Image Motion Estimation I

Outline

1. Introduction to Motion
2. Why Estimate Motion?
3. Global vs. Local Motion
4. Block Motion Estimation
5. Optical Flow Estimation – Basics
6. Optical Flow Estimation – Horn
7. Optical Flow Estimation – Kanade
8. Conclusions
1. Introduction to Motion

Visual motion is very important
Uses of motion

- Estimating 3D structure
  - Far is slow vs. near is fast
- Segmenting objects based on motion cues
- Learning about and tracking objects/people
- Recognizing events and activities

We are tuned for seeing motion

- Even “impoverished” motion data can evoke a strong percept

What are we measuring?
Motion field and parallax

- \( \mathbf{X}(t) \) is a moving 3D point
- Velocity of scene point: \( \mathbf{V} = \frac{d\mathbf{X}}{dt} \)
- \( \mathbf{x}(t) = (x(t), y(t)) \) is the projection of \( \mathbf{X} \) in the image
- Apparent velocity \( \mathbf{v} \) in the image: given by components \( v_x = \frac{dx}{dt} \) and \( v_y = \frac{dy}{dt} \)
- These components are known as the \textit{motion field} of the image

---

**Video Images**

A pair of images from a video scene appears to be very similar:
The Motion Model - Illustration

\[ f(t) \]

\[ f(t-1) \]
The Motion Model - Illustration

The difference image

\[ f(t) \quad f(t-1) \quad |f(t)-f(t-1)| \]

The Motion Model - Description

For a pair of such images, (almost) every pixel in one image can be found in the other in a slightly different location.
The Motion Model - Equation

\[ f(x, y, t) = f(x - v_x(x, y, t), y - v_y(x, y, t), t - 1) \]

- Both \( v_x \) & \( v_y \) may be spatially varying.
- The above equation is called the Brightness Constancy Equation (BCE), implying that brightness is preserved.
- The BCE does not always hold (occlusion, boundaries, illumination changes).

1.5 The Motion estimation Problem

\[ v(x, y, t) = \begin{bmatrix} v_x(x, y, t) \\ v_y(x, y, t) \end{bmatrix} \]
2. Why Estimate Motion?

Application 1 – De-Noising

- Additive noise in video sequences can be removed by spatial & temporal filtering.
- A typical such filter applies a weighted average of the 3D neighborhood in order to compute the corrected pixel value.
- In applying temporal smoothing, the correspondence information is crucial for good performance.
Application 1 – De-Noising

Application 2 – Compression

• In compressing video images, if we have already compressed $f(t-1)$, we know much about $f(t)$.
• Typical approach – building a predicted image for $f(t)$, based on $f(t-1)$.
• Much better compression is achieved if we apply motion compensation.
Application 2 – Compression

Current Image  Predicted Image  Error Image

Previous image Prediction

MC previous image Prediction

Application 3 – Surveillance

• Motion estimation plays an important role in applications such as target tracking and movement/change detection in surveillance systems.

• If the motion is a result of both camera and the imaged objects’ motion, these tasks are far more complex, due to the need to decouple these motion fields.
Application 3 – Surveillance

[Image of a group of children with red boxes and a plus sign drawn over them]
Application 4 – Depth Estimation

- Given a pair of images and the correspondence between them, depth of the imaged objects can be recovered.
- Depth is important for mapping (Digital Elevation Model extraction), segmentation to background/foreground, objects recognition, etc.
Application 4 – Depth Estimation

Sensor

Focal Plane

Field of View

F

H

Point on the surface

Pixel in the Image
Application 4 – Depth Estimation

Second Camera with the same Height & slightly different position

Pixel in the first Image

Pixel in the second Image

D

D₁

D₂

H

h
Application 5 – Super-resolution

- Given a set of low quality images of the same object taken from slightly different locations, a better (super-resolution) image can be recovered.
- The recovery process requires the knowledge of the motion with sub-pixel accuracy.

Example: A set of low quality images
Application 5 – Super-resolution

Each of these images looks like this:

Most of the test data other than a couple of exceptions. The low-temperature solder used is investigated (or some other technology) for nonwetting of 40In40Sn. The microstructural coarse-grained cycling of 58Bi42Sb.

The recovery result:

Most of the test data other than a couple of exceptions. The low-temperature solder used is investigated (or some other technology) for nonwetting of 40In40Sn. The microstructural coarse-grained cycling of 58Bi42Sb.
Other Applications

- There are numerous other applications for motion in images.
- Every application comes with a different set of requirements regarding:
  - Motion representation,
  - Accuracy,
  - Complexity,
  - ....

3. Global vs. Local Motion
The Motion Estimation Problem

\[ f(x, y, t) = f(x - v_x(x, y, t), y - v_y(x, y, t), t-1) \]

Motion Estimation Module

Motion Representation

• The possibilities are:
  – Define a motion vector for every pixel – \( v(x, y, t) \).
  – Define a motion vector for every block of \( M \) by \( M \) pixels.
  – Assume global motion behavior and represent the motion by the parameters of the global mapping.
**Example**

- For an image of 1000 by 1000 pixels:
  - Direct representation requires 2,000,000 unknowns to be computed.
  - Using blocks of 10 by 10 pixels, we need 20,000 numbers.
  - If we assume that the motion consists of rotation and translation, we need 3 numbers only!
- Which is better? Depends on the application.

