SKETCH RECOGNITION USING ZERNIKE MOMENTS

Artificial Intelligence
Abhash Anand
Abhisheek Kar
ALGORITHM OVERVIEW

- Divide the model as well as scene into patches.
- Compute Zernike moments of these patches.
- For every patch in the model find the nearest patch in the scene.
- After obtaining the nearest scene patch for every model patch we compute the proximity scores for every patch.
- The proximity scores are based on neighborhood relationships.
- The patch having the highest score is considered best.
ZERNIKE MOMENTS & PROXIMITY SCORE

- The image being a vector in a very high dimensional space is projected onto a lower dimensional Zernike space.
- The basis vectors are the orthogonal Zernike Vectors. Projection of image along each are the Zernike Moments.
- Two key points used:
  - Neighboring patches should be rotated by the same angle as the current patch we are considering.
  - The scene patch matched by neighboring patches should be at the similar distance as in the model.
**FORMULAE**

- **Zernike Moment**:
  \[
  Z_{pq} = \frac{p + 1}{\pi} \int \int_{x^2 + y^2 \leq 1} V^*_{pq} f(x, y) \, dx \, dy
  \]

- **Zernike Basis Function**
  \[
  V_{pq}(x, y) = V_{pq}(\rho \cos \theta, \rho \sin \theta) = Z_{pq}(\rho) \exp(jm\theta)
  \]

- **Zernike Basis Function (Radial Part)**
  \[
  R_{pq}(\rho) = \sum_{k=|q|, |p-k| \text{even}}^{p} \frac{(-1)^{\frac{p-k}{2}} (p+k)!}{p-k! \frac{k-p}{2} ! \frac{k+q}{2} !} \rho^k
  \]

- **Proximity Score**
  \[
  \alpha_{v,i} = \frac{1}{N_{vu}} \sum_{j:j \neq i, \ \delta_e(h_{vi}, h_{vj}) \leq T_e} \exp \left\{ - \frac{[d_e(h_{vi}, h_{vj}) - d_e(c_{svi}, c_{svj})]^2}{2\sigma_{de}^2} \right\} \times \exp \left\{ - \frac{[d_{za}(h_{vi}, c_{svi}) \circ d_{za}(h_{vj}, c_{svj})]^2}{2\sigma_{za}^2} \right\}
  \]
IMPLEMENTATION DETAILS

- Implementation in MATLAB by us.
- Complete implementation of the algorithm specified in the paper.
- Code for calculating Zernike Moments implemented in C and OpenCV originally.
- MATLAB code for computing Zernike Moment obtained from the web.
- Scale invariant sketch recognition in the range 0.8 to 1.2.
- Scene and model scaled down to 300 (max dimension) for practical computation.
- Affine transformations, scale transformations and partial occlusions taken care of.
RESULTS

- Found the algorithm excellent for calculating the in-plane rotation angle of sketches (error $\approx 2^\circ$).
- Tried variations but found no improvements over the paper.
RESULTS CONT'D.
PROBLEMS FACED & FURTHER WORK

- Difficult to determine the threshold for recognizing the sketch.
- High complexity of the algorithm (NP complete)
- Obtain 2D views of 3D models and get viewpoint
- Try framing heuristics for threshold determination
- Obtain original sketches (as opposed to line drawings from the internet) and evaluate the algorithm on them.