Δp is less than some threshold.

4 Tracker Implementation

Write a function that computes the optimal local motion from frame I_t to frame I_{t+1} that minimizes the error L. The function takes three inputs: the current frame I_t, the next frame I_{t+1}, and an (4 x 1) vector that represents a rectangle on the image frame I_t. The four components of the rectangle are [x, y, w, h], where (x, y) is the top-left corner and (w, h) is the width and height of the bounding box. The rectangle is inclusive, i.e., in includes all the four corners.

Start with the first frame provides I_0, detect the object to be tracked (car/helicopter), and draw the (4 x 1) rectangle over the detected object. Iterate over the rest of the frames and output a tracking video of the object.

5 Notes

You are required to deliver the following:

- Your code.
- A video of your tracker on the car dataset (car.mp4).
- A video of your tracker on the helicopter dataset (helicopter.mp4).
- Report.

You can work in groups of 3.