3D Object Recognition using Multiclass SVM-KNN

R. Muralidharan, C. Chandradekar

April 29, 2014

Presented by: Tasadduk Chowdhury

Problem

- We address the problem of recognizing 3D objects based on various views.
- The objective is to identify a 3D object based on its 2D image taken from any given angle.
- The 3D object recognition has been a prominent research area for last two decades. Recently view-based 3D object recognition has attracted many researchers in the community of machine learning.

Outline

1. Image database used for 3D object recognition.
2. Machine Learning algorithms:
   - Multiclass SVMs
   - K-Nearest Neighbor Algorithm (KNN)
   - SVM-KNN
3. Proposed SVM-KNN based 3D object recognition.
4. Some local and global image features.
5. Experimental results for the proposed method in comparison with other methods.

Image Database

- The COIL (Columbia Object Image Library) database consists of 7,200 images (100 objects, 72 views per object).
- Each image is a color image of size 128×128.
- Objects positioned on a motorized turntable against black background are observed from fixed viewpoint.
- For each object, the turntable was rotated 5° per image.
- Images were normalized to size 128×128 to save disk storage.

Image Database

Figure : Objects from COIL database

(a) View: 0°

(b) View: 45°

(c) View: 300°

(d) View: 0°

(e) View: 45°

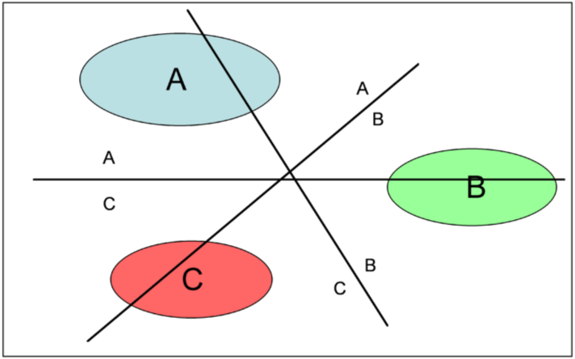
(f) View: 300°

Multi-class SVM

- Many real-world problems deal with more than two classes.
- Multiclass problems involve reformulating them as a number of binary classification problems, and solving these with binary SVMs.
- Two methods to implement a multiclass SVM: one-against-one and one-against-all.

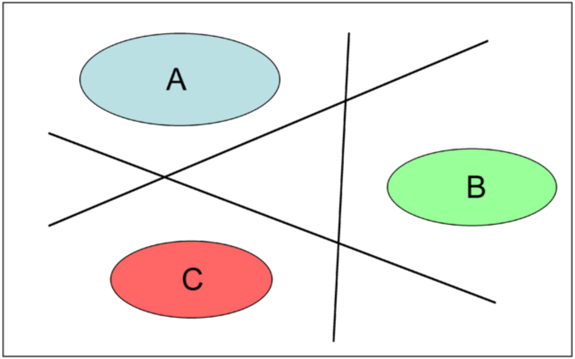
Multi-class SVM

1. *One-against-one:* One binary SVM for each pair of classes ($N(N-1)/2$ SVMs).



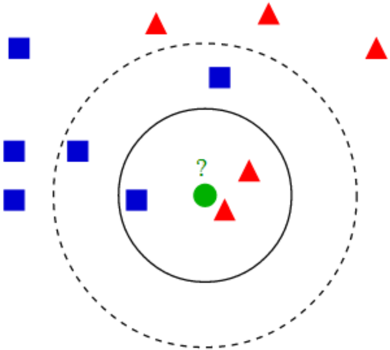
Multi-class SVM

2. *One-against-all:* One binary SVM for each class to separate from the rest (N SVMs).



K-Nearest Neighbor Algorithm (KNN)

Figure : K-nearest neighbors



K-Nearest Neighbor Algorithm (KNN)

- KNN classifies a point based on its k closest neighbors.
- Find the k closest points and choose a label based on majority of its neighbors.
- If $k = 1$, then the object is simply assigned to class of its nearest neighbor.
- Larger value of k reduce the effect of noise on the classification.

