CITS 4402 Computer Vision

Ajmal Mian

Lecture 11 – Optical Flow and Tracking
Objectives of this lecture

- Optical flow and motion field
- Tracking
- Applications
- Optical Flow
- Lucas Kanade Tracking algorithm
- Mean Shift algorithms
What happens when a camera moves?

- Imagine a moving video camera

- There will be relative motion between the camera and the scene

- This motion will appear as displacement of pixels in consecutive images

- If the motion is small between consecutive images, the displacement of pixels will be small
Motion Field

Time t

Displacement field

Time t+1
Optic Flow and Motion Field

- **Motion Field:**
  - Projection of 3D relative velocity vectors onto the image plane.

- **Optic Flow:**
  - Observed 2D displacement of pixels (brightness patterns) in the image.

Most of the time, motion field is what we want to find.
But optic flow is what we actually measure from the image.
Example

- Optic flow $\neq$ Motion field
Aperture Problem

- We can only measure the component of optic flow in the direction of intensity gradient
- We cannot measure the component that is tangent to the intensity gradient
The Optical Flow Constraint

- Intensity at a point in an image is a function of its position and time i.e. $I(x, y, t)$

- To derive the optical flow constraint, we are concerned with its spatiotemporal variation

- To see how $I$ changes after a small change in time and position, we look at its Taylor expansion:

$$I(x + dx, y + dy, t + dt) = I(x, y, t) + \frac{\partial I}{\partial x} dx + \frac{\partial I}{\partial y} dy + \frac{\partial I}{\partial t} dt + \epsilon$$

- If the motion of a point is very small, then the illumination function for that point should not change i.e. the brightness of a point in its new position at time $t + dt$ should not change
The Optical Flow Constraint

- We expect that in
  \[ I(x + dx, y + dy, t + dt) = I(x, y, t) + \frac{\partial I}{\partial x} dx + \frac{\partial I}{\partial y} dy + \frac{\partial I}{\partial t} dt \]

- The intensity does not change therefore
  \[ I(x + dx, y + dy, t + dt) = I(x, y, t) \]

- This means
  \[ \frac{\partial I}{\partial x} dx + \frac{\partial I}{\partial y} dy + \frac{\partial I}{\partial t} dt = 0 \]
  \[ I_x u + I_y v + I_t = 0 \]

- \(u\) and \(v\) are the \(x\) and \(y\) components of the velocity term that we are after.

The optical flow constraint equation is: \(I_x u + I_y v + I_t = 0\)
The Optical Flow Constraint

- We can write \( I_x u + I_y v + I_t = 0 \) as

\[
\begin{bmatrix}
I_x & I_y
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -I_t
\]

\[
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -\frac{I_t}{I_x^2 + I_y^2}
\begin{bmatrix}
I_x \\
I_y
\end{bmatrix}
\]

- So vector \((u, v)\) and \((I_x, I_y)\) are parallel.

- The velocity that we can estimate is in the direction of the intensity gradient.

- This confirms the aperture problem that we raised earlier.
The Optical Flow Constraint

- The optical flow constraint equation
  
  \[ ax + by + c = 0 \]
  
  - Is in the form of the equation of a line \( ax + by + c = 0 \)
  - Thus optical flow is constrained to be on a line.
  - There are two unknowns and one equation per pixel
  - Hence only one direction can be determined (aperture problem)
Techniques for Computing Optical Flow

- Differential techniques (dense flow – all pixels)
  - Based on spatial and temporal variations of the image brightness at all pixels

- Matching techniques (sparse flow – only some reliable pixels)
  - Feature matching
Differential Technique : Lucas-Kanade Motion

\[ I_x u + I_y v + I_t = 0 \]

- Need two or more pixels to solve for both variables
- Include pixels with different gradient directions to solve the aperture problem
Solving the Aperture Problem

One gradient has aperture problem

Two or more gradients can find the correct solution
Solving the Aperture Problem

\[ I_x u + I_y v + I_t = 0 \]

