RoboCup 2010

6. Template Matching

Template Matching

Goal:

- Finding small parts of an image which match a template image
  seam image: I_S, template image: I_T
- **Feature-based**:
  - Derive and search features (edges, corners) in I_S and I_T
  - Advantage: Faster in case of large images
- **Template-based**:
  - Uses entire template (pixel values):
    Move I_T over each point of I_S and calculate
    sum of products / sum of absolute differences / correlation
  - Use position with the highest score
  - Advantage: more robust when no strong features are detected

Computational effort

Example:

- Source image size: 2000x1500 pixels,
- Template size: 200x100 pixels
- Comparisons of pixels:
- (2000-200) * (1500-100) * 200*100 > 50 * 10e9 operations

Speed improvements:

- Downsample I_S and I_T by the same factor (multi-resolution / image pyramids)
- After detection of optimal position, use original image and detect template in the local neighborhood

Quality Improvements

- Use a suitable **color space**: e.g., do not use a gray scale image if you want to distinguish red objects on a green background
- Smooth image (Gaussian) in case of **noise**
- Define a **suitable template**:
  e.g., think about the background color


marker on robot:

- Do you need invariance to scaling and rotation?
  - Keep the distance between camera to objects fixed
  - Use rotation invariant templates (circle)
- Luminance changes?
  - Use correlation instead of pixel differences

Metrics

- Sum of absolute differences

$$SAD(x,y) = \sum_{i=1}^{T_{WIDTH}} \sum_{j=1}^{T_{HEIGHT}} | I_S(x+i, y+j) - I_T(i,j) |$$

- Sum of products

$$SP(x,y) = \sum_{i=1}^{T_{WIDTH}} \sum_{j=1}^{T_{HEIGHT}} [I_S(x+i, y+j) * I_T(i,j)]$$

- Correlation

$$cor(x,y) = \frac{\sum_{i=1}^{T_{PIXEL}} (T_i - \overline{T}) * (S_i - \overline{S})}{\sqrt{\sum_{i=1}^{T_{PIXEL}} (T_i - \overline{T})^2 * \sum_{i=1}^{T_{PIXEL}} (S_i - \overline{S})^2}}$$

$\boldsymbol{T_i}$ / $\boldsymbol{S_i}$: value of pixel ***i*** in template / source image