SVM-KNN

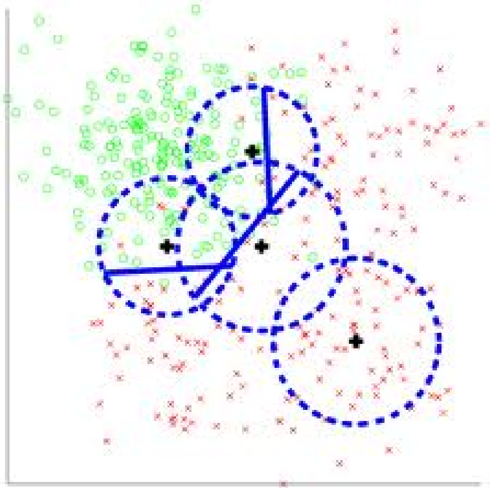
- KNN suffers from the problem of high variance in the case of limited sampling.
- With SVMs, training on a whole dataset can be slow and may not work very well when the number of classes is large.
- Zhang *et al.* (2006) proposed SVM-KNN as a classifier for visual category recognition.
- SVM-KNN can be applied to large multiclass data sets for which it outperforms KNN and SVMs, and remains efficient.

SVM-KNN

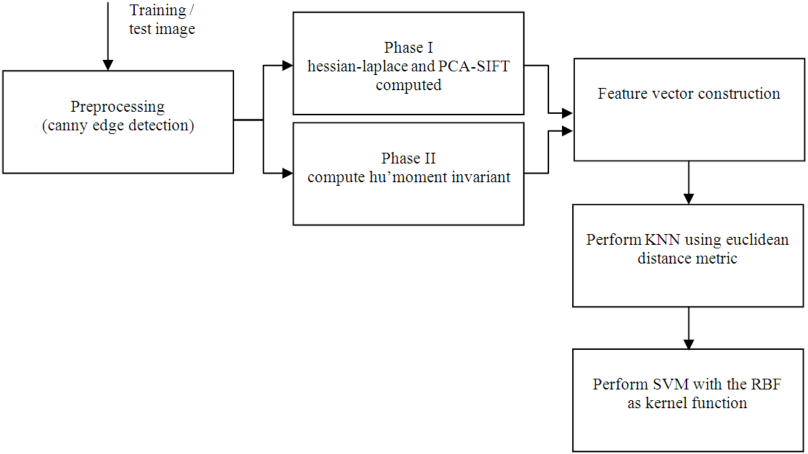
For a query,

1. choose a proper distance function $d(\mathbf{x}, \mathbf{y})$ for the problem. Typically, d is the Euclidean distance function in problems of object recognition.
2. compute distances of the query to all training examples and pick the nearest k neighbors.
3. if k neighbors have all the same labels, the query is labeled and exit; otherwise, compute the pairwise distances between the k neighbors and store in a matrix.
4. convert the distance matrix to a kernel matrix and apply multi-class SVM. A Gaussian function $K(\mathbf{x}, \mathbf{y}) = \exp(-d(\mathbf{x}, \mathbf{y})/\sigma^2)$ can be used as the kernel.
5. use the resulting classifier to label the query.

SVM-KNN



SVM-KNN based 3D Object Recognition



Canny Edge Detection

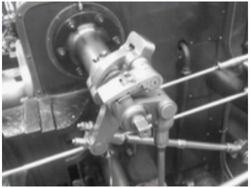
For an image $f(x, y)$:

1. Compute $\partial_x(f * G)$ and $\partial_y(f * G)$ at every point, where $G(x, y)$ is the Gaussian function with parameter σ.

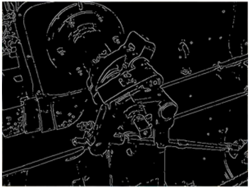
$$\tilde{f}_x = \partial_x(f * G) = f * \partial_x G,$$
$$\tilde{f}_y = \partial_y(f * G) = f * \partial_y G.$$

2. Compute magnitude of the gradient: $M(x, y) = \sqrt{\tilde{f}_x^2 + \tilde{f}_y^2}$.
3. Find local maxima of $M(x, y)$.
4. Apply thresholding/edge linking.

