Introduction

So far we have considered estimating structure and motion from a set of \( n \) images \( \{I_1, \ldots, I_n\} \) in no particular order.

Here we analyse video data, intended as a sequence of images \( I(t), t \geq 0 \) captured by a moving camera.

Furthermore, we assume that the sampling rate is high enough that images change only slightly between a video frame to the next.

In particular, the goals of this lecture are to:

1. Study the properties of the projected motion;
2. Compute the optic flow as an approximation of the image motion;
3. Design trackers to track objects in videos.

Outline

Projected motion

Optical flow and the aperture problem

Lucas-Kanade tracking
Projected motion

A 3D point and its projection are function of time (due to the relative motion between camera and scene):

\[ X(t) = \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix}, \quad x(t) = \frac{f}{Z(t)} X(t) \] (1)

Differentiate \( x(t) \) w.r.t. time to get the projected motion velocity:

\[ \dot{x}(t) = \frac{dx(t)}{dt} = \frac{f}{Z(t)} \dot{X}(t) - \frac{f \dot{Z}(t)}{Z(t)^2} X(t) \]

We define linear \( V \) and angular \( \Omega \) velocities such that \( \dot{X} = V + \Omega \times X \). Hence writing \( \dot{Z} = X \cdot \dot{z} \) (where \( \dot{z} = [0 \ 0 \ 1]^T \)) and then using (1):

\[
\dot{x} = \frac{f}{Z}(V + \Omega \times X) - \frac{f (V + \Omega \times X) \cdot \dot{z}}{Z^2} X
\]

\[
= \frac{f}{Z} V + \Omega \times x - \frac{V \cdot \dot{z}}{Z} x - \frac{(\Omega \times x) \cdot \dot{z}}{f} x
\]

Note that:

Rotation

The depth \( Z \) does not appear with the rotation \( \Omega \):

The rotational component of the motion is independent of the scene depth.

Translation

The depth \( Z \) appears with the translation \( V \) as \( V/Z \):

There is a depth/speed scaling ambiguity.

This means that one cannot tell whether something is large, far away and moving quickly or small, near-to and moving slowly.

Example: a simple motion alarm

In this example, the camera is rotating around a fixed position – hence \( \Omega \neq 0 \) but \( V = 0 \). There is also an independently moving object in the scene.

Let \( \dot{x}(x, y) \) be the velocity of pixel \((x, y)\) (aka “motion field”).

The odometry from the camera axes provides \( \Omega \). This allows to compute the rotational component of the motion field (independent of the scene depth \( Z \)):

\[ \dot{x}_{\text{rotational}}(x, y) = \Omega \times x - \frac{(\Omega \times x) \cdot \dot{z}}{f} x, \quad \text{where} \quad x = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \]

The difference \( \dot{x}(x, y) - \dot{x}_{\text{rotational}}(x, y) \) reveals the moving object.

Approaching and receding motion

In this example, the camera is approaching or receding but not rotating – hence \( V_x = V_y = 0, V_z \neq 0 \) and \( \Omega = 0 \).

\[ \begin{align*}
\text{approaching} & \quad (V_z < 0) \\
\text{receding} & \quad (V_z > 0)
\end{align*} \]

Hence the projected motion is

\[ \dot{x}(x, y) = \frac{f}{Z(x, y)} V - \frac{V_z}{Z(x, y)} x \Rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \end{bmatrix} = -\frac{V_z}{Z(x, y)} \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} \]

where \( Z(x, y) \) is the depth of pixel \((x, y)\) (aka “depth map”). The motion field is a divergent/converged vector field, with a stationary point \( \dot{x}(0, 0) = 0 \) in the middle of the image.
Ego-motion and the focus of expansion

Generalising the previous example, if $V \neq 0$ but $\Omega = 0$, the motion field diverges/converges from a point, called focus of expansion.

To find the coordinates $(x, y)$ of the focus of expansion, set $\dot{x}(x, y) = 0$:

$$0 = \dot{x}(x, y) = \frac{f}{Z(x, y)} V - \frac{V_z}{Z(x, y)} x \Rightarrow \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} = \frac{1}{V_z} \begin{bmatrix} V_x \\ V_y \\ 0 \end{bmatrix}$$

Interpretation: The focus of expansion is found by treating the translational velocity $V$ as a ray, and finding where it hits the image plane.

