Template Matching

Outline
- Template matching problem
- Dissimilarity measures:
  - Sum of squared differences (SSD)
  - Sum of absolute differences (SAD)
- Similarity measures:
  - Cross-correlation (CC)
  - Normalized cross-correlation (NCC)
  - Zero-mean normalized cross-correlation (ZNCC)

Template Matching

Objective:
- Search for a given feature within the source image

Application:
- Feature detection
- Tracking (motion estimation)

Template Matching Algorithm

- Create a small template image $T$ for the feature
- For each pixel $(p,q)$ in source image $F$:
  - Center the template image $T$ over $(p,q)$
  - Calculate similarity/dissimilarity between $T$ and part of $F$
  - Keep the pixel that gives the largest similarity or smallest dissimilarity

Dissimilarity vs. Similarity

- Dissimilarity measures:
  - Calculate the differences between two input images
  - Measuring result referred as cost or energy
  - Lower cost $\rightarrow$ better match

- Similarity measures:
  - Calculate the similarities between two input images
  - Measuring result referred as score
  - Higher score $\rightarrow$ better match

Sum of Squared Differences

- Consider a 2D array of intensities as a multi-dimensional vector
- Measure the square Euclidean distance between two multi-dimensional vectors, $F$ & $T$
- Output range:
  - $0 \leq SSD \leq \#_{\text{pixels}} \times \text{gray\_level} \times \text{gray\_level}$

$$SSD(F, T) = \sum_{u\in T} \sum_{v\in F} (F[p+u, q+v] - T[u, v])^2$$
SSD Example

\[
\begin{align*}
8 & 6 2 3 1 
1 & 2 1 6 4 \cdots [1 2 1] \quad - & 94 & 86 & 11 \cdots \\
2 & 5 3 4 1 \cdots & 2 5 3 & = & 0 & 103 & 73 \cdots \\
1 & 3 0 7 5 \cdots & 1 3 0 & = & - & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{align*}
\]

- \( 49 + 16 + 1 + 9 + 4 + 1 + 4 + 9 = 94 \)
- \( 25 + 0 + 4 + 0 + 16 + 9 + 16 + 0 + 16 = 86 \)
- \( 1 + 1 + 0 + 1 + 1 + 4 + 1 + 1 = 11 \)
- \( 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 0 \) \(- \) lowest cost
- \( 1 + 1 + 25 + 9 + 4 + 1 + 4 + 9 + 49 = 103 \)
- \( 0 + 16 + 9 + 1 + 1 + 4 + 1 + 16 + 25 = 73 \)

Wednesday, October 4, 2017 Minglun Gong 7

Cross-Correlation

- Measure the dot product between two multi-dimensional vectors, \( F \) & \( T \)
- Output range:
  - \( 0 \leq CC \leq \#_\text{pixels} \times \text{gray\_level} \)

\[
CC(F, T)_{(p,q)} = \sum \sum F[p+u,q+v] \cdot T[u,v]
\]

Wednesday, October 4, 2017 Minglun Gong 9

The Problem of CC

\[
SSD(F, T)_{(p,q)} = \sum \sum (F[p+u,q+v] - T[u,v])^2
\]
\[
= \sum \sum (F[p+u,q+v] + T[u,v] - 2F[p+u,q+v] \cdot T[u,v])
\]
\[
= |F| + |T| - 2CC(F, T)_{(p,q)}
\]

- The value of CC is influenced by:
  - The value of SSD
  - \(|T|\), which is constant since the template image is fixed
  - \(|F|\), whose value depends on the location
  - Brighter areas in the image tend to have higher score

Wednesday, October 4, 2017 Minglun Gong 11

Sum of Absolute Differences

- Measure the sum of the absolute differences between corresponding pixels
- Output range:
  - \( 0 \leq SAD \leq \#_\text{pixels} \times \text{gray\_level} \)

\[
SAD(F,T)_{(p,q)} = \sum \sum |F[p+u,q+v] - T[u,v]|
\]

Wednesday, October 4, 2017 Minglun Gong 8

CC Example

\[
\begin{align*}
8 & 6 2 3 1 
1 & 2 1 6 4 \cdots [1 2 1] \quad - & 54 & 54 & 70 \cdots \\
2 & 5 3 4 1 \cdots & 2 5 3 & = & 54 & 50 & 67 \cdots \\
1 & 3 0 7 5 \cdots & 1 3 0 & = & - & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{align*}
\]