Canny Edge Detection



(a) Original



(b) edges detected

Local Image Features: Hessian-Laplace Detector

- The local features are extracted on a small image window around pixel.
- Corners and blobs on an image can be detected by Hessian-Laplace detector.
- The Hessian of an image $f(x, y)$:

$$Hf(x, y) = \begin{bmatrix} f_{xx}(x, y) & f_{xy}(x, y) \\ f_{yx}(x, y) & f_{yy}(x, y) \end{bmatrix}.$$

- The detector chooses points on the image where $det(Hf)$ reaches a local maximum.
- Such points are translation, scale, and rotation invariant.

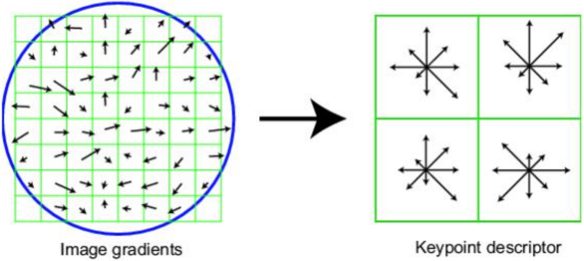
Local Image Features: SIFT

- Scale-invariant Feature Transform (SIFT) is a local feature extraction method (David Lowe, 2004).
- Generally SIFT has a high dimension of 128 features, but Principal Component Analysis (PCA) can be utilized to reduce the number of features.
- Choose an interesting point using the Hessian-Laplace detector and then apply PCA-SIFT on this point.

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor



Adapted from slide by David Lowe

Global Image Feature: Geometric Moments

- For a $M \times N$ image $f(x, y)$, the $(p+q)$'th order *moment* is

$$m_{p,q} = \sum_{y=0}^{M-1} \sum_{x=0}^{N-1} x^p y^q f(x, y), \quad p, q = 0, 1, 2, \cdots .$$

- The *central moment*:

$$\mu_{p,q} = \sum_{y=0}^{M-1} \sum_{x=0}^{N-1} (x - \overline{x})^p (y - \overline{y})^q f(x, y), \quad p, q = 0, 1, 2, \cdots ,$$

where $\overline{x} = m_{1,0}/m_{0,0}$ and $\overline{y} = m_{0,1}/m_{0,0}$.

Global Image Feature: Geometric Moments

- *Normalized moments:*

$$\eta_{p,q} = \frac{\mu_{p,q}}{\mu_{0,0}^{\gamma}}, \quad \gamma = \frac{p+q}{2} + 1.$$

- Hu's Geometric Moments: seven values using the normalized moments up to order three.
- Translation, scale, rotation invariant.
- First three of Hu's geometric moments:

$$\begin{aligned} M_1 &= \eta_{2,0} + \eta_{0,2} \\ M_2 &= (\eta_{2,0} - \eta_{0,2})^2 + 4\eta_{1,1}^2 \\ M_3 &= (\eta_{3,0} - 3\eta_{1,2})^2 + (3\eta_{2,1} - \eta_{0,3})^2. \end{aligned}$$

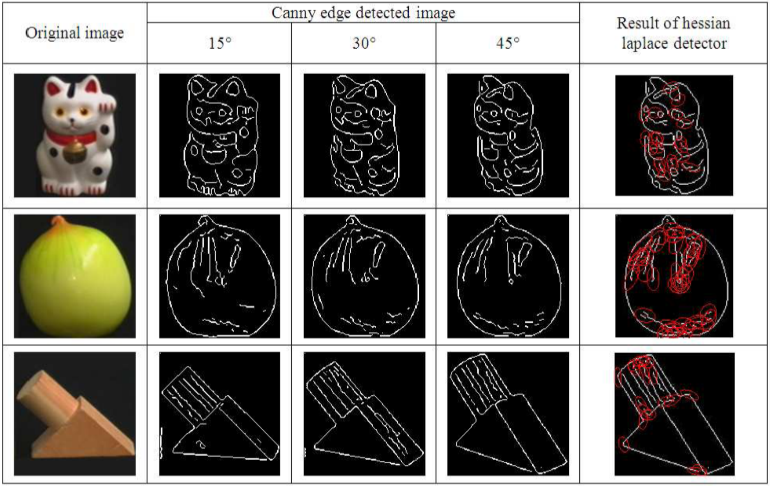
Feature Vector Construction

The following steps are taken to extract a feature vector from the images of a given object:

