Pyramids

Lecture-7
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.
Pyramid

Level 1: 1x1
Level 2: 2x2
Level 3: 4x4
Level 4: 8x8
Level 10: 512x512
Gaussian Pyramids (reduce)

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

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

\[ h(x, y) = f(x+1, y+1)g(-1, 1) + f(x+1, y+1)g(0, 1) + f(x+1, y+1)g(1, 1) + f(x+1, y)g(-1, 0) + f(x, y+1)g(0, 0) + f(x+1, y+1)g(1, 0) + f(x+1, y-1)g(-1, 0) + f(x, y+1)g(0, -1) + f(x+1, y+1)g(1, -1) \]
Reduce (1D)

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

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

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

Gaussian Pyramid

\[ g_0 = \text{IMAGE} \]

\[ g_1 = \text{REDUCE}[g_{L-1}] \]
Gaussian Pyramids (expand)

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

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

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

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

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

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

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

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

\[ g_{1,1} = \text{EXPAND}[g_1] \]
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

\[ a + 2c = 2b \]
Convolution Mask

$$\hat{w}(0) = a$$

$$\hat{w}(-1) = \hat{w}(1) = \frac{1}{4}$$

$$\hat{w}(-2) = \hat{w}(2) = \frac{1}{4} - \frac{a}{2}$$

$$a=.4 \text{ GAUSSIAN, } a=.5 \text{ TRINGULAR}$$
Triangular

The graph shows a Triangular distribution with bars representing the probability density function (PDF) of the distribution. The x-axis represents distinct points labeled as 'c', 'b', 'a', 'b', and 'c'. The y-axis shows the probability density values ranging from 0 to 0.5. The bars indicate the height of the probability density at each point, with 'a' having the highest density.
Approximate Gaussian

[Bar chart with data points labeled c, b, a, b, c and corresponding Gaussian distribution values]
Gaussian

\[ \hat{w}(0) = a \]

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

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

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

<table>
<thead>
<tr>
<th>x</th>
<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>g(x)</td>
<td>.011</td>
<td>.13</td>
<td>.6</td>
<td>1</td>
<td>.6</td>
<td>.13</td>
<td>.011</td>
</tr>
</tbody>
</table>
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.

\[ L_1 = g_1 - \text{EXPAND}[g_2] \]
\[ L_2 = g_2 - \text{EXPAND}[g_3] \]
\[ L_3 = g_3 - \text{EXPAND}[g_4] \]
Fig. 5: First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.
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.

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

- \( g_1 \) is reconstructed image.
Laplacian Pyramid

Fig 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.
Image Compression (Entropy)

7.6

4.4

5.0

5.6

6.2

0.77

1.9

3.3

4.2
Image Compression

(a)  1.58

(b)  .73

(c)  1.0

(d)  .73
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.
• http://ww-bcs.mit.edu/people/adelson/papers.html
Lucas Kanade with Pyramids

• Compute ‘simple’ LK optical flow at highest level
• At level $i$
  • Take flow $u_{i-1}$, $v_{i-1}$ from level $i\!-\!1$
  • bilinear interpolate it to create $u_i^*$, $v_i^*$ matrices of twice resolution for level $i$
  • multiply $u_i^*$, $v_i^*$ by 2
  • compute $f_t$ from a block displaced by $u_i^*(x,y)$, $v_i^*(x,y)$
  • Apply LK to get $u_i'(x,y)$, $v_i'(x,y)$ (the correction in flow)
• Add corrections $u_i'$, $v_i'$, i.e. $u_i = u_i^* + u_i'$, $v_i = v_i^* + v_i'$. 
Pyramids

\[ u_i = u_i^* + u_i', \quad v_i = v_i^* + v_i' \]

\[ f_1 \text{ pyramid} \quad f_2 \text{ pyramid} \]
Interpolation

\[
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
0 & \bullet & \bullet & \bullet & \bullet & \bullet \\
u = 1 & \bullet & \bullet & \bullet & \bullet & \bullet \\
2 & \bullet & \bullet & \bullet & \bullet & \bullet \\
3 & \bullet & \bullet & \bullet & \bullet & \bullet \\
\end{array}
\]

\[
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
0 & \bullet & \bullet & \bullet & \bullet & \bullet \\
v = 1 & \bullet & \bullet & \bullet & \bullet & \bullet \\
2 & \bullet & \bullet & \bullet & \bullet & \bullet \\
3 & \bullet & \bullet & \bullet & \bullet & \bullet \\
\end{array}
\]

\[
\begin{array}{cccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
u^* = 3 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
4 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
5 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
6 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
7 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\end{array}
\]

\[
\begin{array}{cccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
0 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
v^* = 3 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
4 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
5 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
6 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
7 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\end{array}
\]
1-D Interpolation

\[ y = mx + c \]

\[ f(x) = mx + c \]
Lucas-Kanade without pyramids

Fails in areas of large motion
Lucas-Kanade with Pyramids