Matching SIFT

Template
Matching SIFT

Target image
Matching SIFT Using Nearest Neighbor

Matching result
Matching SIFT Using Nearest Neighbor

Matching points whose ratio
(best_match_cost / second_best_match_cost) < 0.7
RANSAC

- Generate matching hypothesis
RANSAC

- Generate matching hypothesis

Randomly choose 3 pairs
RANSAC

- Generate matching hypothesis

Transform all the template Points to the target image

Template

Target
Affine Transformation

Point \((x,y)\) is mapped to \((u,v)\) by the linear function:

\[
\begin{align*}
u &= ax + by + c \\
v &= cx + dy + e
\end{align*}
\]
Affine Transformation

Point \((x,y)\) is mapped to \((u,v)\) by the linear function:

\[
\begin{align*}
    u &= a \cdot x + b \cdot y + c \\
    v &= c \cdot x + d \cdot y + e
\end{align*}
\]

In matrix format:

\[
\begin{bmatrix}
    u \\
    v
\end{bmatrix} = \begin{bmatrix}
    a & b \\
    c & d
\end{bmatrix} \begin{bmatrix}
    x \\
    y
\end{bmatrix} + \begin{bmatrix}
    c \\
    e
\end{bmatrix}
\]
Affine Transformation

Point \((x,y)\) is mapped to \((u,v)\) by the linear function:

\[
\begin{align*}
    u &= a \ x + b \ y + c \\
    v &= c \ x + d \ y + e
\end{align*}
\]

Using homogeneous coordinates:

\[
\begin{bmatrix}
    u \\
    v \\
    1
\end{bmatrix} =
\begin{bmatrix}
    a & b & c \\
    c & d & e \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    x \\
    y \\
    1
\end{bmatrix}
\]
Matlab

t = cp2tform(src_points, target_points, ‘affine’);

src_points: (x1 y1; x2 y2; x3 y3; …)
target_points: (u1 v1; u2 v2; u3 v3; …)

\[ q = [x \ y \ 1] * t.tdata.T; \quad \% q = [u \ v \ 1] \]

Other transformations: ‘similarity’, ‘projective’
Match points to the closest SIFT target points and compute the overall SIFT feature difference.
Demo and Assignment Discussion

Template

Target
Feature-based alignment outline

Source: L. Lazebnik
Feature-based alignment outline

- Extract features

Source: L. Lazebnik
Feature-based alignment outline

- Extract features
- Compute *putative matches*

Source: L. Lazebnik
Feature-based alignment outline

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)

Source: L. Lazebnik
Feature-based alignment outline

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  - *Verify* transformation (search for other matches consistent with $T$)

Source: L. Lazebnik
Feature-based alignment outline

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  - *Verify* transformation (search for other matches consistent with $T$)

Source: L. Lazebnik