

Virtual Shape Recognition using Leap Motion

EC 520: Digital Image Processing and Communication

Aditya Chechani and Harshil Prajapati

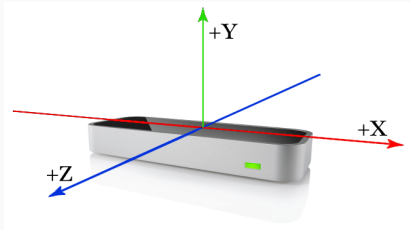
Department of Electrical and Computer Engineering: Boston University

Problem Statement

To develop a robust algorithm to extend Leap Motion capabilities to recognize hand drawn gestures of various 2D shapes (e.g. circle, ellipse, square, triangle, etc.)



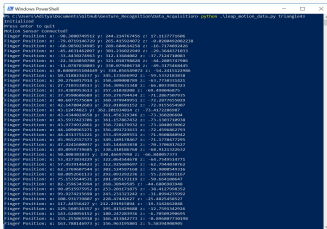
(a) Leap Motion [2]



(b) Coordinates axes of Leap Motion

Data Acquisition

- Leap Motion SDK - Python
- x,y,z coordinates of the tip of the index finger
- Store it in .json file



Shape	Data
Circle	50
Rectangle	50
Triangle	50

Table 1: Generated Data

Figure 2: Capturing Leap data

Data Preprocessing: *Planner Fitting of 3D points*

Shapes will not be on a plane \parallel to XY plane of Leap Motion so we find a best fit plane which has the least squared error with the data points provided [4]. The equation of plane is given by:

$$z = Ax + By + D \quad (1)$$

The coefficients of plane are given by

$$c_x = -\frac{A}{D}; \quad c_y = -\frac{B}{D}; \quad c_z = \frac{1}{D}$$

The normal vector of plane is

$$n_p = \left[-\frac{A}{D} \quad -\frac{B}{D} \quad \frac{1}{D} \right] \quad (2)$$

Normal vector to X-Y plane

$$n_{XY} = [0 \quad 0 \quad 1]$$

Data Preprocessing: *Planner Fitting of 3D points*

Rotation Vector (u):

$$u = \frac{n_p}{|n_p|} \times n_{XY} \quad (3)$$

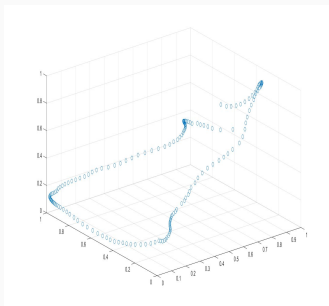
Rotation angle (θ):

$$\theta = \sin^{-1}(|n_p|) \quad (4)$$

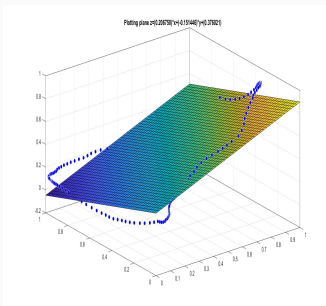
Using the rotation vector and the angle we calculate rotation matrix "R" we get the transformed data

$$[x_{new}, y_{new}, z_{new}]^T = R[x, y, z]^T \quad (5)$$

Data Preprocessing: *Plane Fitting of 3D points*



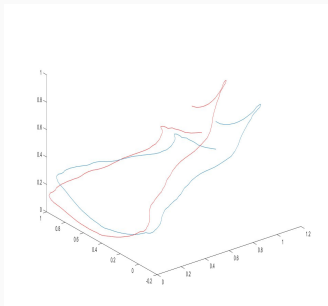
(a) Data points in 3D Space



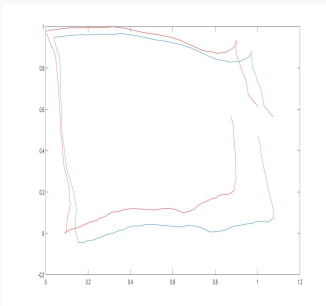
(b) Best Fitted Plane

Figure 3: Plane Fitting of 3D points

Data Preprocessing: *Plane Fitting of 3D points*



(a) Rotated coordinate in 3D space



(b) Coordinates on X-Y plane post processing

Figure 4: Plane Fitting of 3D points

Feature Extraction: *Fourier Descriptor*

Fourier Descriptor: We define

$$s[n] = x_n + iy_n$$

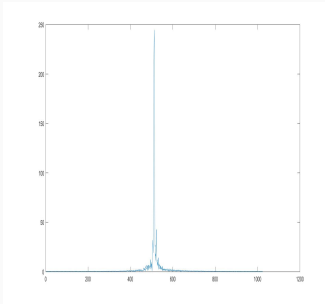
Taking the DFT of $s[n]$

$$a[k] = \frac{1}{N} \sum_{n=0}^{N-1} s[n] e^{\frac{-j2\pi nk}{N}} \quad (6)$$

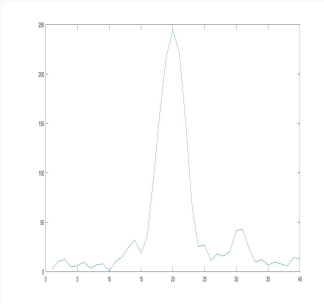
and coefficients can be recovered by

$$s[n] = \sum_{k=0}^{N-1} a[k] e^{\frac{j2\pi nk}{N}} \quad (7)$$

Feature Extraction: *Fourier Descriptor*



(a) Fourier Descriptor of data points.



