Comments

- Algorithm-1 works only for small motion.
- If object moves faster, the brightness changes rapidly, 2x2 or 3x3 masks fail to estimate spatiotemporal derivatives.
- Pyramids can be used to compute large optical flow vectors.

Pyramids

- Very useful for representing images.
- Pyramid is built by using multiple copies of image.
- Each level in the pyramid is 1/4 of the size of previous level.
- The lowest level is of the highest resolution.
- The highest level is of the lowest resolution.
Gaussian Pyramids

\[ g_l(i, j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) g_{l-1}(2i+m,2j+n) \]

\[ g_l = REDUCE[g_{l-1}] \]
**Convolution**

\[
\text{Convolution}
\]

**Gaussian Pyramids**

\[
g_{l,n}(i, j) = \sum_{p=-2}^{2} \sum_{q=-2}^{2} w(p, q) g_{l,n-1}(\frac{i-p}{2}, \frac{j-q}{2})
\]

\[g_{l,n} = \text{EXPAND}[g_{l,n-1}]\]
Reduce (1D)

\[ g_l(i) = \sum_{m=-2}^{2} \hat{w}(m) g_{l-1}(2i+m) \]

\[ g_l(2) = \hat{w}(-2) g_{l-1}(4-2) + \hat{w}(-1) g_{l-1}(4-1) + \hat{w}(0) g_{l-1}(4) + \hat{w}(1) g_{l-1}(4+1) + \hat{w}(2) g_{l-1}(4+2) \]

\[ g_l(2) = \hat{w}(-2) g_{l-1}(2) + \hat{w}(-1) g_{l-1}(3) + \hat{w}(0) g_{l-1}(4) + \hat{w}(1) g_{l-1}(5) + \hat{w}(2) g_{l-1}(6) \]
Expand (1D)

\[ g_{l,n}(i) = \sum_{p=-2}^{2} \hat{w}(p) g_{l,n-1}(\frac{i-p}{2}) \]

\[ g_{l,n}(4) = \hat{w}(-2) g_{l,n-1}(\frac{4-2}{2}) + \hat{w}(-1) g_{l,n-1}(\frac{4-1}{2}) + \]
\[ \hat{w}(0) g_{l,n-1}(\frac{4}{2}) + \hat{w}(1) g_{l,n-1}(\frac{4+1}{1}) + \hat{w}(2) g_{l,n-1}(\frac{4+2}{2}) \]

\[ g_{l,n}(4) = \hat{w}(-2) g_{l,n-1}(1) + \hat{w}(0) g_{l,n-1}(2) + \hat{w}(2) g_{l,n-1}(3) \]

Expand

**Gaussian Pyramid**

\[ g_{l,1} = \text{EXPAND}[g_{1} \ ] \]
Convolution Mask

\[ [w(-2), w(-1), w(0), w(1), w(2)] \]

Convolution Mask

- Separable

\[ w(m, n) = \hat{w}(m)\hat{w}(n) \]

- Symmetric

\[ \hat{w}(i) = \hat{w}(-i) \]

\[ [c, b, a, b, c] \]
Convolution Mask

• The sum of mask should be 1.
  \[a + 2b + 2c = 1\]

• All nodes at a given level must contribute the same total weight to the nodes at the next higher level.
  \[a + 2c = 2b\]
Convolution Mask

\[ \hat{w}(0) = a \]
\[ \hat{w}(-1) = \hat{w}(1) = \frac{1}{4} \]
\[ \hat{w}(-2) = \hat{w}(2) = \frac{1}{4} - \frac{a}{2} \]

a=.4 GAUSSIAN, a=.5 TRINGULAR

Approximate Gaussian
Triangular

Gaussian

\[ \text{Triangular} \]

\[ g(x) \]

\[ x \]

<table>
<thead>
<tr>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>.011</td>
<td>.13</td>
<td>.6</td>
<td>1</td>
<td>.6</td>
<td>.13</td>
<td>.011</td>
</tr>
</tbody>
</table>
Gaussian

\[ g(x) = e^{-\frac{x^2}{2\sigma^2}} \]
Algorithm

- Apply 1-D mask to alternate pixels along each row of image.
- Apply 1-D mask to each pixel along alternate columns of resultant image from previous step.
Gaussian Pyramid

Laplacian Pyramids

- Similar to edge detected images.
- Most pixels are zero.
- Can be used for image compression.

\[ L_1 = g_1 - \text{EXPAND}[g_2] \]

\[ L_2 = g_2 - \text{EXPAND}[g_3] \]

\[ L_3 = g_3 - \text{EXPAND}[g_4] \]
Coding using Laplacian Pyramid

• Compute Gaussian pyramid

\[ g_1, g_2, g_3, g_4 \]

• Compute Laplacian pyramid

\[
\begin{align*}
L_1 &= g_1 - \text{EXPAND}[g_2] \\
L_2 &= g_2 - \text{EXPAND}[g_3] \\
L_3 &= g_3 - \text{EXPAND}[g_4] \\
L_4 &= g_4
\end{align*}
\]

• Code Laplacian pyramid
Decoding using Laplacian pyramid

- Decode Laplacian pyramid.
- Compute Gaussian pyramid from Laplacian pyramid.

\[
g_3 = \text{EXPAND}[g_4] + L_3
\]
\[
g_2 = \text{EXPAND}[g_3] + L_2
\]
\[
g_1 = \text{EXPAND}_1[g_2] + L_1
\]

- is reconstructed image.