Divergence and time to contact

Suppose again that $\Omega = 0$. The motion field is:

$$\dot{x}(x, y) = \frac{f}{Z(x, y)} V - \frac{V_z}{Z(x, y)} x \Rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \end{bmatrix} = \frac{1}{Z(x, y)} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

The divergence of the motion field $\dot{x}(x, y)$ is

$$\nabla \cdot \dot{x} = \frac{\partial}{\partial x} \left( \frac{f}{Z} V - \frac{V_z}{Z} x \right) + \frac{\partial}{\partial y} \left( \frac{f}{Z} V - \frac{V_z}{Z} x \right) = \frac{1}{Z^2} \left( \frac{\partial}{\partial x} (fV_x - xV_z) + \frac{\partial}{\partial y} (fV_y - yV_z) - \frac{2}{Z} V_z \right)$$

- If the camera is approaching/receding, $V_x = V_y = 0$ and at $x = y = 0$

$$\begin{bmatrix} \nabla \cdot \dot{x} \end{bmatrix}(0, 0) = -2 \frac{V_z}{Z(0, 0)} = - \frac{2}{t_c}$$

where $t_c$ is the time to contact.

- If the scene is a fronto-parallel plane $Z(x, y) = Z_{\text{const}}$, then $\frac{\partial Z}{\partial x} = \frac{\partial Z}{\partial y} = 0$, and so anywhere in the image

$$\forall (x, y) : (\nabla \cdot \dot{x})(x, y) = -2 \frac{V_z}{Z_{\text{const}}} = - \frac{2}{t_c}$$

Example: segmentation and motion alarm

Outline

Projected motion

Optical flow and the aperture problem

Lucas-Kanade tracking
Computing the visual motion

- We have considered some uses of projected motion, but not yet discussed how to obtain motion field from imagery.
  - Three ways:
    1. Token based
      - compute corners in successive frames
      - match corners using some proximity and/or similarity scores
    2. Gradient based
      - directly from brightness values
      - great if corner detector or matcher unreliable
    3. Phase based
      - We will look at (2) in some detail
      - For (3) see work of (eg) Heeger, Fleet and others

Observable visual motion and projected motion

- A word of warning:
  - Compare a rotating snooker ball and a stationary light source, with a stationary ball and a moving light source:
    - moves, so visual motion is non-zero.
    - Scene still, but highlight moves.
  - In general, observable motion \( \neq \) projected motion
  - But often its the best we can do (and a reasonable approximation...)

Gradient-based visual motion

- Regard the image as a function of space and time \( I(x, y, t) \).
  - Let \((x(t), y(t))\) be the coordinate of a particular image point over time.
  - Compute the derivative of the function \( I(x(t), y(t), t) \).
  - **Brightness constancy**: \( I(x(t), y(t), t) = \text{const.} \) Why?

Using the chain rule for the total derivative:

\[
\frac{dl(x(t), y(t), t)}{dt} = \frac{\partial l(x, y, t)}{\partial t} + \frac{\partial l(x, y, t)}{\partial x} \frac{dx(t)}{dt} + \frac{\partial l(x, y, t)}{\partial y} \frac{dy(t)}{dt}.
\]

Since \( l(x(t), y(t), t) \) is constant, the total derivative \( dl/dt = 0 \), so that

\[
0 = \frac{\partial l}{\partial t} + \frac{\partial l}{\partial x} \frac{dx}{dt} + \frac{\partial l}{\partial y} \frac{dy}{dt}.
\]

Hence if we write

\[
\nabla I = \left[ \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right], \quad \mu = \left[ \frac{dx}{dt}, \frac{dy}{dt} \right]
\]

we arrive at the **brightness constancy equation**

\[
\nabla I \cdot \mu = -\frac{\partial I}{\partial t}
\]

where \( \mu \) is called the **optic flow**.

Interpretation:

- The optic flow \( \mu(x, y) \) is the apparent velocity of pixel \((x, y)\).
- Ideally, the optical flow will match the projected motion velocity giving \( \mu(x, y) = \dot{x}(x, y) \).
The aperture problem

- The brightness constancy equation is in fact not sufficient to recover the optical flow $\mu$.
- Instead, the brightness constancy only gives us the component of $\mu$ along the direction of $\nabla I$.
- This edge-normal flow is

$$
v = \left( \frac{\mu \cdot \nabla l}{|\nabla l|^2} \right) \cdot \nabla l = - \left( \frac{\partial l}{\partial t} \right) \frac{\nabla l}{|\nabla l|^2}
$$

Thus the solution is to look at information not at a single point, but at a small residual (sum of squared differences):

This is the aperture problem: gradient information is too local, as if the image was seen through a small aperture.