- Getting more equations by imposing additional constraints
- Optical flow is locally smooth i.e. pixels in a small neighborhood have a same displacement \((u, v)\)
- A (5,5) window gives us 25 equations
Lucas-Kanade Optical Flow

- We have more equations than unknowns

\[ A d = b \rightarrow \min|Ad - b|^2 \]

\[(A^T A)d = A^T b\]

\[
\begin{bmatrix}
\sum I_x^2 & \sum I_x I_y \\
\sum I_x I_y & \sum I_y^2
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -
\begin{bmatrix}
\sum I_x I_t \\
\sum I_y I_t
\end{bmatrix}
\]

- When is this solvable?
  - \(A^T A\) is invertible
  - The eigenvalues of \(A^T A\) are large
  - The ratio between the eigenvalues is not too large
Does $A^T A$ seem familiar?

\[
\begin{bmatrix}
\sum I_x I_x & \sum I_x I_y \\
\sum I_x I_y & \sum I_y I_y
\end{bmatrix}
\]

Harris corner detector
Harris corner detector: Revision

**Flat region:**
No change in all directions

**Edge:**
No change along the edge direction

**Corner:**
Change in all directions

Harris corner detector mathematically determines these three cases
Optical flow <-> Harris corner detector

\[ E(u, v) = \sum_{x,y} w(x, y) \left[ I(x+u, y+v) - I(x, y) \right]^2 \]

- **Window function**: Was spatial shift in single frame. Now is displacement across two frames in time.
- **Shifted intensity**: Previous frame
- **Intensity**: Current frame
Implications

- Corners are regions with two dominant gradient directions
  - Both $\lambda_1, \lambda_2$ are big

- Lucas-Kanade works best when $\lambda_1, \lambda_2$ are big

- Lucas-Kanade works best at corners

- No aperture problem at corners

- Corners are good features to compute optical flow
Dense Optical Flow

- Lucas-Kanade will only find optical flow at certain points (corners)

- Sometimes, we need to find optical flow at all pixels

- We will cover one algorithm for dense optical flow in this lecture
  - Farneback’s two frame optical flow
    - Implemented in OpenCV
Farneback’s two frame optical flow

- Based on polynomial expansion of a neighbourhood of pixels

- Approximate each pixel neighbourhood by a polynomial
  \[ f_1(x) = x^\top A_1 x + b_1^\top x + c_1 \]

- Construct a new signal \( f_2 \) by a global displacement \( d \)
  \[
  f_2(x) = f_1(x - d) = (x - d)^\top A_1 (x - d) + b_1^\top (x - d) + c_1 \\
  = x^\top A_1 x + (b_1 - 2A_1 d)^\top x + d^\top A_1 d - b_1^\top d + c_1 \\
  = x^\top A_2 x + b_2^\top x + c_2
  \]

- Thus
  \[
  A_2 = A_1 \\
  b_2 = b_1 - 2A_1 d,
  \]
Farneback’s Optical Flow

- The key observation is

\[ b_2 = b_1 - 2A_1 d \]

- We can calculate the value of the displacement i.e. the optical flow from this

\[ 2A_1 d = -(b_2 - b_1) \]

\[ d = -\frac{1}{2} A_1^{-1} (b_2 - b_1) \]

- If \( A \) is non-singular
Tracking

- Using optical flow, we can track pixels or corners over multiple frames

- Sometimes we don’t want to track every pixel or every corner

- We may want to track a specific object such as
  - A tennis ball
  - Pedestrians
  - Cars
  - Or simple blobs
Tracking

- Matching observations in a new frame to a set of tracked trajectories

How to determine which observations to add to which track?
Tracking Intuitions

1. Predict position along each track.
Tracking Intuitions

1. Predict position along each track.
2. Actual match should be close to the predicted position.

How to determine which observations to add to which track?
Tracking Intuitions

1. Predict position along each track.
2. Actual match should be close to the predicted position.
   - Some matches are highly unlikely.

How to determine which observations to add to which track?
Lucas-Kanade Tracking

- Instead of 5x5 window, we can choose a bigger window around the object and track it with Lucas-Kanade approach.