Gaussian

- Most natural phenomenon can be modeled by Gaussian.
- Take a bunch of random variables of any distribution, find the mean, the mean will approach to Gaussian distribution.
- Gaussian is very smooth function, it has infinite no of derivatives.
Gaussian

- Fourier Transform of Gaussian is Gaussian.
- If you convolve Gaussian with itself, it is again Gaussian.
- There are cells in human brain which perform Gaussian filtering.
  - Laplacian of Gaussian edge detector

Carl F. Gauss

- Born to a peasant family in a small town in Germany.
- Learned counting before he could talk.
- Contributed to Physics, Mathematics, Astronomy,…
- Discovered most methods in modern mathematics, when he was a teenager.
Carl F. Gauss

• Some contributions
  – Gaussian elimination for solving linear systems
  – Gauss-Seidel method for solving sparse systems
  – Gaussian curvature
  – Gaussian quadrature

Separability
Algorithm

- Apply 1-D mask to alternate pixels along each row of image.
- Apply 1-D mask to each pixel along alternate columns of resultant image from previous step.

Gaussian Pyramid
Laplacian Pyramids

- Similar to edge detected images.
- Most pixels are zero.
- Can be used for image compression.
Coding using Laplacian Pyramid

• Compute Gaussian pyramid

\[ g_1, g_2, g_3, g_4 \]

• Compute Laplacian pyramid

\[ L_1 = g_1 - \text{EXPAND}[g_2] \]
\[ L_2 = g_2 - \text{EXPAND}[g_3] \]
\[ L_3 = g_3 - \text{EXPAND}[g_4] \]
\[ L_4 = g_4 \]

• Code Laplacian pyramid

Decoding using Laplacian pyramid

• Decode Laplacian pyramid.
• Compute Gaussian pyramid from Laplacian pyramid.

\[ g_3 = \text{EXPAND}[g_4] + L_3 \]
\[ g_2 = \text{EXPAND}[g_3] + L_2 \]
\[ g_1 = \text{EXPAND}[g_2] + L_4 \]

• is reconstructed image.
Image Compression (Entropy)

7.6

4.4

5.0

5.6

6.2

Huffman Coding (Example-1)

A_1

A_2

A_3

A_4

P=.5

P=.25

P=.125

P=.125

0

0

0

1

1

1

1

A_1 0

A_2 10

A_3 110

A_4 111
Huffman Coding

Entropy  \[ H = -\sum_{i=0}^{255} p(i) \log_2 p(i) \]

\[ H = -0.5 \log .5 - 0.25 \log .25 - 0.125 \log .125 - 0.125 \log .125 = 1.75 \]

Image Compression

1.58

1

.73

(a)   (b)
Combining Apple & Orange

Combining Apple & Orange
Algorithm

- Generate Laplacian pyramid $Lo$ of orange image.
- Generate Laplacian pyramid $La$ of apple image.
- Generate Laplacian pyramid $Lc$ by copying left half of nodes at each level from apple and right half of nodes from orange pyramids.
- Reconstruct combined image from $Lc$.


• ftp://csd.uwo.ca/pub/vision
  Performance of optical flow techniques,
  Barron, Fleet and Beauchemin

Algorithm-2 (Optical Flow)

• Create Gaussian pyramid of both frames.
• Repeat
  – apply algorithm-1 at the current level of pyramid.
  – propagate flow by using bilinear interpolation to the next level, where it is used as an initial estimate.
  – Go back to step 2
Horn & Schunck Method

- Good only for translation model.
- Oversmoothing of boundaries.
- Does not work well for real sequences.

Other Optical Flow Methods
Important Issues

- What motion model?
- What function to be minimized?
- What minimization method?

Minimization Methods

- Least Squares fit
- Weighted Least Squares fit
- Newton-Raphson
- Gradient Descent
- Levenberg-Marquadt
Lucas & Kanade (Least Squares)

- Optical flow eq

\[
fx'u + fy'v = -ft
\]

- Consider 3 by 3 window

\[
\begin{bmatrix}
fx' & fy' \\
\vdots & \vdots \\
fx_9 & fy_9 \\
\end{bmatrix}
= \begin{bmatrix}
-f_{t1} \\
\vdots \\
-f_{t9}
\end{bmatrix}
\]

Lucas & Kanade

\[
Au = ft
\]

\[
A^TAu = A^Tft
\]

\[
u = (A^TA)^{-1}A^Tft
\]

\[
\min \sum_{i=-2}^{2} \sum_{j=-2}^{2} (fx'u + fy'v + f_{ji})^2
\]
Lucas & Kanade

$$\begin{align*}
\min & \sum_{i=-2}^{2} \sum_{j=-2}^{2} (f_{xi} u + f_{yi} v + f_{ii})^2 \\
\implies & \sum (f_{xi} u + f_{yi} v + f_{ii}) f_{xi} = 0 \\
& \sum (f_{xi} u + f_{yi} v + f_{ii}) f_{yi} = 0
\end{align*}$$

Lucas & Kanade

$$\begin{bmatrix}
u \\
u
\end{bmatrix} = \begin{bmatrix}
\sum f_{xi}^2 & \sum f_{xi} f_{yi} \\
\sum f_{xi} f_{yi} & \sum f_{yi}^2
\end{bmatrix}^{-1} \begin{bmatrix}
-\sum f_{xi} f_{ti} \\
-\sum f_{yi} f_{ti}
\end{bmatrix}$$
Lucas & Kanade

\[
\min \sum_{i=-2}^{2} \sum_{j=-2}^{2} w_i (f_{x_i} u + f_{y_i} v + f_{t_i})^2
\]

\[
W A u = W f_t
\]

\[
A^T W A u = A^T W f_t
\]

\[
u = (A^T W A)^{-1} A^T W f_t
\]