- \( 8 + 12 + 2 + 10 + 3 + 2 + 15 + 0 = 54 \)
- \( 6 + 4 + 3 + 4 + 18 + 5 + 9 + 0 = 54 \)
- \( 2 + 6 + 1 + 2 + 30 + 12 + 3 + 12 + 0 = 70 \) \(- \) highest
- \( 1 + 4 + 1 + 4 + 25 + 9 + 1 + 9 + 0 = 54 \)
- \( 2 + 2 + 6 + 10 + 15 + 12 + 3 + 0 + 0 = 50 \)
- \( 1 + 12 + 4 + 6 + 20 + 3 + 0 + 21 + 0 = 67 \)

Wednesday, October 4, 2017 Minglun Gong 10

Illumination Changes

- All measures above assume that the source image and the template image have similar brightness
- They do not work well when the illumination changes
- Need to normalize the pixels' intensities so that illumination changes can be handled

Wednesday, October 4, 2017 Minglun Gong 12
**Normalized CC**

- Measure the dot product between two normalized vectors
- Output range: 0 ≤ NCC ≤ 1

$$NCC(F,T)_{(\alpha)} = CC \left( \begin{bmatrix} F \\ F \cdot \alpha \end{bmatrix}, \begin{bmatrix} T \\ T \cdot \alpha \end{bmatrix} \right) = \sum \sum (F[p+u,q+v] \cdot T[u,v]) \sqrt{\sum \sum F[p+u,q+v]^2 \cdot \sum \sum T[u,v]^2}$$

**Zero-mean Normalized CC**

- Measure the dot product between two zero-mean adjusted vectors
- Output range: -1 ≤ ZNCC ≤ 1

$$ZNCC(F,T)_{(\alpha)} = NCC(F - F_{\alpha},T - T_{\alpha}) \sum \sum (F[p+u,q+v] - F_{\alpha}) \cdot (T[u,v] - T_{\alpha}) \sqrt{\sum \sum (F[p+u,q+v] - F_{\alpha})^2 \cdot \sum \sum (T[u,v] - T_{\alpha})^2} \frac{\text{cov}_{uv}(F,T)}{\sqrt{\text{var}_{uv}(F) \cdot \sqrt{\text{var}(T)}}}$$

**Naïve Variance Calculation**

- Require two passes:
  - First pass computes mean
  - Second pass computes variance

```java
float sum = 0;
for (int v=-h; v<=h; v++)
    for (int u=-w; u<=w; u++)
        sum += T[u][v];
float mean = sum / ((2 * w + 1) * (2 * h + 1));
float var = 0;
for (int v=-h; v<=h; v++)
    for (int u=-w; u<=w; u++)
        var += (T[u][v] - mean) * (T[u][v] - mean);
```

**Updated Variance Calculation**

- Involve only one pass
  - Faster, but same complexity

```java
float sum = 0, square_sum = 0;
for (int v=-h; v<=h; v++)
    for (int u=-w; u<=w; u++)
    {
        square_sum += T[u][v] * T[u][v];
        sum += T[u][v];
    }
float mean = sum / ((2 * w + 1) * (2 * h + 1));
float var = square_sum - sum * mean;
```

**Some Derivations**

$$\text{var}(F) = \sum \sum (T[u,v] - \bar{T})^2 = \sum \sum T[u,v]^2 - 2 T[u,v] \bar{T} + \bar{T}^2 = \sum \sum T[u,v]^2 - N \times \bar{T}^2 = \sum \sum T[u,v]^2 - N \times T^2 \sum \sum (F[p+u,q+v] - F_{\alpha}) \cdot (T[u,v] - T_{\alpha}) \sum \sum (F[p+u,q+v] - F_{\alpha})^2 \cdot \sum \sum (T[u,v] - T_{\alpha})^2 \text{cov}_{uv}(F,T) = \sum \sum F[p+u,q+v] \cdot T[u,v] - N \times F_{\alpha} \cdot \bar{T}$$

**ZNCC Calculation**

- Location independent calculation is same as last slide
- Location dependent calculation is shown on the left