**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 or dissimilarity between $T$ & local patches in $F$
  - Keep the pixels that give the high enough similarity or low enough dissimilarity

**Dissimilarity vs. Similarity**

- Dissimilarity measures:
  - Calculate the differences between 2 input image patches
  - Measuring result referred as matching, costs, matching error, or energy
  - Lower cost $\rightarrow$ better match

- Similarity measures:
  - Calculate the similarities between 2 input image patches
  - Measuring result referred as matching scores
  - Higher score $\rightarrow$ better match

**Sum of Squared Differences**

- Consider intensities in a 2D patch as a multi-dimensional vector
- Measure the square Euclidean distance between 2 vectors
- $SSD(F, T)(p, q) = \sum_{u, q} (F[p + u, q + v] - T[u, v])^2$
- Output range:
  - $0 \leq SSD \leq NL^2$
  - $N$: number of pixels
  - $L$: gray scale level

---

Tuesday, March 27, 2018  Minglun Gong  2

Tuesday, March 27, 2018  Minglun Gong  3

Tuesday, March 27, 2018  Minglun Gong  4

Tuesday, March 27, 2018  Minglun Gong  5

Tuesday, March 27, 2018  Minglun Gong  6
**SSD Example**
- SSD between a 2D array & a template
  \[
  \begin{bmatrix}
  8 & 6 & 2 & 3 & 1 \\
  1 & 2 & 1 & 6 & 4 \\
  2 & 5 & 3 & 4 & 1 \\
  1 & 3 & 0 & 5 & 5 \\
  \end{bmatrix}
  \quad \oplus \quad
  \begin{bmatrix}
  1 & 2 & 1 \\
  2 & 5 & 3 \\
  1 & 3 & 0 \\
  \end{bmatrix}
  \Rightarrow
  \begin{bmatrix}
  94 & 86 & 11 & \cdots \\
  0 & 79 & 61 & \cdots \\
  \cdots & \cdots & \cdots \\
  \end{bmatrix}
  \]
- SSD: 49 + 16 + 1 + 1 + 1 + 9 + 4 + 1 + 4 + 9 = 94
- \(94\) is the lowest cost
- Output range: 0 ≤ SSD ≤ NL

**Sum of Absolute Differences**
- SSD is sensitive to isolated noise
- SSD uses absolute differences instead of square differences
- More robust against noise but is not differentiable

**Cross-Correlation**
- Measure how similar two multi-dimensional vectors are using the dot product between the two
  \[
  CC(F, T)_{(p,q)} = \sum \sum F[p + u, q + v] \times T[u, v]
  \]
- Output range: 0 ≤ CC ≤ NL²

**CC Example**
- CC between a 2D image & a template patch
  \[
  \begin{bmatrix}
  8 & 6 & 2 & 1 \\
  1 & 2 & 1 & 6 \\
  2 & 5 & 3 & 4 \\
  1 & 3 & 0 & 5 \\
  \end{bmatrix}
  \quad \odot \quad
  \begin{bmatrix}
  1 & 2 & 1 \\
  2 & 5 & 3 \\
  1 & 3 & 0 \\
  \end{bmatrix}
  \Rightarrow
  \begin{bmatrix}
  54 & 54 & 70 & \cdots \\
  54 & 50 & 67 & \cdots \\
  \cdots & \cdots & \cdots & \cdots \\
  \end{bmatrix}
  \]
- CC: 8 + 12 + 2 + 2 + 10 + 3 + 2 + 15 + 0 = 54
- \(54\) is the highest match
- Output range: 0 ≤ CC ≤ NL²

**The Problem of CC**
- Relationship between SSD and CC:
  \[
  SSD(F, T)_{(p,q)} = \sum \sum F[p + u, q + v] - T[u, v] = \sqrt{\sum \sum (F[p + u, q + v] - T[u, v])²} = \sqrt{\sum \sum (F[p + u, q + v] + T[u, v] - 2F[p + u, q + v] \times T[u, v])²}
  \]
- The value of CC is influenced by:
  - The value of SSD
  - The template image is fixed
- Brighter areas in the image tend to have higher score

**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
**Zero-mean Normalized CC**

- Measure the dot product between two zero-mean adjusted vectors
  \[ \text{ZNCC}(F, T) \equiv \text{NCC}(F - \bar{F}_u, T - \bar{T}) \]
  \[ \equiv \frac{\sum \sum (F[p+u][q+v] - \bar{F}_u)(T[u][v] - \bar{T})}{\sqrt{\sum \sum (F[p+u][q+v] - \bar{F}_u)^2 \cdot \sum \sum (T[u][v] - \bar{T})^2}} \]

- Output range:
  \[ -1 \leq \text{ZNCC} \leq 1 \]

**Updated Variance Calculation**

- Based on derivation:
  \[ \text{var}(F) = \sum \sum (F[u][v] - \bar{F}_u)^2 \]
  \[ = \sum \sum (F^2[u][v] - 2F[u][v]\bar{T} + \bar{F}_u^2 - \bar{F}_u^2) \]
  \[ = \sum (\sum F^2[u][v] - N \bar{F}_u^2) \]

- Involve only one pass:
  Faster, but same complexity

**ZNCC Calculation**

- Similarity:
  \[ \text{cov}(F, T) = \sum \sum F[p+u][q+v] \cdot T[u][v] - N \bar{F}_u \bar{T} \]

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

**Naïve Variance Calculation**

- Require 2 passes:
  - First pass computes mean
  - Second pass computes variance

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

- float var = 0;
  for (int u=-w; u<=w; u++)
    for (int v=-h; v<=h; v++)
      var += (T[u][v] - mean) * (T[u][v] - mean);

**Normalized CC**

- Measure the dot product between 2 normalized vectors
- \[ \text{NCC}(F, T) = \frac{\sum \sum (F[p][q] \cdot T[u][v])}{\sqrt{\sum \sum F[p][q]^2 \cdot \sum \sum T[u][v]^2}} \]

- Output range:
  \[ 0 \leq \text{NCC} \leq 1 \]