**Global Motion Assumption**

- In the general case, we assume the existence of the following relation:
  \[
  \begin{bmatrix}
  x \\
  y
  \end{bmatrix}_t = p\left(\begin{bmatrix}
  x \\
  y
  \end{bmatrix}_{t-1},\text{Parameters}\right)
  \]
- The function \(p\) is a simple relation that relates previous position to the current one.
- A set of parameters governs the specific type of motion.
Global Motion – Translation

\[
\begin{bmatrix}
x \\
y
\end{bmatrix}_t = \begin{bmatrix}
x \\
y
\end{bmatrix}_{t-1} + \begin{bmatrix}
a \\
b
\end{bmatrix}
\]

Global Motion – Similarity

\[
\begin{bmatrix}
x \\
y
\end{bmatrix}_t = \begin{bmatrix}
c & 0 & \cos(\theta) & -\sin(\theta) \\
0 & d & \sin(\theta) & \cos(\theta)
\end{bmatrix} \begin{bmatrix}
x \\
y
\end{bmatrix}_{t-1} + \begin{bmatrix}
a \\
b
\end{bmatrix}
\]

\begin{align*}
a &= 2.1 \\
b &= 4.7 \\
a &= 5 \\
b &= 5 \\
c &= 0.7 \\
d &= 1.5 \\
\theta &= 10^\circ
\end{align*}
Global Motion – Affine

\[
\begin{bmatrix}
x \\
y
\end{bmatrix}_t = \begin{bmatrix}
c & e \\
f & d
\end{bmatrix} \begin{bmatrix}
x \\
y
\end{bmatrix}_{t-1} + \begin{bmatrix}
a \\
b
\end{bmatrix}
\]

\( \begin{array}{c}
a = -45 \\
b = 15 \\
c = 1 \\
d = 1.3 \\
e = 0.7 \\
f = -0.1
\end{array} \)

Global Motion – Projective

\[
\begin{bmatrix}
x \\
y
\end{bmatrix}_t = \begin{bmatrix}
c & e \\
f & d
\end{bmatrix} \begin{bmatrix}
x \\
y
\end{bmatrix}_{t-1} + \begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix} + \begin{bmatrix}
1
\end{bmatrix}
\]

\( \begin{array}{c}
a = -45 \\
b = 15 \\
c = 1 \\
d = 1.3 \\
e = 0.7 \\
f = -0.1 \\
g = 0.01 \\
h = 0.0062
\end{array} \)
Global Motion Estimation

Parametric Model

Motion Estimation Module

Motion Parameters

4. Block Motion Estimation
**Why Blocks?**

- If we are to find the motion by assigning a vector per pixel then we have
  - Too many unknowns to be estimated,
  - Too much storage space is consumed,
  - Too much data to be transmitted (coding),
  - Ill posed problem – too many unknowns.
- Solution: Divide the image to blocks and assign a motion vector per each block.

**The Basics – Stage 1**

Divide the image \( f(x,y,t) \) into non-overlapping blocks of \( M \) by \( M \) pixels.

\[
B_{[m,n]}[x, y] = f\{M (m-1) + x, M (n-1) + y, t\}, \quad x, y \in [1, M]
\]
The Basics – Stage 2

For every block we perform a search process in the previous image $f(x,y,t-1)$.

For $M=3$:

$$B_{[2,2]}[x, y] = f \{3(2-1) + x, 3(2-1) + y, t\} = f \{3 + x, 3 + y, t\}$$

The Basics – Stage 3

The search in $f(t-1)$ is done by computing a matching factor to every possible displacement.

$$E[m, n] = \sum_{x=1}^{3} \sum_{y=1}^{3} \left[ B_{[2,2]}[x, y] - f \{3 + x + m, 3 + y + n, t - 1\} \right]^2$$
Choose the motion vector $[v_x, v_y]$ which results with the smallest error $E$.

$$[v_x, v_y] = \text{ArgMin}_{[m,n]} E[m,n]$$

**Basic Assumption**

- In the Block-Matching algorithm we assume that all the pixels in every block have the same motion vector.
- The above implies that the block is assumed to have global translational motion.
- We can assume more general parameteric model for the block motion, such as affine.
Complexity - Search Zone

Instead of computing $E$ for every possible position in $f(x,y,t-1)$, assume:

$$-D \leq v_x, v_y \leq +D$$

This way, $E$ is computed only $(2D+1)^2$ times.

Complexity – Error Comp.

Instead of defining the error $E$ as a Mean-Square-Error, define it as Mean-Absolute-Differences:

$$E[m,n] = \sum_{x=1}^{3} \sum_{y=1}^{3} |B_{[2,2]}[x,y] - f[3 + x + m, 3 + y + n, t - 1]|$$
Complexity – Thresholding

Start by computing $E[0,0]$, and if it is smaller than some threshold, assume no motion and do not continue:

$$E[0,0] = \sum_{x=1}^{3} \sum_{y=1}^{3} |B_{2,2}[x, y] - f\{3 + x, 3 + y, t - 1\}| \leq T$$

Complexity – Spiral Search

Start from the [0,0] and compare $E[u,v]$ (while computed) to the minimum so far. If bigger, no need to continue.

$$E[m,n] = \sum_{x=1}^{3} \sum_{y=1}^{3} |B_{2,2}[x, y] - f\{3 + x + m, 3 + y + n, t - 1\}|$$
Complexity – Sub-Optimal

1. Computing $E[u,v]$ at 9 locations around the [0,0].
2. Find the minimum point and compute 3 additional $E$ near it.
3. Search the new minimum point.

Example 1
Example 2

Example 3
Applications

- Block Motion Estimation (BME) is extensively used in video coding (MPEG1, MPEG2, MPEG4, H261, & H263).
- Typically blocks of 8 by 8 or 16 by 16 are used.
- In regular video coding, motion estimation consumes 50-80% of the CPU time.
- There are chips that perform BME.