Overcoming the aperture problem

Intuition tells us that there should be no aperture problem if there is a brightness gradient in “two directions”; i.e. at corners.

Thus the solution is to look at information not at a single point, but at a small patch containing a corner. The optical flow can then be recovered fully by tracking this corner.

We do so by comparing $I(x, y, t)$ to $I(x + u, y + v, t + \delta t)$ in a small patch $\Omega$ and find the relative translation $(u, v)$ as the minimiser of the comparison residual (sum of squared differences):

$$
E(u, v) \approx \sum_{(x, y) \in \Omega} [I(x + u, y + v, t + \delta t) - I(x, y, t)]^2
$$

$$
\approx \sum_{(x, y) \in \Omega} \left[ \frac{\partial I(x, y, t)}{\partial x} u + \frac{\partial I(x, y, t)}{\partial y} v + \frac{\partial I(x, y, t)}{\partial t} \right]^2
$$

Overcoming the aperture problem

$$
E(u, v) \approx \sum_{(x, y) \in \Omega} \left[ \frac{\partial I(x, y, t)}{\partial x} u + \frac{\partial I(x, y, t)}{\partial y} v + \frac{\partial I(x, y, t)}{\partial t} \right]^2
$$

Differentiating w.r.t. $u, v$ yields

$$
\frac{\partial E}{\partial u} = \sum_{(x, y) \in \Omega} \left[ \frac{\partial I(x, y, t)}{\partial x} u + \frac{\partial I(x, y, t)}{\partial y} v + \frac{\partial I(x, y, t)}{\partial t} \right] = 0
$$

$$
\frac{\partial E}{\partial v} = \sum_{(x, y) \in \Omega} \left[ \frac{\partial I(x, y, t)}{\partial x} u + \frac{\partial I(x, y, t)}{\partial y} v + \frac{\partial I(x, y, t)}{\partial t} \right] = 0
$$

Rewrite as

$$
\begin{bmatrix}
\frac{\partial I(x, y, t)}{\partial x} \\
\frac{\partial I(x, y, t)}{\partial y} \\
\frac{\partial I(x, y, t)}{\partial t}
\end{bmatrix}
\begin{bmatrix}
u \\
v \\
\delta t
\end{bmatrix}
\approx
\begin{bmatrix}
\frac{\partial I(x, y, t)}{\partial x} \\
\frac{\partial I(x, y, t)}{\partial y} \\
\frac{\partial I(x, y, t)}{\partial t}
\end{bmatrix}
\begin{bmatrix}
u \\
v \\
\delta t
\end{bmatrix}
$$

Computing the optic flow

$$
H \mu = - b
$$

$H_\Omega$ is the matrix describing the local autocorrelation. Thus, if the region $\Omega$ contains:

- A constant patch, then $H_\Omega$ has rank 0 and we cannot solve for the optic flow.
- An edge, then $H_\Omega$ has rank 1, and we can find the normal component of the optic flow.
- A corner, then $H_\Omega$ has full rank (i.e. 2), and we can solve the equations to find both components of the optic flow.
Computing the optic flow

Notes

▶ The first order approximation used only holds for very small values of \( \mu \), \( |\mu| < 1 \).
▶ To recover bigger flows, use idea of a Gaussian pyramid:
  ▶ Successively smooth and subsample images

Visual tracking

The aim of visual tracking is to locate the pose of a target in an image successively over an extended sequence of images.

Target: Requires a model of the “appearance” of the tracked object. This could be for example:
  ▶ An image patch or a colour histogram;
  ▶ A 3D CAD model;
  ▶ A learned discriminator function (see deep learning).

Pose: This describes the location of the target object, such as:
  ▶ The \((x, y)\) of the object center
  ▶ A 2D deformation representing translation, scale, rotation, etc;
  ▶ The pose \((R, T)\) of the underlying 3D object;
  ▶ The full articulation of a deformable 3D object.

Tracking as optimisation

We seek the image position \( p = \begin{bmatrix} t_x & t_y \end{bmatrix}^T \) which maximises the similarity to the template
▶ Exhaustive search possible but undesirable
▶ Better to do gradient ascent/descent
Suppose the current object pose (translation of region $\Omega$) is $(t_x, t_y)$. Given a new frame $I$, we want to find an update $(t_x + \delta t_x, t_y + \delta t_y)$ of the pose that minimises the sum of squared differences:

$$E(\delta t_x, \delta t_y) = \sum_{(x,y) \in \Omega} (I(x + t_x + \delta t_x, y + t_y + \delta t_y) - T(x, y))^2.$$