1. Preprocess: convert the color images to greyscale and resize to 100×100; apply edge detection on the images and extract the following features.
2. Local features: extract 36 PCA-SIFT descriptors from a point detected by Hessian-Laplace detector.
3. Global features: Hu's seven geometric moments.
4. All 43 features are stored in a vector. These vectors are used to train the SVM.

Feature Vector Construction

Figure : Feature Extraction.



Training Phase

- Images of objects are given as input to the system.
- Preprocess the images.
- Extract local and global features.
- Constructed feature vector is stored with object label.
- Train the SVM.

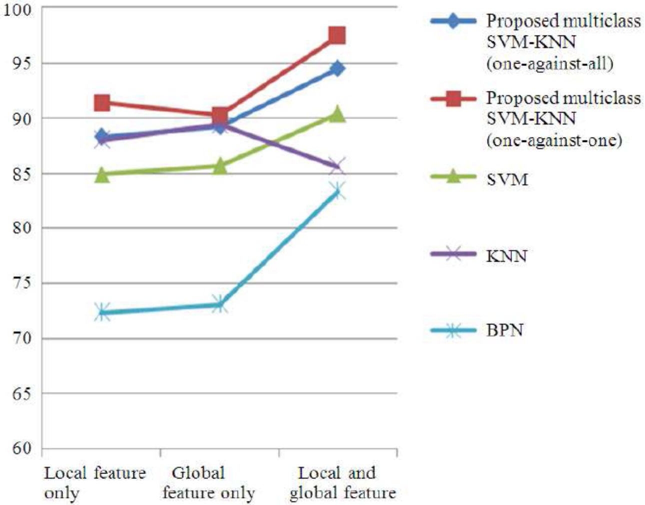
Testing Phase

- Input: image of an object.
- Preprocess and construct a feature vector as in the training phase.
- Find k closest feature vectors from the training set.
- Use multiclass SVM for classification.

Experimentation

- To experiment the proposed method, the COIL image database was used.
- 100 objects were selected for object recognition.
- For training set, 10 different views per object. (size of the training set $= 1000$).
- For test set, 10 alternate views of the same objects were chosen.
- For comparision, SVM, KNN, and BPN were experimented using the same training and test set.

Results



100
95
90
85
80
75
70
65
60
Local feature only
Global feature only
Local and global feature
Proposed multiclass SVM-KNN (one-against-all)
Proposed multiclass SVM-KNN (one-against-one)
SVM
KNN
BPN

Results

Table 1: Performance rate of the proposed classier for various types of features and compared with other classifiers for COIL-100

Classifier/types of Features	Local feature only	Global feature only	Local and Global features
Proposed multiclass SVM-KNN (one-against-all)	88.30	89.25	94.5
Proposed Multiclass SVM-KNN (one-against-one)	91.40	90.24	97.4
SVM	84.90	85.70	90.4
KNN	88.00	89.40	85.6
BPN	72.40	73.10	83.4

Conclusion

- Combining local and global feature provides better results.
- SVM-KNN classifier has greater accuracy than the traditional methods (SVM, KNN, BPN).
- Future work will include the process of increasing the efficiency by adding more features.

Bibliography

- R. Muralidharan, C. Chandrasekar *3D Object Recognition using Multiclass Support Vector Machine-K-Nearest Neighbor Supported by Local and Global Feature.* 2012.
- H. Zhang, A.C. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. 2006
- D. Lowe. Distinctive Image Features from Scale-invariant Keypoints. 2004.
- M. Hu. Visual Pattern Recognition by Moment Invariants. 1962.