- Problem: The appearance of object can change over a few frames.

- We can update our template.

- Or introduce an affine warp into the equation.
Tracking with Mean-Shift

- For tracking objects whose appearance is defined by histograms
- Such as colour histograms, gradient histograms or any other histogram
- Basic idea is to shift the center towards the mean of the data in feature space
Mean Shift

- A tool for finding modes in a set of data samples manifesting an underlying Probability Density Function (PDF) in $R^N$

- PDF in feature space
  - Colour space
  - Scale space
  - Any feature space you can conceive
Mean Shift
Intuitive Description

Objective: Find the densest region
Intuitive Description

**Objective**: Find the densest region

- Region of interest
- Center of mass
- Mean Shift vector
Intuitive Description

Objective: Find the densest region
Intuitive Description

**Objective**: Find the densest region

- Region of interest
- Center of mass
- Mean Shift vector
Intuitive Description

**Objective**: Find the densest region
Intuitive Description

**Objective**: Find the densest region
Mean Shift and Colour Models

Two approaches

1. Create a colour “likelihood” image with pixels weighted by similarity to the desired colour (best for single coloured objects).

2. Represent colour distribution with a histogram. Use Mean Shift to find region with most similar distribution of colours.
Mean Shift on Weight Images

Compute likelihood maps where the pixel values are proportional to the likelihood that the pixel comes from the object we are tracking.

Computation of likelihood can be based on:

- Colour
- Texture
- Shape (e.g. boundary)
- Predicted location
- Their combination
Mean Shift on Weight Images

- The brightness of the pixel represents the “likelihood” that the pixel belongs to the object we are tracking.

- Perform standard mean shift on these pixels.

- Notice how the search window moves towards the center of the brightest pixels.
Example: CAMSHIFT (Continuously Adaptive Mean Shift)


- Likelihood is skin colour
- Location is found by mean shift
- Size and Roll found by fitting an ellipse to the points
- Implemented in OpenCV
Mean Shift and Colour Models

Two approaches

1. Create a colour “likelihood” image with pixels weighted by similarity to the desired colour (best for single coloured objects).

2. Represent colour distribution with a histogram. Use Mean Shift to find region with most similar distribution of colours.
Mean Shift Object Tracking: General Framework

Choose a reference model in the current frame → Choose a feature space → Represent the model in the chosen feature space
Mean Shift Object Tracking: General Framework

1. Start from the position of the model in the current frame.
2. Search in the model’s neighborhood in the next frame.
3. Find the best candidate by maximizing a similarity function.
4. Repeat the same process in the next pair of frames.
Mean Shift Object Tracking: PDF Representation

\[ \hat{q} = \{ q_{\mu} \}_{\mu=1}^{m} \quad \sum_{\mu=1}^{m} q_{\mu} = 1 \]

\[ \hat{p}(y) = \{ p_{\mu}(y) \}_{\mu=1}^{m} \quad \sum_{\mu=1}^{m} p_{\mu} = 1 \]

Similarity Function:

\[ f(y) = f(\hat{q}, \hat{p}(y)) \]
RANSAC: RANdom SAmple Consensus

- An iterative method to estimate the parameters of a mathematical model from a set of observed data containing outliers.

- It can be faster than the Hough Transform because it does not have to search through the entire solution space like Hough Transform.

Algorithm:

- For n iterations
- Define the model with random samples (usually minimum required)
- Find all data samples that agree with the model
- Save the model with the largest consensus (no of samples agreeing)
RANSAC: Line fitting example

- One of the lines (estimated with two random samples) will give the highest consensus.

- This line would be the best fit to the data. The remaining would be treated as outliers.
Demos

- Lucas-Kanade feature tracking
- CAMSHIFT face tracking
Summary

- Optic flow and motion field
- Tracking
- Optical Flow
- Lucas-Kanade algorithm
- Mean Shift algorithm

Acknowledgements: Material for this lecture was taken from Robert Collins Computer Vision course, and previous lectures delivered by Du Huynh and Peter Kovesi.