**38. Template Matching**

- Cross correlation

**Metric - Candidate metric:**

1. \( M_1 : p^* = \sum_{i=1}^{N} (P_o - W_i) \), \( N = \# pixels \) in window

2. \( M_2 : p^* = \sum_{i=1}^{N} \frac{(P_o - W_i)^2}{N} \)

3. \( M_3 : p^* = \frac{N}{\sum_{i=1}^{N} W_i P_i} \) (Unnormalized Correlation)

4. \( M_4 : p^* = \frac{N}{\sum_{i=1}^{N} W_i P_i} \) (Normalized Correlation)

\( \sqrt{\sum_{i=1}^{N} W_i} \) constant

5. \( M_5 : p^* = \frac{\sum_{i=1}^{N} (W_i P_i)}{\sqrt{\sum_{i=1}^{N} W_i^2 \sigma^2_i}} \) "Normalized" - Detection of peaks may be difficult, also noise may mask peak correlation.

\( M_6 \) overcomes these problems

\( M_6 : p^* = \frac{N}{\sum_{i=1}^{N} (W_i - \bar{W})(P_o - \bar{P})} \) "Statistical Correlation"

\( \sigma_x^2 = \frac{N}{\sum_{i=1}^{N} (W_i - \bar{W})^2} \)

\( \sigma_y^2 = \frac{N}{\sum_{i=1}^{N} (P_i - \bar{P})^2} \)

\( \sigma_{xy} = \frac{N}{\sum_{i=1}^{N} (W_i - \bar{W})(P_i - \bar{P})} \)

\( p^* \) are called Correlation Coefficients

\( M_1 \) to \( M_5 \) are more effective if image is normalized to \( M_6 \)

AVERAGE = PICTURE BRIGHTNESS (Global/local)

**Note:**

- To avoid squaring, use \( p^* \) instead of \( p^2 \)

- \( \bar{W} \) and \( \bar{P} \) are average of pixels of image under window

- \( p^* \) is normalized, \( -1 < p^* < 1 \)
FIGURE 12.9
(a) Image.
(b) Subimage.
(c) Correlation coefficient of (a) and (b). Note that the highest (brighter) point in (c) occurs when subimage (b) is coincident with the letter “D” in (a).
3. 25 pts. Find a correlation coefficient for each of the four mask positions relative to the image area of interest.

\[
P = 0.35 \quad \text{and} \quad \bar{y} = 0.06
\]

\[
\text{Object}\quad 0\quad 0\quad 0\quad 0
\]

\[
\text{Image}\quad 0\quad 1\quad 0\quad 1\quad 0\quad 0\quad 0\quad 0
\]

\[
\text{Mask}\quad 0\quad 1\quad 0\quad 1\quad 0\quad 0\quad 0\quad 0
\]

\[
\text{Object of Interest}\quad 1\quad 2\quad 3\quad 4\quad 5\quad 6\quad 7\quad 8\quad 9
\]

\[
\delta = \frac{4}{9}
\]

\[
\Sigma = \frac{w - \bar{w}}{w - \bar{w}} \cdot \sum (f - \bar{f})^2
\]

\[
M_b = \frac{\sum (w - \bar{w}) (f - \bar{f})}{\sum (w - \bar{w})^2}
\]

\[
R = \frac{\sum (w - \bar{w}) (f - \bar{f})}{\sqrt{\sum (w - \bar{w})^2 \cdot \sum (f - \bar{f})^2}}
\]

\[
R_a = \frac{1.728}{(2.222)^{\frac{1}{2}}} = 0.80
\]

\[
R_b = \frac{0.776}{(2.222)^{\frac{1}{2}}} = 0.35
\]

\[
R_c = \frac{0.111}{(1.556)^{\frac{1}{2}}} = 0.11
\]
If the image $I(k, j)$ has only two gray levels, it is a binary, or black-and-white, image. For binary images, an alternative technique called corner-point encoding might be used to extract the vertex pixels directly. Each of the interior pixels has eight adjacent pixels called its neighbors. We determine whether a given pixel is a corner point, or vertex, by examining the intensity pattern of its neighbors. This can be done by scanning over the image with a set of $3 \times 3$ corner-point templates, or masks, which represent all possible types of corners in objects that are at least 2 pixels wide. If $I(k, j)$ is a binary image with 0s representing background pixels and 1s representing foreground pixels, the family of eight corner-point templates shown in Fig. 8-8 can be used.

![Corner-point templates](image_url)

Figure 8-8 Corner-point templates.

Note that the set of corner-point templates is generated by taking the corner pattern appearing in the upper right portion of the first template and rotating it counterclockwise by multiples of $\pi/4$ to generate the remaining seven templates. To search for corner points, we can scan the image with the templates using the normalized cross-correlation function in $\mathcal{NCC}$. Note from Fig. 8-8 that the center pixel of each template is a foreground pixel. Thus the search for corner points can be made considerably more efficient if we first scan the $(m - 2)(n - 2)$ pixels in the interior of $I(k, j)$ to find candidate pixels, pixels with a value of 1. Once a candidate pixel is located, its eight neighbors can then be examined by cross-correlating with the $3 \times 3$ templates. If the normalized cross correlation evaluates to 1 for some $i$, where $1 \leq i \leq 8$, then the process can be terminated, since a corner-point pixel has been identified.

**Exercise 8-3-3: Holes in Parts.** Suppose an algorithm is available for finding the corner points of a part. Indicate how this same algorithm could be used, without modification, to find the corner points of a hole in the part. Hint: The complement of a binary image is a binary image.

**Exercise 8-3-4: Whiskers.** Suppose we want to locate the end pixels of line segments or curves in the foreground that are only 1 pixel wide. Sketch a set of templates that could be used to identify the end points of these whiskers.

Once the edge pixels have been determined, they can be fitted with straight lines. The intersections of the straight lines then determine a set of $M$ vertices. Since the object is assumed to be a polyhedron, it is completely specified by its vertices. Note that in this case we have taken the original information contained in the $mn$ pixels and reduced it to $2M$ integer coordinates which specify the locations of the $M$ vertices of the polyhedral object.
**Binary Images - Correlation**

\[
R^* = \frac{\sum_{i=1}^{N} (P_i - w_i) \oplus \text{XOR}}{\sum_{i=1}^{N} (P_i \oplus w_i)}
\]

- \(R^*\) small \(\Rightarrow\) high correlation
- \(R^*\) large \(\Rightarrow\) low correlation

**Matching Algorithm Which Is Invariant To**

- Position, Scale, and Orientation
- Invariant to Translation, Rotation, and Scale


\(f \rightarrow F(f)\) Invariant to Trans

**Mellin Transform** (Correlation in Frequency Domain)

\[
\begin{align*}
F(xy) & \rightarrow \text{FFT} \\
|F(u,v)| & \rightarrow \text{ABS} \\
R & \rightarrow \text{Rectangular to Polar} \\
F(\rho \cdot e^{j\theta}) & \rightarrow \text{BO SCALE} \\
M(u,v) & \rightarrow \text{FFT} \\
\end{align*}
\]

\[y \xrightarrow{\rho \theta} x \xrightarrow{F(xy)} \]

\[x \xrightarrow{\rho \theta} y \xrightarrow{F(xy)} \]

\[M(u,v) \rightarrow \text{Match Correlation} \]

\[\text{warp}^R\]