(b) Considering only 40 high energy points

Classification: k-NN

For a Data set $\mathcal{D} = (\mathbf{x}_1, y_1), \dots (\mathbf{x}_n, y_n)$ with \mathbf{x} as 20 point feature vector from feature extraction and test point \mathbf{x} Let $(\mathbf{x}_1, y_1), \dots (\mathbf{x}_n, y_n)$ be reordered such that

$$d(\mathbf{x}, \mathbf{x}_1) \leq d(\mathbf{x}, \mathbf{x}_2), \dots d(\mathbf{x}, \mathbf{x}_n)$$

where $d(\mathbf{x}, \mathbf{x}_i)$ can be l_p distance

$$\|\mathbf{x} - \mathbf{x}_{test}\|_p := \left(\sum_{i=1}^d |\mathbf{x}_i - \mathbf{x}_{test}|^p \right)^{\frac{1}{p}}$$

For our project we are considering the Euclidean distance (l_2)

$$h_{k-NN} = \underset{y=1..n}{\operatorname{argmax}} \sum_{j=1}^k 1(y_{(j)} = y) \quad (8)$$

where $1(y_{(j)} = y)$ is the number of k-NNs of \mathbf{x} with label =y [5]

Classification: k-NN

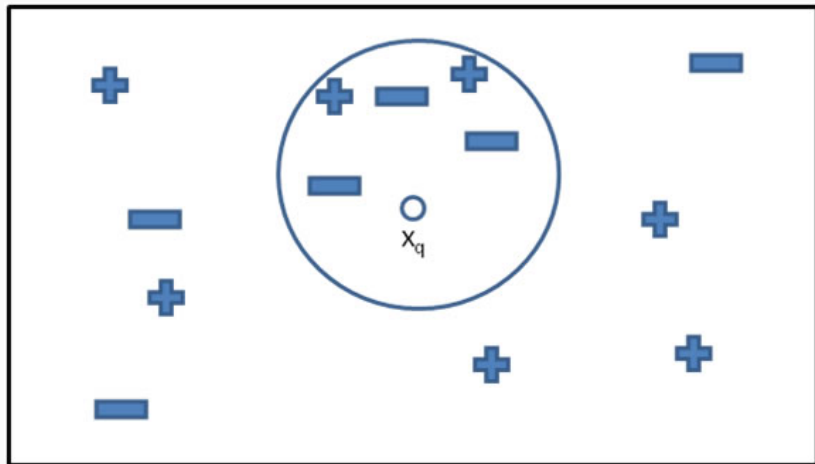


Figure 6: k-NN Example [1]

Classification: k-NN

Predict	Ground Truth		
	10	0	0
	0	9	3
	0	1	7

Table 2: 80:20 Split (NN)

CCR: 86.67%

Predict	Ground Truth		
	10	1	1
	0	8	4
	0	1	5

Table 4: 80:20 Split (3-NN)

CCR: 76.67%

Predict	Ground Truth		
	19	1	1
	1	18	3
	0	1	16

Table 3: 60:40 Split (NN)

CCR: 88.33%

Predict	Ground Truth		
	10	1	1
	0	7	6
	0	2	3

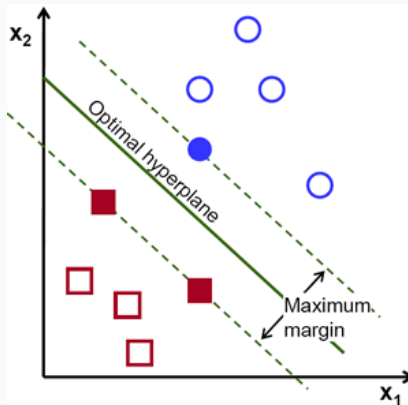
Table 5: 80:20 Split (5-NN)

CCR: 66.67%

Classification: SVM

For multiclass classification we extend binary SVM, using One v/s All (OVA) or One v/s One (OVO) approach.

Figure 7: SVM Example [3]



Results: SVM Classification

One v/s One:

Predict	Ground Truth		
	8	0	0
	2	10	6
	0	0	4

Table 6: $C=16, \sigma 16$

CCR: 73.33%

Predict	Ground Truth		
	10	0	0
	0	9	2
	0	1	8

Table 7: $C=16, \sigma 32$

CCR: 90%

Results: SVM Classification

One v/s All:

- Mean CCR = 71.11% ($C = 2, \sigma = 16$)
- Mean CCR = 88.9% ($C = 2, \sigma = 32$)

C1	Ground Truth		C2	Ground Truth		C3	Ground Truth	
Predict	8	0	Predict	7	1	Predict	6	0
	2	20		3	19		4	20

Table 8: OVO ($C = 2, \sigma = 32$)

Thank You

References i



k-NN Example.

<http://www.d.umn.edu/~deoka001/knn.html>.

Accessed: 2018-04-18.



Leap Motion Development SDK.

<https://developer.leapmotion.com>.

Accessed: 2018-04-09.



Open CV Tutorials: Introduction to Support Vector Machines.

https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html.

Accessed: 2018-04-18.



D. Eberly.

Least squares fitting of data.

Chapel Hill, NC: Magic Software, 2000.



P. Ishwar.

EC 503 - Learning from Data , January - May 2018.